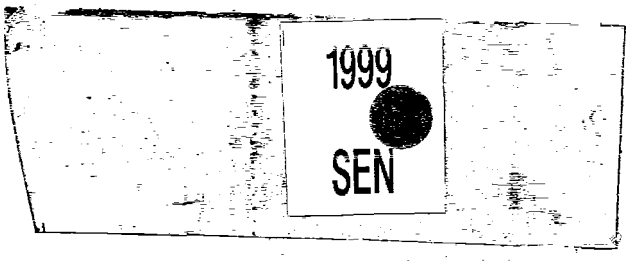
Université Paris 7 - CNRS
UFR d'informatique

Laboratoire
d'Automatique
Documentaire
et Linguistique



LADL, 2 Place Jussieu, F - 75251 Paris Cedex 05

Outils de reconnaissance d'expressions linguistiques complexes dans des grands corpus.

Thèse de doctorat
Jean Senellart
janvier 1999

1999
SEN

UNIVERSITÉ PARIS 7 - DENIS DIDEROT
UFR D'INFORMATIQUE
Laboratoire d'Automatique Documentaire et Linguistique

THÈSE

pour l'obtention du Diplôme de

DOCTEUR DE L'UNIVERSITÉ PARIS 7
SPÉCIALITÉ INFORMATIQUE FONDAMENTALE ET APPLICATIONS

présentée et soutenue publiquement

par

Jean SENELLART

le 19 janvier 1999

Titre :

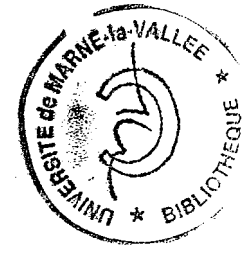
Outils de reconnaissance d'expressions linguistiques complexes dans des grands corpus

---

Directeur :

Maurice Gross

JURY :
M. Robert Cori
M. Maxime Crochemore
M. Gaston Gross
M. Franz Guenthner
M. André Salem
M. Max Silberztein



1

REMERCIEMENTS

Je tiens exprimer en premier lieu ma gratitude envers le Professeur Maurice Gross qui m'a initié à cette discipline prometteuse qu'est la linguistique informatique et m'a fait partager sa passion. Sa présence, son soutien et les discussions que nous avons eues m'ont permis de mener cette étape de mon travail à son terme.

Cette étude n'aurait pu être menée à terme sans le bénéfice de l'environnement exceptionnel du LADL et sans le contact stimulant de ses différents membres. En particulier, mes remerciements vont à Max Silberztein pour de nombreuses discussions et ses conseils, à Christian Leclère pour sa patience et ses réponses à de nombreuses interrogations linguistiques et à Éric Laporte. Mes remerciements vont également à Blandine Courtois, Cédrick Fairon, Philippe Vasseux, Annie Meunier et à toute l'équipe pour leur accueil chaleureux et pour des échanges fructueux.
Je tiens également à remercier l'équipe de l'IGM et en particulier Maxime Crochemore, Dominique Perrin, et Mathieu Raffinot pour le soutien et les conseils qu'ils m'ont apportés.
Je dois aussi beaucoup à l'équipe d'Antoinette Renouf à Liverpool qui m'a accueilli et fait découvrir un autre regard sur le langage naturel.

Enfin, je remercie tous les membres du jury d'avoir bien voulu assister à ma soutenance.

à Pascale et à ma petite famille qui s'agrandit d'Agathe

Pour cette version définitive (...rien n'est jamais définitif) du manuscrit, je dois beaucoup à Éric Laporte qui a consciencieusement relu et vérifié les lemmes, formules et propositions techniques. Merci également à mon ami Franck qui a aussi eu le courage de faire une très complète relecture en affrontant malgré son *emploi du temps de ministre* (mot composé ternaire !). Merci également à Anne D. qui a, comme d'ordinaire, fait preuve de beaucoup de zèle...

Table des matières

1 Introduction 5
1.1 Contexte . . . 5
1.2 Convention et guide de lecture . . . 7
1.2.1 Conventions linguistiques . . . 7
1.2.2 Typographie . . . 7
1.2.3 Guide de lecture . . . 7

I Méthodologie, outils 9

2 Les Ambiguïtés 13
2.1 L'existence des ambiguïtés . . . 14
2.1.1 Exemples . . . 14
2.1.2 L'importance statistique des ambiguïtés . . . 15
2.1.3 L'origine des ambiguïtés . . . 17
2.2 La levée d'ambiguïté. Les approches locales . . . 18
2.2.1 Grammaire locale de désambiguïsation . . . 18
2.2.2 Les approches statistiques . . . 22
2.2.3 Conclusion . . . 25
2.3 Source des ambiguïtés, le langage ou la description? . . . 26
2.3.1 La grammaire de *majuscule* . . . 26
2.3.2 Le genre des adjectifs . . . 27
2.3.3 Le déterminant *beaucoup* . . . 27
2.3.4 Les étiquetages inutiles . . . 29
2.3.5 Conclusion . . . 29
2.4 Orientation générale . . . 30
2.4.1 Les ambiguïtés résiduelles . . . 30

3 Les ressources linguistiques 31
3.1 L'objectif . . . 31
3.2 Les unités élémentaires . . . 36
3.3 La grammaire . . . 37
3.3.1 L'utilisation du dictionnaire . . . 38
3.3.2 Les transformations . . . 41
3.3.3 Le lexique-grammaire . . . 48
3.3.4 Grammaire d'analyse et grammaire de description . . . 50
3.3.5 Interdépendance des phénomènes linguistiques . . . 52

3.3.6 Conclusion 55
3.4 Les contraintes pratiques 56

4 Les outils informatiques **57**
4.1 La représentation du texte 58
4.2 Les graphes 61
4.2.1 Les automates, représentation duale 61
4.2.2 L'utilisation du dictionnaire 62
4.2.3 Les sous-graphes, ou les graphes récursifs 62
4.2.4 Transformations 65
4.2.5 Gestion de contraintes 72
4.2.6 Les graphes patrons 72
4.2.7 Les opérateurs spéciaux 74
4.2.8 Les opérations sur les numéraux 76
4.3 Projection 83
4.4 Conclusion 86

II Les expressions figées **91**
4.5 Introduction 93
4.6 La reconnaissance 94
4.7 L'objectif 94
4.8 La normalisation 95
4.9 Expressions composées et expressions simples. 96
4.10 La grammaire de reconnaissance 97
4.11 Les limites actuelles 98

5 Les adverbes figés **101**
5.1 Les adverbes figés des tables P 103
5.1.1 Quelques chiffres 110
5.2 Les autres adverbes 115
5.2.1 Les adverbes provenant des tables EPX 115
5.2.2 Les adverbes provenant des tables A 115
5.2.3 Les adverbes de date, de durée, et de fréquence 117
5.3 La mobilité adverbiale 118
5.3.1 Vérification de contraintes 118
5.3.2 Mobilité restreinte 121
5.3.3 Les adverbes portant sur des groupes nominaux 121
5.4 Conclusion 124

6 Négation **127**
6.1 La négation *ne...pas* 128
6.1.1 Structure 128
6.1.2 Portée de la négation, interprétation 129
6.1.3 Négation et groupes nominaux 132
6.2 Les autres négations 137
6.2.1 *ne...plus* 137
6.2.2 *ne...jamais* 138

6.2.3 *ne...point* . . . 138
6.2.4 *ne...(aucunement+nullement)* . . . 138
6.2.5 *ne...guère* et *ne...plus guère* . . . 139
6.2.6 *ne...que* . . . 140
6.2.7 Négations composées . . . 141
6.2.8 Autres négations non compositionnelles . . . 142
6.3 Conclusion . . . 145

7 Groupe nominal et structure verbale figée **147**
7.1 Le groupe nominal . . . 147
7.1.1 Introduction . . . 148
7.1.2 Les différents niveaux de groupes nominaux . . . 158
7.2 Les structures verbales figées . . . 159
7.2.1 La table `C1IPN` . . . 159
7.2.2 Limites de la reconnaissance . . . 163
7.2.3 Les différentes tables appliquées . . . 165
7.2.4 Extrait de texte transformé . . . 168
7.3 Conclusion . . . 170

III Indexation **173**

8 Introduction **175**

9 L'indexation de textes **179**
9.1 Définitions . . . 179
9.1.1 Désambiguïsation terminologique . . . 181
9.2 Structure induite par l'index . . . 182
9.3 Opérations de base . . . 183
9.4 Propriétés des index et des opérateurs . . . 186
9.4.1 Le calcul de $I_\tau^* = I_{\tau^*}$ et de I^n . . . 189
9.4.2 Quelques index spéciaux . . . 190
9.5 Complexité réelle et complexité théorique . . . 191
9.6 Pattern matching à l'aide des index . . . 192
9.6.1 Algorithmes existant de *pattern-matching* sur un texte utilisant une structure précalculée. . . . 193
9.6.2 Recherche de séquences formées de lettres . . . 195
9.6.3 Recherches de mots décrits par des index de longueur variable . . . 206
9.6.4 Recherche d'expression régulière . . . 208
9.7 Indexation d'automate. . . . 223
9.8 Limites de l'indexation . . . 225
9.9 Conclusion . . . 225

10 Flot **227**
10.1 Introduction . . . 227
10.2 Rappels, définitions . . . 228
10.3 Coupe minimale et ensemble pivot . . . 231
10.3.1 La matrice chemin-transition . . . 231

10.3.2 Les nouvelles contraintes du problème (min_2). 233
10.3.3 Chemins améliorants et flots propres 237
10.3.4 Nouvelle définition d'un flot admissible 237
10.3.5 L'algorithme des lettres pivots 238
10.3.6 Flux maximal et équilibrage. 239
10.4 Conclusion 243

IV Expressions figées et Statistique **247**

11 Expressions figées et Statistique **249**
11.1 Introduction 249
11.2 Statistique textuelle, outils et utilisation. 252
11.2.1 Les outils : du texte aux valeurs numériques 252
11.2.2 L'utilisation des outils : des valeurs numériques au sens 256
11.3 Nature de l'information 256
11.3.1 Information et entropie 256
11.3.2 Entropie et langage naturel 258
11.4 La composante linguistique 260
11.4.1 Le décompte des unités d'information 260
11.4.2 La formation des phrases 262
11.4.3 L'approximation de la grammaire 264
11.4.4 Les figés 265
11.5 Conclusion 271

V Conclusion, Perspectives **275**

A Grammaire utilisée par les transducteurs étendus **279**
A.1 Valeurs numériques 279
A.2 Formes 280
A.3 Mot 280
A.4 Nombre 280
A.5 Séquence 281

B La loi de Zipf **283**

Chapitre 1

Introduction

1.1 Contexte

Soit S une séquence formée par la concaténation de lettres de l'alphabet. À la question suivante : «Est-ce que S forme une phrase acceptable en français ?», tout interlocuteur français peut répondre d'une manière binaire par «oui» ou «non». Un des enjeux du traitement automatique du langage naturel est de construire une procédure permettant de simuler ce test d'acceptabilité d'une séquence au langage. D'une manière formelle, le français est un ensemble potentiellement infini de phrases. Simuler le test d'acceptabilité consiste à décider de l'appartenance d'un élément à cet ensemble et nécessite donc de pouvoir décrire cet ensemble.
Historiquement, ce problème a été traité en intégrant la grammaire des langues naturelles dans le cadre de formalismes informatiques sur lesquels les procédures de calcul sont bien définies. Parmi ces formalismes, on distingue les langages définis par des expressions régulières et les langages dits *context-free*. Les premiers ont fait l'objet de vives critiques quant à leur incapacité à traiter le langage naturel. Les seconds, beaucoup plus «puissants», posent des problèmes informatiques de manipulation. De nombreux formalismes intermédiaires ont alors vu le jour, intégrant des capacités supplémentaires par rapport aux expressions régulières, des facilités de calcul informatique et la modélisation de phénomènes linguistiques [1] ; la problématique reste basée autour de la recherche d'un compromis «puissance du formalisme»/«efficacité». Cette approche très théorique du langage naturel a été critiquée, car l'apport linguistique à ces formalismes réside essentiellement en des exemples ponctuels et non sur une étude systématique du langage, comme le nécessiterait une approche scientifique rigoureuse. Il faut ajouter que la partie linguistique de ces formalismes est fortement inspirée de celle des grammaires à usage «humain».
Au cœur du débat, se trouve la séparation faite entre les grammaires et le lexique. La tentative d'implémentation d'une grammaire se fait au niveau des catégories grammaticales, chaque catégorie pouvant ensuite être substituée par des classes entières de mots du lexique.
L'approche systématique du LADL sur le français a conduit à l'élaboration de ressources constituées de tables de lexique-grammaire et d'une description de grammaires dites locales basées sur le formalisme des automates finis. Les unités élémentaires dans ces res-

1. Comme les transformations permettant de rendre équivalentes des structures très différentes et d'opérer des manipulations sur ces structures.

sources sont essentiellement des mots du lexique et non pas des catégories grammaticales.

Cette différence radicale d'approche peut être vue comme l'opposition entre l'exception dans une grammaire «générale» et unité figée dans une grammaire basée sur le lexique. En effet, dans une grammaire que nous dirons «générale», les règles sont basées sur des catégories grammaticales. Un problème important se pose: celui des exceptions, abondantes dans les grammaires scolaires. Une exception est «une structure ne se comportant pas comme les autres». À cette notion d'exception est donc liée la notion de comportement général. On ne peut donc dire qu'une forme d'une construction est une exception que si on a examiné l'ensemble des autres formes et non pas un extrait d'exemples bien choisis. D'un autre côté, l'approche basée sur le lexique répertorie l'ensemble des formes correspondant à une construction et sépare les caractéristiques partagées et celles qui diffèrent pour chacune de ces constructions. Cette approche décrit le comportement individuel de mots simples comme les verbes simples, mais aussi le comportement de groupes de mots qui forment des expressions figées. La notion de figement à laquelle nous faisons référence est générale «*est figée toute séquence composée dont la composition syntaxique ou sémantique n'est pas libre*». La notion de liberté faisant alors référence à la grammaire des formes simples c'est-à-dire définies en termes de catégories grammaticales. Le figement est donc exprimé d'une manière analogue à la notion d'exception. Mais le critère de décision est maintenant basé sur une étude exhaustive de la grammaire de référence. L'opposition entre «exception dans une grammaire générale» et «unitée figée» est aussi une opposition de fréquence: l'exception est censée être ... exceptionnelle donc rare. Nous montrerons qu'en fait, les expressions figées correspondent à la moitié des formes du langage.

Notre travail se situe dans ce cadre. Puisque le comportement des formes figées n'est pas celui des formes «libres», il faut que leur reconnaissance soit spécifique. Nous proposons un formalisme basé sur les automates finis permettant de reconnaître[2] ces expressions figées. Pour les ressources exprimées sous formes de tables du lexique-grammaire, ce formalisme s'inspire du formalisme décrit par E. ROCHE [19] pour reconnaître les entrées du lexique-grammaire des verbes simples du français. Il se définit par ailleurs comme une extension naturelle de la description par automate fini et sa construction est totalement guidée par les besoins linguistiques. La séparation de l'outil informatique correspondant à ce formalisme et de la grammaire est impossible à effectuer. Notre présentation sera basée sur une abondance d'exemples concrets[3] qui présentent un caractère scientifique rigoureux à cette science expérimentale qu'est la linguistique et imposent au formalisme une contrainte bien plus importante que sa puissance théorique d'expression: sa «**puissance d'expression pratique**». Notre description est basée sur des ressources existantes: nous n'introduisons pas de nouvelle formalisation linguistique, mais nous détaillons celles existantes pour aboutir à un résultat concret. Ce résultat est exprimé par un programme, mais surtout par des chiffres donnant le nombre de séquences localisées. D'un point de vue algorithmique, nous sommes confrontés à un problème nouveau et propre à notre

2. Nous définirons précisemment ce que nous entendons par reconnaissance. L'idée principale est que l'étape de reconnaissance n'est qu'une étape intermédiaire à interprétation ultérieure, et donc pour cela, la structure obtenue doit contenir suffisamment d'information pour ne pas nécessiter un retour à des outils syntaxiques dans cette phase d'interprétation.

3. Que nous extraierons autant que possible de ressources textuelles existantes.

discipline: nous devons reconnaître un nombre très important d'expressions régulières (de l'ordre de 50 000) dans un graphe acyclique et ce, de manière itérative et non ordonnable. La structure naturelle permettant cette reconnaissance est la structure d'index de texte et d'automate. Nous présentons des algorithmes nouveaux de reconnaissance de motifs dans ces structures. L'algorithmique exposée n'est donc pas un point de départ du formalisme mais une contrainte d'implémentation.

1.2 Convention et guide de lecture

1.2.1 Conventions linguistiques

Si nous avons essayé d'éviter au maximum les termes techniques linguistiques, plusieurs notations subsitent qui permettent d'alléger la description linguistique. Les phrases notées P sont souvent représentées à l'aide de leur structure comme N_0 V N_1 *Prép* N_2. Les N_i représentent les différents groupes nominaux principaux de la phrase. N_0 est le groupe nominal sujet, N_1 est le groupe nominal objet, et N_2 représente un second complément du verbe. *Prép* représente une préposition quelconque. Le V représente le verbe principal de la phrase. Nous empruntons aux grammaires formelles l'écriture de règles de réécriture de la forme suivante *du*$\longrightarrow$ *de le*. Cette notation montre une règle transformant la forme simple *du* en deux unités *de* et *le*. De ces règles se déduisent un parenthésage des séquences reconnues que nous utilisons également.

1.2.2 Typographie

Nous donnons au fil des chapitres un grand nombre de phrases exemples pour illustrer nos propos. Ces exemples sont normalisés sous la forme suivante:

(1) Phrase extraite du corpus.
(2) *Phrase fabriquée de toute pièce.*
(3)* *Phrase incorrecte, ou dont l'analyse ne correspond pas à celle attendue.*
(4)? *Phrase douteuse.*

La police de caractère différencie les exemples extraits d'un corpus (quotidien) comme la phrase (1) et les exemples «théoriques» construits pour mettre en valeur un phénomène comme dans les phrases (2) à (4). Les symboles «*» et «?» permettent d'indiquer si les phrases sont incorrectes (suivant l'interprétation en cours) ou douteuse.

Comme illustration, nous présentons aussi des données autres que des phrases de la manière suivante:

Exemple : Ceci est un exemple.

Dans chaque chapitre, chaque terme nouveau est imprimé en **gras**, et un index des différents termes à la fin de l'ouvrage renvoit aux définitions de chaque terme utilisé. À chaque partie est associée une bibliographie propre.

1.2.3 Guide de lecture

Cette thèse est organisée en trois parties principales, d'une partie servant d'ouverture, et de deux annexes. La première partie «*Méthodologie, outils*» permet de situer la démarche

informatique que nous avons adoptée dans le contexte du traitement automatique de langue naturelle. Nous y présentons d'une manière constructive le formalisme choisi.
La seconde partie «*Les expressions figées*» présente le problème linguistique en détail et montre la manière de «programmer» la grammaire des expressions figées à l'aide du formalisme présenté.
La troisième partie «*Indexation*» décrit la structure d'index naturellement associée au formalisme et montre les algorithmes permettant de la manipuler.
Dans la dernière partie, «*Statistique et unités figées*», nous étudions les liens entre les unités figées et une application courante : l'extraction d'information. Nous y montrerons que la notion d'unité figée peut naturellement se combiner à des méthodes statistiques d'analyse, et qu'il s'agit même d'une nécessité pour approfondir ces analyses.
Enfin, nous conclurons en comparant les résultats obtenus aux objectifs initiaux et nous donnerons plusieurs pistes de travail et les perspectives possibles.

Première partie

Méthodologie, outils

Introduction

Le LADL a commencé en 1967 une description complète de la langue française, cette description comporte désormais différentes ressources, dont des dictionnaires électroniques, le lexique-grammaire, et un grand nombre de grammaires locales sous forme de graphes. Les dictionnaires éléctroniques sont complètement normalisés et contenus dans des fichiers électroniques. Le lexique-grammaire consiste en une série de tables associant à des entrées représentant des structures élémentaires, des informations binaires. Cependant, la description de ces tables, ainsi que des ajouts encore nécessaires sont essentiellement sous une forme descriptive (livres).

Notre objectif ne sera donc pas de construire une grammaire du français qui existe déjà, mais d'étudier la manière de l'utiliser, et d'en montrer des domaines d'utilisation. Le travail d'Emmanuel Roche [19] propose des outils qui incorporent le lexique-grammaire des verbes simples. Notre travail, essentiellement basé sur l'utilisation des formes figées et sur la reconnaissance de groupes nominaux simples met en évidence la différence entre la *grammaire descriptive* possédée, et la *grammaire d'analyse* que nous cherchons à implémenter.

Au premier plan de la description se trouve le dictionnaire électronique. Sa forme purement électronique en a fait le meilleur candidat pour une première phase d'analyse du langage. Nous avons consacré notre premier chapitre à la description du problème de l'*ambiguïté* qui met en évidence le rôle de ce dictionnaire et les problèmes causés par son utilisation exclusive. Cette analyse donnera une orientation aux chapitres suivants, en montrant que la séparation entre la grammaire et le lexique est très floue et qu'il ne faudra pas la forcer artificiellement.

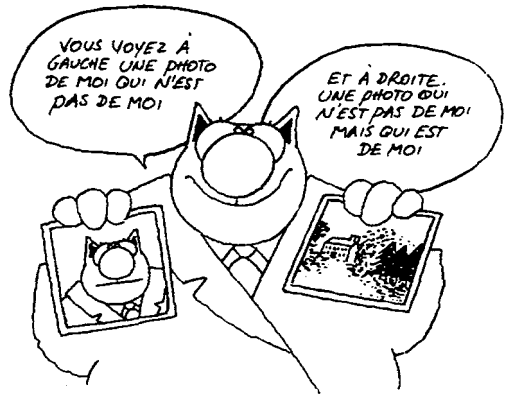
Dans le deuxième chapitre, nous décrivons le cadre formel dans lequel s'inscrit la description du LADL, nous décrivons également avec précision les ressources disponibles, et nous cherchons à définir *a priori* les différentes contraintes imposées par la description sur la *grammaire d'analyse*. Pour bien mettre en avant le caractère pratique du formalisme auquel nous aboutirons pour cette grammaire d'analyse, nous discutons aussi dans ce chapitre des contraintes effectives imposées à cette grammaire (en particulier, il n'est pas envisageable d'avoir une grammaire unique décrivant toute la grammaire du français, et le fait même de devoir la segmenter est une contrainte qui doit influer sur le formalisme lui-même).

Indépendamment des différents formalismes existant consacrés à l'analyse du langage naturel, nous pensons que le formalisme utilisé doit être dérivé de la description que nous possédons. Les considérations de puissance formelle d'une grammaire sont évidemment utiles pour déterminer s'il existe des phénomènes qui ne seront pas descriptibles par telle ou telle grammaire, mais ces considérations ne sont pas primordiales, dans le sens où elles ne doivent pas guider la forme que la grammaire aura. Dans cette optique, nous

présentons dans le troisième chapitre les différents outils développés pour implémenter notre grammaire, avant d'étudier la puissance du formalisme sous-jacent. De plus, par rapport au formalisme de base choisi qui est celui des langages rationnels, nous nous permettons de grandes libertés. Dans la partie III, nous étudierons les algorithmes adaptés à ce formalisme.

Chapitre 2

Les Ambiguïtés



[7]

L'AMBIGUÏTÉ est réputée être un des plus importants problèmes du traitement du langage naturel. À «cause» de ce phénomène, il n'est pas possible d'associer une étiquette grammaticale ou sémantique unique à chaque mot (cette opération étant appelée l'étiquetage automatique). De grandes applications comme la traduction automatique, l'indexation automatique, l'analyse statistique et même l'analyse syntaxique en pâtissent. Pour cette raison l'activité de désambiguïsation est devenue un domaine de recherche complet.

Une tentative de définition de l'ambiguïté consiste à dire qu'un mot ou une séquence est ambiguë quand ce mot ou cette séquence peuvent avoir plusieurs analyses différentes suivant le contexte.

Plusieurs observations s'impose sur la notion d'ambiguïté dans la langue ordinaire. Étonamment, le phénomène d'ambiguïté est loin d'être perçu lors de la lecture humaine d'un texte[1], alors qu'une lecture automatisée en fait apparaître énormément. Par ailleurs, l'expression *lever l'ambiguïté* n'est pas utilisée dans l'apprentissage des langues maternelles ou des secondes langues par les parents ou professeurs. La seule ambiguïté rencontrée dans ce cadre est l'ambiguïté de sens, utilisée par de grands auteurs et conduisant à des quiproquos mais dont les fondements sont sémantiques et dont la portée dépasse souvent

1. Dans la mesure où : quand on lit *le clocher est haut.*, on n'a pas la moindre hésitation quand à la nature du mot *clocher* qui pourrait aussi bien être un verbe.

le cadre de la phrase. De plus, dans un contexte d'analyse de texte, on ne dit qu'un mot est ambigu que quand il existe deux interprétations différentes de la phrase à cause de différentes interprétations que l'on peut faire de ce mot. Enfin, un fait important que nous mettrons en évidence est que plus le dictionnaire utilisé est précis, plus les ambiguïtés sont nombreuses, ce qui n'est pas intuitif puisque cela signifierait que mieux la langue est décrite moins bien elle est comprise.

Pour comprendre la problématique attachée à ce concept, nous chercherons à répondre aux trois questions suivantes: que sont les ambiguïtés, comment apparaissent-elles dans le traitement automatique de la langue, et pourquoi?

Dans une première partie, nous montrerons donc l'existence et la diversité des ambiguïtés, nous évaluerons leurs fréquences, et nous mettrons en évidence les processus qui les engendrent. Dans une seconde partie, nous présenterons différentes stratégies locales de désambiguïsation[2], leurs résultats, et leurs limites. Enfin, nous montrerons que la relation entre l'activité d'étiquetage et la notion d'ambiguïté est étrange. Si l'ambiguïté est mise en évidence par l'étiquetage, l'étiquetage lui-même est défini par les ambiguïtés (le problème est donc engendré par le traitement du problème).
L'analyse de cette situation donnera une orientation générale à tout notre travail: en particulier l'utilisation du dictionnaire sous sa forme actuelle ne sera plus considérée comme la première étape de tout traitement.

2.1 L'existence des ambiguïtés

2.1.1 Exemples

Les exemples d'ambiguïté abondent dans la littérature, nous en avons classé quelques-uns pour en montrer l'étendue. Les définitions que nous donnons à chaque catégorie ne prétendent pas être complètes, elles servent simplement à montrer la diversité du phénomène.

1. Ambiguïté dite morphologique (homonymie): coïncidence typographique[3] entre deux mots dont la nature est différente.
   - *est*: verbe *être* à la troisième personne ou le nom signifiant l'un des quatre points cardinaux.
   - *par*: préposition, ou nom signifiant le nombre minimum de coups qu'il faut pour réaliser un parcours de golf donné.
   - *président*: verbe *présider* à la troisième personne ou nom, titre de personalité.
2. Ambiguïté sur les informations flexionnelles: par exemple, le présent de l'indicatif et le présent du subjonctif ne se distinguent pas pour les verbes du premier groupe aux trois personnes du singulier. De même, pour les verbes du premier groupe,

2. On utilisera indifféremment les termes **désambiguïser** (provenant vraisemblablement de l'anglais), **désambigüer** et **lever les ambiguïtés**.

3. Nous ne considérons en effet que les ambiguïtés typographiques, en vue d'un traitement automatique de texte. Certaines ambiguïtés disparaissent quand on les considère du point de vue phonologique.

la première et la troisième personne du présent sont identiques. Par exemple, la forme *mange* correspond à *je mange*, à *il mange*, à *que je mange*, à *qu'il mange* et à *Mange !*[4].

3. Ambiguïté de sens (polysémie[5]) : un même mot, dont «l'étiquette syntaxique» est la même se réfère à plusieurs entités sémantiques possibles. Par exemple le nom *sens* qui détermine l'orientation d'une direction (le sens de la marche, d'une flèche, d'un axe) , la signification d'un objet (le sens d'une question, d'une phrase, d'une remarque), ou l'un des cinq sens.

4. Ambiguïté au niveau des parties du discours : *le savant allemand* interprétable comme *le savant*$_N$ *allemand*$_A$ (=*le savant qui est allemand*), ou *le savant*$_A$ *allemand*$_N$ (=*l'allemand qui est un savant*).

5. Ambiguïté au niveau des phrases : «*La petite brise la glace.*», analysable comme :
   (la petite[6]*)*$_{N_0}$ *(brise)*$_V$ *(la glace)*$_{N_1}$
   ou bien comme :
   (la petite brise)$_{N_0}$ *(la)*$_{N_1}$ *(glace)*$_V$
   De même *Max couvre la pomme de terre* peut théoriquement avoir la double analyse :
   (Max)$_0$ *(couvre)*$_V$ *(la pomme de terre)*$_1$.
   (Max)$_0$ *(couvre)*$_V$ *(la pomme)*$_1$ *(de terre)*$_2$.

6. La précédente catégorie présuppose qu'il est possible de segmenter automatiquement un texte en phrase. En fait le processus qui effectue cette segmentation va lui aussi buter sur des problèmes d'ambiguïté. Il arrive couramment qu'on désigne quelqu'un dans un texte par son prénom, voire l'initiale de son prénom. La séquence suivante est alors *a priori* ambiguë : ... *il aperçoit A. Ce dernier* .... La première interprétation est qu'il y a deux phrases, la seconde est que *A. Ce* est un nom propre.

7. Ambiguïté au niveau du traitement de la référence :
   Le professeur a envoyé le cancre chez le proviseur, parce qu'il en avait envie.
   Le pronom *il* peut se référer aussi bien à :
   Le professeur que *le cancre* ou *le proviseur*.

2.1.2 L'importance statistique des ambiguïtés

Dans [4], B. COURTOIS évalue le nombre d'ambiguïtés présentes dans le dictionnaire des formes fléchies du LADL : le DELAF dans sa version de 1994 comprenant près de 710,000 entrées. B. COURTOIS différencie l'ambiguïté dans son sens traditionnel, qui est «plusieurs interprétations pour une même phrase», et l'ambiguïté lexicale, qui est l'existence de plusieurs entrées dans un dictionnaire pour une forme donnée. Pour faire son évaluation, B. Courtois ne considère que ce deuxième type d'ambiguïté, et elle prend comme critère de définition de l'ambiguïté que deux formes sont ambiguës si elles ont la même orthographe

4. Sans rapport avec la dédicace de ce livre :).

5. La différence entre polysémie et homonymie n'est en fait pas très claire : à partir de quand décide-t-on qu'il s'agit du même mot avec un sens différent ou de deux mots différents ? Cette question est discutée dans [4].

6. Avec ici, *la petite* qui est *la petite fille*

et si elles différent par un des point suivants: le lemme, la catégorie grammaticale, les variations morphologiques, le genre, et le nombre. Le nombre d'ambiguïtés trouvé est d'environ 75,000, soit à peu près un mot du dictionnaire sur dix. B. COURTOIS met de plus en évidence les principaux types d'ambiguïté lexicale:

- Nom - Adjectif { *diplomate*, *religieuse*, *villageois*
- Verbe - Nom { *clocher*, *voyant*, *prévenu*, *guide*
- Verbe - Adjectif { *argenté*, *criant*, *fier*, *agrégé*

Ces ambiguïtés correspondent à la première des catégories (ambiguïté morphologique) définies précédemment (les marques sémantiques ont été supprimées, de même que les informations d'ordre flexionnel). Nous pouvons assez facilement donner une estimation du nombre des autres ambiguïtés lexicales (catégories 2 et 3) pour certaines catégories grammaticales:

- Ambiguïté sur les informations flexionnelles: il y a environ 100 000 formes d'adjectifs différentes dans le DELAF (elles différent par la forme, ou par le lemme, ou par le genre, ou par le nombre): environ 50 000 proviennent d'adjectifs ayant la même forme au masculin et au féminin (et 500 sont complètement invariables). Cela signifie qu'*une forme sur deux*, identifiée comme un adjectif dans un texte est ambiguë suivant ce critère, puisqu'on ne peut pas connaître son genre par simple consultation du dictionnaire.

- Ambiguïté sur le sens: la version actuelle du DELAF contient un certain nombre d'étiquettes sémantiques [10] associées aux noms: par exemple +Hum caractérise les noms humains, +Conc caractérise les noms concrets non humains, +Anl caractérise les noms d'animaux, +Abst caractérise les noms abstraits, etc. Le DELAF ne contient pas de définition proprement dite correspondant aux différents sens possible pour chaque nom, mais les sépare selon ces grandes catégories. Par exemple *aplomb* peut être +Conc (la direction verticale indiquée par un fil à plomb) ou +Abst (ne pas manquer d'aplomb), *adaptateur* peut-être +Hum (d'un roman, d'une pièce de théâtre), ou +Conc (adaptateur électrique). Sur les 150 000 entrées correspondant à des noms, environ 7 000 sont ambiguës selon ce critère (un sur vingt environ).
  Le lexique-grammaire des verbes simples du français différencie tous les verbes suivant leurs constructions syntaxiques. Cette différenciation est très proche d'une différenciation sémantique. (*Pierre lit un livre*, et *Pierre lit une lettre à Marie* correspondent nettement à deux constructions différentes, et aussi à deux sens différents, même s'il y a visiblement une partie commune). Le nombre d'entrées du lexique-grammaire des verbes simples est d'environ 17 000 alors que le nombre de verbes simples différents est seulement d'environ 11,000. Le taux d'ambiguïté des verbes suivant ce critère n'est donc pas loin de un sur deux.

Tous ces chiffres montrent qu'avec un dictionnaire à très large couverture comme le DELAF le nombre d'ambiguïtés lexicales prévisibles est très important (entre 25% et 50%). Cet ordre de grandeur du nombre de formes ambigües dans le dictionnaire, ne reflète cependant pas exactement le nombre d'ambiguïtés que l'on va rencontrer en moyenne dans un texte. En effet, les taux d'ambiguïté diffèrent [4] en fonction des **couches** du DELAF (les couches correspondent aux différents niveaux d'usage du vocabulaire [6]) et de plus la répartition des mots est non-aléatoire sur le texte. Par exemple, un texte technique dans lequel beaucoup de termes sont propres au domaine est vraisemblablement moins ambigu qu'une conversation de bistro où tous les termes sont d'usage courant et présentent donc un grand nombre d'interprétations différentes.

2.1.3 L'origine des ambiguïtés

Nous venons de voir quelques exemples d'ambiguïtés et nous avons évalué leur nombre en français. Ce phénomène est très vaste et très général. Pour mettre en évidence la différence entre l'approche naturelle (qui ne considère presque jamais les ambiguïtés) et l'approche automatique (qui doit désambigüer plus d'un mot sur deux), explicitons la source des ambiguïtés.
Notre point de départ est un texte sous une forme linéaire comme celui de la figure 2.1[7]. On possède un dictionnaire que l'on applique au texte : on cherche chaque mot du texte dans le dictionnaire, et on représente toutes les interprétations trouvées comme des chemins parallèles. Cela nous fournit un graphe acyclique (cf figure 2.2) appelé **automate du texte**. Le nombre de chemins différents partant du début et allant à la fin du texte (ou de la phrase) est donc le produit du nombre d'interprétations différentes de chaque mot. Ce nombre augmente très vite (M. GROSS discute dans [9] de la manière de compter les ambiguïtés d'une phrase).
Dans notre exemple de 8 mots qui ne forment même pas une phrase complète, il y a 720 chemins possibles. Tout raffinement au dictionnaire va faire augmenter ce nombre d'interprétations. Par exemple, dans l'exemple présenté, le mot *que* n'a pas été étiqueté. La consultation d'un dictionnaire usuel nous donne plus de 10 entrées différentes pour ce mot. Si on décide de les différencier, le nombre de chemins sera tout de suite porté à 7 200, ce qui veut dire que cet extrait de phrase pourrait avoir 7 200 interprétations possibles suivant le contexte !

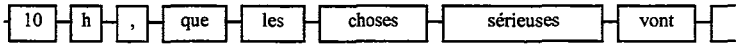


FIG. 2.1 – *Représentation linéaire des formes du texte*

L'opération de **désambiguïsation** consiste à «élaguer» cet automate en supprimant les transitions qui ne sont pas permises (par le contexte, ou même jamais permises) de manière à réduire ce nombre de chemins.
Nous présentons dans la prochaine partie, différentes méthodes conduisant à cette désambiguïsation, puis nous reviendrons sur le bien-fondé même de l'entreprise d'étiquetage.

7. Nous avons volontairement choisi une séquence incomplète sans contexte, pour bien montrer que le processus est local.

7. **choser** (populaire et régional) : *s'occuper de quelque chose..*

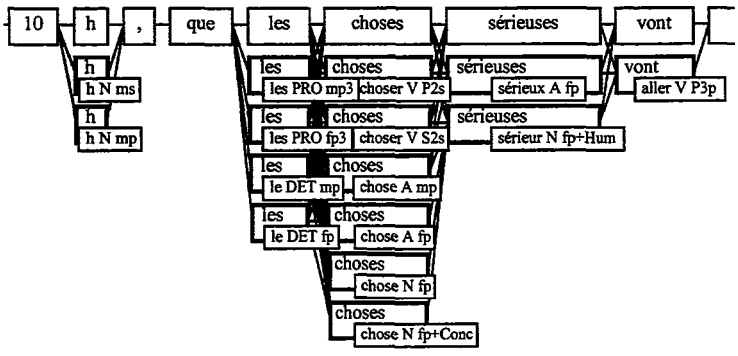


FIG. 2.2 – *Automate du texte, après passage d'un dictionnaire.* **choses** *a six interprétations possibles :* **«tu choses*»**, **«que tu choses»**, **«leurs visages tous choses»**, **«leurs figures toutes choses»**, **«laissez les choses suivre leur cours»**, **«ces choses-là !»**

2.2 La levée d'ambiguïté. Les approches locales

Pour résoudre le problème de levée d'ambiguïté, plusieurs approches existent :

- les grammaires locales de désambiguïsation (voir par exemple [1], [13], [18]) ;
- les approches statistiques (voir par exemple [5], [17]);
- l'approche basée sur l'heuristique de la **longest match** (occurrence la plus longue).

Toutes ces méthodes sont des méthodes locales et négatives (au sens où elles conduisent toutes à dire que tel chemin de l'automate du texte n'est pas possible, ou d'une manière équivalente, qu'un seul chemin est possible.).
Nous resterons assez vague sur l'évaluation des résultats des différentes méthodes. Il est en effet assez difficile d'évaluer individuellement chaque méthode, et encore plus de les comparer parce qu'elles n'opèrent pas toutes sur les mêmes objets ni sur les mêmes données. Nous dirons seulement que les résultats obtenus sont «bons» quand les résultats sont voisins de résultats souhaités, ou «partiels» quand il y a des cas non traités.
Nous verrons dans la partie suivante que l'activité d'étiquetage a un objectif essentiellement statistique. Dans cette optique, l'important n'est pas de savoir désambigüer tous les mots, mais de savoir traiter toutes les ambiguïtés fréquentes dans un texte. Parmi les plus fréquentes, on trouve les ambiguïtés des formes du verbe *avoir*, de la forme *par*, de la forme *est*, des formes *les*, *le*, *la* (qui peuvent être pronoms ou déterminants), etc.

2.2.1 Grammaire locale de désambiguïsation

La forme *est*

Une des formes les plus fréquentes des textes en français est la forme *est*. Pour la désambiguïser, nous pouvons affirmer sans risque que dans le contexte *nord-est* ou *sud-est*, il ne peut pas s'agir de la forme verbale. Cette règle est exprimée à l'aide d'une

grammaire locale de désambigüation [21] (figure 2.3) dont la forme est celle d'un transducteur. L'information contenue dans la sortie du transducteur a pour but d'attribuer d'une manière sûre une certaine catégorie à un mot, ou de lui supprimer une autre. Par exemple, la règle exprimée dans ce transducteur peut être formulée ainsi : *si la forme est est précédée de nord- ou de sud-, alors cette forme ne peut pas être une forme du verbe être.* On ne peut pas plus préciser si *est* est un nom ou un adjectif dans ce contexte. Voyons l'effet d'une telle règle sur l'automate du texte de la figure 2.4.
Seul le premier *est* est concerné, et son interprétation *être* est supprimée (en fait, seule la transition de [-] à [*être*] est supprimée, mais comme cela conduit à un noeud non relié : le noeud lui-même est supprimé.). Ce qui donne le nouvel automate du texte de la figure 2.5.

Les différentes formes du verbe *avoir*

Considérons maintenant quelques formes du verbe *avoir* au présent : l'infinitif *avoir* est homonyme avec le nom *avoir* (*un avoir*). Pour la deuxième personne du présent : *as*, il y a une ambiguïté avec le nom *as* (*as de pique*). À la troisième personne, il y a ambiguïté avec la lettre *a*. À la deuxième personne du pluriel, il y ambiguïté avec le nom *avion* au pluriel. Tous ces mots sont extrêmement fréquents dans les textes. Le verbe *avoir* est, en particulier, fréquemment employé comme auxiliaire de temps composé. Dans ce cas, il est suivi d'un verbe au participe passé noté *Vpp* :

Mais cette réunion a également été l'occasion d'aborder...
...ce patron qui avait pourtant su diversifier ses fabrications...
Il sera réputé ne jamais avoir été à la retraite.
Pour le livre que vous avez récompensé...
Nous avions également popularisé une technologie permettant d'opérer...

Il est tentant de proposer une règle qui affirme que si une forme du verbe *avoir* est suivie (éventuellement séparée par un adverbe) d'une forme verbale au participe passé, alors il s'agit effectivement du verbe *avoir*. On peut, de plus, déduire que la séquence qui sépare *avoir* du participe passé est réellement un adverbe, et que la forme proposée par le dictionnaire comme participe passé est réellement un participe passé. Cette règle est exprimée à l'aide de la grammaire locale de la figure 2.6.

Cette règle fonctionne parfaitement, et sur un corpus d'une année du journal Le Monde, nous n'avons pas trouvé d'erreurs dans son application. Cependant, il est très facile de la mettre en défaut sur les exemples suivants :

(1)* *des avions sûrement volés*
(2)* *l'avoir bien mal acquis*
(3)? *...d'avoir bien mal acquis...*

(1) et (2) sont deux groupes nominaux parfaitement naturels, qui sont des «exceptions» à la règle précédente. Dans (1), on peut utiliser le fait qu'il ne peut pas y avoir de déterminant, ici *des*, devant la forme verbale *avions*. Dans (2), le *l* peut aussi bien être un déterminant qu'un pronom, et il ne peut être pronom que si *avoir* est la forme verbale, il ne peut être déterminant que si *avoir* est la forme nominale. Ce qui nous permet d'enrichir un peu la règle. Par contre, dans (3), on ne peut pas résoudre localement puisque les deux contextes suivants sont possibles :

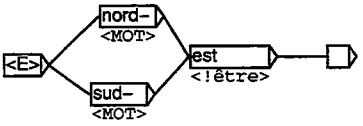


FIG. 2.3 – *Grammaire locale de désambiguïsation de la forme est utilisable dans le système INTEX©*

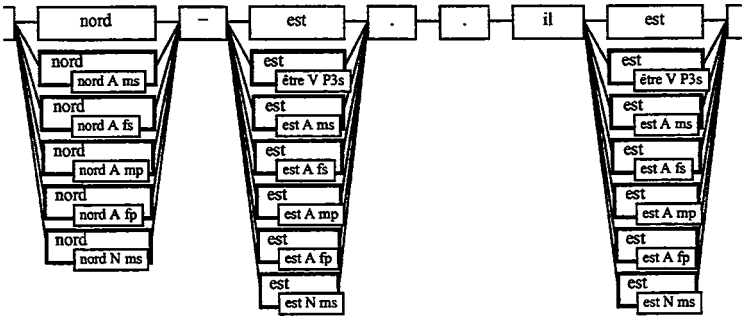


FIG. 2.4 – *Deux occurrences de est non désambiguïsées.*

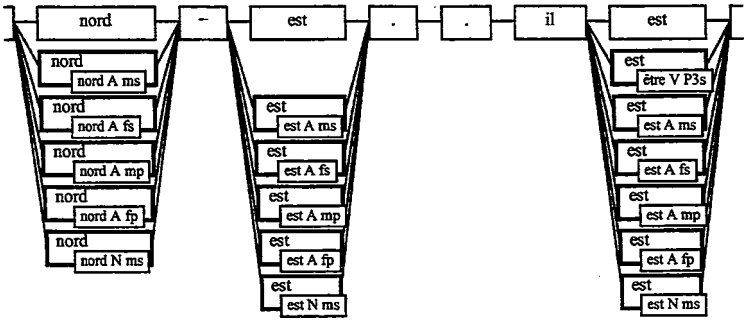


FIG. 2.5 – *Les mêmes occurrences de est partiellement désambiguïsées.*

FIG. 2.6 – *Grammaire locale de désambiguïsation d'une construction avoir Vpp.*

(4) *on lui reproche d'avoir bien mal acquis...*
(5) *je connais un exemple d'avoir bien mal acquis...*

Distinguer ces deux structures commence à devenir extrêmement compliqué localement. Dans (4), il faut en effet trouver le verbe admettant un complément en *de* suivi d'une construction infinitive, ce qui ressort d'une analyse syntaxique complète de la phrase (le verbe n'est pas forcément juste à gauche du *de* comme dans (6)) et cela ne serait toujours pas suffisant puisque reproche est aussi un nom, etc. Dans (5), il faut connaître tous les noms possédant la construction *de N*, et ce nom principal peut ici aussi être éloigné du groupe *d'avoir...*, par des modifieurs (adjectifs) ou par une relative comme dans (7). Nous n'arriverons donc pas à désambiguïser localement le mot *avoir* dans le cas général.

(6) *D'avoir bien mal acquis sa fortune lui est reproché.*
(7) *je connais un exemple, dont tout le monde parle, d'avoir bien mal acquis...*

Ces exceptions ne sont pas des anecdotes concernant un mot particulièrement ambigu. Elles sont le reflet du comportement général de la description par **règles négatives** (c'est à dire qui détruisent des chemins). En effet, l'existence de règles négatives est basée sur l'hypothèse qu'il est possible localement de trouver la nature d'un mot et elle implique que son auteur connaisse et maîtrise tous les contextes possibles d'un mot donné. De plus, la démarche n'est pas très satisfaisante: il suffit de trouver une seule exception inattendue à une règle négative, et elle devient fausse.

Considérons maintenant la règle d'accord entre sujet et verbe. La règle de la figure 2.7 dit que si le pronoms *elle* précéde un verbe (éventuellement précédé d'autres pronom pré-verbaux), alors ce verbe est forcément conjugué à la troisième personne. Cette règle permet de lever l'ambiguïté systématique première personne/troisième personne pour les verbes du premier groupe. Il est encore une fois possible de trouver une exception:

(8) *Jean et elle m'ont...*

Pour contourner ce problème, on peut imposer que *elle* doit être au début de la phrase. Cela empêchera alors le traitement de beaucoup d'autres cas comme:

(9) *Ce matin, elle mange...*

dont la construction est exactement la même. De plus, quand nous approfondissons la complétude d'une telle règle, nous nous rendons compte que beaucoup d'autres cas dont le traitement devrait être le même sont passés sous silence dans cette règle, comme les trois cas suivants:

(10) *elle, me dis-tu souvent, est différente des autres.*
(11) *elle seule connaît ce problème.*
(12) *elle, sans aide extérieure, peut résoudre ce problème.*

Compléter alors la grammaire pour tenir compte de tous ces cas de figure nécessite souvent de la réécrire entièrement.

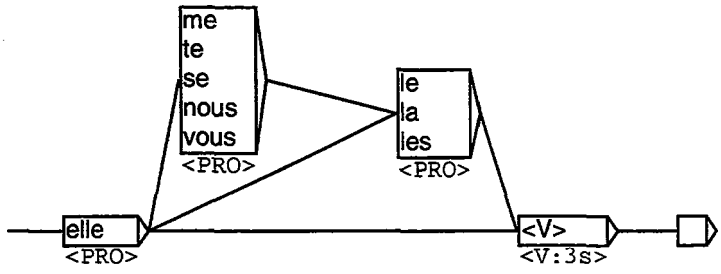


FIG. 2.7 – *Désambiguïsation*

Il y a donc un second problème théorique intrinsèque à la notion de règle négative : deux structures grammaticales très proches peuvent ne pas être désambiguées pareillement, car la description locale de l'une ne sera pas possible comme dans (10) ou car l'ajout d'une contrainte pour contourner une «exception» comme dans (8) conduira à omettre de nombreux autres cas comme (9).

En conclusion, précisons que le comportement statistique des grammaires locales est excellent, parce qu'elles permettent de lever, d'une manière très simple, la très grande majorité des cas d'ambiguïté. «*Toutefois, les intuitions grammaticales peuvent se révéler inexactes, à cause bien souvent d'une construction ou d'une ambiguïté à laquelle on n'aura pas pensé.*» [13]. Or le traitement d'une seule exception à une règle peut prendre une importance démesurée par rapport à la simplicité de la règle, pouvant même aller jusqu'à l'invalider complètement et la supprimer. De plus, nous avons vu que pour écrire une règle négative, il fallait supposer que toute la grammaire était connue[8], ce qui n'est pas le cas. De nombreux éléments imprévus comme les noms propres peuvent survenir. Par exemple, un article de journal parlant de la revue *Elle*, risque d'être mal interprété, car la forme *Elle* nom de revue n'aura pas été incorporée à la grammaire[9]. Enfin, l'incomplétude de telles grammaires (c'est à dire l'incapacité à traiter deux cas syntaxiquement très proches) n'est pas satisfaisante du point de vue théorique.

2.2.2 Les approches statistiques

Il serait difficile de parler de levée d'ambiguïté sans mentionner l'approche statistique qui est de loin l'approche la plus courante dans les algorithmes de désambiguïsation. D'une certaine manière, la méthode par grammaire locale en est une puisque nous avons vu qu'il est très difficile d'éviter des erreurs théoriques, alors que pratiquement (ou *statistiquement*) la grammaire se comporte bien. Nous la séparons néanmoins des méthodes présentées ici puisque son fondement est analytique et n'est pas basé sur des comptages empiriques.

Nous regroupons dans les approches statistiques, des méthodes réellement statistiques

8. Il est très difficile pour un linguiste, d'affirmer qu'il n'existe aucune situation où on observe tel ou tel phénomène.

9. Et il ne s'agit pas de rajouter simplement une nouvelle interprétation pour ce mot, l'existence de cette interprétation peut modifier profondément les règles existantes.

(résolution d'ambiguïté lexicale à l'aide d'un modèle de Markov caché) dans le sens où elles manipulent des probabilités, mais aussi une méthode d'approximation (longest-match) basées sur des comportements statistiques.

Levée d'ambiguïté lexicale

Le principe de l'étiquetage statistique est simple. Si on possède un corpus manuellement étiquetté, il est possible de calculer la probabilité de présence de chaque catégorie grammaticale en fonction du contexte (le plus souvent gauche). On construit une chaîne de Markov sur les catégories. En présence d'un nouveau texte et d'un mot possédant plusieurs étiquetages possibles on peut connaître la probabilité de chaque mot, et faire le choix le plus probable.
Ceci est un schéma général, dont le comportement est assez frustre. Il est possible de le raffiner en utilisant un modèle de Markov caché [2], qui a pour effet de pouvoir fournir une prédiction même dans les cas les plus rares.
Ensuite d'autres améliorations sont possibles en ne cherchant plus à maximiser la probabilité locale, mais à maximiser la probabilité tout le long de la phrase (réduction de l'entropie) [2]. Le gros inconvénient de cette approche est de nécessiter un gros corpus d'entraînement. Cette méthode suppose que les catégories grammaticales soient homogènes : toute information obtenue sur des représentants d'une catégorie peut servir à l'ensemble de la classe, c'est-à-dire à chacun de ses représentants. Cela signifie que l'on suppose que les grammaires de chaque élément de la classe sont, si ce n'est identiques, du moins très proches.

Les résultats obtenus par cette méthode sont très bon et tout à fait comparables à ceux obtenus par les grammaires locales [3]. Un des problèmes de cette méthode et qu'il est difficile de pouvoir améliorer ses capacités : comme elle ne manipule aucune donnée extérieure à part le dictionnaire et l'étiquetage, les seules modifications possibles sont des modifications sur les formules statistiques utilisées et il n'y a pas de raison *a priori* qu'une méthode soit fondamentalement plus puissante qu'une autre. La limite atteinte par les meilleures méthodes est vraisemblablement une limite linguistique qu'il n'est pas possible de franchir sans plus de description.
Cet argument est contesté car l'incomplétude des résultats obtenus est toujours justifié par une taille non suffisante du corpus d'entraînement. Cela conduit à la question philosophique : «*Est-il possible de construire une grammaire complète du langage uniquement par des méthodes statistiques basées sur un corpus ?*» Nous donnons des éléments de réponse à cette question en 11.

Le «longest-match»

La **«longest-match»** (plus longue occurrence) est la plus longue occurrence qu'un automate peut reconnaître à partir d'un point donné du texte. Cette notion est incorporée au programme INTEX© depuis 1992[10]. Cette méthode classique de «pattern-matching» peut-être utilisée comme heuristique de désambiguïsation d'une manière très naturelle. Elle dit que dans le cas d'une ambiguïté entre deux structures différentes, on ne conserve

10. On trouve une reformulation dans [12] en 1996. Elle est appelée alors left-to-right longest-match.

que la plus longue. Elle n'est donc pas comparable aux méthodes lexicales de désambiguïsation. Considérons la date suivante :

le 3 janvier 1998 à 7 heures du soir

Si nous appliquons l'automate de reconnaissance des dates, nous reconnaîtrons 10 dates possibles :

le 3
le 3 janvier
janvier 1998
le 3 janvier 1998
1998
à
7 heures
à
7 heures du soir
le soir
le 3 janvier 1998 à 7 heures
le 3 janvier 1998 à 7 heures du soir

Toutes correspondent bien à une date : *cette histoire date de 1998, du 3 janvier, de janvier 1998, ?de à 7 heures*[11], *du soir...*

Il est pourtant probable que dans son contexte, la seule date valide est la plus longue ; mais cette intuition pourrait être infirmée en choisissant correctement le contexte :

* *Cette alerte a durée de 3 heures du matin, (le 3 janvier 1998, à 7 heures du soir)*$_{DATE}$ *le lendemain.*

Ici, l'interprétation serait donc mauvaise (la présence de la virgule ne change rien). On s'en sort cependant dans ce cas, en disant que *de 3 heures du matin, le 3 janvier 1998, à 7 heures du soir le lendemain* est décrit dans la grammaire des intervalles de temps `de Date à Date`, et que le longest match s'appliquera encore.

Nous ne voulons pas justifier le longest-match à tout prix, mais il est évident même sans regarder un texte, que l'approximation faite, sera la plupart du temps correcte. Cette approximation sera d'ailleurs d'autant plus correcte que le motif reconnu sera long.

Nous avons appliqué cette méthode systématiquement à la reconnaissance des expressions verbales figées sans leurs éventuels compléments libres, et le taux d'erreur provoqué par cette méthode est inférieur à 0,1%. En effet, il est difficile de trouver des cas où les séquences *est-elle enfin sur le point de* et *sera de moins en moins en mesure de*, par exemple, ne correspondent pas à la structure figée que l'on trouve dans les phrases suivantes :

La France [est-elle enfin sur le point de] construire un dispositif législatif ambitieux?
... qui [sera de moins en moins en mesure de] remplir ses missions

Voici quelques cas d'analyses défectueuses :

(13)* Il [a lancé un appel à] témoins.
(14)* L'omnibus [rend déjà un service public], ...
(15)* On se [rend compte à] présent ...
(16) On [se rend compte] à présent ...

(13) a été localisé par l'automate de l'expression N_0 *lancer un appel à* N_1. (14) par N_0 *rendre un service à* N_1 et (15) par N_0 *rendre compte à* N_1.

11. Pour cette date, ce test ne fonctionne pas, mais on peut en trouver un autre *Cette réunion est à 7 heures.*

(14) est vraisemblablement un jeu de mots, donc nous ne pouvons pas vraiment le compter comme faux. Par contre, même si l'origine étymologique de *lancer un appel à témoins* est la construction N_0 *lancer un appel à* N_1, elle est complètement figée[12] maintenant. Nous ne l'avions pas dans nos tables, sinon le longest-match aurait ici fonctionné. Par contre dans (15) l'analyse faite est mauvaise, mais de même taille que (16). Nous pourrions choisir l'unité qui commence la première, mais il est évident que cela reporterait le problème à un autre endroit, il nous faut ici analyser le pronom *se* qui ne peut provenir que d'un complément *à* N_2, auquel cas la préposition n'appartient pas à l'expression verbale.
Les vrais cas d'erreurs provoqués par l'heuristique de longest-match pour cette application sont donc très rares et en plus limités à un petit nombre de constructions qu'il serait possible de ne pas reconnaître si on recherche un taux d'erreur nul.
Le longest-match utilise le fait que si les mots simples sont fortement ambigus, ils le sont moins quand on les reconnaît dans des séquences, et deviennent de moins en moins ambigus à mesure que la séquence s'allonge.

2.2.3 Conclusion

Nous avons présenté différentes approches locales au problème de la désambiguïsation des mots. Ces approches sont dites locales, car même si elles mettent en jeu localement le contexte, elles n'opèrent pas au niveau de la phrase[13] ce qui nécessiterait une analyse syntaxique complète. On voit ainsi que l'étape de désambiguïsation ne peut pas être la première étape logique de l'analyse syntaxique : au mieux elle peut diminuer la quantité de calcul nécessaire à l'analyse complète. Chacune des méthodes présente des points forts, mais elles font toutes des approximations (dont la principale est de ne regarder que le contexte local), et on pourra toujours fabriquer des exemples les mettant en défaut. Cependant, sur le plan pratique les résultats obtenus sont bons (100% de désambiguïsation observée de la forme *avoir* avec la première méthode par exemple). Les limites atteintes sont autour de 90% d'efficacité. Il faut cependant voir ce que signifie ce fort pourcentage : cela signifie qu'il y a un mot sur dix dont l'analyse est fausse, donc plus d'une phrase sur deux. La limite atteinte n'est pas un problème de raffinement de formule statistique ou de formalisme pour les grammaires.

En plus de l'approximation locale qui conduit forcément à des erreurs, nous montrons dans la prochaine partie qu'il y un problème de définition formel de l'étiquetage des mots simples, et que plus cette définition est à la source même de beaucoup d'ambiguïtés.

L'approche locale ne peut être qu'une approche intermédiaire et dans un but de résultat approché : lors d'application pratiques, par exemple la localisation des titres de personnalité, où on exploite justement le fait qu'une analyse locale suffit à extraire beaucoup d'information, on utilisera certaines grammaires *ad-hoc*, pour par exemple, ne pas confondre *ils le président*, avec *le Président de la République*. Pour ce type d'application, on admettra le caractère approximatif de la démarche.

12. En effet, le nom *témoins* sans déterminant ne peut pas être analysé par une règle générale, sinon on devrait accepter **lancer un appel à voisins*.
13. Ces méthodes sont aussi désignées par shallow parsing.

2.3 Source des ambiguïtés, le langage ou la description?

Nous avons vu précédemment comment apparaissaient les ambiguïtés, et comment on pouvait les supprimer par des méthodes locales. Nous avons vu qu'aucune de ces méthodes n'est formellement satisfaisante, mais que certaines ont de très bons résultats pratiques. Nous étudions dans cette partie, sur quelques exemples, pourquoi la plupart des ambiguïtés existent.

2.3.1 La grammaire de *majuscule*

La forme *majuscule* possède 3 entrées différentes dans le DELAF, et 27 entrées dans le DELAC (dictionnaire des noms composés). Il est donc 30 fois ambigu.

Forme	Catégorie	Genre/Nombre
majuscule	ADJ	ms
majuscule	ADJ	fs
majuscule	NOM+Conc	fs
A majuscule	NOM	ms
...		
Z majuscule	NOM	ms
lettre majuscule	NOM	fs

Cherchons maintenant dans quels contextes peuvent être employés ces différents usages:

(17) *un A majuscule*
(18) *une lettre majuscule*
(19) *un (mot+phrase) (tout+ϵ) en majuscule*
(20)? *un (caractère+symbole) majuscule*
(21) *la lettre A (en+ϵ) majuscule*
(22) *une majuscule*

On comprend pourquoi *majuscule* a été mis dans le dictionnaire comme adjectif: pour pouvoir analyser correctement les groupes nominaux (17) et (18). On voit aussi pourquoi il a été codé dans le dictionnaire comme nom: pour pouvoir analyser correctement le groupe nominal (22). Ces contextes sont de plus très limités, on pourra étendre l'énumération des lettres (17) à d'autres systèmes d'alphabet (grec par exemple), mais mis à part l'adjonction d'autres modifieurs comme *accent circonflexe* dans *un A accent circonflexe majuscule*, on ne pourra vraisemblablement pas trouver d'autres contextes.
La même analyse pourrait s'appliquer à la forme *A*: dans (17) il s'agit d'un nom, dans (21), il s'agit d'un adjectif.

Remarquons que dans tous les exemples, on peut restituer le mot *lettre*:

un A majuscule → une lettre A majuscule
une lettre majuscule → une lettre majuscule
un mot en majuscule → un mot en lettres majuscules
une majuscule → une lettre majuscule

Ceci nous montre qu'on a en fait seulement deux objets différents contenant la forme *majuscule*:

- le modifieur *majuscule* modifiant nécessairement le nom *lettre* (qu'il faut faire apparaître), ce modifieur prend aussi la forme équivalente *(lettre) en majuscule*.

- le modifieur *en majuscule* modifiant un nom différent de *lettre* (*mot, texte, paragraphe, message...*) , ce modifieur est alors équivalent à *(texte) en lettres majuscules.*

Ces deux catégories sémantiques différentes ne recoupent pas les catégories présentes dans le dictionnaire. Ceci n'est pas étonnant, on a attribué différentes catégories à la forme *majuscule* pour que cette forme entre dans le cadre d'une grammaire générale. Et donc quand on utilise le dictionnaire pour effectuer cette analyse générale on crée des ambiguïtés totalement artificielles provenant de la grammaire et non pas de la forme elle-même.
L'alternative proposée ici est d'associer une unique règle de grammaire propre à ce mot, qui désambiguë vraiment le mot *majuscule* (puisqu'elle permet de choisir entre les deux seules utilisations différentes), alors que l'étiquetage basé sur le mot, conduisait à associer deux étiquettes différentes qui ne reflète pas cette vraie ambiguïté.

2.3.2 Le genre des adjectifs

Blandine Courtois dans [4] fait remarquer que *la tradition orthographique et grammaticale considère qu'il existe deux genres et deux nombres pour les adjectifs. Cette attitude est justifiée par des paires comme : beau et belle, et national et nationaux pour lesquels la distinction est phonique. Cependant, de nombreux adjectifs comme aigre, rouge ou capable ne présentent pas de telles variations phoniques. Et orthographiquement, seul le nombre est distinguable. Pour ces mots, on pourrait adopter une représentation sans indication de genre, ce qui changerait complètement le décompte des ambiguïtés.* De fait, cette tradition orthographique et grammaticale s'exprime par la règle «les adjectifs s'accordent en nombre et en genre avec les noms qu'ils modifient», et se retrouve logiquement dans la règle

$$GN_{ms} = (A_{ms} + \epsilon)\ N_{ms}\ (A_{ms} + \epsilon)^*$$

L'ambiguïté flexionnelle dans *une maison rouge*, a donc été créée par l'utilisation d'une règle grammaticale générale, et par la volonté que l'étiquetage soit conforme à cette règle. Si on décide plutôt que certains adjectifs ne portent pas l'expression du genre (comme en anglais), on supprime instantanément ce type d'ambiguïté.

2.3.3 Le déterminant *beaucoup*

D'après les dictionnaires, *beaucoup* est appelé adverbe, mais peut avoir une fonction de déterminant dans (23), pronom dans (24) et (25), et adverbe dans (26).

(23) *(Beaucoup)$_{ADV}$ (de)$_{PREP}$ mes amis qui ont été voir ce film sont déçus.*
(24) *(De)$_{DET}$ mes amis qui ont été voir ce film, (beaucoup)$_{ADV}$ sont déçus.*
(25) *Mes amis ont été voir ce film. (Beaucoup)$_{PRO}$ sont déçus.*
(26) *Ils ont bu (beaucoup)$_{ADV}$ (de)$_{PREP}$ vin.*
(27) *Il y avait du vin.Ils ont (beaucoup)$_{ADV}$ bu.*
(28) *Ils ont (beaucoup)$_{ADV}$ bu.*
(29) *Ils ont (beaucoup)$_{ADV}$ bu de quelque chose.*
(30) *Ils ont bu (beaucoup)$_{ADV}$ de quelque chose.*
(31)* *(De)$_{DET}$ mes amis (...), (beaucoup)$_{ADV}$ de mes amis sont déçus.*
(32)?* *Ils ont (beaucoup)$_{ADV}$ bu (beaucoup)$_{ADV}$ de quelque chose.*

L'étiquetage donné dans ces exemples est celui produit par un étiqueteur du français répandu (celui en démonstration sur le site de Xerox `http://www.rxrc.xerox.com/grenoble`).

Il est cependant assez clair que (23) et (24) sont exactement les mêmes phrases: la construction est la même, il y a juste eu une permutation du groupe prépositionnel et de l'adverbe, ce que l'absence de différence de sens entre ces deux phrases confirme, en plus du fait qu'il n'est pas possible de rétablir la structure initiale après *beaucoup* en laissant le complément prépositionnel (31). Notons que l'étiquetage de *de* a changé entre ces deux phrases.
Dans (25), *beaucoup* a effectivement une rôle de pronom, mais celui-ci n'est pas très différent de celui de la phrase précédente. On peut d'ailleurs remplacer dans cette phrase *beaucoup* par *beaucoup d'entre eux* sans modifier aucunement ni le sens ni la construction du reste de la phrase. Et dans ce cas, c'est *d'entre eux* qui portera la référence au premier groupe nominal, *beaucoup* retrouve son étiquette d'adverbe (dixit l'étiqueteur).
Pour passer de (26) à (27), on a déplacé l'adverbe *beaucoup* avant le participe passé. (Le déplacement d'adverbe est commun). L'étiquetage de *beaucoup* ne change pas. Maintenant si on supprime l'objet direct du verbe boire, et qu'on le rend implicite par la phrase (27), *beaucoup* n'est pas plus étiqueté pronom, mais on voit que le lien avec le vin n'est pas aussi clair que si on avait *Ils en ont beaucoup bu.* Cependant, il y a quand même une référence à un objet liquide qui se boit. On peut donc reformuler (28) en (30). On remarque qu'il s'agit bien du même *beaucoup*, puisque (32) est interdit.

Ces exemples montrent que dans tous les cas, il n'y avait qu'une seule forme *beaucoup*, qui est *beaucoup de N.* Le *N* pouvant être un nom (*vin*), un groupe nominal défini (*mes amis*), ou être absent mais dans ce cas là représenter *quelque chose* (qui est en accord avec le verbe de la phrase). Cette simple règle fait que l'étiquetage de *beaucoup* ou de *de* ne varierait pas d'une manière un peu arbitraire en fonction de sa position, et de la position et de la présence de son complément.
De plus, cela permettrait d'arriver à ne pas dire que *de* peut-être un déterminant, ce qui est difficilement justifiable. Notons que nous avons fait le test d'étiquetage sur les phrases précédentes en remplaçant la forme *beaucoup* par la forme *hier...* Le résultat était exactement le même à part pour la phrase (25). Ce qui prouve que la volonté de vouloir étiqueter la forme *beaucoup* conduit à décider qu'il s'agit d'un adverbe (au même titre que n'importe quel autre adverbe) ou d'un pronom, et c'est cette information qui est mise dans le dictionnaire.
Dire que *beaucoup* n'est pas une unité élémentaire (dans nos exemples[14]), mais qu'il est toujours dans une séquence *beaucoup de*, où il a alors systématiquement une fonction de déterminant, résoud le problème de changements artificiels de catégorie observés. Bien sûr, dans ce cas, il faut décrire les opérations permettant de passer de (23) à (24), et de (23) à (25). Cette description doit être contenue dans une grammaire propre à *beaucoup* qui n'a évidemment pas le même comportement qu'un adverbe comme *hier.*

14. Cette analyse semble générale sauf dans des cas comme *j'ai beaucoup rêvé.* En fait, on associe naturellement cette phrase à *j'ai fait beaucoup de rêves.* La forme verbale, **j'ai rêvé beaucoup de rêves* n'existe pas. Nous ne pensons pas que cela doive modifier l'analyse générale, même si une des formes équivalentes (la forme que l'on dirait canonique) n'existe pas. Nous prouvons ceci dans 3.3.2, page 43.

2.3.4 Les étiquetages inutiles

Nous montrerons, dans le chapitre sur la reconnaissance des expressions figées, que le dictionnaire ne sera utilisé que pour donner des informations flexionnelles. Nous montrerons aussi que la surface totale des expressions figées (verbales et autres) dans un texte est plus d'un mot sur deux. Cela relativise *a priori* grandement l'importance de l'opération d'étiquetage. Ainsi on reconnaîtra les expressions suivantes sans essayer de trouver la catégorie individuelle de chaque mot :

(33) Ces essais sont loin de faire l'unanimité.
(34) ... mais j'ai faim.
(35) Le gouvernement n'avait pas su mesurer la portée de la fatwa.
(36) ... le tunnel dont on ne voit toujours pas le bout.
(37) Ils devraient faire preuve de plus de clarté.
(38) Il repère des signes qui nourissent ses soupçons.

En fait, on peut étiqueter ces phrases, mais cet étiquetage serait inutile pour une analyse syntaxique car les unités soulignées forment des unités minimales (par exemple le sens de *tunnel* n'a plus rien à voir avec la phrase (36)). De fait cet étiquetage conduirait à une généralisation abusive de la grammaire. Dans (37), par exemple, on a la structure de phrase

ils	*devraient faire*	*preuve*	*de*	*plus de clarté*
$(PRO)_0$	V	$(N)_1$	$PREP$	$(DET\,N)_2$

La construction $V\,N$ n'a rien de générale : on peut l'interpréter en disant que les groupes nominaux peuvent ne pas avoir de déterminant, ce qui est faux dans le cas général. De plus, cela ne ferait pas apparaître que *preuve* ne peut pas être mis au pluriel, même si le sujet est pluriel, que la seule préposition possible est *de*. La décision de rattachement [15] de *de plus de clarté* au nom *preuve* ou au verbe *faire* relèverait d'un jugement totalement arbitraire.

2.3.5 Conclusion

Pour une forme donnée, nous venons de montrer que les différents étiquetages donnés par un algorithme d'étiquetage quelconque basé sur l'utilisation d'un dictionnaire scolaire, ne peuvent pas décrire les différents objets linguistiques et/ou sémantiques représentés par cette forme. La plupart des ambiguïtés reposent sur un choix de conventions dont l'intérêt linguistique n'apparaît pas.

Nous avons montré que les ambiguïtés de nos exemples sont causées par la nécessité d'étiqueter qui entraîne l'attribution systématique de catégories ou d'information de genre/nombre à tous les mots (ce qui assure une cohérence inutile avec une grammaire générale intuitive). Cet étiquetage est imposé par l'existence même des dictionnaires de mots simples. L'opération d'application de tels dictionnaires au texte engendre des ambiguïtés inexistantes exigeant l'application de différentes méthodes de désambiguïsation qui cadre les données dans la grammaire générale précédemment utilisée. Il s'agit donc

15. Le rattachement de groupe prépositionnel est une des activités en vogue dans le domaine de levée d'ambiguïté automatique.

d'un cercle vicieux.

2.4 Orientation générale

Le cercle vicieux décrit précédemment est basé sur la proposition qu'il existe une grammaire générale sur les catégories syntaxiques: nous avons vu deux exemples de formes nécessitant une grammaire propre pour leurs analyses. Cette remarque peut être étendue à tous les mots du lexique. En effet, même les mots dont les combinaisons semblent «libres» ne sont pas analysables par une grammaire «générale». Vouloir étiqueter les mots simples c'est se concentrer sur cette partie superficiellement «régulière» du langage et identifier la reconnaissance à l'analyse.
Nous utiliserons cette analyse pour définir une première ligne directrice dans le travail que nous proposons: l'opération d'étiquetage en elle-même est naturelle, mais doit s'appliquer à des unités beaucoup plus complexes que les mots simples. Pour cette raison, notre démarche ne sera pas basée sur l'existence des ambiguïtés: nous proposerons une structure permettant d'ignorer totalement ce phénomène. Aucune des grammaires produite ne tiendra compte de l'éventuelle ambiguïté de ses constituants.

2.4.1 Les ambiguïtés résiduelles

Nous pensons que l'analyse complète d'une phrase, c'est à dire la description précise de ses constituants, supprime la question même de levée d'ambiguïté, sauf au niveau des phrases (mais là seule une analyse des phrases en contexte pourrait aider). M. GROSS décrit dans [9] des ambiguïtés de niveaux beaucoup plus fins que ceux présentés précedemment. Parmi ces ambiguïtés résiduelles restantes, il existe un grand nombre de structures conduisant systématiquement à des ambiguïtés (différenciées dans les dictionnaires par sens propre et sens figuré). En particulier, de nombreuses constructions figées ont une lecture complètement «libre»:

Jo a pris le taureau par les cornes.
Jo compte les moutons pour s'endormir.
Jo a avalé la pilule.
Jo paye avec sa carte bleue.
L'acier trempé est très résistant.

Ces phrases recevront donc systématiquement une double analyse, qu'aucune méthode syntaxique ne pourra lever. D'autres outils à un niveau plus élevé de description devront être établis.
Considérons également les deux phrases suivantes correspondant à deux constructions verbales différentes:

(Jo a construit un lit) en bois.
Jo a construit (un lit en bois).

Il est difficile d'entendre une différence de sens entre ces deux analyses et pourtant les constructions syntaxiques sont nettement différentes. D'ailleurs la seconde construction est équivalente à *Jo a construit (un lit de bois)* qui n'admet la première analyse. Pour ce type d'ambiguïté, il faudra que le processus d'interprétation sémantique puisse montrer l'équivalence entre les deux interprétations pour ignorer l'ambiguïté.

Chapitre 3

Les ressources linguistiques

Nous voulons établir une *grammaire de reconnaissance* de certaines formes linguistiques. Pour mettre en évidence les opérations pertinentes sur le langage naturel, et expliciter ce que nous entendons par reconnaissance, nous commençons par définir l'objectif ultime vers lequel l'analyse doit tendre ; objectif pour lequel la *reconnaissance* n'est qu'une étape intermédiaire. Nous définirons ensuite les objets sur lesquels les grammaires doivent porter : traditionnellement il s'agit des mots, mais en explicitant ce choix, nous montrerons qu'il n'est pas unique et qu'il a des implications. Nous étudierons ensuite les différentes composantes constituant la grammaire du français dans le cadre d'une description par le lexique-grammaire comme effectuée par le LADL, et nous en ferons ressortir les différents traits devant guider la construction d'une *grammaire de reconnaissance* automatique.

3.1 L'objectif

Nous voulons construire une grammaire qui devra «reconnaître» certaines formes linguistiques. Pour appréhender ce que le terme *reconnaissance* veut dire, prenons deux modèles extrêmes de grammaire de reconnaissance. La première est basée sur la règle simple : *toute séquence de mots entre deux points est une phrase.* Le taux d'exactitude de cette règle sera très élevé, les phrases seront donc très bien reconnues, mais ne recevront *aucune analyse.*
À l'autre extrême se trouve l'analyse par le lexique-grammaire, qui identifiera chaque composant et sa compatibilité avec les autres composants. Une telle grammaire a actuellement un taux de reconnaissance des phrases limité, car la description du français n'est pas terminée, mais une *grande profondeur d'analyse.* Pour que la distinction entre ces deux exemples ait un sens, il faut nécessairement définir ce que nous entendons par analyse.
On distingue traditionnellement deux étapes dans l'analyse (manuelle) d'une phrase : la distinction des différents composants, et ensuite, la distinction de leurs fonctions. Nous avons vu dans le chapire précédent, que la volonté d'étiqueter à tout prix (c'est-à-dire de trouver la nature de tous les composants élémentaires) était un faux problème, et ne facilitait en aucun cas la recherche des fonctions des différents constituants.
La démarche que nous présentons ici n'aboutit pas, en apparence, à un résultat plus riche. Nous reconnaîtrons aussi des groupes nominaux, des adverbes, nous grouperons certains mots de la phrase, et au bout du compte nous obtiendrons aussi une représentation ar-

borescente des phrases (sur lesquelles nous aurons opéré un certain nombre d'opérations, dont des permutations et des regroupements). Pour comprendre la différence, il nous faut détailler plus précisément les étapes ultérieures du traitement.

Un des buts finaux de l'analyse syntaxique d'une langue comme le français est d'arriver à décomposer tout texte en une suite de phrases élémentaires. La notion de phrase élémentaire est une notion relativement [1] simple, définie avec précision à partir de schémas de phrases élémentaires définis dans le lexique-grammaire.
Schématiquement, car nous reviendrons sur la description du concept de lexique-grammaire : *Jean lit ce livre* correspond à un schéma de phrase élémentaire décrit dans la table notée 32R2 (l'organisation des différentes tables du lexique-grammaire est décrite dans [15]) : la structure de cette phrase élémentaire est $N_0\ V\ N_1$ et ses propriétés sont décrites dans la table.
Jean a lu cette phrase dans ce livre de structure apparente $N_0\ V\ N_1\ Prép\ N_2$ correspond à un autre schéma de phrase élémentaire décrit dans la table 38LS.
Cela signifie que les deux verbes *lire* dans ces deux phrases sont des mots syntaxiquement différents. Nous percevons bien, de plus, la différence sémantique des deux actions *lire un livre* ou *lire une phrase*: la première action est une action qui s'inscrit dans la durée, la seconde est ponctuelle [2].
Par contre, *Jean lit ce livre dans le salon*, de même structure apparente que *Jean lit cette phrase dans ce livre*, ne correspond pas à un schéma de phrase élémentaire. Il s'agit de la même phrase que la première à laquelle est rajoutée *dans le salon*, qui est un complément circonstanciel de lieu, comme *à voix haute* serait un complément circonstanciel de manière (ou plutôt un complément circonstanciel de lecture, car *à voix haute* dont la nature est adverbiale, ne modifie que les verbes *lire*, *penser*, *réfléchir*, *rêver*). Le critère de choix, permettant de dire si tel complément dans telle construction est essentiel (tout en étant essentiel il peut être facultatif), ou bien s'il est circonstanciel est parfois délicat à établir. De fait, une telle question ne se pose pas dans notre travail puisque le lexique-grammaire des verbes simples a été construit et que cette information y est contenue.
L'idée principale de l'analyse par phrase élémentaire et que les **phrases complexes** [3] peuvent être décomposées en combinaison de phrases élémentaires. *Jean lit ce livre dans le salon* s'analyse ainsi :

Jean lit ce livre.(32R2)
Jean se trouve dans le salon.(35ST)

En fait, on peut pousser un peu loin plus cette analyse: de la même manière que le complément circonstanciel de lieu est extrait du schéma élémentaire, on peut aussi extraire le complément circonstanciel de temps, qui est ici implicite à cause de l'emploi du présent, ce que l'on peut exprimer ainsi :

1. Une complexité réside dans la difficulté qu'il y a à séparer les compléments essentiels des compléments circonstanciels [10].
2. Dans notre exemple, la distinction était nette, par contre dans la phrase *Jean a lu cette histoire dans ce livre*, il y a une ambiguïté. On peut forcer les deux interprétations, en modifiant le contexte. Le premier sens est *Jean a lu pendant quelques heures cette histoire qui est dans ce livre, il ne l'a pas encore fini.* Le second sens est *C'est dans ce livre que Jean a lu cette histoire.* Ici, le processus est achevé, et l'accent est mis sur l'endroit où a été lue l'histoire. L'ambiguïté existera toujours, mais dans certains contextes, la différenciation entre les deux sens ne sera pas pertinente.
3. Celles ne correspondant pas à un schéma de phrase élémentaire).

(1) *Jean lire ce livre.*(32R2)
(2) *Jean se trouver dans le salon.*(35ST)
(3) *Cela* (=(1)+(2)) *se passer dans le présent*[4].(31R)

Cette décomposition se justifie bien, la phrase (1) non conjuguée est explicite. Si un adverbe de temps comme *maintenant* était présent, elle le serait plus encore. Et les phrases suivantes :

(4)* *Jean lit ce livre hier.*
(5)* *Jean lisait ce livre maintenant.*

sont incorrectes, ce qui prouve bien que l'adverbe de temps, et le temps avec lequel est conjugué le verbe ont la même fonction (i.e. il s'agit bien du même objet). Cette représentation n'est pas très loin d'une représentation logique, comme nous l'illustrons sur l'exemple suivant :

Christian Ramussen, jeune Danois de 18 ans, n'a pas eu de chance. Né dans un milieu défavorisé, il n'a pas été à le bout de sa scolarité. Après un accident en rollers, qui lui a brisé le genou droit, il a dû interrompre sa formation de cuisinier.

Ces trois phrases peuvent être décomposées selon les phrases élémentaires suivantes. Nous avons indiqué une pseudo-décomposition logique (ou fonctionnelle) que l'on pourrait leur associer. Cette décomposition ne prétend pas suivre de règles formelles précises, elle est juste ébauchée à titre indicatif en vue d'un traitement ultérieur. Elle permettra de contourner les formes théoriques quand nous parlerons des transformations. Nous avons aussi négligé l'expression du temps pour alléger les notations.

CR est un homme.	homme(CR)
CR est de nationalité Danoise.	Nationalité(CR, *Danoise*)
CR a 18 ans.	âge(CR, 18)
CR n'a pas eu de chance.	¬(avoir_de_la_chance(CR))
CR est né dans un milieu défavorisé.	né_dans_un_milieu_défavorisé(CR)
(P_0): CR suit une scolarité.	P_0 = suivre_scolarité(CR)
CR n'a pas été au bout de P_0.	¬(aller_au_bout_de(CR, P_0))
(P_1): CR faisait du roller.	P_1 = faire_du_roller(CR)
(P_2): CR a eu un accident.	P_2 = avoir_accident(CR)
P_2 a eu lieu pendant P_1.	avoir_lieu_pendant($P2, P1$)
GD est un partie du corps de CD.	partie_du_corps(CR, GD)
GD est un genou droit.	genou_droit(GD)
(P_3): GD a été brisé.	P_3 = briser(X, GD)
P_2 a fait que P_3.	cause(P_2, P_3)
CR suivait une formation(= F).	suivre_formation(CR, F)

En considérant cet objectif qui est de générer automatiquement ces schémas de phrases sous-jacentes, comparons maintenant l'étiquetage (figure 3.1) obtenu par une grammaire régulière, et l'étiquetage que nous allons présenter à partir du chapitre 4.5 (figure 3.2). Il est clair que le deuxième est plus proche de l'objectif que le premier[5]. Notre objectif sera donc de montrer comment un tel étiquetage peut être obtenu. Dans ce cas, l'ensemble

4. Qui peut être lié au moment de l'énonciation, au temps auquel se passe le discours donc au verbe performatif sous-jacent : *je dis que ....*
5. En fait, il n'est même pas clair que le premier étiquetage nous rapproche de l'objectif.

des opérations suivantes auront été nécessaires :

1. reconnaissance du nombre *18* (page 76) ;
2. reconnaissance de *Danois* comme adjectif de nationalité ;
3. transformation (très productive) :
   $Adj-nationalité \longrightarrow (homme)_N\ (Adj\text{-}nationalité)_{modif}$;
4. reconnaissance de *jeune* N + *Hum de 18 ans* (table ENPC (page 154) ;
5. reconnaissance avec dictionnaire de *Christian Ramussen* comme nom propre ;
6. reconnaissance de *(Christian Ramussen), (jeune ...)* comme groupe nominal décrivant un nom propre ;
7. regroupement $n'...pas \longrightarrow (n_pas)_{NEG}$ (chapitre 6) ;
8. regroupement $a\ eu \longrightarrow (avoir)_{PC}$;
9. reconnaissance N_0 *avoir de la chance* (table A?) ;

La représentation finale ayant été choisie, l'étape suivante consistera à associer une fonction à chacune des transformations en jeu.

Donnons un autre exemple de comparaison entre les deux approches. Considérons la reconnaissance des groupes N *Prép* N (nom-préposition-nom). Toute grammaire dira qu'un groupe nominal (GN) a la forme :

$$Dét\ N\ Prép\ (Dét + \epsilon)\ N$$

Une telle règle reconnaîtra dans le texte précédent : *le bout de sa scolarité, un accident en rollers* et *sa formation de cuisiner.* Ces trois groupes nominaux seront étiquetés indifféremment GN. Les seules analyses «logiques» automatiques possibles sur cette reconnaissance superficielle du groupe nominal sera *bout(scolarité)*, *accident(roller)*, et *formation(cuisiner).* Cela est loin de l'analyse que nous avons présentée précédemment. L'analyse des groupes nominaux que nous proposons en 7.1 ne reconnaîtra pas la séquence *le bout de sa scolarité* seule, cette séquence sera reconnu avec le verbe *aller* comme étant une expression verbale figée : *aller au bout de. un accident en roller* ne sera pas non plus marqué comme étant un groupe nominal, *en roller* sera noté comme un adverbe modifiant la phrase figée sous-jacente *avoir un accident.* Enfin *sa formation de cuisinier* sera reconnu comme étant une forme dérivée de la construction à séquence support : N_0 *(avoir une formation de)* N_1 où *avoir une formation de* $\equiv$ *être.*
Chacun des exemples est reconnu comme une entrée du lexique-grammaire propre, l'association d'une fonction logique à chacun de ces exemples est donc directe.

Conclusion : Nous avons présenté deux niveaux différents de reconnaissance. Le premier type utilise une grammaire classique[6]. Basée sur un faible nombre de règles cette

6. Ce type de grammaire n'existe toujours pas à large couverture et nous ne pensons pas qu'elle puisse exister.

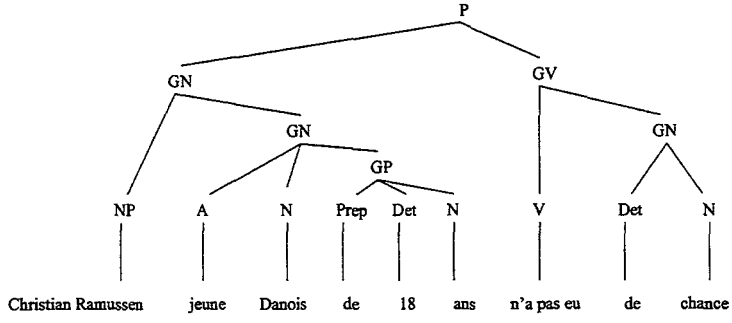


FIG. 3.1 – *Étiquetage par une grammaire générale*

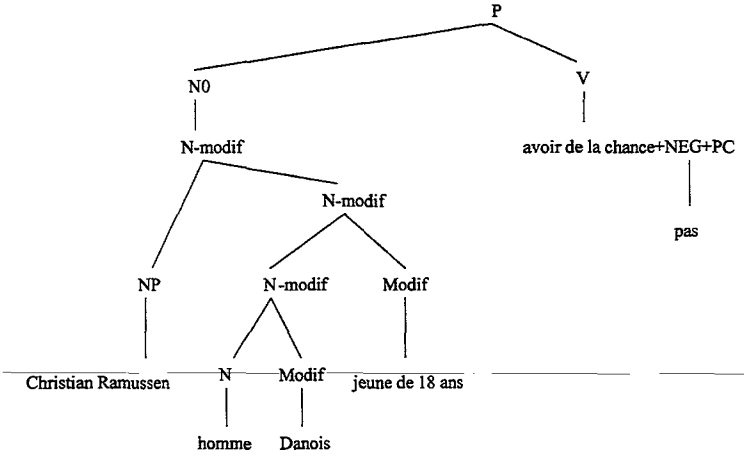


FIG. 3.2 – *Reconnaissance de schémas élémentaires. La suite des mots formés par les feuilles de l'arbre ne représente plus directement la phrase. Certains mots sont apparus (*`homme`*), d'autres ont été regroupés (*`avoir de la chance`*).*

méthode a un pouvoir de reconnaissance relativement étendu, mais les séquences reconnues sont peu distinctives, aussi, la reconnaissance ne peut pas être poursuivie par un travail ultérieur.
Le deuxième type de reconnaissance est l'analyse par le lexique-grammaire dont nous montrerons l'application et qui associe pratiquement à chaque forme un comportement syntaxique précis. *sa formation de cuisinier* sera reconnu, alors que *sa formation en cuisinier*, ou *sa formation du cuisinier* ne le seront pas. À chaque séquence reconnue, il est donc possible d'associer des informations utilisables dans un traitement ultérieur.

3.2 Les unités élémentaires

Nous venons de voir que les phrases (donc les textes) résultent de la composition d'unités minimales appelées phrases élémentaires. Seules ces unités sont linguistiquement pertinentes : un mot isolé comme *un*, *mot* ou *isolé* ne représente que la manifestation dite de surface de la présence d'une phrase élémentaire. C'est donc sur ces unités minimales que la grammaire est définie.
Nous avons cependant une démarche inverse de celle où une représentation de surface est engendrée à partir de phrases élémentaires : partant d'une phrase nous voulons retrouver les différentes phrases élémentaires qui la composent. Nous opérons donc sur des sous-séquences de phrases et nous voulons reconstituer ces phrases élémentaires. Le choix actuel de ces sous-séquences est essentiellement lié à la segmentation naturelle des phrases en mots[7], et ne repose pas sur des critères formels généraux. D'autres solutions pourraient être envisagées :

- sous-unités : décomposition en morphèmes. Par exemple *mangera* serait analysé comme l'union du morphème *manger* et du morphème *-a* (c'est-à-dire futur simple, troisième personne du singulier) ce qui aurait comme avantage de rapprocher naturellement des mots comme *déménagement* et *déménager* mais aussi de faire apparaître directement que le *-a* du futur, et l'adverbe *demain* ont la même fonction grammaticale (voir page 32) ;
- sur-unités : décomposition en mots composés. Par exemple, on dirait que *pendant ce temps* est une seule unité qui pourrait très bien être écrite sans espace : *pendant-cetemps*, cela reviendrait à ne plus considérer les trois mots simples *pendant*, *ce* et *temps*,
- décomposition en n-grammes (suite de n lettres) ;
- suite de mots comprise entre deux points.

Ces méthodes sont en fait équivalentes mais sont plus ou moins difficiles (lourdes) à appliquer. En fait, plus on choisit des unités petites, plus le dictionnaire sera petit, et plus la grammaire décrivant les relations entre les unités s'agrandira. À la limite, la décomposition en phrases ne nécessite plus de grammaire mais demande un dictionnaire

7. D'ailleurs les dictionnaires usuels sont exclusivement des dictionnaires de mots simples : il ne serait pourtant pas difficile d'imaginer un dictionnaire de tétragrammes (suites de 4 lettres) dont les entrées aurait une structure équivalente comme la suivante : **ubso** `adj` et `n.f` **1.** `adj` Inférieur à la vitesse du son. *un vol subsonique* **2.** `n` Action de subsumer. *subsomption*.
Il est évident qu'un tel dictionnaire serait moins «naturel» et non utilisable.

de taille infinie qui contiendrait toutes les phrases possibles.
On observe ce phénomène dans une mesure moindre, dans l'hypothèse d'une décomposition en morphèmes : de grandes ressources lexicales sont nécessaires pour savoir que *déménager* est constitué d'un seul morphème, alors que *dé-ranger* de deux[8]. Le «gain» que nous avons présenté au niveau de la grammaire sera contrebalancé par une plus grande lourdeur au niveau du dictionnaire.
Dans le cas d'une segmentation en mots simples, le lien entre *déménager* et *déménagement* sera fait à l'aide des tables de nominalisation (voir 7.1.1), la simple consultation du dictionnaire ne pourra pas nous donner cette information, il faudra aller la chercher dans le lexique-grammaire.
Toutes les ressources pourront donc être disponibles pour n'importe quelle description précédente, mais leur expression sera plus ou moins lisible et naturelle. Choisir le mot (suite de caractères séparés par des caractères de ponctuations) comme unité minimale semble être la meilleure solution. L'intuition de *mot* est fidèle : elle conduit cependant à constater que certains mots n'existent pas vraiment (*escampette*, *hui*) mais doivent être incorporés à de plus larges unités ; on est également conduit à s'apercevoir que des mots sont artificiellement ambigus : par exemple, nous avons montré en 2.3.3, page 27 que *beaucoup* est un mot ambigu, alors qu'il n'existe qu'un seul objet sémantique qui peut avoir plusieurs positions syntaxiques. Cet exemple n'est pas une exception : tous les «mots» devront recevoir une telle analyse[9].
En conclusion, la grammaire n'est définie que sur les phrases élémentaires, et nous chercherons donc à travailler sur des unités de plus en plus proches de celles-ci (qui physiquement dans le texte pourront être des séquences connexes ou non de mots, des mots simples, ou des parties de mots). Notre point de départ de l'analyse étant le texte, nous commencerons néanmoins forcément l'analyse au niveau des mots simples, points d'entrée convenus dans les dictionnaires. Les exemples d'autres segmentations possibles que nous avons présentés ont de plus mis en évidence le lien non trivial entre la grammaire et le dictionnaire. Ces deux notions sont totalement dépendantes du choix de la segmentation, et les informations appartenant au dictionnaire pour une segmentation donnée peuvent appartenir à la grammaire pour une autre segmentation. Cela relativise donc l'écart théorique entre la notion de dictionnaire et la notion de grammaire.

3.3 La grammaire

L'idée sous-jacente à la construction d'une grammaire d'un langage naturel est qu'il existe des règles générales régissant le comportement de familles d'objets, et que la mise en évidence de ces règles permet de décrire de façon compacte le comportement de chaque famille d'objets. L'apprentissage classique de la langue maternelle ou d'une langue étrangère met l'accent sur les grandes catégories grammaticales *verbes*, *noms*, *adjectifs*, *déterminants*, etc. Ces grandes catégories sont éventuellement divisées en quelques sous-catégories, souvent au moyen de fonctions grammaticales : par exemple, on distingue traditionnellement les verbes transitifs des verbes intransitifs. Avec des familles aussi glo-

8. La morphologie ne pouvant pas, en effet, être un critère suffisant ni nécessaire pour segmenter une phrase en morphème.

9. Sinon nous devrons nous poser des questions artificielles suivantes : «Est-ce que le *23* de *23 mai* est le même *23* que dans *23 millions*?», «Est-ce que le mot *lire* pris en dehors de ses schémas de phrase propres est un verbe?»

bales, la tentation est grande de construire des règles à ce niveau de généralité. Nous avons vu en 2.3 que l'utilisation de ces catégories relève plus d'une tradition que de propriétés formelles, ce qui doit inciter à une grande prudence vis-à-vis de règles contenant des symboles aussi généraux que <*A*>, <*V*>, <*N*>, <*DET*>...
Par ailleurs, le travail effectué sur les verbes simples au LADL depuis 1966, a mis en évidence que sur les 12 000 verbes simples du français, très peu[10] ont le même comportement syntaxique. Cela ne signifie pas qu'aucun travail d'unification n'est possible: au contraire, ce travail a mis en évidence un certain nombre de phénomènes linguistiques (transformations), et a regroupé par classes des verbes ayant des propriétés syntaxiques communes (ces propriétés sont dites définitionnelles). Les classes ainsi créées sont appelées des tables du lexique-grammaire. À l'intérieur de ces tables, la capacité de chaque entrée à entrer ou non dans une structure donnée est notée par les symboles binaires + et −. Ce travail a donc conduit à la construction d'une «grammaire» de chaque verbe, ce qui est en contradiction avec la volonté d'une grammaire unificatrice et simplificatrice. Nous retrouvons ce phénomène dans la description de nombreux phénomènes linguistiques: la description d'un titre de personnalité par exemple relève d'une même démarche ; chaque profession possède sa grammaire propre que l'on ne peut pas espérer décrire par des méta-règles syntaxiques générales (cf. 7.1.1). *Plus généralement, chaque entrée lexicale, dispose d'au moins une grammaire propre.* L'élaboration d'une grammaire du langage est moins dans le travail de factorisation, que dans celui de mise en évidence des différentes propriétés permettant de caractériser chaque entrée, et de l'étude de la combinaison des différentes règles.

3.3.1 L'utilisation du dictionnaire

Nous disposons d'un dictionnaire de toutes les formes fléchies du français, appelé le DELAF [10]. Ce dictionnaire comporte plus de 900 000 entrées, qui sont des mots simples, et il associe à chaque entrée son **lemme**, sa **catégorie grammaticale**, des **informations flexionnelles**, et des **marques sémantiques**. La table 3.2 en donne un extrait.
Les différentes catégories grammaticales sont principalement *V* (verbe), *A* (adjectif), *N* (nom), *ADV* (adverbe), *PRO* (pronom), *DET* (déterminant), *PREP* (préposition), *XI* (forme non définie grammaticalement, appartenant à une unité plus longue, par exemple le *hui* de *aujourd'hui*). La notion de **lemme** est définie par la tradition lexico-grammaticale: le lemme d'un *V* est sa forme à l'infinitif, le lemme d'un *A* ou d'un *N* est la première forme de l'adjectif (ou du nom) de la liste suivante qui existe: masculin singulier, singulier, forme elle-même. Par exemple, le lemme de *vieilles* est *vieil*, le lemme de *aliquante* (adjectif féminin singulier)[11] est *aliquante* (féminin singulier) car l'adjectif n'existe pas au masculin. Le lemme d'un nom ou d'un adjectif est le représentant d'une classe d'éléments sémantiquement équivalents qui diffèrent entre eux par le nombre et le genre. Il faudra donc faire attention lors de l'utilisation des lemmes: ceux-ci offrent une méthode pour désigner tous les éléments d'une même classe qui fait abstraction de catégories flexionnelles comme le genre ou le nombre.

10. En fait, pas deux verbes ont exactement le même comportement syntaxique.

11. Une *partie aliquante* est une partie qui n'est pas contenue un nombre exact de fois dans un nombre (contraire de *aliquote*). En fait *aliquante* n'existe que dans ce mot composé, mais il est étiqueté comme adjectif dans les dictionnaires.

Pour les verbes, on observe aussi de telles singularités, mais le traitement en est différent : *apparoir* est l'infinitif d'un verbe que l'on utilise uniquement dans les locutions *il appert (que+de)*[12]. Cette forme incorrecte grammaticalement a été rajoutée aux dictionnaires pour fournir un lemme à *appert* et ce, parce que dans *il appert que*, on attribue à la forme *appert* une étiquette verbale. Donc, même si *apparoir* existe dans le dictionnaire pour des raisons de cohérence, il n'est pas possible de le trouver dans un texte.
Nous pourrions multiplier les exemples qui montrent que si le dictionnaire est un outil très utile, il faudra être très vigilant lors de son utilisation, car son contenu est aussi basé sur des traditions pédagogiques étrangères au traitement automatique du langage naturel. Aussi, la décision d'étiquetage individuel de chaque mot, et les informations relatives à chaque entrée reflètent cette tradition.

a	avoir	V	P3s
à	à	PREP	
aa	aa	N	ms:mp
ab	ab	XI	
abaissaient	abaisser	V	I3p
abaissait	abaisser	V	I3s
abaissant	abaissant	A	ms
abaissant	abaisser	V	G
abaisse	abaisse	N	fs
abaisse	abaisser	V	P1s:P3s:S1s:S3s:Y2s
abaissé	abaissé	A	ms

TAB. 3.2 – *Extrait du dictionnaire DELAF*

Nous avons indiqué que chaque entrée du dictionnaire possédait un comportement syntaxique différent : un symbole générique comme <*V*> qui correspond à toutes les formes dont la catégorie dans le dictionnaire est *V* est donc inutilisable. Une règle grammaticale qui comporterait ce symbole serait valable indifféremment pour toutes les formes de verbe. Nous ne connaissons aucune règle de ce type. Nous montrerons au chapitre 6 qu'on ne peut pas écrire une règle générale pour la négation d'un verbe, on ne peut pas non plus décrire par une règle générale la conjugaison d'un verbe à un temps composé. Quand nous le ferons, les règles produites seront donc des approximations.
Notre stratégie quant à l'utilisation du dictionnaire dépendra cependant des catégories :

- <*V*> (verbe). Nous différencions les verbes simples du lexique-grammaire (que nous n'avons pas traités puisqu'ils font l'objet de la thèse de E. ROCHE [19]) et les verbes entrant dans des structures verbales figées, que nous avons traités dans des grammaires individuelles dérivées du lexique-grammaire des structures verbales figées. Le rôle du dictionnaire des mots simples sera d'associer chaque forme à son lemme, et à ses informations flexionnelles.
- <*PRO*> (pronom). Nous n'avons pas du tout utilisé le dictionnaire pour leur catégorisation. A priori, il devrait être encore plus clair pour les pronoms que pour les verbes simples, qu'il n'y en a pas deux présentant le même comportement syntaxique, ni ayant une même fonction (comme nous pouvons le constater sur cet échantillon :

12. *il appert que* : il est évident que. – *il appert de* : il résulte de.

il, *lui*, *aucun*, *auquel*, *autrui*, *c*, *ç*, *ce*, *ceci*, *en*, *y*, etc.) Cependant, par commodité[13] nous utilisons le dictionnaire pour avoir accès à leurs informations flexionnelles (ce n'est pas une obligation, vu le nombre de pronoms et vu le fait que nous ferons au moins une règle particulière par pronom, nous aurions pu ajouter ces informations dans les règles mêmes.

- <*DET*> (déterminant). Nous avons adopté une approche par lexique grammaire et par graphes. Ceux-ci sont identifiés uniquement en présence du reste du groupe nominal. Cela supprime les ambiguïtés systématiques déterminant/pronom. Nous unifierons tous les groupes nominaux sous la forme *DET AN*. Une fois unifiés, nous pourrons alors faire référence à de tels groupes par les symboles génériques *DET* et *AN* (en fait nous n'opérerons plus alors au niveau du nom, mais à un niveau supérieur).

- <*ADV*> (adverbe). Les seuls adverbes analysables pour lesquels nous avons utilisé le dictionnaire DELAF sont les adverbes en *-ment*, tous les autres sont mieux décrits dans les tables d'adverbes figés. De plus, du fait de l'invariabilité des adverbes, le dictionnaire ne peut apporter aucune information flexionnelle.

- <*PREP*> (préposition). Dans cette catégorie d'éléments n'ayant pas de possibilité de flexion, aucun élément ne peut être substitué à un autre d'une manière régulière. Il ne serait donc pas non plus cohérent d'écrire des règles contenant le symbole <*PREP*>. Cependant pour localiser des groupes adverbiaux libres potentiels (sans vouloir les reconnaître), ou des compléments locatifs pour lesquels nous n'avons que peu de descriptions disponibles nous serons amenés à écrire des règles sommaires comme <*PREP*> <*GN*>, ou <*PREP*> <*N*>.

- <*N*> (nom). Cette catégorie très importante demande à être subdivisée du fait des comportements très différents. Les noms humains et les noms concrets non humains se distinguent des noms de procès décrits par les tables de nominalisations et de prédicats nominaux (tables F, E, et A). D'autres sont décrits par les dictionnaires de mots composés, et d'autres par des graphes spécifiques (par exemple les professions, voir 7.1.1). Pour toutes ces catégories, le dictionnaire ne sert encore qu'à donner des informations flexionnelles, la reconnaissance elle-même étant faite par des grammaires. Les noms concrets (+Conc) qui ont un comportement très homogène[14] et d'autres catégories de noms ne sont pas encore décrites. Nous utiliserons donc le symbole <*N*> pour reconnaître ces noms d'une manière provisoire.

- <*ADJ*> (adjectif). Même situation que pour les noms.

- <*XI*>. Cette catégorie ne sera pas du tout utilisée[15], puisque par définition, toutes les formes de cette catégorie font partie d'unités plus longues décrites ailleurs.

13. Et aussi pour ne pas se priver d'un moyen efficace et dédié pour représenter les informations flexionnelles.

14. Mais pas totalement, par exemple *rue* et *chemin* sont deux noms concrets sémantiquement très proches, et pourtant on peut les distinguer syntaxiquement par * *sur la rue* et *sur le chemin*, et *dans la rue* et * *dans le chemin*. Cette différence devra donc être indiquée quelque part : le dictionnaire ne contient donc pas toute l'information sur ces deux noms.

15. Dans une étape superficielle de correction de fautes typographiques, elle est toutefois indispensable, puisqu'elle permet de distinguer les formes qui n'existent pas telle quelle, ni comme partie d'une unité plus grande.

À plus forte raison, nous n'utiliserons pas le symbole «joker» <*MOT*> qui remplace n'importe quel autre mot.

Conclusion : Avant de présenter la première grammaire, nous fixons un cadre très restrictif et totalement indépendant du formalisme. Aucune règle de grammaire ne devra contenir de références aux dictionnaires à l'aide de symboles aussi généraux que <*V*>, <*ADV*>, <*PRO*> ou <*DET*>.

3.3.2 Les transformations

Nous présentons ici les transformations de HARRIS [15], et nous étudions ici leur comportement ainsi que la manière de les prendre en compte dans nos grammaires.

Considérons le schéma de phrase N_0 *V* N_1 (*Max lire*[16] *ce livre* en est un représentant). On peut appliquer à cette phrase une «opération» de passivation : *ce livre être lu par Max.* Appliquée au schéma général N_0 *V* N_1, cette opération est une transformation que l'on note **[passif]** :

$$N_0\ V\ N_1 \xrightarrow{\textbf{[passif]}} N_1\ est\ Vpp\ par\ N_0$$

On peut ainsi énumérer toutes les formes dérivées de notre schéma de base par des processus analogues (table 3.3 page 42) et les représenter sur un diagramme comme celui de la figure 3.3 en faisant ainsi apparaître les relations de commutativité entre les différentes transformations.

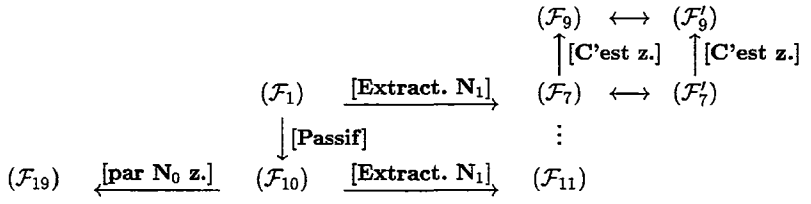


FIG. 3.3 – *Par convention, les transformations sont notées entre les crochets* [*et*]. *Ainsi la transformation associant* $\underline{N_0\ V\ N_1}$ *à* $\underline{C'est\ N_1\ que\ N_0\ V}$ *s'appelle l'extraction de* N_1 *et se note* [**Extract.** $\mathbf{N_1}$].

À chaque transformation, on associe une opération qui modifie l'analyse en phrases élémentaires des phrases auxquelles elle s'applique.
Par exemple, si on applique **[hier]** à *Max lire ce livre*, on obtient *Hier, Max lisait ce livre* et la décomposition en phrases élémentaires est :

(P_1) =:*Pierre lire ce livre.*
P_1 *se déroule hier.*

Par contre, la décomposition en phrases élémentaires d'une phrase est invariante par la transformation **[passif]**.
D'autres transformations sont binaires, elles combinent deux phrases pour en obtenir une

16. Nous avons en effet vu précédemment (page 32) que les schémas de phrases correspondaient à des verbes sans temps, ce que nous représentons ici par un infinitif. Pour des raisons de lisibilité nous représenterons aussi les phrases élémentaires au présent.

$(\mathcal{F}_1)$	$(N_0\ V\ N_1)_P$		Max lit ce livre.
$(\mathcal{F}'_1)$	$(N_0\ V\text{-}IL\ N_1?)_P$		Max lit-il ce livre?
$(\mathcal{F}_2)$	$(N_1,\ N_0\ V)_P$	*	Ce livre, Max lit.
$(\mathcal{F}_3)$	$(C'est\ N_0\ qui\ V\ N_1)_P$		C'est Max qui lit ce livre.
$(\mathcal{F}_4)$	$(qui\ V\ N_1\ ?)_P$		Qui lit ce livre?
$(\mathcal{F}_6)$	$(N_1,\ qui\ V?)_P$	*	Ce livre, qui lit?
$(\mathcal{F}_6)$	$(N_0\ qui\ V\ N_1)_{GN}$		(Max qui lit ce livre)$_{GN}$
$(\mathcal{F}_7)$	$(C'est\ N_1\ que\ N_0\ V)_P$		C'est ce livre que Max lit.
$(\mathcal{F}'_7)$	$(C'est\ N_1\ que\ V\ N_0)_P$		C'est ce livre que lit Max.
$(\mathcal{F}_8)$	$(que\ V\ N_0\ ?)_P$		Que *lit* Max?
$(\mathcal{F}'_8)$	$(que\ N_0\ V\text{-}IL?)_P$		Que Max lit-il?
$(\mathcal{F}_9)$	$(N_1\ que\ N_0\ V)_{GN}$		(ce livre que Max lit)$_{GN}$
$(\mathcal{F}'_9)$	$(N_1\ que\ V\ N_0)_{GN}$		(ce livre que lit Max)$_{GN}$
$(\mathcal{F}_{10})$	$(N_1\ est\ Vpp\ par\ N_0)_P$		Ce livre est lu par Max.
$(\mathcal{F}_{11})$	$(C'est\ N_1\ qui\ est\ Vpp\ par\ N_0)_P$		C'est ce livre qui est lu par Max.
$(\mathcal{F}_{12})$	$(qui\ est\ Vpp\ par\ N_0\ ?)_P$		Qu'est-ce qui est lu par Max?
$(\mathcal{F}_{13})$	$(N_1\ qui\ est\ Vpp\ par\ N_0)_{GN}$		(ce livre qui est lu par Max)$_{GN}$
$(\mathcal{F}_{14})$	$(N_1\ Vpp\ par\ N_0)_{GN}$		(ce livre lu par Max)$_{GN}$
$(\mathcal{F}_{15})$	$(C'est\ N_1\ qui\ est\ Vpp)_P$		C'est ce livre qui est lu.
$(\mathcal{F}_{16})$	$(N_1\ qui\ est\ Vpp)_{GN}$		(ce livre qui est lu)$_{GN}$
$(\mathcal{F}_{17})$	$(qui\ est\ Vpp\ par\ N_0\ ?)_P$		Qu'est-ce qui est lu par Max?
$(\mathcal{F}_{18})$	$(C'est\ par\ N_0\ que\ N_1\ est\ Vpp)_P$		C'est par Max que ce livre est lu.
$(\mathcal{F}_{19})$	$(N_1\ est\ Vpp)_P$		ce livre est lu.
$(\mathcal{F}_{20})$	$(N_0\ PRO\ V,\ N_1)_P$		Max le lit, le livre.

TAB. 3.3 – *Quelques transformations de la structure élémentaire* $N_0\ V\ N_1$ *(par exemple «Max lit ce livre»)*

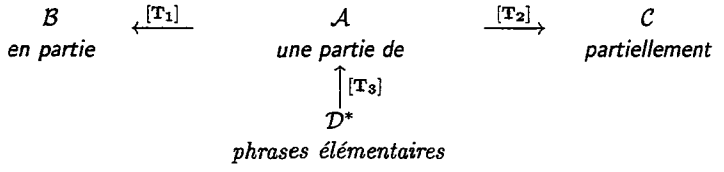


FIG. 3.4 – *Transformations définissant une classe de déterminants.*

troisième.
Le rôle des grammaires transformationnelles est d'arriver à décomposer toute phrase complexe en phrases élémentaires en reconstituant l'ensemble des transformations lui ayant été appliquées. Pour cela, on choisit dans tout diagramme de transformation une forme dite canonique (dans le diagramme de la figure 3.3, il s'agit clairement de la forme $(\mathcal{F}_1)$). Cette forme canonique est la phrase élémentaire à laquelle l'analyse doit aboutir.

Nous voulons montrer que l'analyse par transformation met en évidence des phénomènes linguistiques importants en associant des structures syntaxiques différentes, mais elle ne permet pas d'aboutir à ce résultat sans introduire la notion de forme théorique. Pour mieux cerner le problème, considérons quelques exemples :

(6) *Max a en partie lu ce livre.*

en partie est un adverbe provenant d'un déterminant adverbial [10] dont l'analyse est la suivante :

(7) *Max a lu une partie de ce livre.*

Une autre transformation sur (7) existe qui donne :

(8) *Max a partiellement lu ce livre.*

(9) *Max a lu ce livre partiellement.*

Il s'agit encore de la même phrase avec la même interprétation. Nous pouvons la décomposer en deux phrases élémentaires (en ajoutant un démonstratif qui ne change pas fondamentalement notre propos) :

(10) *Ceci est une partie de ce livre.*

(11) *Max a lu ceci.*

Si on note $[\mathbf{T_1}]$ la transformation permettant de passer de (7) (que l'on note forme $\mathcal{A}$) à (6) (forme $\mathcal{B}$), et $[\mathbf{T_2}]$ la transformation permettant de passer de (7) à (8) (forme $\mathcal{C}$), on peut définir une classe de déterminants vérifiant ces propriétés dont l'analyse (transformation $[\mathbf{T_3}]^{-1}$) donne (10), (11) (forme $\mathcal{D}^*$, l'étoile indique que c'est la forme canonique dans notre schéma transformationnel). On peut représenter ces différentes formes sur le diagramme de la figure 3.4.
Notre objectif est de systématiser le passage de $\mathcal{A}$, $\mathcal{B}$, ou $\mathcal{C}$ vers $\mathcal{D}^*$ pour tous les membres de la classe vérifiant les propriétés énoncées. Par exemple, *la totalité de* reçoit la même analyse : ses formes transformées sont *en totalité* ($\mathcal{B}$) et *totalement* ($\mathcal{C}$).
Considérons maintenant le déterminant composé *une (grande+petite) partie de* provenant de $\mathcal{B}$ par une transformation notée $[\mathbf{T_4}]$. Ce déterminant composé se transforme par $[\mathbf{T_1}]$ en *en grande partie*, mais ne se transforme pas par $[\mathbf{T_2}]$ en $\mathcal{C}$. Il vérifie donc le diagramme partiel (figure 3.3.2). Nous avons donc trouvé un déterminant faisant partie de la même classe. Sémantiquement, c'est clair ; syntaxiquement, ce déterminant dérive d'un élément de la classe, et de plus il vérifie trois des quatre formes définitionnelles de la classe. Nous

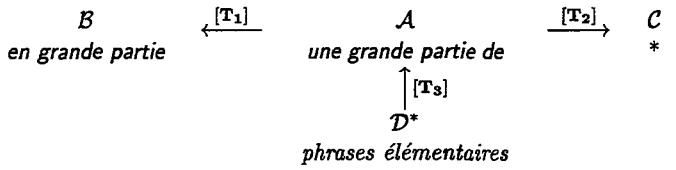


FIG. 3.5 – *Elément vérifiant seulement une partie des propriétés de la classe.*

étendons donc la définition de la classe aux formes vérifiant «la plupart» des relations. En fait, ici l'absence de la forme C n'est pas très gênante puisque la forme de base $\mathcal{D}^*$ (celle vers laquelle on veut dans tous les cas arriver) existe.
Si on continue le recensement des adverbes et des déterminants présentant ces caractéristiques, on trouve aussi *en entier* (forme B), et *complètement* (forme C). Testons l'existence des autres formes :

(12) *Max a en entier lu ce livre.* (B)
(13) *Max a complètement lu ce livre.* (C)
(14) *Max a entièrement lu ce livre.* (C)
(15)? *Max a lu l'intégralité de ce livre.* (A)
(16)? *Max a lu ce livre au complet.* (B)
(17)* *Max a lu la complétude de ce livre.* (A)

La forme (16) ne sonne pas très bien, nous verrons page 151 que *au complet* modifie plutôt un collectif (*J'ai complètement lu cette trilogie*, *J'ai lu cette trilogie au complet*). Cette forme est bien la forme B dans notre schéma, même si le déterminant n'est plus le même, ce qui est confirmé par le fait qu'on observe aussi la propriété de modification par l'adjectif *grand* : *au grand complet.*
(15) est correcte, mais il ne s'agit pas[17] de la forme A de (12). En effet, la forme B *en intégralité* existe, ce qui confirme qu'il existe les deux schémas différents :

en intégralité ↔ *l'intégralité de* ↔ *intégralement*

en entier ↔ **l'entier de* ↔ *entièrement*

De plus, nous observons l'existence de la forme E *dans son intégralité*, qui n'existait pas pour *en partie*, mais qui existe pour *en totalité.*
Enfin (17) est impossible.
Ces exemples prouvent qu'il existe les formes B et C, indépendamment de la forme A, et donc *a fortiori* de la forme dite élémentaire $\mathcal{D}^*$. Cela prouve qu'il ne sera pas possible d'obtenir les phrases élémentaires $\mathcal{D}^*$ pour ces deux exemples.
Notre schéma de base peut être étendu avec les nouvelles transformations observées. On note $[\mathbf{T_4}]$ la transformation associant *en partie* à *en grande partie*, et $[\mathbf{T_5}]$ la transformation associant *la totalité de* à *dans sa totalité* et E' l'état *dans sa grande majorité* (figure 3.6).
Si la forme de base $\mathcal{D}^*$ n'est pas commune à tous les représentants de cette classe, il faut envisager une autre manière d'interpréter ces déterminants/adverbes.
Nous pouvons exprimer ceci autrement : nous avons remarqué que certaines formes (A, B et C) reliées entre elles par des transformations pouvaient être représentées par une forme

17. Sémantiquement, on pourrait trouver des rapprochement ; morphologiquement il faudrait revenir au latin pour trouver un lien, mais ces deux critères sont de toute façon sans valeur.

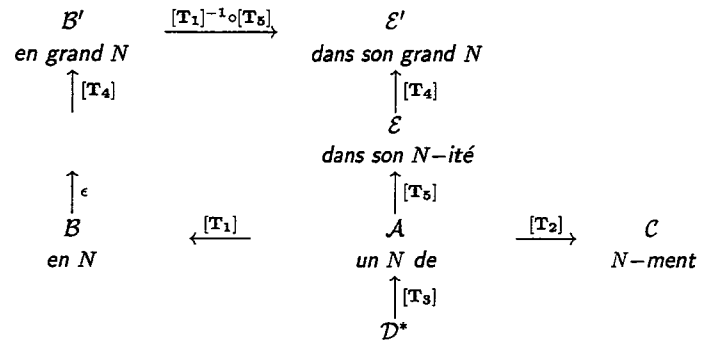


FIG. 3.6 – *Extension de la classe à d'autres transformations.*

de base $\mathcal{D}$. Cette forme plus simple semble être le bon candidat aux phrases élémentaires décrivant nos formes initiales. Il se trouve que pour d'autres éléments de la classe, cette forme n'existe pas. On peut opérer de la même manière que dans la définition du DELAF quand une forme n'était pas régulière:

- pour les adjectifs, on décide que la forme la plus proche de la forme canonique est le lemme. Cela revient à choisir un autre représentant de la classe, ce qui pose un problème de cohérence pour l'ensemble des phrases élémentaires obtenues: en effet, *ce livre en entier* serait un schéma élémentaire, alors que *ce livre en partie* ne le serait pas.

- pour les verbes, on invente un lemme (*apparoir*) permettant de compléter notre classe. Notons *entièreté* le nom permettant de compléter le diagramme de transformation de *ce livre en entier*. Nous avons donc la séquence *l'entièreté de ce livre*, qui est la forme $\mathcal{A}$ du diagramme. Ceci permet de définir un lemme pour notre forme, le prix en est introduction de formes théoriques non acceptées qui vont se mélanger avec des formes acceptées.

Ici, une autre solution pourrait être envisagée: pour chacun des éléments, on choisit comme représentant de la classe, un objet extérieur au langage (par exemple, une fonction logique qui «interprète» les phrases élémentaires). On introduirait alors des formes théoriques, c'est-à-dire un métalangage bien distinct du langage lui-même.

Ce que nous venons d'observer sur un cas particulier révèle un fait très général que l'on peut exprimer ainsi:

> Soit $\mathcal{S}_1$, une famille homogène de phrases sur lesquelles s'appliquent une transformation $\mathcal{T}$ donnant la famille de phrase $\mathcal{S}_2$. $\mathcal{S}_2$ présente un certain nombre de propriétés syntaxiques communes dérivées de $\mathcal{S}_1$. Nous pourrons souvent trouver une phrase S, dont les propriétés syntaxiques seront communes aux membres de $\mathcal{S}_2$ et n'ayant pas d'image par T^{-1}. (C'est-à-dire ne correspondant pas à une phrase attestée).

Pour montrer la généralité de cette «règle», considérons encore quelques exemples sur des transformations plus classiques:

1. La transformation **[Passif]** :

   (18) *Max est pris par le temps.*
   (19)* *Le temps prend Max.*
   (20)? *Max est pris de court par le temps.*
   (21) *Le temps prend Max de court.*
   (22) *Léa est atteinte par LE (maladie+vieillesse+mal...)*
   (23) *Léa est (très+gravement+ϵ) atteinte.*
   (24)* *LE (maladie+vieillesse+mal...) atteint Léa.*
   (25) *Le car est bondé de touristes.*
   (26) *On a bondé ce car de touristes.*

   (18) est une phrase de structure passive, le «complément d'agent» *par le temps* est indispensable et la forme active (19) n'est pas acceptable. En rajoutant *de court*, la phrase (20) est étrange car *de court* est redondant, mais la forme active (21) est acceptable.
   (22) est une construction beaucoup plus productive (donc moins figée), ici le complément d'agent est facultatif (23) et *atteinte* est modifiable par des adverbes comme *très* (spécifique de verbe) ou *gravement*. La forme active est interdite.
   (25) est une construction passive courante du verbe *bonder*, alors que la construction active (26) est attestée mais beaucoup moins naturelle.
   Ces trois exemples montrent qu'il est possible de trouver des formes verbales passives sans correspondant actif, et qu'il y a aussi une certaine continuité entre les deux : certaines formes sont préférées au passif, mais on peut toujours en faire l'analyse en partant d'un actif. Pour d'autres formes dont l'actif semble impossible, on remarque que légèrement modifiée la forme à l'actif est acceptable, mais plus celle au passif (phrases (20) et (21)).

2. La transformation **[Négation]** :

   (27) *Luc ne manquera pas de venir nous voir.*
   (28)* *Luc manquera de venir nous voir.*
   (29) *Il a été brûlé vif.*
   (30) *Il a été brûlé.*
   (31)?* *Il n'a pas été brûlé vif.*
   (32) *Il n'a pas été brûlé.*

   On retrouve le même phénomène, (27) n'existe qu'avec une négation, (29) n'est pas correcte avec la négation[18].

3. La transformation **[Nominalisation]**. Nous donnons plus de détails sur cette opération en 7.1.1. Brièvement elle associe des phrases construites avec des «verbes pleins», avec des phrases à verbe support dans laquelle le verbe est transformé en nom. Les arguments du verbe sont conservés, ainsi que de nombreuses propriétés syntaxiques : (33)→ (34)→(35)→ (36) :

18. On peut accepter (31) en contraste : *Il n'a pas été brûlé vif, il a été écartelé.* Alors que (32) est complètement naturel. Cette analyse par contraste est propre à la négation. Nous décrivons ce phénomène au chapitre 6

(33) *Luc propose une balade à Léa.*
(34) *Luc fait une proposition de balade à Léa.*
(35) *La proposition de Luc a Léa...*
(36) *Luc fait une proposition à Léa.*
(37) *Luc fait une proposition intéressante à Léa.*
(38) *Luc fait une plaisanterie à Léa.*
(39) *La plaisanterie de Luc à Léa...*
(40)* *Luc plaisante à Léa.*
(41) *Luc plaisante.*
(42) *Luc fait des plaisanteries.*

Alors que (42) peut provenir de (41), la forme équivalente (40) n'est pas acceptable quand on rajoute l'argument *à Léa* (38). De plus, si on ajoute l'adjectif *intéressante* à *proposition* comme dans (37), alors il n'est plus possible de trouver un correspondant à la forme verbale.

Tous ces exemples prouvent que s'il est possible d'appliquer des transformations à des phrases simples pour obtenir des phrases plus complexes, il n'est pas toujours possible de décomposer les phrases complexes en phrases simples «canoniques». Les raisons de ce phénomène peuvent être diachroniques: ainsi, certaines formes disparaissent dans certaines classes, ce que des recherches diachroniques pourraient faire apparaître. Mais il peut aussi s'agir de faits complètement formalisables: par exemple on peut modifier presque tout passif par *très*, puisque la grammaire de *très* veut qu'il s'incorpore entre un auxiliaire et un participe passé. Cependant, en revenant à l'actif la modification ne peut plus s'appliquer si le temps n'est pas composé:

(43) *Luc regarde cette émission.*
(44)? *Cette émission est regardée par Luc.*
(45) *Cette émission est très regardée par Luc.*
(46)* *Luc très regarde cette émission.*

Plus généralement, cela est dû au fait qu'une transformation ne s'applique pas à des phrases particulières, mais à une classe homogène définie par des propriétés (cette classe peut-être réduite à un seul élément). La classe image par la transformation est définie par l'image des propriétés de la première classe. Dans cette nouvelle classe, on peut trouver des éléments n'ayant pas d'antécédent par la transformation.

Cette explication englobe d'une manière très générale le phénomène des exceptions. Le processus de conjugaison est aussi une transformation : par exemple à toute forme verbale infinitive, on peut associer une forme verbale à la troisième personne du singulier du présent de l'indicatif. Notons $[\mathbf{d_{3sP}}]$ cette «transformation» (que nous décrivons sur un mot mais qui devrait être définie dans le cadre d'une phrase). Cette transformation associe l'ensemble des formes verbales $\mathcal{P}_{3s}$ à l'ensemble des formes verbales infinitives $\mathcal{W}$ ce que nous pouvons représenter par le diagramme suivant:

$$\begin{array}{ccc} \mathcal{P}_{3s} & \xleftarrow{[\mathbf{d_{3sP}}]} & \mathcal{W}^* \\ N_0 \textit{ mange } N_1 & & N_0 \textit{ manger } N_1 \end{array}$$

Dans le cas général, nous voulons associer à tout élément de $\mathcal{P}_{3s}$, sa forme infinitive, *i.e.*, sa forme transformée dans $\mathcal{W}$. Dans ce cas, le principe de «dépendance» des deux classes s'applique puisque nous avons vu qu'il existait un élément de $\mathcal{P}_{3s}$, *appert*, n'ayant pas d'équivalent par $[\mathbf{d_{3sP}}]$ dans $\mathcal{W}^*$, et pour remédier à cela, $\mathcal{W}^*$ est complété dans le

dictionnaire à l'aide de la forme théorique *apparoir*.

Conclusion : Les transformations sont des moyens très simples et très naturels pour transformer des schémas de phrase en d'autres. On arrive ainsi à associer naturellement des schémas de phrases différents entre eux, de la même manière qu'on associe les noms pluriels aux noms singuliers correspondants. Cependant, de la même façon qu'il n'est pas possible d'avoir un représentant canonique pour une famille de nom (le lemme), il ne va pas être possible de trouver un représentant ayant une forme canonique (par exemple l'actif) pour une famille de schémas de phrases en relation par des transformations. Aucune des familles de phrases n'est assurée d'être complète vis-à-vis des autres familles associées. Vouloir revenir à une forme canonique (la phrase élémentaire) va donc buter sur ce problème.

Deux solutions permettent *a priori* de le contourner. La première est d'introduire des familles de phrases théoriques (comme pour les verbes dont l'infinitif n'existait pas, on peut créer un infinitif théorique assurant la fermeture du langage par rapport à la grammaire). Cette solution n'est guère satisfaisante à ce niveau d'analyse, parce qu'elle conduit à manipuler des objets non grammaticaux. La deuxième solution, uniquement descriptive, consiste à représenter par un graphe comme celui de la figure 3.7 l'ensemble des structures possibles et de ne pas essayer de «défaire» les transformations. De toute façon, chaque chemin correspondant à un schéma différent, il suffira de leur associer individuellement une représentation logico-sémantique. Formellement, cela revient à dire que le représentant choisi de la famille de schémas de phrase, est un objet non linguistique : le graphe lui-même. C'est cette solution que nous avons adoptée.

3.3.3 Le lexique-grammaire

Le lexique-grammaire désigne l'ensemble des tables de schémas de phrases élémentaires élaborées au LADL. Ce lexique-grammaire se décompose en diverses catégories : le lexique-grammaire des verbes simples, le lexique-grammaire des verbes figés, le lexique-grammaire des adverbes figés, etc.

Chaque table de verbes simples du français regroupe un ensemble de verbes au comportement syntaxique homogène (les propriétés communes à toutes les entrées de chaque table, sont appelées propriétés définitionnelles). Il se trouve que certaines de ces tables correspondent à peu près avec des catégories sémantiques, mais ce n'est pas un critère de constitution des tables. Par exemple, la table appelée 38L1 (dont nous donnons un extrait dans le tableau 3.6) contient des verbes exprimant un déplacement sans que l'on puisse poser la question *où ?* sur le complément de lieu. Par définition, ce sont des constructions transitives, de la forme N_0 *V* N_1*lieu*.

Parmi les propriétés de la table, $N_0 =: Nhum$ indique la possibilité de sujet humain. *V statique* indique si le verbe a une interprétation statique (donc sans déplacement : par exemple, *le chemin monte la colline*).

Considérons les deux entrées *gravir* et *grimper* :

Paul gravit la colline

Paul grimpe l'escalier

Ces deux verbes semblent très proches, ils sont cependant distingués dans la table par la propriété $N_0 = N - hum$. Le verbe *gravir* n'accepte pas, en effet, de sujet non humain [19]

19. Il faut néanmoins préciser cette opposition catégorie humain/non humain, dans : *le robot gravit la colline*, le sujet est une extension de humain.

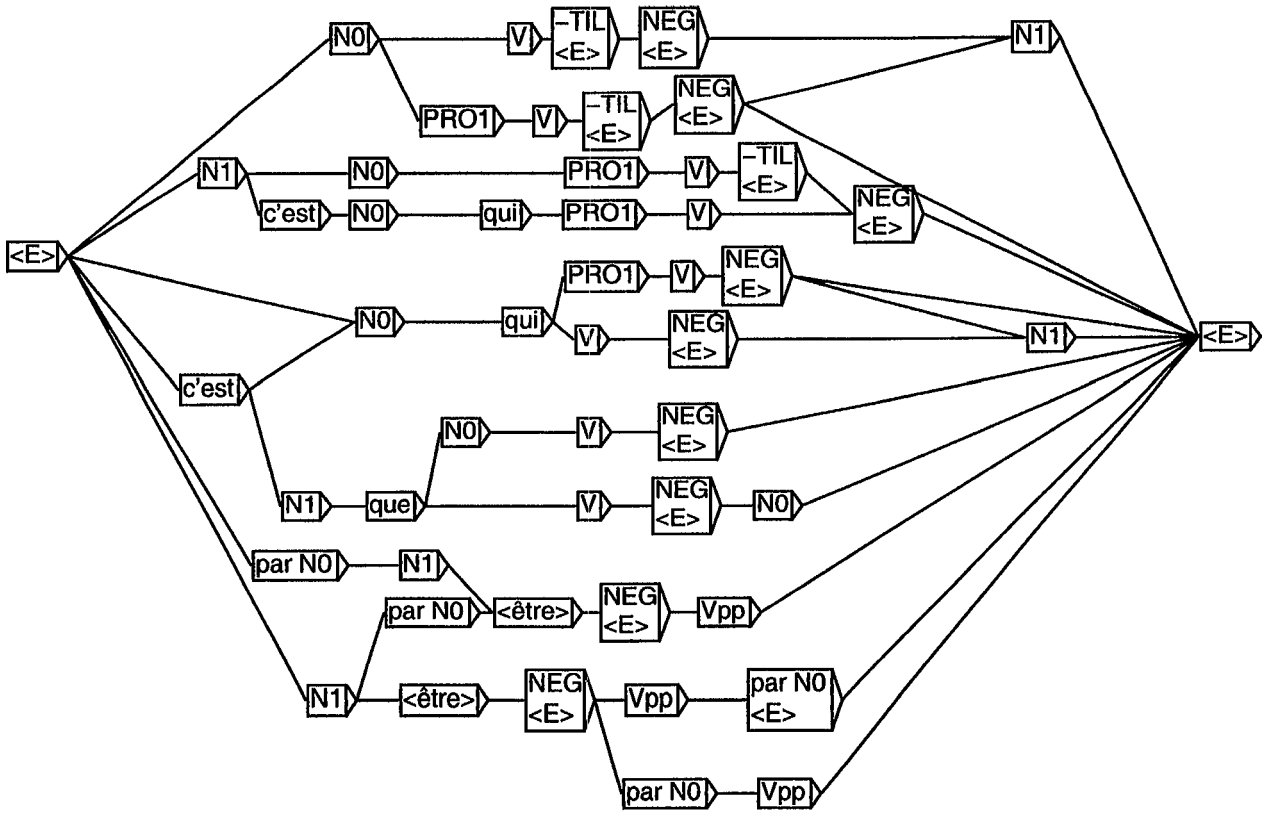


FIG. 3.7 – *Représentation des différentes structures dérivées de la structure canonique* $N_0\ V\ N_1$

$N_0 =: Nhum$	$N_0 =: N-hum$	V statique	V mouvement	$N_0 =: chemin$		$N_0\ V\ Loc\ N_1$	$Prép =: dans$	$Prép =: sur$	$Prép =: contre$	$Prép =: à$	$Prép =: par$	$V - inf\ W$	N_1 destination	N_1 source	N_1 lieu du procès	N_1 lieu de passage	N_1 en mouvement sur combien?
+	−	−	+	−	abandonner	−	−	−	−	−	−	+	−	+	−	−	−
+	−	−	+	−	aborder	+	−	−	−	+	−	+	+	−	−	−	−
+	−	+	−	−	accaparer	−	−	−	−	−	−	−	+	−	−	−	−
+	−	+	+	+	gravir	−	−	−	−	−	−	+	−	−	+	−	+
+	+	+	+	+	grimper	−	−	−	−	−	−	+	−	−	+	−	+

TAB. 3.6 – *Extrait de la table* 38*L*1

alors que le verbe *grimper* le permet.

Les symboles binaires − et + se prêtent parfaitement à une utilisation informatique (telle structure est autorisée ou interdite).

3.3.4 Grammaire d'analyse et grammaire de description

Toutes les transformations que nous venons de décrire opèrent sur la structure profonde de la phrase, c'est-à-dire que la fonction de chaque constituant est connue. Ces transformations ne s'appliquent donc pas à des suites de mots, mais à des structures. Quand toutes les transformations ont été appliquées, cette structure est effacée et ne retient que les mots (la surface de la structure). Nous notons cette opération |**Réduction Phrase**|, il ne s'agit pas d'une transformation. Cette opération est la dernière opération s'appliquant à la séquence. En effet, même les règles qui semblent être les plus proches de la «surface» comme les règles de contractions (*de le*→ *du*) ou d'accord vocalique (*ma armoire*→ *mon armoire*) s'appliquent à des objets identifiés (c'est-à-dire ayant une fonction bien définie) et non pas à des suites de caractères. Dans le premier cas, il s'agit du *le* déterminant et non pas du *le* pronom, dans le deuxième cas, le *ma* est déterminant[20]. Ces opérations sont donc appliquées avant cette opération |**Réduction Phrase**|.
Nous avons représenté sur la figure 3.8 les opérations conduisant à la formation de la phrase (47) (en simplifiant un peu).

(47) Les dortoirs de l'internat garçons datent en grande partie du début du siècle.

Sur cette figure, nous voyons en particulier que l'adverbe *en partie*, provenant du déterminant adverbial *une grande partie de N*, est mobile (comme tous les adverbes), et que la transformation [**Mobilité**] le déplace après le verbe.

Supposons que nous voulions appliquer l'inverse de toutes les transformations au texte pour retrouver la décomposition en phrases élémentaires. La première étape consistera à appliquer |**Réduction Phrase**|$^{-1}$. Cette première étape représente en fait l'analyse syntaxique complète de la phrase: en particulier il faut déterminer que *ces dortoirs* est le

20. Même si ce mot n'est apparemment pas ambigu, la phrase *Ma arme son fusil*, présente la même succession vocalique que *Ma armoire se fissure*. Et pourtant, la règle d'accord vocalique ne s'applique qu'à la deuxième phrase!

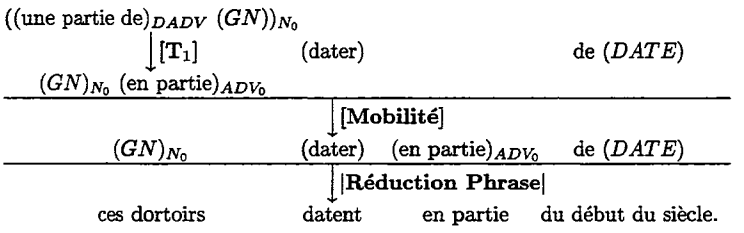


FIG. 3.8 – *Décomposition de la formation de la phrase* (47). [$\mathbf{T_1}$] est la transformation décrite précédemment, [**Mobilité**] est une transformation de mobilité adverbiale.

groupe nominal sujet de la phrase, que *en partie* est rattaché à ce groupe nominal... Nous ne pouvons donc pas espérer défaire simplement cette réduction: puisque tout le travail de l'analyse sera justement d'y arriver!
Pour pouvoir appliquer [**Mobilité**]$^{-1}$, la connaissance de la structure de la phrase est cependant indispensable: l'adverbe ne peut être déplacé qu'entre deux constituants de la phrase élémentaire.
Ceci montre que les opérations que nous allons appliquer au texte, ne pourront pas être l'inverse des opérations l'ayant construit. **La grammaire d'analyse** sera distincte de **la grammaire de description**. Nous serons conduit à passer par des analyses non naturelles: dans notre exemple, il faudra analyser *en partie* comme un adverbe, et vérifier ensuite s'il peut compléter un groupe nominal, sans quoi cette analyse serait fausse.
Par exemple dans la phrase (48), nous devons commencer par marquer *en partie* comme un adverbe (49) (parallèlement à d'autres analyses qui reconnaîtront *partie de tennis*). L'analyse des autres mots de la phrase donne (50), et comme le schéma de phrase élémentaire (51) n'existe pas, l'analyse échoue. Dans cette analyse, nous avons donc dû «remonter» jusqu'à la première étape du diagramme de transformation 3.8 avant d'appliquer [**Mobilité**]$^{-1}$.

(48) *Notre jeu de balle s'est vite transformé en partie de tennis.*
(49) *Notre jeu de balle s'est vite transformé (en partie)$_{ADV}$ de tennis.*
(50) *(notre jeu de balle)$_{N_0}$ se transformer (vite)$_{ADV}$ (en partie)$_{ADV}$ de (tennis)$_N$.*
(51)* *N_0 se transformer de N.*

Bien sûr, l'ordre que nous proposons ici n'est pas le seul: après avoir reconnu le sujet du verbe *se transformer* dans (50), on pourrait appliquer [**Mobilité**]$^{-1}$ (cette transformation sera seulement partielle puisqu'on ne pourra déplacer l'adverbe que dans la partie reconnue) pour obtenir (52), on applique ensuite [$\mathbf{T_1}$]$^{-1}$ qui conduit à (53). Ensuite, l'analyse de l'adverbe *vite* donne (54). L'analyse échoue ensuite, car aucun schéma de phrase élémentaire ne correspond à (51).

(52) *(notre jeu de balle)$_{N_0}$ (en partie)$_{ADV}$ se transformer (vite)$_{ADV}$ de (tennis)$_N$.*
(53) *(une partie de notre jeu de balle)$_{N_0}$ se transformer (vite)$_{ADV}$ de (tennis)$_N$.*
(54) *(une partie de notre jeu de balle)$_{N_0}$ se transformer de (tennis)$_N$. Cela s'est fait vite.*

Dans ces deux stratégies d'analyse, l'ordre et la nature des opérations diffèrent de l'ordre et de la nature des transformations «naturelles» sous-jacentes. Le choix des «règles de grammaire» est fixé par l'auteur de la grammaire, en partie indépendamment des ressources grammaticales existantes. L'analyse, elle, se fait «en aveugle» et il faut faire des

approximations (qui seront ensuite vérifiées) : le choix de la stratégie est libre. Aussi, malgré les descriptions très détaillées des phénomènes linguistiques que nous décrirons dans la partie II, nous avons dû faire des choix dans la stratégie de la reconnaissance. Ces choix sont personnels, peut-être discutables, et les critères pour les juger sont leur lisibilité et leur efficacité.
Traditionnellement en informatique, on distingue, pour une grammaire formelle, l'analyse montante et l'analyse descendante. La première part du texte pour arriver aux règles, la seconde part des règles pour arriver au texte. Nous ne pouvons pas effectuer la seconde car les transformations étant des opérations n'augmentant pas nécessairement la taille de la séquence (et même pouvant la diminuer), le nombre des opérations pouvant générer une séquence de longueur finie peut être aussi grand que l'on veut, donc l'appartenance d'un mot au langage n'est pas décidable. Nous ferons donc la première[21], mais comme les règles de la grammaire de description ne sont pas toutes inversibles, nous devrons en définir de nouvelles permettant de retrouver les règles de la vraie grammaire.
Nous montrons deux autres situations, engendrée par |**Réduction Phrase**|, et compliquant le travail de construction de la grammaire.

3.3.5 Interdépendance des phénomènes linguistiques

Nous venons de voir qu'il est possible d'associer à toute phrase complexe :

- un ensemble de phrases élémentaires ;
- un ensemble de transformations ;
- et une «formule» donnant le schéma d'application de ces transformations sur les phrases élémentaires pour obtenir la phrase complexe initiale.

Vu le nombre de transformations, il serait souhaitable de décrire les phénomènes correspondant à chacune de ces transformations d'une manière indépendante. Par exemple, dans le graphe de la figure 3.7, nous faisons référence aux sous-unités N_0, N_1 et NEG, ce qui sous-entend que nous pouvons les décrire séparément. Si nous voulions incorporer à ce graphe la description des groupes nominaux, celui-ci deviendrait gigantesque et de plus nous dupliquerions beaucoup des informations présentes dans les autres schémas de phrases élémentaires, ce qui est contraire à la factorisation espérée par la description linguistique. Nous étudions ici l'indépendance entre les règles décrivant les groupes nominaux, la négation appliquée à un verbe, et les schémas de phrases élémentaires.

Supposons que nous disposions des trois règles suivantes : (R_1) qui décrit la négation appliquée à un verbe, (R_2) qui décrit les groupes nominaux, et (R_3) qui décrit un schéma de phrase élémentaire (par exemple N_0 *lit* N_1 table **32R2**). Nous allons montrer qu'il n'est pas possible d'ordonner l'application de ces trois règles, et même, que ces trois règles ne peuvent pas vraiment être définies indépendamment. Considérons les phrases :

21. En fait, ce n'est pas si simple : quand nous ferons apparaître un verbe support effacé, nous irons chercher le verbe support approprié dans la table adaptée plutôt que de faire réapparaître tous les verbes supports possibles ce qui conduirait à un grand nombre d'analyses parallèles, dont une seule serait validée *in fino*. L'analyse montante est donc contrôlée par «le haut».

(55) *Luc lit ce livre.*
(56) *Luc ne lit pas ce livre.*
(57) *Luc ne lit pas de livre.*
(58) *Luc ne lit pas de livres.*
(59) *Luc n'a lu ce livre à personne.*

Détaillons la démarche à suivre pour analyser chacune de ces phrases à partir des trois règles précédentes :

- Pour analyser (55), nous devons appliquer d'abord (R_2) qui construira
  $(Luc)_{GN}$ *lit* $(ce\ livre)_{GN}$
  puis (R_3) qui construira
  $N_0 = (Luc)_{GN}\ V = lire\ N_1 = (ce\ livre)_{GN}$
  L'ordre d'application des règles est donc (R_2) $\longrightarrow$ (R_3).

- Pour analyser (56), nous devrons appliquer d'abord (R_1) qui reconnaîtra la négation :
  Luc lit $(ne_pas)_{NEG}$ *ce livre*
  puis (R_2) pour obtenir :
  $(Luc)_{GN}$ *lit* $(ne_pas)_{NEG}$ $(ce\ livre)_{GN}$
  et ensuite (R_3). En fait, on aurait pu commencer par (R_2) et appliquer ensuite (R_1), les deux règles sont ici indépendantes. L'ordre d'application des règles est donc (R_1) $\longrightarrow$ (R_2) $\longrightarrow$ (R_3) ou (R_2) $\longrightarrow$ (R_1) $\longrightarrow$ (R_3)

- Dans les phrases (57) ou (58), par contre, il est indispensable de reconnaître d'abord la négation qui ici est en trois parties *ne...pas...de*. En effet, le *de* de *de livre* provient de *ne lit pas un livre* et celui de *de livres* provient de *ne lit pas des livres*. Remarquons que *de livre* n'est pas forcément situé juste après la négation, nous pouvons avoir des insertions d'adverbes, ou une permutation de l'objet direct[22] comme dans les phrases suivantes :
  (60)? *De livre, Luc ne lit pas.*
  (61) *Luc ne lit pas, aujourd'hui, de livre.*
  Cela appelle plusieurs remarques : le choix du déterminant *un* ou *des* à substituer à *de* nécessite de connaître le nombre du groupe nominal. De plus, la localisation du groupe nominal concerné par la négation nécessite l'analyse complète de la phrase. Pour appliquer (R_1), nous avons donc besoin des résultats des règles (R_2) et (R_3), (R_3) ne pouvant pas être appliqué sans l'application préalable de (R_1) !

  On pourrait éventuellement dire que *de livre* est un groupe nominal et on pourrait alors commencer par (R_2). Mais cette analyse n'est valide que s'il s'agit du complément d'objet d'un verbe portant une négation. Pour savoir s'il s'agit bien d'un complément d'objet direct d'un verbe portant une négation, il faudrait que (R_3) soit appliqué. Nous retrouvons la même contradiction que précédemment puisque cela est contradictoire avec l'hypothèse que nous commençons l'analyse par (R_2).

22. Cela est décrit comme nous l'avons vu dans la partie précédente dans la règle de reconnaissance du schéma élémentaire

Nous résolvons ce problème au chapitre 6 en utilisant les deux règles suivantes :
(R_1) *ne V pas* $\longrightarrow$ V:+Neg *ne_pas*
(R_2) *de N*:s $\longrightarrow$ $(un\ N)_{GN}$:+N_1+Neg
et en vérifiant ensuite lors de la reconnaissance de la phrase élémentaire (R_3), que si un groupe nominal possède l'information +N_1 il ne peut être qu'en position d'objet direct, et que s'il possède l'information +Neg, alors le verbe doit porter lui aussi une négation.
Cela signifie que les formes suivantes :

* *Je lis de livre.*

* *D'enfants ne lisent pas.*

Recevront les analyses partielles :

Je lis (de livre)$_{GN}$+N_1+Neg.

(D'enfants)$_{GN}$+N_1+Neg *lisent*+Neg *ne pas.*

Mais l'analyse s'arrêtera là, car dans la première phrase le verbe ne porte pas de négation, et dans le deuxième cas, il n'y a pas de N_0 (puisque *d'enfants* est marqué +N_1). Nous avons donc conservé une indépendance relative entre les trois règles, mais (R_3) comportera quand même des tests concernant l'application des deux règles précédentes.

Une alternative à cette méthode localiserait le complément N_1 dans la règle de description de la négation. Pour cela, il faudrait dans la règle décrivant la négation, reprendre les structures de la phrase élémentaire donnée par la règle (R_3), les règles ne seraient donc pas plus indépendantes.

– Dans (59), *personne* est un groupe nominal (négatif) complément du verbe, il implique la présence du *ne* devant celui-ci. Pour pouvoir rattacher le *ne* à *personne*, il faut encore une fois utiliser un artifice : on reconnaît le groupe nominal *personne*[23] isolément dans un premier temps, on reconnaît *ne V* dans un deuxième temps, et dans la phase de reconnaissance de la phrase, on vérifie que les deux parties sont présentes. Associer le *ne* à *personne* directement est possible dans le cas d'une phrase à compléments libres, mais n'est pas réalisable pour des constructions figées comme celles des exemples suivants[24] :

(62) *Il donne lecture de ce journal à Max.*
(63) *Il en donne lecture à Max.*
(64) *Il n'en donne lecture à personne.*
(65) *Lecture n'en est donné à personne.*
(66) *À personne, lecture n'en est donné.*
(67) *Le mérite de ce travail revient à Max.*
(68) *Le mérite de ce travail ne revient pas à Luc.*
(69) *Le mérite de ce travail ne revient à personne.*
(70) *À personne, le mérite de ce travail ne revient.*
(71)? *C'est à personne que le mérite de ce travail ne revient.*

23. Il peut être beaucoup plus compliqué comme *personne d'autre que GN*.

24. Pour que le phénomène soit net, les phrases (66), (70) et (74) sont difficilement acceptables sans contexte. Cependant, avec un contexte montrant l'insistance sur *personne*, elles deviennent tout à fait correctes. Par exemple, *Puisqu'il en est ainsi, à personne d'autre que Luc, lecture n'en sera faite !*

(72)? *Ce n'est à personne que le mérite de ce travail ne revient.*
(73) *Grand bien fasse à Luc.*
(74)? *À personne, grand bien ne fasse.*

Dans (66), (70) et (74) le groupe nominal séparant le complément *à personne* du verbe portant le *ne* n'est pas descriptible par une grammaire générale (sinon en acceptant que *lecture* seul est un groupe nominal, cela conduirait à accepter que tout nom seul est aussi un groupe nominal potentiel ce qui n'est vrai que dans des cas très restreints. La seule solution pour «sauter» ce groupe nominal serait donc d'utiliser une séquence indéterminée (<MOT>*) ce qui est contraire à notre méthodologie principale. Ceci montre qu'il existe des cas pour lesquels il n'est pas possible d'associer directement le *ne* à *personne*, ce qui impose la démarche en deux temps proposée auparavant (reconnaissance indépendante des deux parties, puis quand l'analyse a suffisamment progressée, vérification de la présence simultanée des deux parties).

Conclusion : Quel que soit le type de la grammaire que nous adopterons, nous avons montré que certains phénomènes simples ne mettant en jeu qu'un seul phénomène linguistique devront être reconnus en plusieurs étapes, et que l'ordre d'application de règles décrivant plusieurs phénomènes distincts ne peut pas être défini *a priori*. Les exemples présentés montrent que la séparation des différents phénomènes n'est pas simple. Nous devrons souvent, au cours de l'analyse, faire des choix locaux qui nécessiteront une vérification ultérieure, ce qui contredit le principe d'indépendance des différents phénomènes.

3.3.6 Conclusion

Nous avons présenté le cadre de la description du français à l'aide des transformations et du lexique-grammaire. Nous avons montré la différence entre cette *grammaire de description* et une *grammaire d'analyse*. Le processus de formation de phrases à partir de schémas de phrases élémentaires est bien décrit : il s'agit de la série de transformations et la réduction de la structure à sa surface. Cependant l'opération réciproque n'est pas faisable directement pour plusieurs raisons : d'une part, l'opération de réduction fait perdre les informations liées à la structure, et on ne sait plus alors comment appliquer les transformations inverses. D'autre part, certaines transformations comportent des effacements de séquences redondantes avec le contexte. Ces transformations ne sont pas non plus inversibles puisqu'il faudrait connaître le contexte, avant même d'avoir analysé la phrase courante. De ce fait, les grammaires d'analyse que nous construirons ne seront pas aussi naturelles que les grammaires de description. Comme elles sont basées sur un texte sans structure, elles devront parfois faire des approximations qui nécessiteront des vérifications ultérieures. L'indépendance des règles décrivant les phénomènes ne pourra pas toujours être conservée. De plus, il n'est pas possible de définir un ordre application général de ces transformations ce qui nécessite des algorithmes fonctionnant en plusieurs passes.
Nous avons d'autre part montré que l'utilisation du dictionnaire doit être modérée : les informations d'ordre grammatical associées à chaque entrée ne peuvent pas se substituer à une grammaire. Cela signifie que chaque entrée du dictionnaire n'est utilisable que s'il existe aussi une entrée dans la grammaire pour le mot considéré.

3.4 Les contraintes pratiques

Nous avons vu qu'à chaque entrée lexicale est associée une grammaire. Le nombre de grammaires que nous devrons manipuler est probablement d'un ordre de grandeur de quelques centaines de milliers. La construction de ces grammaires se heurtera à des contraintes pratiques d'organisation.
Tout d'abord, il faut gérer efficacement un aussi grand nombre de grammaires. Cela nécessitera des interfaces graphiques efficaces. Ensuite, pour garder un graphe lisible, il faudra que celui-ci tienne sur un écran ou une feuille de papier. Cette contrainte pratique n'est pas liée au matériel, il serait possible de gérer des graphes sur plusieurs écrans, mais la lisibilité diminuerait alors. Il faut donc imposer une réorganisation des gros graphes en faisant des appels à des sous-graphes sémantiquement intéressants (et pertinents?). Cette décomposition permet en plus de factoriser des séquences syntaxiquement homogènes pouvant être réutilisées dans d'autres graphes.
Nous verrons que cette décomposition est en outre extrêmement efficace du point de vue de la programmation, et que réciproquement, si on pouvait factoriser automatiquement des sous-motifs syntaxiquement homogènes dans un graphe quelconque, on pourrait en accélérer le traitement (voir page 217).

De même, il serait possible de réunir toutes les tables en une seule, en mettant en parallèle toutes les transformations contenues dans toutes les tables, et de rajouter les propriétés définitionnelles dans de nouvelles colonnes. Il n'y a pas de difficultés techniques à cela[25], mais pratiquement la table ne serait plus lisible. Nous allons retrouver cette contrainte pratique dans la construction des grammaires.
Aussi bien dans la description des verbes, que dans celle de titres de personnalité, il n'est pas possible d'établir de grandes classes d'unités linguistiques ayant exactement le même comportement syntaxique. La mise au point d'une grammaire consiste à regrouper les objets au maximum, en indiquant leurs différences à l'intérieur de la classe, de manière à obtenir des ensembles naturels, c'est-à-dire n'ayant pas trop d'entrées, et à l'intérieur de chaque ensemble pas trop de différences de comportement.

Enfin, il faudra que le passage de ces grammaires sur un texte dispose d'outils de suppression d'erreurs. On ne peut pas en effet imaginer qu'on puisse maîtriser directement une bibliothèque de si grande taille. En fait, nous verrons que l'application des grammaires est analogue à un processus de programmation pour lequel des outils de mise au point sont nécessaires.

25. La table résultant n'aurait que quelques dizaines de milliers d'entrées, et quelques centaines de colonnes.

Chapitre 4

Les outils informatiques

Dans ce chapitre, nous présentons les outils que nous avons développés en vue de manipuler des textes et des grammaires satisfaisant les besoins linguistiques décrits dans le chapitre précédent. Ces outils ne sont pas tous originaux: certains sont basés sur d'autres outils réalisés par M. SILBERTZEIN dans le cadre du logiciel INTEX©. À des fins de recherche, nous avons modifié l'aspect de certains et nous y avons ajouté des fonctionnalités. Cependant, la partie importante de ces outils n'est pas tellement l'interface, mais la puissance pratique rajoutée à l'outil de base qui est la reconnaissance par automate fini. Par exemple, nous montrons qu'il est indispensable de manipuler des graphes récursifs, de rajouter de nouvelles opérations au mécanisme de transduction et de pouvoir gérer diverses contraintes d'accord en genre et en nombre parallèlement à la reconnaissance d'une expression. Nous montrons aussi qu'il est pratique de factoriser des sous-séquences non initiales ni terminales dans un ensemble de séquences, ce qui n'est pas possible dans les automates finis. Ce problème a été partiellement résolu par l'introduction de contraintes. Nous donnons finalement un ensemble d'opérateurs indispensables qui permettent d'effectuer des opérations non triviales indispensables.
Notre présentation est basée sur des exemples. Dans ces exemples, il faudra bien distinguer l'outil permettant de coder les faits grammaticaux et la grammaire elle-même. Nous avons fait le maximum pour séparer ces deux aspects, et dans ce chapitre, nous mettons l'accent sur l'outil informatique et ses capacités. Il est courant de rechercher une grammaire totalement déclarative, approche qui s'oppose à la manière procédurale qui «code» des éléments grammaticaux. Dans notre utilisation, cette différence est pertinente, puisque la plupart des données grammaticales sont dans des graphes, des listes, des tables, mais elle n'est pas complète. En effet, c'est notre analyse des besoins linguistiques qui nous a conduit à créer ou à modifier les outils, or cette analyse est loin d'être universelle. Il existe donc une dépendance entre grammaire et outil.
Nous avons également suivi une certaine tradition grammaticale qui privilégie une linéarité dans l'analyse des phrases: la phrase initialement est une suite linéaire d'objets, et toutes les étapes du traitement conserveront cette linéarité[1]. Ce choix n'est pas universel: certaines analyses, en particulier les grammaires TAG [11] effectuent des calculs sur une forme arborescente).

1. Même quand nous considérons l'automate du texte: un chemin dans l'automate est une suite linéaire de mots.

Indépendamment de ces outils, des problèmes de complexité liés à la reconnaissance d'un très grand nombre de graphes sur un texte se posent. Nous traiterons ce problème dans un chapitre indépendant, où nous présenterons une approche du texte par index, cette approche conduisant à plusieurs algorithmes de reconnaissance à haute performance.

4.1 La représentation du texte

Formellement, le texte $\mathcal{T}$, est un mot de $(\mathcal{A}+\mathcal{P}+\mathcal{C})^*$, où $\mathcal{A}$ est l'alphabet, $\mathcal{P}$ est l'ensemble des signes de ponctuations, et $\mathcal{C}$ est l'ensemble des chiffres. En absence d'information supplémentaire sur la nature des unités linguistiques minimales, le texte est fractionné au niveau des espaces, des changements typographiques, et entre chaque caractère de ponctuation. Il peut donc être reformulé sous la forme $((\mathcal{A}^+ + \mathcal{P} + \mathcal{C}^+)_)^*(\mathcal{A}^+ + \mathcal{P} + \mathcal{C}^+)$, le caractère spécial «_» indiquant la séparation entre les différents objets, que nous appelons **formes**. Avec cette convention, nous pouvons aussi représenter le texte de manière graphique comme une suite de formes (cf. figure 4.1).
À partir de maintenant, on opposera la forme, qui est une séquence de lettres, de ponctuations, ou de chiffres au **mot** qui est une séquence «reconnue». La démarche consistant à appliquer des dictionnaires au texte linéaire met en parallèle des séquences de mots avec des séquences de formes. Cette représentation (voir figure 4.2) est appelée **automate du texte**[2]. Dans ce graphe, nous avons placé sur la ligne supérieure, les formes du texte, et en parallèle sur les autres lignes, nous avons fait figurer les différents mots. Sur cet exemple, 720 chemins mènent du début de la ligne à sa fin. L'opération de désambiguïsation décrite dans le chapitre 2 consiste à «élaguer» ce graphe en supprimant les transitions qui ne sont pas permises par le contexte. Notre démarche est différente mais se basera aussi sur cette notion d'automate du texte. Nous utiliserons également en première étape de l'analyse un dictionnaire, mais nous le passerons sur le vocabulaire du texte, et non pas sur le texte, et ce dictionnaire sera fortement réduit. Les mots ne seront construits que par des grammaires. Nous ne dirons que nous aurons une ambiguïté dans le texte, que quand il existera plusieurs chemins entre le début et la fin d'une phrase. Les chemins construits localement et non terminés ne seront donc pas considérés comme des ambiguïtés. De plus, les formes seront conservées jusqu'à la fin de l'analyse: nous verrons d'ailleurs qu'une grande partie des grammaires que nous décrirons travaillent sur les formes et non pas sur les mots. Conserver les formes permet de définir des grammaires qui fonctionnent sur des unités non prévues dans les dictionnaires (voir page 22) ou qui n'utilisent pas intentionnellement le dictionnaire.

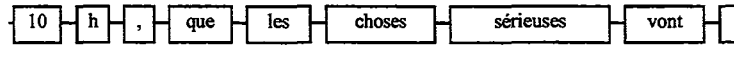


FIG. 4.1 – *Représentation linéaire des formes du texte*

À chaque mot sont associées plusieurs informations dans le format du dictionnaire. Ce sont - la **forme**, - le **lemme**, - la **catégorie grammaticale**, - des **informations syntaxiques et sémantiques**. Nous les identifions sur la figure 4.3, qui est un «zoom» de

2. Cette désignation est traditionnelle mais il faut noter que si le graphe concerné est direct, il est nécessairement acyclique.

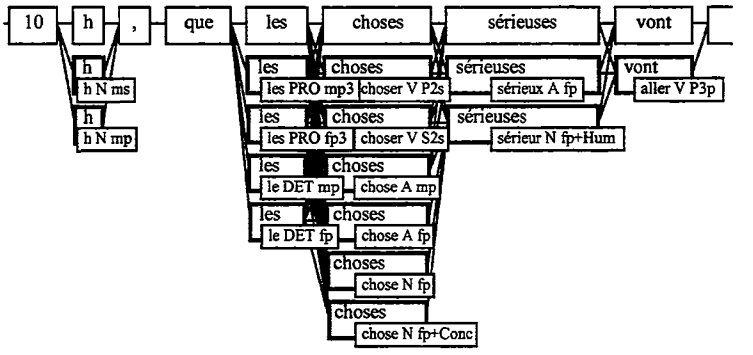


FIG. 4.2 – *Automate du texte, après passage d'un dictionnaire*

l'automate du texte précédent.

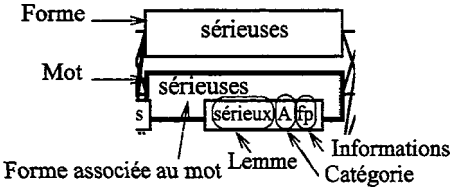


FIG. 4.3 – *Zoom sur l'automate de la figure 4.2. La boîte supérieure est la forme initiale du texte linéaire. La boîte inférieure est un mot dont nous voyons les différents champs : Forme=***sérieuses***, Lemme=***sérieux***, Catégorie=***A***, et Informations=***fp***.*

La notion de **lemme** est une notion traditionnelle dépendante de la catégorie et des choix faits lors de l'établissement de la grammaire. Par exemple, pour les groupes nominaux que nous aurons localisés, le lemme sera le nom-tête associé au groupe nominal (ceci est un choix fait au niveau de la construction de la grammaire). Ainsi les séquences *un groupe d'étudiants*, *ces étudiants*, et *l'étudiant dont je me souviens* auront tous le lemme commun ***étudiant***.
La **catégorie** est toujours représentée en majuscule : elle permet de regrouper sous une même étiquette plusieurs unités ayant le même comportement grammatical. Ainsi, pour les groupes nominaux, nous utiliserons la catégorie *GN*.
Les informations supplémentaires seront les mêmes que dans le dictionnaire : il peut s'agir d'informations flexionnelles comme `m` et `f` qui indiquent les deux genres `s` et `p` indiquent les deux nombres, ou d'informations sémantiques diverses : `Hum`, `Conc`, etc.
Ces différents champs sont autant de variables, qu'il est possible d'utiliser à sa guise. Nous chercherons cependant à toujours conserver pour ces champs une même signification.
La manière de représenter un mot est la suivante[3] :

3. Par rapport à la forme des entrées du dictionnaire DELAS, les informations morphologiques et sémantiques sont mises sur le même plan.

$$\{\boxed{\text{forme}},\boxed{\text{lemme}},\boxed{\text{Catégorie}}:\boxed{\texttt{Infos}}\}$$

Le champ `Infos` est structuré de la manière suivante : $I_1 I_2 ... I_k (+info_i)^*$
Les I_j sont représenté par un seul caractère, les $info_j$ peuvent en comporter plusieurs.
Par exemple, {*cet étudiant*, *étudiant*.*GN* :ms+Hum+Ddéf} est le mot dont la forme est cet étudiant, le lemme (le nom-tête pour un groupe nominal) est *étudiant*, la catégorie est *GN*, et les informations supplémentaires sont :

- `m` genre masculin, `s` nombre singulier ;
- `Hum` marque sémantique humain ;
- `Ddéf` déterminant défini (cette information est importante lorsqu'on combine plusieurs déterminants par exemple).

Par rapport au texte initial, nous avons ajouté des informations à chaque mot. Chacun des champs d'un mot est une variable dans lequel nous stockerons des informations diverses (mais dont la signification restera sensiblement la même : le lemme correspondra toujours à une forme canonique pour le mot). Le champ *Informations* permet d'ajouter toutes les étiquettes sémantiques ou syntaxiques voulues. Son comportement est celui d'un ensemble : ses éléments ne sont pas ordonnés et ne peuvent pas se répéter.
Les mots sont créés par les règles de transformation, comme nous le verrons plus loin. Il est vraisemblable que nous aurons besoin d'enrichir plus encore les informations rattachées à chaque unité. Prenons un exemple : aujourd'hui quand nous avons reconnu qu'un verbe porte une négation[4], nous lui ajoutons la marque `+Nég`. Schématiquement :

Luc ne lit pas de livres $\longrightarrow$ *Luc* {*ne lit pas*, *lire*.*V* :P3s+Nég} *de livres*

Et on a la règle : *V*+Nég *de* *N*+p $\longrightarrow$ *V des N* qui rétablit le bon déterminant au complément principal du verbe.
Avec cette règle la phrase précédente se transforme en :
Luc {*ne lit pas*, *lire*.*V* :P3s+Nég} *de livres*$\longrightarrow$

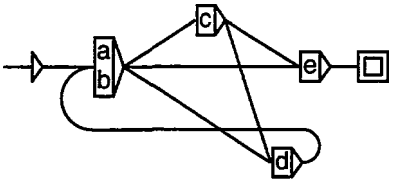
Luc {*ne lit pas*, *lire*.*V* :P3s+Nég} *des livres*

En fait, pour l'analyse sémantique ultérieur, il ne sera pas suffisant de savoir qu'il y a une négation qui porte sur le verbe, mais il faudra aussi savoir quelle est cette négation. Il faudra alors lier au trait[5] +Nég la négation elle-même comme sur l'arbre de la figure 3.2, page 35.
Nous avons contourné ce problème de la même façon que E. ROCHE l'a fait dans [19], en rajoutant directement au texte les informations supplémentaires à la suite du mot. Par exemple, nous avons placé à la suite de tout verbe portant une négation, la négation elle-même :

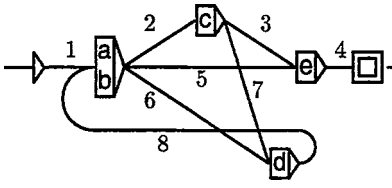
Max {*ne lit pas*, *lire*.*V* :P3s+Nég} {*ne pas*, .*NEG*} *de livre*.

4. Nous avons besoin de cette information, car quelle que soit la négation, la phrase contenant ce verbe a un comportement propre, par exemple vis-à-vis de la conjonction de coordination *ni*. Ou encore, le déterminant de l'objet direct d'un verbe portant une négation peut être modifié.

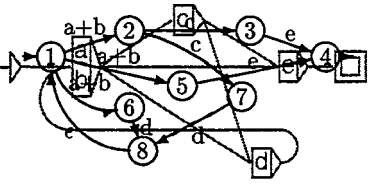
5. Le terme **trait** est traditionnellement utilisé pour désigner des informations supplémentaires de ce type (trait sémantique, morphologique ou syntaxique) [16].



(a) Représentation duale



(b) Numérotation des transitions



(c) Représentation canonique

FIG. 4.4 – *Passage de la représentation duale à la représentation canonique*

4.2 Les graphes

Toutes les grammaires que nous allons développer ont comme point de départ les automates finis. Ce choix implémenté dans INTEX©par M. SILBERZTEIN tient dans l'existence d'un outil dont l'interface est extrêmement pratique d'utilisation et permet de créer, modifier, imprimer des graphes, faire des alignements entre les différents composants, naviguer entre les sous-graphes, etc. Ce choix ne reflète cependant pas une volonté de restreindre la description aux langages rationnels. Cette base est enrichie de divers mécanismes permettant d'avoir une puissance suffisante, tout en gardant une représentation claire.

4.2.1 Les automates, représentation duale

Les graphes utilisés sont des représentations duales de la représentation canonique des automates finis. : les transitions sont représentés dans des boîtes, et les nœuds ne sont pas représentés; entre chaque boîte existent des arcs orientés par la flèche à droite de chaque boîte. Cette représentation est complètement équivalente à la représentation canonique des automates finis: pour passer de la première (ex. figure 4.4(a)) à la seconde (figure 4.4(c)), il suffit, par exemple[6], de considérer un nœud au milieu de chaque arc de la représentation duale (figure 4.4(b)), et de restaurer les transitions entre ces différents nœuds. La transformation réciproque est aussi simple. L'avantage de cette représentation

6. En fait, pour conserver les propriétés de factorisation du graphe, il faut procéder autrement; à chaque boîte on associe un nœud d'entrée, et un nœud de sortie, et on remplace les arcs entre les différentes boîtes par des transitions ϵ. Il est intéressant de noter que la factorisation visuelle des deux types de graphes ne correspond pas au même schéma.

pour notre utilisation est de mettre en valeur et d'ordonner (physiquement) les transitions. En même temps, les nœuds qui n'ont aucune signification linguistique n'apparaissent plus.
Plusieurs lignes apparaissant dans une même boîte signifient plusieurs transitions parallèles.

4.2.2 L'utilisation du dictionnaire

Le contenu des transitions des graphes que nous développons peut être soit des formes (lettres de $\mathcal{A}$), soit des références à des mots de la manière suivante :

- <*CAT*> correspond à tous les mots étiquetés CAT ;
- <*lemme*> correspond à toutes les mots du texte portant le lemme <*lemme*> ;
- <*lemme.CAT*> correspond à l'intersection des deux ensembles précédents ;
- <*X.ms*:+Hum> correspond à l'intersection de <*X*> et des mots dont l'information sémantique est +Hum, et dont les informations flexionnelles sont ms.

Par convention, et pour qu'il n'y ait pas d'ambiguïté entre les catégories et les lemmes, les catégories seront toutes en majuscules.
Il est possible aussi de faire référence à toutes les entrées qui ne correspondent pas à une définition, en utilisant un symbole de négation logique : «!». Ainsi !<*V*> correspond à tous les mots qui ne sont pas étiquetés V. Nous n'utiliserons pas cette possibilité seule (qui est aussi puissante que le symbole <*MOT*> correspondant à n'importe quelle forme), mais en combinaison avec la conjonction représentée «&» qui permet d'affiner le contenu d'une transition, par exemple :

<*manger.V*> & !<*V*:W> & !<*V*:K>

correspond à toutes les formes du verbe *manger* non infinitive (W) et non participé passé (K).

Quand nous appliquerons un dictionnaire au texte, nous ne développerons pas tous les mots possibles figurant dans les articles du dictionnaire, mais pour chaque entrée, nous ajouterons aux formes correspondantes, les définitions du dictionnaire. Le dictionnaire en lui-même n'ajoutera donc aucun nœud à l'automate du texte. Pour reprendre l'exemple de la figure 2.1 (page 17), nous avons vu que la construction des interprétations induites par un dictionnaire[7], conduisait à un automate de texte comme celui de la figure 2.2 (page 18). Si nous appliquons le même dictionnaire sans introduire les interprétations, le texte garde la même forme (et nous le représentons par l'automate de la figure 4.5.). Si nous recherchons maintenant les groupes nominaux définis décrits par le graphe de la figure 4.13, un nouveau chemin est construit qui tient compte des données du dictionnaire : figure 4.6.

4.2.3 Les sous-graphes, ou les graphes récursifs

Chaque transition peut aussi contenir une référence à un autre graphe désigné par son nom. Ces transitions particulières sont représentées en grisés comme dans le graphe de

7. Limité aux formes verbales, nominales, et adjectivales.

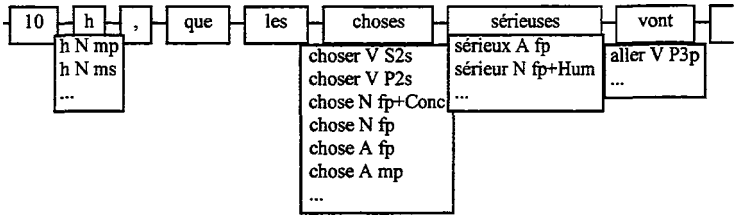


FIG. 4.5 – *Texte linéaire, avec informations provenant du dictionnaire.*

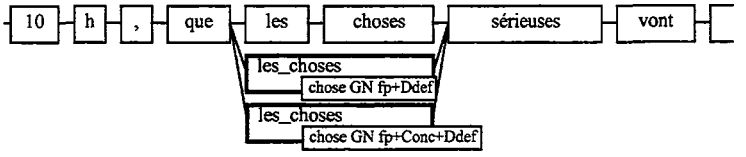


FIG. 4.6 – *Application du graphe de la figure 4.13.*

la figure 4.7. Ce procédé permet de construire des graphes reconnaissant des grammaires context-free. Il ne s'agit donc plus d'automates finis. En fait, tant que les sous-graphes ne provoqueront pas d'appel récursifs (en remplaçant physiquement tout le sous-graphe dans le graphe initial), ou s'ils ne provoquent que des appels récursifs mais terminaux (dans ce cas on dispose d'un arc de l'état final vers un état quelconque) ou initiaux[8], il y aura possibilité de revenir à un automate fini.
Les appels de sous-graphes constituent un moyen élégant de factoriser des sous-motifs dans de gros graphes (ce qui est une de nos contraintes pratiques). Il ne s'agit pas que d'offrir une facilité d'écriture, mais aussi de décrire des séquences complexes du type :

$$(M.X)_1,\ \underline{(\textit{le fils de M.Y, le directeur})_{NP}},\ (\textit{et président de ...})_{NP}$$

Pour ces séquences, la récursivité n'est ni terminale, ni initiale, et le schéma est intrinsèquement infini (même si on ne trouvera jamais plus de 3 ou 4 niveaux d'imbrication), et donc la puissance d'une grammaire context-free est nécessaire.

Factorisation

Les automates finis offrent un moyen de factoriser naturellement dans des séquences différentes des sous-séquences initiales ou finales. Il n'y a pas de raison théorique qui limiterait la factorisation de séquences en langage naturel à ces deux types de factorisation. On voudrait, par exemple, représenter de manière concise les différentes manières de représenter un groupe nominal décrivant un nom propre associé au titre de président en anglais. Nous voulons donc reconnaître les séquences suivantes :

(1) *the French President Jacques Chirac*
(2) *the President Jacques Chirac of France*
(3) *Jacques Chirac, the French President*
(4) *Jacques Chirac, the President of France*

8. Ce qui correspond aux langages linéaire droit et linéaire gauche.

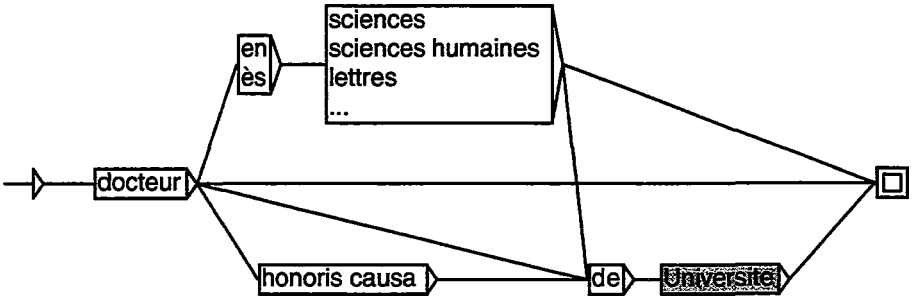


FIG. 4.7 – *Description du titre de docteur. Ce graphe fait appel à un sous-graphe : le graphe* Université *qui décrit la désignation des universités.*

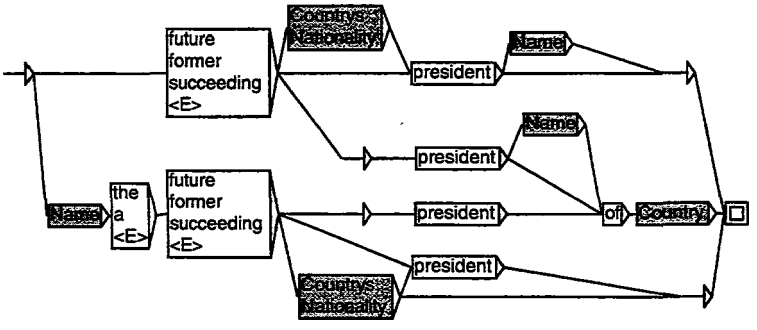


FIG. 4.8 – *Graphe décrivant la grammaire de president. Il n'est pas possible de le factoriser plus, à cause des dépendances début/fin.*

Ces séquences sont reconnues par le graphe de la figure 4.8. Nous voyons qu'il n'est pas possible de factoriser ce graphe plus avant, car les séquences (5), (6) sont incorrectes : nous devons donc avoir un chemin associé à chacune des quatre constructions précédentes.

(5)* *the French President Jacques Chirac of France*

(6)* *Jacques Chirac, the President Jacques Chirac of France*

À chaque enrichissement de la grammaire de *president* (par exemple, ajout de l'option *vice-president*), nous devrons dupliquer l'information dans chacun de ces quatre chemins. Ceci n'est pas satisfaisant du point de vue de la maintenance.

La représentation par automate fini n'offre pas de moyen simple de résoudre ce problème. Nous n'avons pas introduit de fonctionnalités permettant d'effectuer ces factorisations. Néanmoins, nous montrons (page 72) que nous pouvons contourner ce problème[9] par l'ajout de contraintes.

9. qui se généralise aux problèmes d'accord.

4.2.4 Transformations

Ce que nous appelons ici transformation est une représentation informatique des modifications de séquences associées aux «transformations linguistiques». Mais il ne s'agit pas de la même notion (en cas d'ambiguïté possible entre ces deux types de transformation, nous appellerons celle que nous décrivons ici: **règle de transformation**). Les opérations que nous voulons effectuer sont des permutations de séquences, les formations des mots, et diverses manipulations de leurs différents champs. Pour ne pas introduire de terminologie supplémentaire, nous continuerons à appeler les «graphes» effectuant ces transformations: des transducteurs (ou **transducteur étendu** en cas d'ambiguïté, par opposition à transducteur simple).
Il est possible par application continue de transducteur jusqu'à un point fixe, d'opérer des permutations sur les mots, mais cela n'est intéressant ni du point de vue de la lisibilité, ni du point de vue de l'efficacité, puisque le nombre de «passes» du transducteur dépend de la distance entre les deux mots à permuter. Nous avons donc délibérément choisi d'ajouter des fonctionnalités de déplacement de mot à nos graphes. Cette fonctionnalité ressemble un peu à celle du programme **sed**, qui permet de reconnaître des expressions rationnelles et de les remplacer par des expressions dans lesquelles on peut reprendre des sous-parties de l'expression reconnue.
MAX SILBERTZEIN a incorporé cette fonctionnalité à INTEX©comme indiqué sur la figure 4.9.

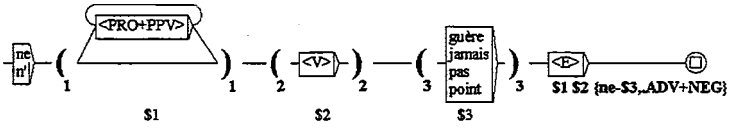
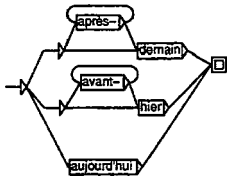


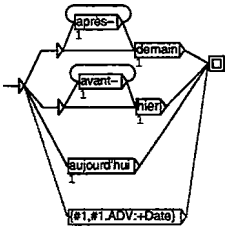
FIG. 4.9 – *Représentation graphique d'une transformation sur des sous-séquences d'une expression reconnue. La syntaxe est analogue à celle de* **sed**.

Le «mécanisme» que nous avons adopté diffère légèrement de ce standard et permet de généraliser la notion de transduction en donnant une forme fonctionnelle à la «sortie» des graphes: **la sortie d'un transducteur étendu, pour une certaine séquence reconnue, sera une fonction des différentes parties de cette séquence.** Elle permet, de plus, de proposer plusieurs sorties pour un même chemin, et d'associer des contraintes à ces sorties. Ces transducteurs s'appliquent directement à l'automate du texte, et ont pour effet de rajouter des états et des chemins dans celui-ci.
Pour ne pas surcharger l'exposé de détails techniques, nous présentons ici seulement leur fonctionnement général sur quelques exemples, et nous donnons en annexe A, la définition formelle de ces transducteurs étendus.

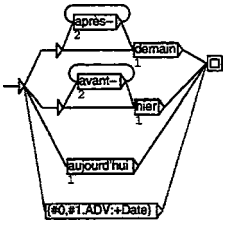
1. Un transducteur étendu est représenté comme un transducteur fini (représentation duale), dont le vocabulaire d'entrée est l'ensemble des formes, des mots, et des opérations spéciales sur les mots que nous avons présentées auparavant. Le vocabulaire de sortie est $I\!N$. À ce transducteur, on ajoute une ou plusieurs «transitions spéciales» (représentées en grisé) et indiquant les chemins à ajouter à l'automate du texte.



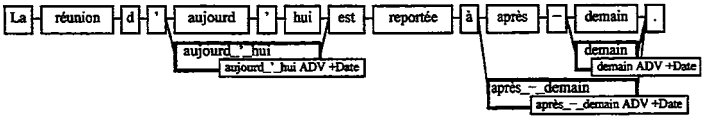
(a) Graphe reconnaissant certaines dates



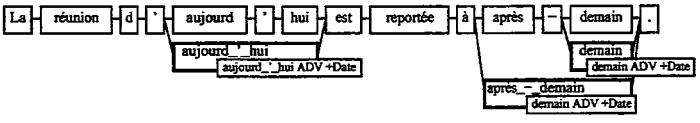
(b) Première transformation sur les séquences reconnues



(c) Transformation plus évoluée



(d) Automate du texte après transformation par 4.10(b)



(e) Automate du texte après transformation par 4.10(c)

FIG. 4.10 – *Reconnaissance des dates et transformations.*

Le graphe de la figure 4.10(a) reconnaît les séquences *aujourd'hui*, *demain*, *hier*, *avant-hier*, *avant-avant-hier*...

Dans le graphe de la figure 4.10(d), nous avons ajouté à chacune des transitions du graphe précédent la même sortie : le nombre 1. Ce graphe est donc maintenant considéré comme un transducteur. En grisé, entre l'état initial et l'état final, nous avons représenté une de ces «transitions spéciales» : $\boxed{\{\#1, \#1.ADV{:}\texttt{+Date}\}}$. Pour une séquence donnée, reconnue par un chemin du graphe, le symbole #1 est une variable dont la valeur est la sous-séquence associée à la sortie 1 du transducteur. Ici, il s'agit donc de toute la séquence reconnue.

Supposons par exemple que nous considérons la séquence *avant-hier*, cette séquence est reconnue par le transducteur et la sortie associée pour toute la séquence est 1_1. La variable #1 vaut donc *avant-hier*.

L'application de ce graphe à un texte ajoutera, dans l'automate du texte, le mot formé par $\boxed{\{\#1, \#1.ADV{:}\texttt{+Date}\}}$ en parallèle avec les séquences reconnues (figure 4.10(d)). À chaque séquence reconnue par le graphe, un nouveau nœud dans l'automate du texte a été créé, étiquetant cette séquence adverbe (ADV) et ajoutant comme information que c'est une date (+Date).

Supposons maintenant que nous voulions affiner la sortie : on veut maintenant associer à chaque adverbe un des trois lemmes *hier*, *aujourd'hui*, et *demain*. Pour cela, on utilise le graphe de la figure 4.10(c). Les séquences reconnues sont scindées en deux variables #1 et #2. #1 correspond à *aujourd'hui*, *hier* et *demain* alors que #2 correspond aux modifieurs *avant** ou *après**.

La transformation est $\{\#0, \#1.ADV{:}\texttt{+Date}\}$. #0 est une autre variable, mais dont la valeur est, par convention, toujours la totalité de la séquence reconnue. L'application de ce graphe au même texte donne l'automate du texte de la figure 4.10(e).

La recherche du motif <*demain*> dans ce texte donne les deux occurrences :

$\{demain, demain.A{:}\texttt{+Date}\}$ et $\{après\text{-}demain, demain.A{:}\texttt{+Date}\}$

2. La transformation associée au graphe de la figure 4.11 est plus complexe : $\boxed{\#2{+}\texttt{Nég}\ \{\#1\ \#3, \#3.NEG\}}$. Elle est composée de deux parties : $\boxed{\#2{+}\texttt{Nég}}$ et $\boxed{\{\#1\ \#3, \#3.NEG\}}$. La première partie reprend le mot de la sous-séquence numérotée 2 dans la séquence reconnue, et lui ajoute l'information +Nég. La sous-séquence numérotée est le verbe de l'expression, donc on ajoute +Nég au verbe portant une négation. La deuxième partie est analogue aux transformations précédentes. L'application de cette transformation à un texte conduit à l'automate du texte de la figure 4.12. On remarque que le graphe fait appel au dictionnaire puisqu'il recherche <*V*> : la forme *aime* possède trois entrées dans le dictionnaire correspondant à ce motif.

3. Un autre exemple est la reconnaissance et le marquage des déterminants définis. Les déterminants ne seront reconnus qu'en présence du nom associé[10]. Cela signifie que toute séquence reconnue comme étant un déterminant est suivie d'une

10. Cela permet d'éviter de parler de déterminant en absence de nom. Il faudra cependant parfois refaire apparaître ce nom comme dans *J'en veut une* qui sera analysé comme *Je veux une* *N*+REF. L'information +REF indiquant la référence à un autre nom.

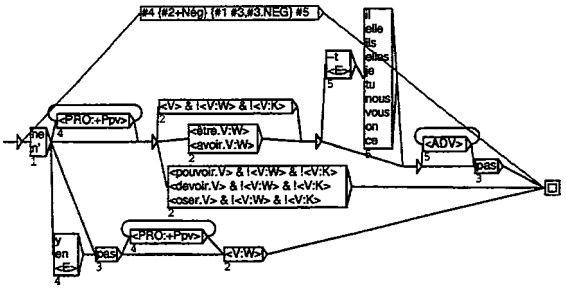


FIG. 4.11 – *Regroupement de la négation : on distingue le cas où la forme verbale est à l'infinitif, et le cas où la forme verbale est ni à l'infinitif, ni au participe passé. L'ordre des constituants de la négation n'est pas le même dans les deux cas.*

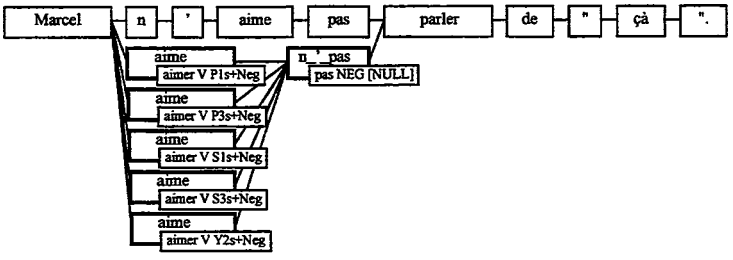


FIG. 4.12 – *Reconnaissance et regroupement des parties de la négation par le graphe 4.11.*

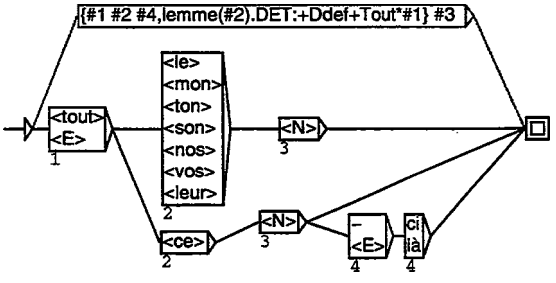


FIG. 4.13 – *Reconnaissance des déterminants définis.*

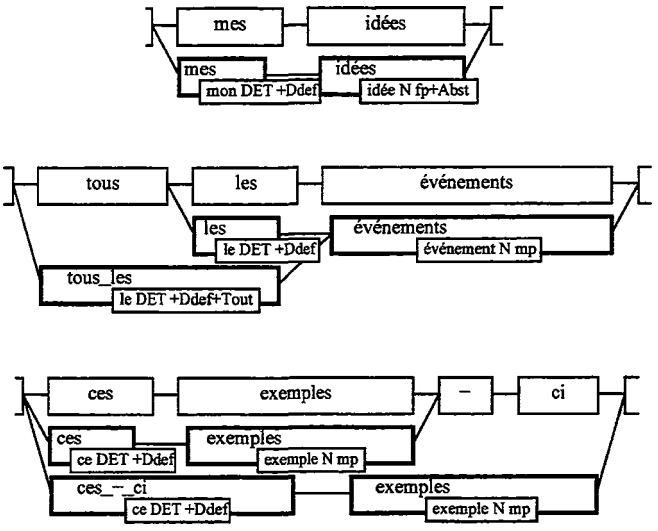


FIG. 4.14 – *Déterminants définis reconnus par le graphe 4.13.*

séquence représentant le reste du groupe nominal. D'une manière simplifiée[11] pour la démonstration, le graphe de la figure 4.13 reconnaît tous les déterminants définis (éventuellement modifiés par *tout*, et par *ci* ou *là* pour les adjectifs démonstratifs). Ce graphe reconnaît donc les trois séquences suivantes ainsi que les sous séquences entre parenthèses:

tous les événements (les événements)
cet exemple-ci (cet exemple)
mes idées

À chacune de ces cinq séquences, il correspond un chemin unique dans le graphe et les variables associées à chacun de ces chemins sont:

$\#1 = \epsilon$	$\#2 =$ *les*	$\#3$, *événements*	$\#4 = \epsilon$
$\#1 =$ *tous*	$\#2 =$ *les*	$\#3$, *événements*	$\#4 = \epsilon$
$\#1 = \epsilon$	$\#2$, *cet*	$\#3 =$ *exemple*	$\#4 =$ *-ci*
$\#1 = \epsilon$	$\#2$, *cet*	$\#3 =$ *exemple*	$\#4 = \epsilon$
$\#1 = \epsilon$	$\#2 =$ *mes*	$\#3 =$ *idées*	$\#4 = \epsilon$

La transformation associée est :

$$\{\#1\ \#2\ \#4, lemme(\#2).DET\texttt{:+Ddef+Tout*\#1}\}\ \#3$$

Dans le premier mot de la transformation, le champ correspondant au lemme est *lemme*(#2). *lemme* est un opérateur sur les mots (nous décrivons les différents types d'opérateurs et le moyen de les définir plus loin), donnant le lemme d'un mot. Dans les informations de ce mot, il y a aussi la valeur `Tout*#1`. Cette valeur est nulle si #1 vaut ϵ, sinon elle vaut `Tout`. Cette fonctionnalité permet de rajouter des informations syntaxiques ou sémantiques selon le chemin suivi.

11. Nous ne considérons pas ici l'accord en genre et en nombre.

Cette transformation nous produit donc les cinq sorties associées aux cinq chemins :

{*les*, *le*.DET:mp+Ddef}	{*événements*, *événement*.N:mp}
{*tous_les*, *le*.DET:mp+Ddef+Tout}	{*événements*, *événement*.N:mp}
{*cet_-ci*, *ce*.DET:+Ddef}	{*exemple*, *exemple*.N:ms}
{*cet*, *ce*.DET:+Ddef}	{*exemple*, *exemple*.N:ms}
{*mes*, *mon*.DET:fp+Ddef}	{*idées*, *idée*.N:fp}

C'est ce que nous retrouvons dans les automates du texte correspondant à ces séquences (figure 4.14).

4. Un dernier exemple permet de mettre en évidence la possibilité d'associer des informations différentes suivant le chemin de la séquence reconnue. Le graphe de la figure 4.15 reconnaît deux contextes du mot *président*, un contexte *Politique* (*Président de la République Française*), et un contexte *Économique* (*président de Elf-Aquitaine*). Nous avons associé deux transformations différentes, regroupant tous les mots des deux expressions en un seul nom[12]. Pour que seule la transformation pertinente soit associée à la séquence reconnue, les transformations ne sont appliquées que si elles sont connectées à des nœuds appartenant au chemin reconnu.
   Ainsi à *président français* est associé le mot
   {*président français*, *président*.N:+Politique}
   à *président directeur général* est associé le mot
   {*président directeur général*, *président*.N:+Eco}
   et à la forme *président* seule, sont associées les deux mots
   {*président*, *président*.N:+Politique} et {*président*, *président*.N:+Eco}

Nous avons ébauché, avec ces quelques exemples, la méthode générale qui permet de reconnaître et d'identifier des séquences d'un texte. Les graphes de transformation reconnaissent des motifs auxquels s'appliquent naturellement plusieurs types d'opérations qui créent de nouveaux nœuds et chemins dans l'automate du texte. Dans les chapitres suivants, cette méthode sera intensivement utilisée. Nous donnons le détail du formalisme en annexe, et nous présenterons dans la partie 8, les algorithmes qui gèrent de tels objets.

12. *Président de la république* est un nom composé, donc l'application du dictionnaire des noms composés permettra de faire l'opération de regroupement. Mais avec l'adjectif de nationalité, l'expression n'est plus complètement figée et donc la description ne peut plus se faire avec un dictionnaire.

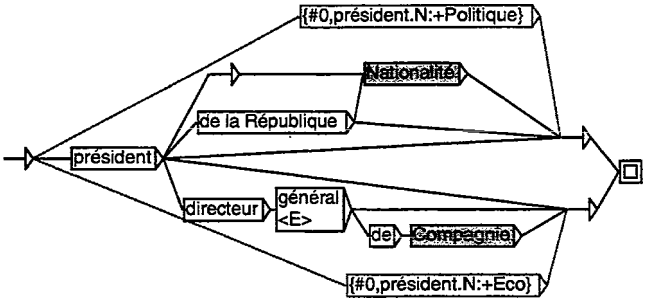


FIG. 4.15 – *Reconnaissance du mot président dans deux contextes différents.*

4.2.5 Gestion de contraintes

Dans certaines situations, des graphes généraux reconnaissent certains motifs mais laissent de côté certaines vérifications de contraintes. Par exemple la vérification de l'accord n'est pas effectuée par le graphe de la figure 4.13. Pour tenir compte de cette contrainte, nous pouvons envisager de dupliquer le graphe en autant d'exemplaires qu'il y a de combinaison de genre/nombre possibles (voir figure 4.16). Ce genre d'opération est relativement facile pour le français, puisque les différents cas d'accord sont relativement peu nombreux ; mais cette solution pose des problèmes de lisibilité et de maintenance. D'un point de vue formel, la solution n'est pas très satisfaisante, on aimerait bien pouvoir indiquer la forme, d'une part, et les contraintes d'autre part.
La solution que nous avons adoptée consiste à attacher à chaque transformation une contrainte exprimée comme une formule logique (cf. 4.17). Si pour un chemin du graphe et pour une transformation associée à ce chemin, la formule logique associée à la transformation est vraie, alors la transformation est appliquée, sinon elle ne l'est pas. Ces formules logiques utilisent les mêmes opérateurs que les transformations que nous définissons plus loin.
Graphiquement, cette «formule logique» apparaît «sous» la transformation à laquelle elle s'applique.
Cette fonctionnalité permet aussi de faire des vérifications sur le chemin suivi. Nous avons vu page 63, que la possibilité de factoriser des parties de chemins est limitée avec la représentation par automate finie. Reconsidérons l'automate non factorisable de la figure 4.8. L'automate de la figure 4.18 reconnaît toutes les séquences décrites par l'automate précédent, plus certaines séquences non correctes comme : *the French President of Belgium*. La transformation associée à ce graphe est #0, c'est-à-dire le chemin d'entrée. Nous avons introduit comme contrainte que le terme de nationalité ne peut pas figurer simultanément à gauche et à droite : `non(#1 et #3)`, et il en va de même pour le nom propre : `non(#2 et #4)`. Les séquences réellement reconnues par cet automate très simple sont donc exactement les séquences reconnues par l'automate que nous avions signalé comme non factorisable.

4.2.6 Les graphes patrons

Pour convertir les données contenues dans les tables du lexique-grammaire en données de nos programmes, nous avons adopté un système associant à chaque table, un graphe dit **graphe patron**, contenant toutes les structures permises par la table. Chaque entrée de la table est ensuite automatiquement spécifiée en un graphe propre dérivant de ce graphe patron.
Le graphe patron contient alors trois types de symboles spéciaux : les symboles `%`i et les symboles `@`i et ~`@`i. Les symboles `%1`, ..., `%`k sont des variables représentant les différentes parties (1 à k) de l'entrée de la table. Les symboles `@1`, ..., n et ~`@1`, ..., ~`@`n représentent les n colonnes de la table.
Quand une entrée est calculée à partir du graphe patron, les variables `%`i sont remplacées par les valeurs correspondantes à l'entrée. Les symboles `@`i, sont eux, soit remplacés par ϵ, soit par une séquence ZZZ_ZZZ [13] selon que la colonne associée contient le symbole +

13. Cette séquence est inexistante dans le texte, cela bloque donc l'analyse des chemins contenant la séquence.

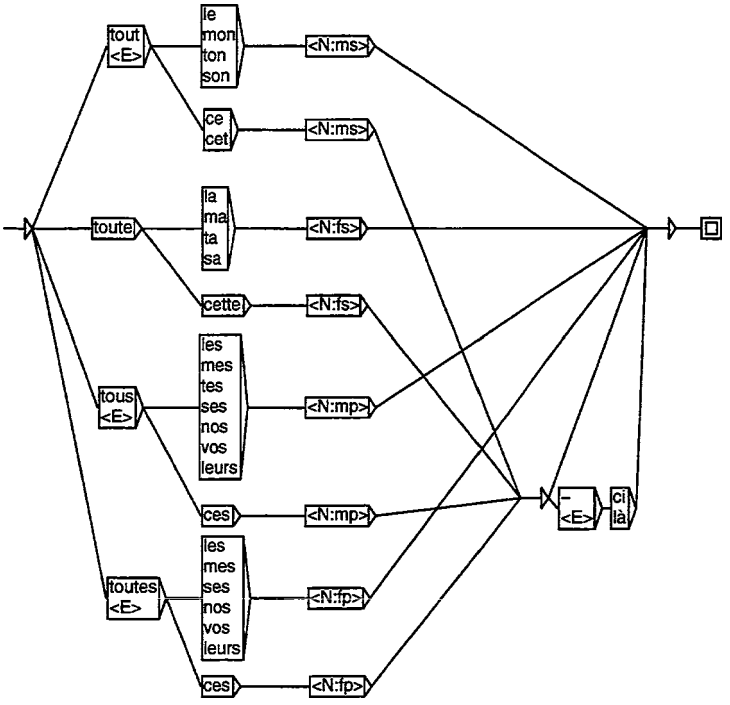


FIG. 4.16 – *Graphe reconnaissant les déterminants définis, avec la vérification des accords*

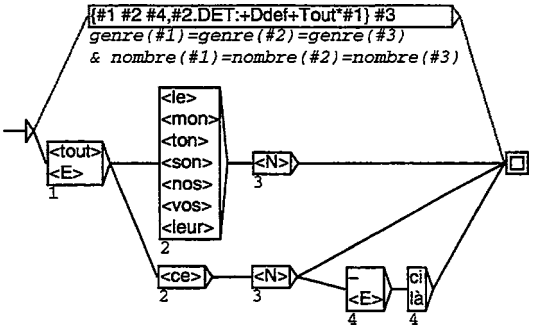


FIG. 4.17 – *Graphe reconnaissant les déterminants définis, une vérification du genre et du nombre est faite avant d'opérer la transformation*

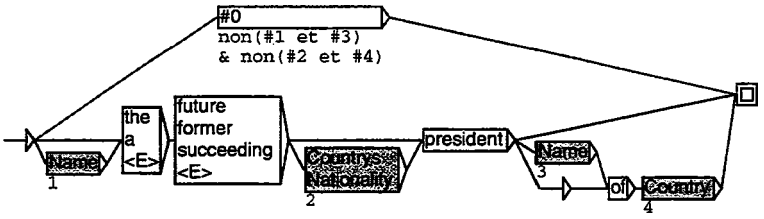


FIG. 4.18 – *Transducteur étendu, appliquant la transformation #0 (c'est-à-dire la transformation identité) aux seules séquences reconnues par l'automate 4.8.*

ou le symbole −. L'effet d'une telle substitution dans le graphe est d'autoriser tous les chemins contenant des symboles @*i* dont les colonnes associées contiennent le symbole + et d'interdire tous les chemins dont les colonnes associées contiennent le symbole − (en effet, introduire dans le chemin un caractère inexistant dans le texte, revient à couper ce chemin). Enfin, les symboles ~@*i* ont le comportement inverse de celui des symboles @*i*, ce qui permet d'autoriser des chemins quand une certaine colonne contient − et de les interdire si cette colonne contient +.

Pour illustrer cette procédure considérons l'extrait de la table 4.1. Cette table représente quelques structures verbales figées de la table `C1IPN` du lexique-grammaire. Nous détaillons cette table et son traitement en 7.2, page 159.
Pour cette table, chaque entrée possède 5 champs : %1 correspondant à `PPV` (le pronom préverbal), %2 correspondant à `V` (le verbe), %3 correspondant à `DET` (un déterminant exemple), %4 correspondant à $\mathtt{N}_1$ (le nom-tête du premier complément) et %5 correspondant à `Prép` (la préposition du deuxième complément).
Pour la première entrée, nous avons donc :

$$\%1 = \epsilon \quad \%2 = \text{accorder} \quad \%3 = \text{une} \quad \%4 = \text{valse} \quad \%5 = \text{à}$$

Les séquences correspondant aux structures verbales canoniques de ces entrées sont donc reconnues par le graphe patron de la figure 4.19.
Certaines de ces entrées admettent un passif, ce qui est indiqué dans la dernière colonne de la table. Le graphe patron reconnaissant la structure passive est celui de la figure 4.20.
Nous pouvons maintenant regrouper ces deux graphes. On autorise le chemin passif s'il est autorisé par la table et, ceci, en insérant le symbole @7 dans le chemin représentant le passif (graphe patron de la figure 4.21).
Les graphes patrons permettent donc de représenter sous une forme d'automate, des structures complexes décrites par les tables du lexique grammaire. Nous donnerons des exemples complets d'utilisation au chapitre 5.

4.2.7 Les opérateurs spéciaux

Extraction des différents champs d'un mot

La figure 4.13 donne un exemple de transformations dont la sortie est le résultat d'un calcul non trivial sur les séquences reconnues. La description des contraintes d'accord fait aussi appel à des fonctions donnant le genre et le nombre de mots. Dans ce cas,

PPV	V	DET	N_1	Prép	N_0 V N_1	N_2 =: $Nhum$	N_2 =: $N - hum$	$Prep\ Nhum$ = ppv	$Prep\ N - hum$ = ppv	$sur\ N_2$ = $dessus$	$Passif$
ϵ	accorder	une	valse	à	−	+	−	+	−	−	+
ϵ	apporter	un	secours	à	−	+	−	+	−	−	+
ϵ	accumuler	des	preuve	contre	+	+	−	−	−	−	+
ϵ	acheminer	des	secours	:Loc	−	+	+	−	+	−	+
ϵ	acheminer	une	aide alimentaire	:Loc	−	+	+	−	+	−	+
ϵ	acheminer	une	aide	:Loc	−	+	+	−	+	−	+
ϵ	adresser	un	sourire	à	−	+	−	+	−	−	+
ϵ	adresser	un	ultimatum	à	+	+	+	+	−	−	+
ϵ	adresser	une	critique	à	−	+	−	+	−	−	+
SE	faire	de la	bile	pour	+	+	+	−	−	−	−
SE	faire	de la	mousse	pour	+	+	+	−	−	−	−

TAB. 4.1 – *Extrait de la table* C1IPN

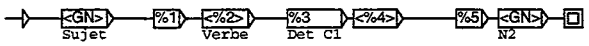


FIG. 4.19 – *Graphe patron décrivant la structure canonique des entrées de la table 4.1.*

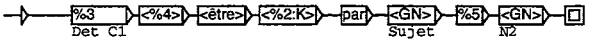


FIG. 4.20 – *Graphe patron décrivant la structure passive associée aux entrées de la table 4.1.*

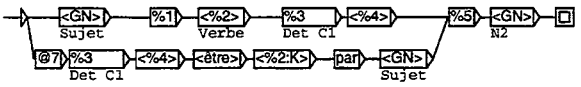


FIG. 4.21 – *Graphe patron décrivant la structure active, et si elle est permise par la table : la structure passive.*

l'utilisation des fonctions permet d'alléger considérablement la grammaire, mais celles-ci ne sont pas indispensables. Par contre, dans la description des déterminants définis, il n'y a pas d'autre moyen pour arriver au résultat présenté.
Ces informations sont dépendantes de la grammaire. En français, les notions de genre et de nombre sont pertinentes; dans d'autres langues, elles ne le sont pas, mais la notion de «cas» peut l'être.
C'est pour cette raison que nous avons défini ces fonctions d'une manière déclarative: le transducteur classique[14] de la figure 4.22 décrit la structure des mots et associe à chaque champ un nom. Ce nom définit la fonction qui extrait le champ d'un mot.
Les symboles `[a..z]` correspondent à l'alphabet, défini comme l'alphabet utilisé par INTEX© [20]. Il ne s'agit donc pas d'un ordre ASCII, puisque les lettres accentuées sont présentes et ordonnées de manière naturelle.

Traitement des débuts de phrase

Pour traiter les majuscules, et en particulier pour traiter la majuscule de début de phrase, nous avons ajouté une fonction spéciale `minusculise` définie sur l'alphabet. Cette fonction transforme le première caractère d'une forme commençant par une majuscule, en son ou ses équivalents minuscules. Cette fonction est utilisée dans le graphe de la figure 4.23(a). Le résultat de la transformation est le symbole {S} marquant le début d'une phrase ainsi que d'une forme résultant de la fonction `minusculise`. Un exemple d'application de ce graphe sur un texte est donné figure 4.23(b). Dans cet exemple, un alphabet avec majuscules non accentué est utilisé. La forme *A* qui suit un point suivi d'un espace, possède donc deux transformations, *a* ou *à*. Avec un alphabet à majuscules accentuées[15], seul l'interprétation *a* aurait été produite.
On remarque que l'interprétation locale «. *Loïc*→{S} loïc» a été construite, mais que la séquence initiale *M. Loïc Josse* n'a pas été modifiée, son interprétation se fera par le graphe des noms propres.

Cette fonction `minusculise` est définie de manière procédurale, ce qui est assez légitime car la notion de majuscule est assez universelle. Elle utilise le fichier alphabet de la langue considérée. Ce sont les règles d'utilisation de l'alphabet qui sont codées.

4.2.8 Les opérations sur les numéraux

Dans les graphes décrivant les dates, il existe des «contraintes d'accord» entre le quantième et le mois. Le quantième peut être écrit en chiffres ou en lettres.
L'expression de l'année dans une date possède une grammaire différente de l'expression d'un valeur monétaire: en effet on écrit *le 2 décembre 98* ou *le 2 décembre 1998*. Il n'est pas possible d'insérer un espace entre le chiffre des milliers et la séquence *98* est équivalente à *1998*[16].

14. Remarquons que le transducteur n'est pas déterministe en sortie. Il est rendu déterministe avec la définition suivante: partant d'un nœud donné, les transitions `[a..z]` sont interprétées comme *toutes les lettres de l'alphabet pour lesquelles il n'existe pas d'autre transitions*. Cela permet d'éviter de faire l'énumération des lettres n'ayant pas d'autres interprétations.
15. Ce qui n'est pas encore répandu dans tous les quotidiens français.
16. Le bug du millénaire changera peut-être les habitudes journalistiques.

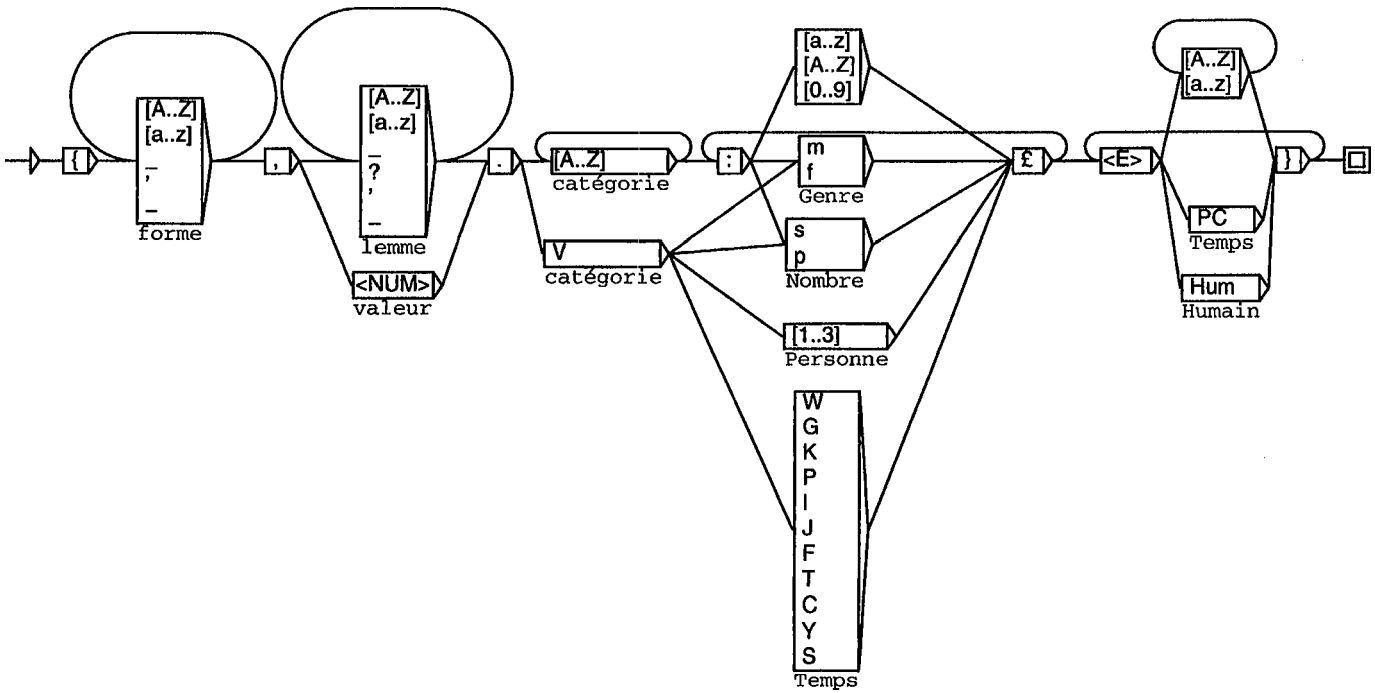
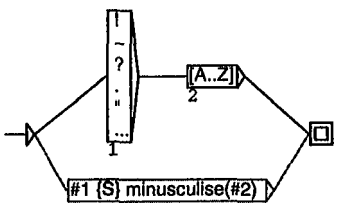
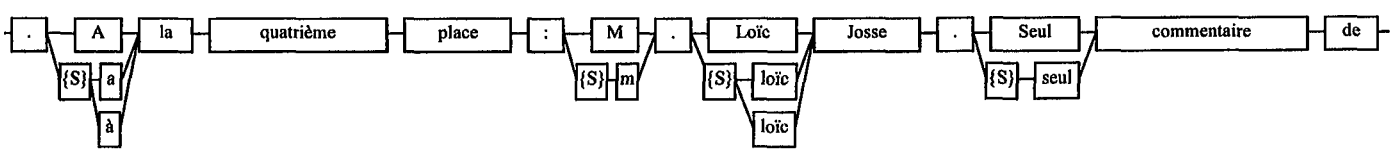


FIG. 4.22 – *Graphe décrivant la structure des mots et identifiant ses différents champs par un nom. Ce nom sera le nom de la fonction permettant d'extraire le champ concerné.*



(a) Transducteur étendu localisant le début de phrase, et mettant le premier mot en minuscule.



(b) Automate du texte après application du graphe précédent.

FIG. 4.23 – *Traitement des débuts de phrase.*

Les numéros de téléphones sont décrits par une autre grammaire, il s'agit d'une combinaison de chiffres de longueurs fixes en France, dont on ne cherchera pas à évaluer la valeur globale. La segmentation est aussi différente puisque les chiffres sont groupés deux par deux[17].

Pour implémenter ce type de grammaire, nous avons besoin de manipuler des nombres en pouvant contrôler leurs formes (littérale, numérique, chiffres groupés deux par deux...) ainsi que leurs valeurs (*le 31 septembre* sera une faute syntaxique[18]). Nous avons défini pour cela un nouveau type de nœud dans l'automate du texte. Ce nœud est organisé en quatre champs comme un mot : la **forme**, la **valeur**, la **catégorie**, et des **informations** syntaxiques et sémantiques éventuelles. La valeur se substitue donc au lemme du mot. Nous appelons ces nœuds des **nombres**. Ces nœuds permettent de stocker par exemple les valeurs suivantes :

forme	valeur	catégorie	Info
23	23	*NUM*	p + `Chiffre`
vingt − trois	23	*NUM*	p + `Lettre`
12 453	12453	*NUM*	p + `Chiffre`

Il faut ensuite des règles de transformation permettant de constuire ces nœuds. Par exemple :

- Dans le graphe 4.25(a), nous décrivons la reconnaissance des nombres sous formes de chiffres. Le symbole <*DIG*:*n*> correspond à une séquence de n chiffres consécutifs. Ce graphe reconnaît donc tous les chiffres de la forme $([0..9]^{\leq 3}[0..9]^3) + [0..9]^*$. Par exemple *1*, *23*, *10 234*, *10234*, etc.
  Une des deux règles de transformation associée est :
  $$\{\#1\ \#2, [\#1]*[1000]+[\#2].DNUM\texttt{:mfp+Chiffre}\}$$
  L'opération $[xxx]$ associe une valeur numérique à une séquence de chiffres contigus xxx. L'expression *[#1]*[1000]+[#2]* représente donc la valeur numérique du nombre reconnu. `mfp` indique que le nombre reconnu est pluriel (p) et le genre est indéterminé (`mf`). Cette transformation ne doit donc s'appliquer que si le nombre est supérieur à 1, ce qui est indiqué dans la contrainte liée à la transformation. L'information `+Chiffre` indique que le nombre construit provient d'une séquence en chiffres.
- Dans le graphe 4.25(b), nous décrivons la reconnaissance des nombres en lettres compris entre 1 et 10. La transformation associée à chaque chemin reconnu est simple, elle crée la valeur numérique de la séquence reconnue, la valeur est sa valeur numérique, sa catégorie est encore *DNUM*, et les informations font maintenant apparaître `+Lettre` indiquant que le nombre provient d'une séquence littérale.
- Le graphe 4.25(c), reconnaît les nombres compris entre 0 et 99, en utilisant les nombres créés par la règle précédente.

17. Dans le journal Le Monde, le séparateur est le trait d'union. Cette grammaire dépend vraisemblablement du journal.

18. La portée de cette correction syntaxique est évidemment moindre, mais nous permet d'illustrer simplement la notion de contrainte.

Nous pouvons ainsi écrire les grammaires de nombres aussi grands que l'on désire. Les nombres reconnus sont ensuite utilisés dans des graphes comme celui de la figure 4.24, où on effectue des vérification entre le quantième et le mois. Dans ce graphe, on ne précise pas que le quantième provient d'une séquence numérique. De la même façon, en écrivant la grammaire des mois, on pourrait associer à chaque mois une valeur numérique, et faire correspondre ainsi des dates purement numériques aux dates où le mois est écrit en toutes lettres.
Nous montrons un résultat d'application des graphes de reconnaissance de nombres à un texte d'exemple sur l'automate de la figure 4.26.

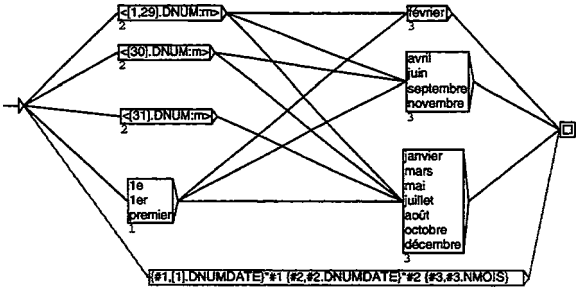
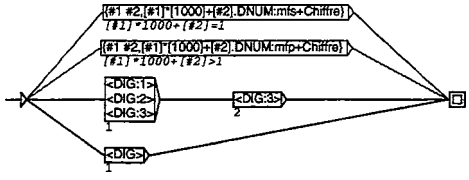
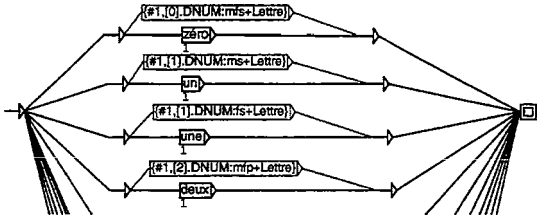


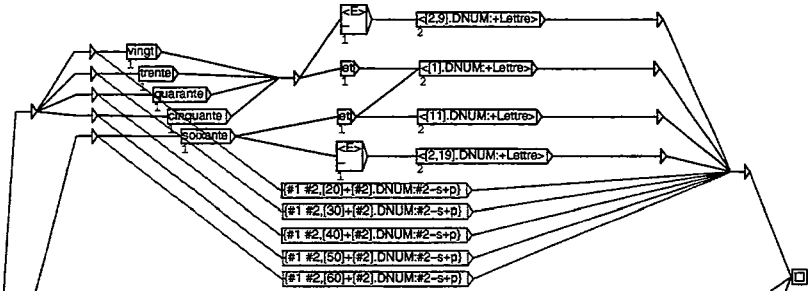
FIG. 4.24 – *Graphe reconnaissant les dates quantième mois avec contraintes du quantième par rapport au mois.*



(a) Reconnaissance des nombres de la forme *nn nnn*, ici une contrainte sur le nombre permet de fixer le nombre singulier/pluriel



(b) Reconnaissance des nombres en lettres (unités)



(c) Reconnaissance des nombres en lettres (dizaine) et calcul de leurs valeurs.

FIG. 4.25 – *Graphes reconnaissant les nombres sous leurs formes numériques et littérales. Dans tous les cas, la valeur représentée est calculée.*

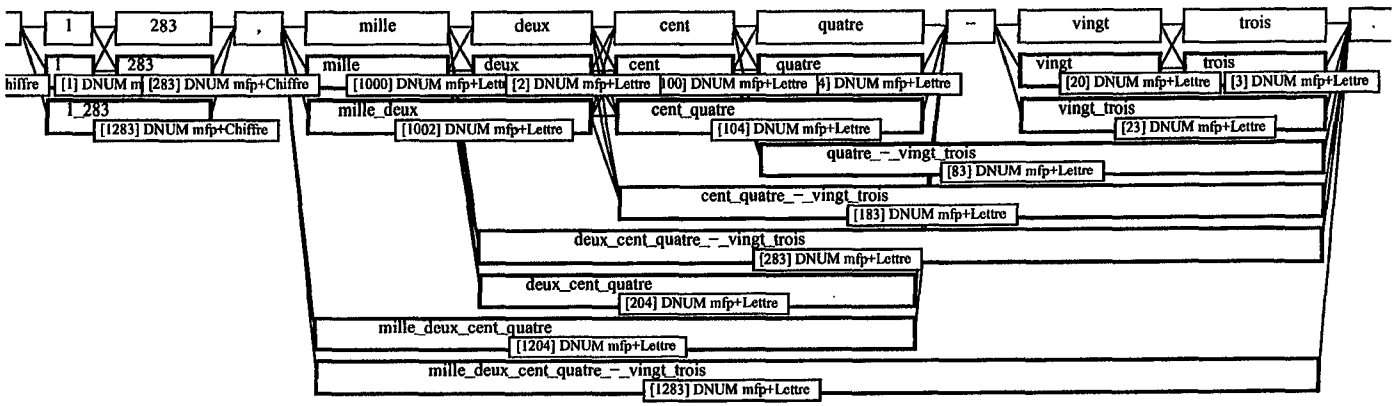


FIG. 4.26 – *Automate de texte, après application de grammaires de reconnaissance de nombres. On remarque que toutes les sous-chaînes correspondant à des nombres sont reconnues. De plus, les nombres sous forme numérique portent l'information* `+Chiffre`*; les autres, l'information* `+Lettre`.

4.3 Projection

Opérer sur l'automate du texte permet de mener parallèlement plusieurs analyses. Tant que l'analyse syntaxique complète d'une phrase donnée n'est pas terminée[19], il y aura plusieurs analyses possibles pour chaque mot de la phrase. Les travaux d'analyse statistique ou d'indexation automatique[20] sont difficiles à définir sur l'automate du texte : on veut donc pouvoir travailler sur une forme linéaire même si l'analyse syntaxique de la phrase n'a été que partielle. Nous devons donc faire une sorte de **projection** qui produit une séquence linéaire représentant «au mieux» l'analyse faite. Il est évident que ce processus engendrera une perte d'information importante et nous retrouverons le problème des ambiguïtés locales, puisque nous devons faire un choix pour l'analyse de chaque mot. Il ne peut pas y avoir de solution exacte à ce problème.
Nous présentons ici sur un exemple, notre algorithme de projection. Il utilise une heuristique de choix (analogue à celle de longest-match (voir 2.2.2). Cette heuristique n'est pas suffisante, en particulier dans le cas de **chevauchement de motifs** ou dans le cas de chemins de mêmes longueurs. Dans ces situations, nous prenons l'intersection de toutes les projections possibles, quitte à revenir à la forme initiale du texte.

Considérons par exemple l'automate de la phrase (7), après une analyse partielle reconnaissant des adverbes figés et des groupes nominaux[21] (figure 4.27).

(7) Tous ses partenaires passent de plus en plus au second plan.

Nous pouvons envisager la projection comme étant le choix d'un chemin particulier parmi tous les chemins de l'automate. Pour notre automate, il y a 4 950 chemins. Ce choix peut-être guidé par plusieurs critères :

1. Minimisation du nombre de nombre de formes dans ce chemin : en effet, les formes correspondent à des unités non analysées. Le chemin le moins bon est celui qui n'a reçu aucune analyse. Ce critère appliqué à notre phrase exemple nous restreint à 6 chemins différents (figure 4.27(b)).

2. Longest-match[22] : quand plusieurs chemins décrivent une séquence, on choisit le plus long (la longueur étant mesurée par rapport au texte initial). Dans notre exemple, cela restreint le nombre de chemins à 4 seulement (voir figure 4.27(c)).

3. Minimisation du nombre d'objets dans la phrase : dans notre exemple, on remplace *à le* par *au*. Ce critère n'est donc valide que pour les transformations d'automate diminuant le nombre des objets.

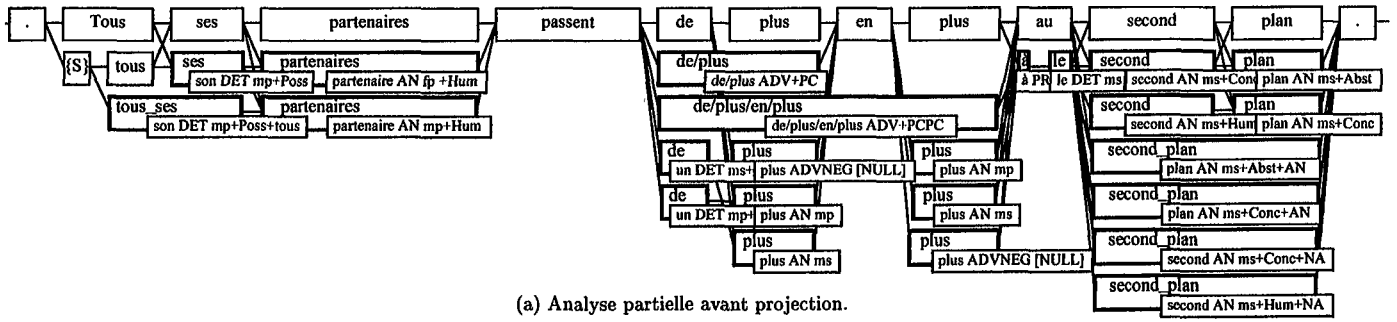
Il est possible de mettre en défaut chacun de ces critères sur un exemple particulier. Chacun de ces critères peut être retenu comme le meilleur dans d'autres cas bien choisis. Cependant, en pratique, ces trois critères fournissent des projections comparables.

19. En fait, même quand l'analyse complète sera effectuée, il pourra rester des ambiguïtés au niveau de la phrase, qui feront qu'il y a plusieurs chemins entre le début et la fin de chaque phrase.

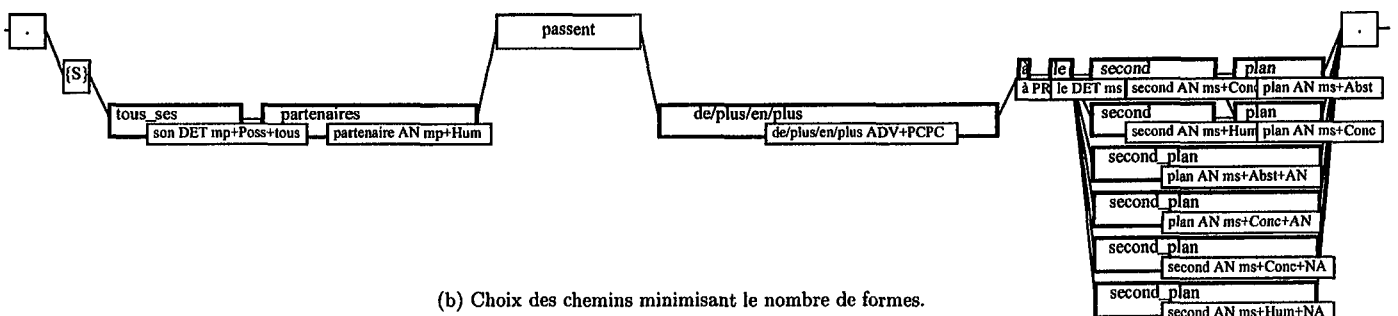
20. Indexation signifiant ici extraction de mots clefs.

21. Les séquences reconnues comme étant des groupes nominaux sont mises sous la forme canonique *DET AN*, la catégorie *AN* représente un nom éventuellement modifié à droite ou/et à gauche. Nous justifions cette écriture dans 7.1.

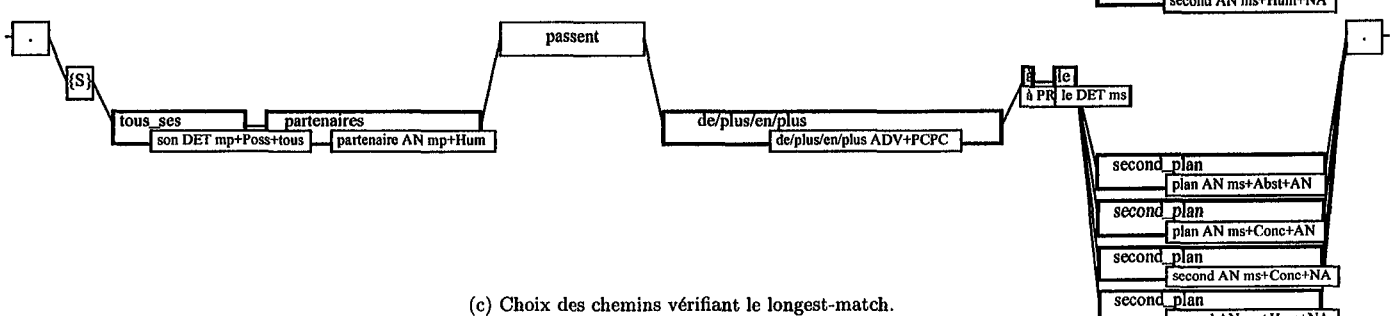
22. Il ne s'agit pas du **longest-match** décrit page 24. Dans notre exemple, chaque chemin différent est créé par un automate différent. Et donc l'heuristique considérant à choisir un chemin à chaque application d'automates est inopérante.



(a) Analyse partielle avant projection.



(b) Choix des chemins minimisant le nombre de formes.



(c) Choix des chemins vérifiant le longest-match.

FIG. 4.27 – *Projection de 4.27(a) en 4.27(b) ou 4.27(c).*

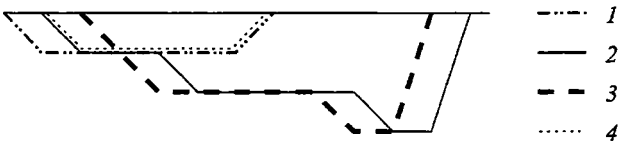


FIG. 4.28 – *Trois différents chemins de l'automate du texte après analyse partielle.*

Nous avons choisi d'appliquer un critère de longest-match que nous exprimons ainsi : on considère l'ensemble des sous-chemins dont les nœuds initiaux n_i et les nœuds finaux n_f sont sur le texte linéaire. On ordonne les nœuds du texte linéaire de la gauche vers la droite (ordre naturel de lecture).
Pour tout sous-chemin $\mathcal{C}_1$ dont les nœuds initiaux et finaux sont n_i^1 et n_f^1, s'il existe un chemin $\mathcal{C}_2$ dont les nœuds initiaux et finaux sont n_i^2 et n_f^2 tels que : $\begin{cases} n_i^2 \leq n_i^1 \\ n_f^2 \geq n_f^1 \\ n_i^2 \neq n_i^1 \text{ ou } n_f^2 \neq n_f^1 \end{cases}$
alors on supprime le sous-chemin $\mathcal{C}_1$.
Nous avons représenté sur la figure 4.28 un automate de texte, sur lequel nous avons distingué quatre sous-chemins. On voit facilement que le sous-chemin 3 doit être supprimé puisque le chemin 2 vérifie la condition ci-dessus. De même le sous-chemin 4 vérifie la condition précédente avec les chemins 2 et 1. La condition de suppression est définie sur des chemins, et non pas sur des transitions. L'algorithme codant ces suppressions n'est donc pas direct. En particulier, nous voyons que pour supprimer le sous-chemin 4 en conservant le sous-chemin 1, il faudra dupliquer ce dernier.

Cet algorithme va donc supprimer un certain nombre de chemins de l'automate. Cependant des sous-chemins parallèles peuvent subsister dans deux cas de figure :

1. Cas de sous-chemins complètement parallèles : leurs nœuds initiaux et finaux sont identiques. Par exemple *le second plan* dans notre exemple précédent reçoit les deux analyses parallèles $le_{DET}\ second_N\ plan_A$ et $le_{DET}\ second_A\ plan_N$.

2. Cas de chevauchement entre deux sous-chemins : leurs nœuds initiaux et finaux sont alternés (celui qui commence avant l'autre, se termine aussi avant). Les chemins 1 et 2 de la figure 4.28 sont ainsi. Nous rencontrerons ce phénomène dans la reconnaissance des expressions verbales figées quand le complément d'une structure verbale figée coïncide avec une partie d'un nom composé. Par exemple dans la phrase suivante, *donner des leçons* est reconnu comme une forme dérivée de la structure verbale figée N_0 *donner une leçon à* N_1. Et *leçons de morale* est reconnu comme un nom composé.
   «Il serait mieux qualifié pour (donner des {leçons)$_1$ de morale}$_2$ si son septennat n'avait pas été ...»

Dans ces deux cas, nous calculerons l'intersection de la projection de tous les chemins parallèles. Ainsi, nous voyons que *second plan* a 4 interprétations différentes dans l'automate du texte de la figure 4.27(a). En effet, les deux mots peuvent être nom ou adjectif, et chacun des deux noms possède deux entrées différentes :

{*plan*, *plan*.*N*:ms+Abst} (passer au second plan)
{*plan*, *plan*.*N*:ms+Conc} (le plan géométrique)
{*second*, *second*.*N*:ms+Hum} (le second, adjoint du capitaine)
{*second*, *second*.*N*:ms+Conc} (le second étage[23])

Cela donne donc quatre groupes nominaux possibles, représenté[21] par 4 séquences *DET AN* différentes. Il n'est pas possible ici de décider quel groupe choisir ; l'intersection des quatre chemins conduit au groupe nominal incomplet :

{*le*, *le*.*DET*:ms+Ddéf} {*second_plan*, *???*.*AN*:ms}

Nous avons perdu le lemme dans l'opération, mais pas le genre ni le nombre, ni la structure globale de groupe nominal.

Il se peut que l'intersection de la projection soit vide, par exemple :

{*donner des leçons*, donner une leçon à.*V*:+C1IPN}
{ *leçons de morales*, leçon de morale.*N*:fp+NDN}

Dans ce cas la forme du texte initiale est utilisée.

Pour l'exemple (7), la projection de l'automate du texte (figure 4.27) selon cette méthode est alors :

{*S*} {*tous_ses*, *son*.*DET*:mp+Poss+tous} {*partenaires*, *partenaire*.*AN*:mp+Hum} *passent* {*de/plus/en/plus*, .*ADV*:+PCPC} {*à*, .*PREP*} {*le*, *le*.*DET*:ms+Ddéf} {*second_plan*, *???*.*AN*:ms}.

Il est clair que des analyses linguistiques plus profondes peuvent améliorer certains résultats de l'analyse de cette phrase.

4.4 Conclusion

Nous avons développé plusieurs outils basés sur la description des besoins et des contraintes linguistiques du chapitre précédent. Ces outils permettent de gérer un texte sous la forme *automate de texte*, ainsi que des dictionnaires, grammaires sous la forme de graphes simples, de transducteur étendus, de *graphes patrons* permettant d'utiliser les tables du lexique-grammaire. Ces outils correspondent aux objectifs généraux donnés dans le chapitre précédent :

- Les graphes simples sont basés sur un formalisme bien connu, les automates finis. Mais nos graphes ne sont pas limités aux expressions rationnelles. D'une manière très libre, plusieurs fonctionnalités pertinentes linguistiquement ont été ajoutées. Le formalisme déborde donc du domaine rationnel (introduction du traitement de nombres, ajout de différents champs variables aux unités du texte, utilisation de sous-graphes).

- Les graphes de transformation permettent d'opérer des opérations complexes sur les différents objets d'un texte. Nous avons intentionnellement omis de ce chapitre la description complète du formalisme définissant ces graphes[24]. D'une part, cela nous a permis d'alléger la description de ces graphes et d'autre part d'en montrer

23. En fait d'une manière générale : *le second*, est une forme pronominale dont le nom de référence peut être quelconque. *Tu vois ces livres ? Donne moi le second !*. Cependant, d'une manière assez figée, on dit *rendez-vous au second.*, sans faire référence directe à *étage*.

24. Cette description est donnée en annexe A.

les capacités sur des exemples. L'étude de la puissance théorique peut être faire séparément.

- L'utilisation de dictionnaire n'est pas une étape préalable indispensable à l'utilisation de grammaires. Cette particularité s'exprime d'une manière «procédurale», puisque l'application d'un dictionnaire au texte linéaire ne crée pas de chemins dans l'automate du texte. Nous montrerons en particulier, qu'un grand nombre de structures sont reconnues sans une seule référence au dictionnaire.

- Toutes les opérations se font sur des *mots*, qui sont aussi bien des séquences de plusieurs formes non connexes que des formes isolées.

- La notion d'ambiguïté n'est présente nulle part dans nos outils, ni dans nos grammaires. Celle-ci est, un résultat potentiel d'une analyse, et non pas un point de départ conduisant à la création de grammaires.

- Il est toujours possible, à tout moment de l'analyse, de transformer l'automate du texte en une séquence linéaire représentant au mieux les transformations déjà appliquées à l'aide de la projection.

Pour coder ces outils d'une manière efficace, nous avons dû développer des algorithmes nouveaux de «pattern-matching». Cependant, nous avons fait passer l'efficacité algorithmique du formalisme après sa complexité théorique. En particulier, nous n'avons pas essayé de modifier le formalisme quand nous rencontrions des difficultés algorithmiques dues à des phénomènes nouveaux, cela n'aurait pas été cohérent avec notre démarche basée essentiellement sur la description linguistique[25].
Ce chapitre n'a de sens, que si les outils proposés sont réellement utilisés dans des cas concrets. Nous proposons dans la partie suivante la description de grammaires permettant de reconnaître des unités figées et des groupes nominaux.
La séparation outil/grammaire que nous avons opérée dans ces deux chapitres est essentiellement due à un souci de lisibilité. L'interaction est en fait beaucoup plus importante. Cela signifie que la description d'autres phénomènes, ou la description d'une autre langue nous obligera vraisemblablement à introduire d'autres fonctionnalités *ad-hoc*. Cela ne nous paraît pas fondamentalement gênant, et c'est même le reflet de notre méthodologie de création d'outils dont la base est la description linguistique.

25. D'une manière étonnante, nous verrons que le contraire s'est produit. Le traitement des sous-graphes dans le «pattern-matching» nous a conduit a considérer la factorisation de sous-unités dans des automates finis quelconques. La résolution de ce problème conduirait à améliorer les performances du pattern-matching avec n'importe quel automate.

Bibliographie

[1] E. Brill. A simple rule-based part of speech tagger. In *3rd Applied ACL*, pages 152–155, Trente, 1992.

[2] E. Charniak. *Statistical Language Learning.* MIT press, Massachusetts, 1993.

[3] A. Chrobot. Tagging statistique sans ambiguïtés ou tagging partiellement ambigu mais sans erreur. Rapport technique, LADL, 1997.

[4] B. Courtois. Formes ambiguës de la langue française. *Linguisticæ Investigationes*, XX(1):167–202, 1996.

[5] S.J. DeRose. Grammatical category disabuiguation by statistical optimization. *Computational Linguistics*, 14:31–39, 1998.

[6] M. Garrigues. *Méthode de paramétrage des dictionnaires et grammaires éléctroniques.* Thèse de doctorat, Université Paris 7, Paris, 1993.

[7] P. Geluck. *LE CHAT*, chapter 1, page 56. Casterman, 1986.

[8] M. Gross. *Grammaire transformationnelle du Français – Syntaxe de l'adverbe*, volume 3. Cantilène, 1986.

[9] M. Gross. Les ambiguïtés. Rapport technique, LADL, 1994.

[10] Z.S. Harris. *Mathematical Structure of Language.* Wiley - Interscience, New-York, 1968.

[11] A.K. Joshi, L. Levy, et M. Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 1975.

[12] L. Karttunen. Directed replacement. In *Proceedings of ACL'96*, Santa Cruz, California, 1996. Une présentation peut aussi être trouvée à `http://www.rxrc.xerox.com/grenoble/mltt/fst/articles/jnle-97/rele.html`.

[13] E. Laporte. Levée d'ambiguïtés par grammaires locales. *Linguisticæ Investigationes*, Supplementa 17:97–114, 1995.

[14] C. Leclere. Dictionnaire électroniques du français. In *Langue française*, volume 87, chaptitre Un système de dictionnaires électroniques pour les mots simples du français, pages 11–22. Larousse, 1990.

[15] C. Leclere. Dictionnaire électroniques du français. In *Langue française*, volume 87. Larousse, 1990.

[16] J. Pitrat. *Textes, ordinateurs et compréhension.* Eyrolles, 1985.

[17] A. Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of the First Empirical Methods in NLP*, Philadelphia, 1996.

[18] E. Roche. Text disambiguation by finite-state automata, an algorithm and experiments on corpora. In *COLING'92*, Nantes, 1992.

[19] E. Roche. *Analyse syntaxique transformationnelle du français par transdcteurs et lexique-grammaire.* PhD thesis, Université Paris 7, 1993.

[20] M. Silberztein. *Dictionnaires électroniques et analyse automatique de textes. Le système INTEX.* Masson, 1993.

[21] M. Silberztein. Expérience d'étiquetage avec levée d'ambiguïtés. In *Actes des Premières journées INTEX*, Paris, 1996. LADL, Université Paris 7.

Deuxième partie

Les expressions figées

4.5 Introduction

LE MOT (suite de caractère entre deux séparateurs) ne correspond pas *a priori* à une unité syntaxique élémentaire comme nous l'avons vu au chapitre 3. Par exemple *aujourd'hui* est structurellement composé de deux mots mais forme une seule unité syntaxique. On dira alors que *aujourd'hui* est une unité syntaxique minimale[26] composée[27]. Nous voulons reconnaître toutes les «unités minimales composées» du français. Cette notion d'unité composée est intimement liée à la notion d'expression figée. En effet (voir [8]) : une **expression figée** est une expression composée dont la structure et/ou le sens ne peut être calculé à partir de la structure et/ou du sens des différents composants. Une expression figée est donc une unité minimale composée.

Exemple : les différentes formes figées suivantes sont des unités minimales composées dont nous donnons le type ainsi qu'un équivalent non composé en anglais[a] quand nous en trouvons un :

carte postale	nom	postcard
noir et blanc	adjectif	a monochrome photo
de moins en moins de	déterminant	
au nez de	préposition	
entre-temps	adverbe	meanwhile
vraiment pas	négation	
s'entre-tuer	verbe	
par conséquent	conjonction	therefore
faire les frais de la conversation	verbe	
lui-même	pronom	himself
Pierre qui roule n'amasse pas mousse	phrase	

[a] Le fait qu'un même concept s'exprime à l'aide d'un seul mot dans une autre langue renforce l'idée que la séparation en mots n'est pas significative.

Cependant, nous ne nous restreignons pas aux expressions figées : certains phénomènes linguistiques minimaux sont intrinsèquement composés, dans la mesure où leur expression utilise plusieurs mots. C'est le cas essentiellement de la négation et de la flexion des verbes aux temps composés. En effet, en français, l'introduction de la négation passe par l'utilisation de deux mots : le *ne* et la négation elle-même (*Il n'est pas là*). Cette caractéristique de la négation ne reflète pas que celle-ci est composée de deux unités minimales. En allemand, la même négation s'exprime à l'aide d'un seul mot : *nicht* (*Er ist nicht da.*). En anglais, le phénomène est moins clair, puisque la négation peut faire ressortir l'auxiliaire (*He is not here, He does not want.*). De la même façon, l'expression des temps composés comme le passé composé passe par un auxiliaire et un verbe au participe passé. Il ne fait cependant pas de doute que seul l'ensemble auxiliaire et participe passé forme l'unité syntaxique minimale.

Il y a donc deux types différents d'unités minimales composées, pour lesquelles la démarche ne sera pas la même. Les expressions figées par définition **ne** sont pas syntaxiquement régulières. On ne peut pas dire la même chose de la négation puisque le phénomène de la négation est unique : cela aurait peu de sens de montrer que ce phénomène a un comportement irrégulier et donc qu'il est «figé». Cependant par souci de simplification, nous désignerons désormais ces deux types différents par «expressions figés» (cela n'est que

26. Minimale car ici les deux mots *aujourd* et *hui* n'existent pas. D'une manière plus générale, on dira qu'une unité est minimale quand sa décomposition n'apporte aucune information.

27. Composée car comportant plusieurs «mots».

terminologique).
Pour bien préciser les fondements, l'optique de notre démarche ainsi que les problèmes soulevés nous commençons par rappeler brièvement ce que nous entendons par reconnaissance dans ce cadre précis des unités minimales composées. Nous montrons ensuite les enjeux de la démarche et la manière dont elle s'intègre dans un objectif plus large d'analyse syntaxique.

4.6 La reconnaissance

Nous appellerons la reconnaissance des unités composés, la *localisation physique* dans le texte de telles séquences à l'aide de ressources *spécifiques*. Il y a donc deux aspects différents: la localisation, et l'utilisation pour cette localisation d'une ressource spécifique. Montrons la différences entre ces deux aspects sur un exemple: il existe des noms composés de type <*N*>-<*N*> (*homme-grenouille*). Nous pouvons très simplement les localiser avec un taux d'erreur quasiment nul en faisant une recherche systématique de tous les motifs correspondant à <*N*>-<*N*> à l'aide d'un dictionnaire de mots simples. Cette procédure les *localiserait*, mais ne les reconnaîtrait pas, dans la mesure où si on voulait leur associer une valeur sémantique, une traduction dans une langue étrangère ou des propriétés syntaxiques, la procédure de localisation ne pourrait apporter aucune information supplémentaire (vu que toutes les occurrences auront été reconnues indifféremment). Dans cet objectif, la seule solution pour ces applications est d'avoir une liste exhaustive de ces noms composés. Dans cette liste, il est alors immédiat de rajouter les informations voulues. Pratiquement, pour nos unités composées en général, les ressources que nous utilisons sont des dictionnaires (listes d'entrées associées à des informations flexionnelles et sémantiques), des tables du lexique grammaire mais aussi des bibliothèques de graphes. Une unité composée est reconnue par un graphe si elle est associée à un chemin propre la différenciant des autres unités composées. L'ajout d'information dans un graphe se fait alors simplement à l'aide de transduction. Nous verrons qu'en fait, ces trois types de ressources différents sont transformés (ou seront à terme transformés) en une bibliothèque de graphes. Nous avons déjà détaillé le mécanisme de transcription des tables du lexique-grammaire en ensemble de graphes, nous verrons que le devenir des dictionnaires de mots composés est aussi le même.

4.7 L'objectif

On peut voir la localisation des unités minimales composées de deux points de vue différents. D'un côté, il s'agit d'une opération permettant d'obtenir des unités syntaxiques pertinentes, ce qui donne un but en soi à la démarche. De l'autre, du fait du caractère figé de ces unités minimales, il n'y aurait pas moyen de reconnaître[28] des phrases contenant de telles unités. La reconnaissance des unités figées est donc un moyen de «normaliser» un texte vis-à-vis de la même grammaire qui avait déclaré le caractère non-compositionnel ou non syntaxique de ces unités.
La question découlant de la définition des expressions figées, est «*Quelles sont la grammaire et la procédure de calcul sémantique nous permettant de décider de leur caractère*

28. Par une grammaire régulière et/ou une procédure de calcul de sens aussi «régulière».

figé ?». Nous décrivons cette opération de normalisation dans la partie suivante, et montrons quelle est la grammaire libre de référence, en montrant la limite de son application.

4.8 La normalisation

Comme nous venons de le voir, un des buts/effets de la reconnaissance d'expression composée est de normaliser le texte. Par exemple la phrase (1) est non grammaticale[29] parce que *Question diffusion* est une séquence <N> <N>, de plus *donner lecture* est une séquence <V> <N> et *à tout le* est une séquence *à* <ADV> <DET> (de la même façon que **à lundi trois villages*). Aucune de ces séquences ne sont régulières. Après reconnaissance des expressions figées, la phrase aura la forme plus régulière (2) :

Prép N, N V à Dét N de Dét N

(1) *Question diffusion, Pierre a donné lecture à tout le village de ce message.*

(2) *(question)$_{PREP}$ diffusion, Pierre (a donné lecture)$_{V}$ à (tout le)$_{DET}$ village de ce message.*

Pour que cet étiquetage soit cohérent par rapport à cette «grammaire régulière», il faut que celle-ci soit clairement définie. En fait, nous ne croyons pas que celle-ci existe vraiment. Nous avons déjà vu que chaque phrase dérive d'un schéma de phrase élémentaire, et que la reconnaissance de ce schéma élémentaire doit passer par le lexique-grammaire (où chaque schéma élémentaire possède une entrée distincte). Cette grammaire ne peut donc pas décrire ces schémas, mais au moins faire un ensemble d'opérations minimales sur les phrases. En particulier, dans la phrase précédente, il ne serait pas possible de voir que *Question diffusion* est un adverbe et qu'il possède donc vraisemblablement une certaine mobilité sans avoir procédé à l'opération de normalisation décrite. Cela entraînerait d'une manière très générale, des blocages dans l'analyse des phrases à cause de ce composant dont il n'est même pas possible de deviner la nature.
Nous avons donc choisi d'adopter un certain nombre de conventions que nous indiquons ici. Dans cette «grammaire régulière» :

- Tous les groupes nominaux <*GN*> auront la forme (voir 7.1):

  <*DET*> <*AN*>

  Le symbole <*AN*> sera défini dans 7.1 et regroupe le nom tête et ses modifieurs, mis à part le déterminant que nous avons conservé indépendant pour des raisons que nous préciserons.

- Tous les adverbes <*ADV*> auront la forme :

  <*PREP*> <*DET*> <*AN*>

- Les verbes auront la forme unique <*V*> (en particulier, les auxiliaires sont incorporés au verbe) ;

- Une phrase élémentaire <*P*> aura la forme :

  <*GN*:+N0> <*V*> ((<*PREP*>) <*GN*:+N1> ((<*PREP*>) <*GN*:+N2>))

29. En comparaison avec une «grammaire régulière».

– Une phrase totalement reconnue aura la forme :

$$\{S\}\ \langle P\rangle\ \langle ADV\rangle^*\ \{S\}$$

Tous les adverbes sont donc mis en fin de phrase.

4.9 Expressions composées et expressions simples.

Considérons l'expression *découvrir le pot aux roses.* Nous pourrons la trouver sous différentes formes dans les textes, mais la variation de ces formes se limite à la conjugaison du verbe comme dans (3), à la négation comme dans (4), à différentes permutations dans l'ordre de ces éléments comme dans (5) et éventuellement à l'insertion d'éléments libres comme des adverbes comme dans (6).

(3) *Max (découvre+a découvert) le pot aux roses.*
(4) *Max ne découvre toujours pas le pot aux roses.*
(5) *Le pot aux roses a enfin été découvert par Max.*
(6) *Max a découvert, en fouillant dans les affaires de Léa, le pot aux roses.*

Dans tous les cas, nous n'avons pas besoin d'analyser la séquence *le pot aux roses* qui sera toujours sous sa forme initiale. Cela signifie en particulier, que nous n'aurons pas besoin d'effectuer une analyse syntaxique sur cette expression. Nous pouvons dire que *pot* est un nom, que *le* est un déterminant dans un but de classification de l'expression (et c'est d'ailleurs comme cela que les expressions ont été réparties dans les différentes tables), mais cela ne servira pas pour reconnaître l'expression. Cette caractéristique est très intéressante dans la mesure où elle va permettre de situer l'analyse de certaines formes d'expressions figées en amont de l'analyse syntaxique. Cependant, la séparation n'est pas toujours aussi simple : dans (6), nous avons l'insertion d'une phrase complète *en fouillant dans les affaires de Léa* entre les deux composantes de l'expression figée. Nous avons vu dans la partie précédente, que nous n'utiliserions pas de symboles de reconnaissance génériques comme <*MOT*> qui pourraient nous permettre de «sauter» l'insertion libre pour reconnaître l'expression complète à l'aide du motif de localisation suivant :

$$\langle \textit{découvrir}\rangle\ \langle MOT\rangle^*\ \textit{le pot aux roses}$$

Cela nous conduit donc à dire que cette étape de reconnaissance d'expressions figées ne pourra pas être complètement indépendante de l'étape d'analyse syntaxique générale. Dans le cas de figure précédent, la même structure sera décrite par :

$$\langle \textit{découvrir}\rangle\ \langle ADV\rangle^*\ \textit{le pot aux roses}$$

où <*ADV*> représente les adverbes reconnus dans une phase précédente.
Nous avons représenté sur le schéma de la figure 4.29 page 100, les différentes relations entre les différents objets que nous devrons manipuler. Nous avons distingué deux types de relations. Les relations fortes en trait continu (par exemple, pour reconnaître les structures verbales figées, il faut savoir reconnaître la négation), et les relations plus faibles en pointillés indiquant que certaines objets d'une catégorie, dépendent de la reconnaissance d'une autre catégorie (par exemple, nous verrons que pour reconnaître des adverbes figés comme *d'après une source bien informée*, il faut reconnaître les groupes nominaux ayant *source* comme nom-tête : *d'après une source que je tiens à garder secrète, d'après la*

même source, etc.). Ce schéma n'est donné qu'à titre indicatif[30]. Mais il permet de se rendre compte de l'interdépendance de tous les phénomènes, et d'effectuer un premier ordonnancement des différentes opérations que nous devrons effectuer :

- Les adverbes : ceux-ci sont les plus sollicités par les autres objets, et dépendent peu, eux-même, d'autres objets. Nous verrons de plus, que la reconnaissance des adverbes libres se base sur la reconnaissance des groupes nominaux, mais que ceux-ci (les adverbes libres) sont extrêmement rares dans les textes.

- La négation : nous avons indiqué sur le schéma de dépendance une relation forte entre la négation et la reconnaissance complète des phrases élémentaires. En fait, nous verrons dans 6 que cette relation indique que pour identifier la portée de la négation ainsi que pour vérifier que le verbe associé à un groupe nominal négatif porte bien le *ne*, nous devrons attendre d'avoir analysé toute la phrase. Cependant, la reconnaissance des différentes composantes de la négation peut se faire localement. Il y aura donc deux étapes dans la reconnaissance de la négation : l'étape locale reconnaissant les différentes composantes de la négation (*Pour autant, il n'a lu à Pierre aucun livre*), et une étape à un niveau d'analyse beaucoup plus poussé les regroupant ou vérifiant leurs dépendances.

- Les auxiliaires de temps des verbes, les pronoms préverbaux, l'inversion du sujet pronominal, les auxiliaires de modalité.

- Les déterminants composés, puis les groupes nominaux.

- Les structures verbales figées.

Il sera possible de trouver des schémas de phrase dont l'analyse ne suit pas cet ordre de préférence, ou qui nécessitent l'application répétitive de ce schéma. Dans la phrase (7) suivante par exemple, l'analyse de *en écoutant France-Inter* bloquera l'analyse de la phrase principale qui ne pourra être analysée que lors d'une deuxième passe.

(7) *Il lit, en écoutant France-Inter, un livre de poésie.*

4.10 La grammaire de reconnaissance

La description que nous allons faire suit le schéma ci-dessous basé sur l'ordre que nous venons de donner :

1- Reconnaissance des phrases et «minusculisation» ;

2- Ouverture des différents dictionnaires de mots simples ;

3- Reconnaissance des noms composés à l'aide du dictionnaire des noms composés ;

4- Résolution des contractions ;

5- Résolution des élisions ;

30. Il est évident qu'il serait possible de trouver des relations entre chacun des objets

6- Reconnaissance des formes numériques et calcul de leurs valeurs ;

7- Reconnaissance des pronoms préverbaux ;

8- Reconnaissance des formes verbes-pronom sujet ;

9- Reconnaissance des auxiliaires de verbes ;

10- Reconnaissance des <*AN*> ;

11- Reconnaissance des adverbes ;

12- Reconnaissance des négations ;

13- Reconnaissance des groupes nominaux ;

14- Reconnaissance des structures verbales figées ;

15- Répétition des étapes 10 à 15 ;

16- Projection sous forme linéaire.

Nous rappelons que dès l'application de la première grammaire (reconnaissance des phrases), nous passons d'un texte linéaire à un automate de texte. La notion d'ambiguïté n'est donc pertinente à aucune de nos grammaires. Si nous avons besoin, en sortie, d'un texte analysé sous forme linéaire, nous effectuerons l'étape 16 qui fera éventuellement ressortir ce problème (voir discussion page 83).
L'ensemble du processus dans sa forme actuelle représente l'application d'environ 400 graphes différents. Nous ne pourrons évidemment pas tous les décrire, et l'intérêt serait moindre. Nous insisterons cependant sur le recensement de toutes les ressources linguistiques utilisées dans le processus.

Nous présentons maintenant les différents types de problèmes que rencontre la reconnaissance actuelle des unités figées ; nous illustrerons le traitement de chacun de ces problèmes de manière détaillée dans les chapitres suivants, et nous donnerons des exemples complets de la méthodologie adoptée.

4.11 Les limites actuelles

Actuellement, un grand nombre d'expressions figées sont déjà reconnues dans INTEX© par l'utilisation de dictionnaires de mots composées (le `DELAC`), et par l'application de graphes (par exemple, les graphes de date). Le `DELAC` qui est au même format que le `DELAS`, est une liste de mots composés provenant de plusieurs sources (noms composés, adverbes figés, ...).

L'utilisation de listes pour reconnaître des unités composés se heurte à plusieurs problèmes :

- diversité des formes prise par une même structure figée ;
- existence de structures figées non connexes ;

- interdépendance des différentes descriptions.

Pour mettre en évidence ces problèmes et montrer la manière de les résoudre, nous commençons par décrire les adverbes figés, dont la dépendance avec les autres descriptions est assez faible, et dont les structures sont quasiment toutes connexes. Cette description nous permet donc de nous concentrer sur la variation des formes prises par une même structure. Nous montrons en particulier des premiers exemples de l'utilisation des graphes patrons.
Nous montrerons ensuite comment la description de la négation (simple et composée) peut-être effectuée. Pour celle-ci, aucune forme n'est connexe, et la diversité des formes n'est pas si importante. Le problème de dépendance avec les autres descriptions sera détaillé.
Nous montrerons ensuite une première description des groupes nominaux dont nous aurons besoin pour reconnaître les structures verbales figées. Enfin, nous détaillerons la description des formes verbales figées, et nous en donnerons des résultats concrets.

BIBLIOTHEQUE * UNIVERSITE

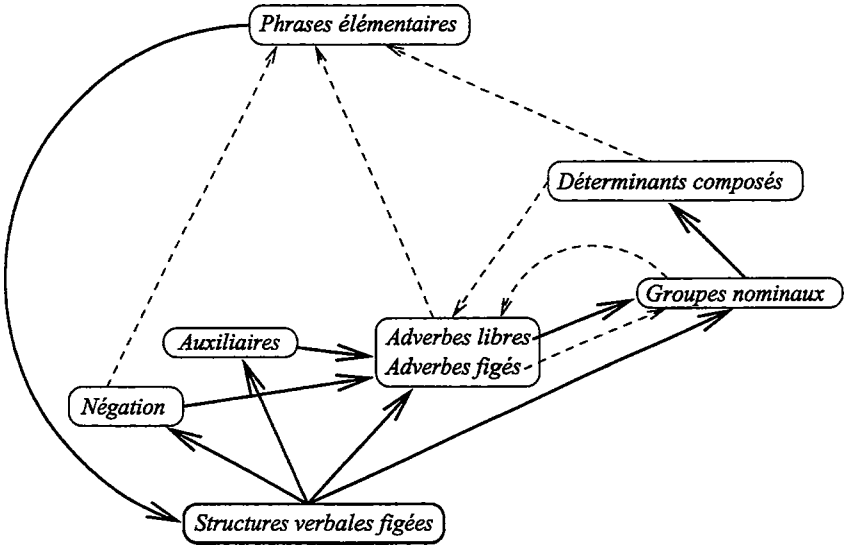


FIG. 4.29 – *Dépendance entre les différentes descriptions. Les traits pleins montrent des dépendances fortes, les traits en pointillés montrent des dépendances occasionnelles.*

Chapitre 5

Les adverbes figés

LA NOTION D'ADVERBE dont nous parlons ici, est celle définie dans le cadre du lexique-grammaire (cf. [10]). Il s'agit *approximativement* des compléments de verbes non incorporés dans les schémas de phrase élémentaire. Les adverbes ont la forme générale *Prép Dét N Modif*, ce qui s'exprime avec nos notations <*PREP*> <*DET*> <*AN*>. D'une manière générale, les adverbes ont une certaine mobilité dans la phrase : cette mobilité est responsable de la majorité des cas d'unités composées non connexes que nous observons dans les chapitres suivants. Nous discuterons donc des moyens pour traiter cette mobilité.

L'analyse syntaxique des adverbes [10] nécessite de faire ressortir un schéma de phrase sous-jacent : dans la phrase (1) par exemple, il faut faire ressortir la construction «conduire en état d'ivresse» pour effectuer l'analyse (2) et comprendre la phrase.

(1) *En état d'ivresse, cet homme est rentré dans un arbre.*
(2) *Cet homme est rentré dans un arbre (avec sa voiture), alors qu'il conduisait en état d'ivresse.*

Cette analyse est très complexe : pour les adverbes provenant de construction en *être*, le rétablissement de la phrase sous-jacente est relativement simple : il «suffit» de réintroduire le verbe *être* effacé (si on a pu résoudre le problème de portée). Pour les adverbes des tables P que nous détaillons plus loin, cette analyse est impossible pour le moment, la construction verbale support indiquée dans la table n'étant qu'un exemple possible. Enfin, sans faire ressortir cette phrase sous-jacente, il n'est pas possible de résoudre le problème de portée de l'adverbe, comme nous pouvons le voir dans les phrases (3) à (6).

(3) *En hiver, cet homme est rentré dans cet arbre.*
(4) *Avec sa voiture, cet homme est rentré dans cet arbre.*
(5) *En état d'ivresse, cet homme est rentré dans cet arbre.*
(6) *A plus de 200 kilomètre/heure, cet homme est rentré dans cet arbre.*

Reconnaître un adverbe aura donc, à ce niveau d'analyse, un sens local sauf pour certaines familles d'adverbes comme ceux de négation ou ceux provenant de déterminants adverbiaux que nous étudierons ailleurs plus en détail. Cette «reconnaissance» d'adverbes nous permettra donc essentiellement, en les déplaçant, de reconnaître d'autres expressions figées. Il y a donc un double enjeu à la reconnaissance des adverbes.
Nous nous intéressons dans l'optique de la reconnaissance d'expressions composées à la seule reconnaissance des adverbes figés. Cependant, pour reconnaître des constructions verbales figées incorporant une insertion d'adverbes quelconques, nous aimerions pouvoir

décrire tous les adverbes. En fait, la distinction entre adverbe libre et adverbe figé n'est pas simple. Dans un texte, les différents adverbes que nous pouvons trouver peuvent être en partie classées dans les catégories «scolaires» de compléments circonstanciels de date, de lieu, et de manière.

- Adverbes de date : tard dans la soirée de jeudi 30 décembre, A partir du 1 janvier, peu après, en 1986, le 26 avril 1981, pendant ces neuf mois ;
- Adverbes de lieu : en France, au Bangladesh, au Proche Orient, dans cette ville de Béthune, dans cette ville de près de cent mille âmes, sur les Champs Elysées ;
- Adverbes de manière : plus particulièrement, volontairement, dans le détail, par balles, avec certitude ;

Parmi ces adverbes, certains sont certainement figés comme *dans le détail, par balle* au sens où leur analyse ne peut pas relever d'une analyse générale. En effet, *dans le détail* n'admet pas d'autre déterminant que *le* ou *les. par balles* pourrait provenir d'une passivation *il a été blessé par une balle*, mais l'effacement du déterminant n'est pas régulier : *Il a été heurté par une voiture* (**Il a été heurté par voiture*).

Pour d'autres d'adverbes comme *le 17 avril 1981*, il semble y avoir une certaine productivité, mais celle-ci est néanmoins contrôlée par une grammaire [17]. Pour les locatifs, la liberté sémantique semble être totale, mais le choix de la préposition est très fortement liée au nom concret associé (*dans la rue*, *sur la route*, *en Ardèche*, *dans le Nord*). Cela signifie qu'il n'est pas possible de «reconnaître» ces adverbes par une formule aussi simple que *Prép Dét N Modif*.
Il est encore plus difficile d'essayer de produire un adverbe de manière qui soit libre, à moins d'introduire la construction *de manière Adj* approprié ou une construction verbale :

(7) *Il lit son livre de manière étonnante.*
(8) *Il lit son livre en (faisant du vélo+conduisant+jouant du saxophone).*

Dans ces deux exemples, la reconnaissance se fait encore par une grammaire simple *de (une+<E>) manière Modif*, et *en V-ant*. Ce que nous voulons mettre en valeur dans ces exemples est qu'on peut produire «aléatoirement» un groupe nominal suivant le schéma *Dét N Modif* [1], mais qu'il est beaucoup plus difficile de produire un adverbe avec le même processus : la préposition étant un élément de contrainte syntaxique supplémentaire forte. Pour cette raison, il est difficile de dire qu'un adverbe est libre, à part peut-être pour des adverbes comme *(devant+dessus+dessous+à côté de+loin de)* <*N*:Conc>.
Dès lors que nous posséderons une grammaire opérationnelle décrivant une construction adverbiale donnée, nous considérerons donc que celle-ci est figée (justement grâce au fait que nous pouvons la décrire, en opposition à une construction libre) et nous la reconnaîtrons. Cette extension va nous assurer une excellente couverture des adverbes dans les textes.
Pour chaque adverbe reconnu (par exemple *de tous les âges*), nous ajouterons de l'information concernant l'origine de son analyse, ce qui permettra de procéder à des opérations ultérieures de traduction, d'analyse syntaxique... Cet information se présentera de deux

1. Avec néanmoins des contraintes que l'on peut dire «sémantique» comme dans **les voitures inspirées*.

manières différentes :

- le lemme contient la structure verbale complète donnée par la table :

$$\{\textit{de tous les âges}, \textit{être de tous les âges.ADV}\}$$

- ou il porte comme information le nom de la table ou du graphe l'ayant repéré :

$$\{\textit{de tous les âges}, \textit{de tous les âges.ADV}:\text{+EPDETC}\}$$

Les graphes patrons que nous présentons produisent la structure verbale complète, mais pour des raisons de lisibilité (entrée plus courte), les exemples ne font référence à la table que par les informations.

5.1 Les adverbes figés des tables P

M. GROSS, dans *Syntaxe de l'adverbe* [10], décrit d'une manière détaillée les tables P du lexique-grammaire, contenant les adverbes figés ne recevant pas d'analyse décrite par d'autres processus (en particulier les tables EPX que nous utiliserons aussi).
Chaque entrée de ces tables est associée dans les tables à un verbe, indiquant une construction verbale support dans laquelle intervient l'adverbe. Regardons son utilisation possible sur quelques exemples :

(9) ... après avoir été grièvement blessé par balles par des inconnus, (il) a annoncé...
(10) On se réunit même entre voisins pour gratter en groupe.
(11) Il convient, estime-t-il en substance, de ne pas choisir immédiatement parmi les quatre départements candidats.
(12) Et voilà qu'à nouveau, il faisait nuit.
(13) L'équipe Clinton saura, à nouveau, vaincre les résistances du Congrès.
(14) Anodine en apparence, cette disposition peut se révéler dangereuse.
(15) Ces instructions tendent à une revalorisation des salaires des fonctionnaires ainsi qu'à une diminution sans délai de la TVA.
(16) Le communiqué rappelle, par ailleurs, les décisions déjà prises en août.
(17) Il a fait valoir les efforts déployés actuellement, en particulier la décision de l'OTAN d'agir par la force...

- *par balles* est associé à *blesser Nhum*, et le sujet du verbe est forcément humain. C'est exactement le cas de la phrase (9) : la construction verbale sur laquelle porte l'adverbe est effectivement celle attendue.

- *en groupe* est associé à *travailler*, et le sujet est forcément humain $N_0 =: Nhum$. Dans la phrase (10), on le trouve avec le verbe *gratter* (une guitare). Il a le même sens (ensemble). Ici, en fait, presque toutes les verbes d'activités humaines pourraient convenir (*manger en groupe*, *faire du jogging en groupe*...).

- *en substance* est associé à *dire N à N* avec un sujet humain. On le trouve dans (11), avec la construction N_0 *estimer que P*. Il s'agit d'un autre verbe d'énonciation.

à

- *nouveau* est associé à *se fâcher* avec un sujet humain. On le trouve associé à *il faisait nuit*. En fait, il pourrait porter sur toute événement susceptible d'être répété.

- *en apparence* est associé à *dormir*. On le trouve dans (14) associé à l'adjectif *anodine*, avec la phrase sous-jacente *Cette disposition est anodine. en apparence* s'applique donc à ce verbe *être* implicite.

- *sans délai* est associé à *se produire*. On le trouve dans (15) associé à *diminution*. Il faut faire réapparaître la relation de nominalisation dans laquelle *diminution* intervient : *procéder à une diminution*.

Comme pour le cadre général, l'analyse de ces adverbes est donc complexe, et nous ne l'avons pas ébauchée. Dans quelques situations, il aurait été possible de reconnaître la construction verbale support exemple associée à chaque entrée, mais cette reconnaissance aurait été marginale. Nous n'avons pas non plus tenu compte des indications sémantiques précisant le sujet de la phrase sur laquelle porte l'adverbe, puisque ces indications sont relatives au verbe donné comme exemple, et non pas intrinsèques à l'adverbe comme nous venons de le voir.

Nous avons supprimé de ces tables, tous les adverbes intervenant dans une négation. Leur description est faite d'une manière plus complète dans le chapitre 6.

Pour certains adverbes, des indications sont données dans la table précisant une lemmatisation possible (tout du moins, une équivalence avec une construction «régulière» comme *de façon Modif*). Dans ces cas, nous n'avons pas seulement reconnu les adverbes, mais nous leur avons attribué comme lemme cette construction «régulière». Ainsi, *en toute candeur* et *candidement* sont reconnus et lemmatisés pareillement : *d'une façon candide*. Pour cela, nous avons dû ajouter dans les tables les adjectifs associés, ceux-ci ne pouvant pas être obtenus par des méthodes entièrement morphologiques.

Nous présentons la reconnaissance des adverbes provenant des tables `P` [10], des tables `A` ([16], [6]), des tables `EPX` ([13] et [1]). Pour ces tables, nous détaillons la reconnaissance de manière suffisamment approfondie pour donner un exemple complet de l'utilisation des graphes patrons.
Nous montrons ensuite des problèmes d'organisation et de traitement de données pour la reconnaissance des dates.
Enfin, nous mettrons en évidence que la mobilité adverbiale est une des causes principales de la non connexité d'unités élémentaires (provenant de la négation, de déterminants, ou de structures verbales). Nous étudierons sur trois exemples les différents cas de figure possibles, et la méthode que nous utiliserons pour résoudre ce problème général de la non connexité.

Les `PADV`

Les `PADV` (voir table 5.1) est une classe dont chaque élément est un mot simple. Leur reconnaissance est donc relativement simple.
Les propriétés que nous avons exploitées sont :

- *très Adv*, la propriété de modification par *très* (par exemple *très communément*) que nous avons étendu aux adverbes *bien*, *fort*, *plus* et *moins* ;

- *tout Adv*, la propriété de modification par *tout* (par exemple *tout dernièrement*) ;

- et *de façon C – a* (*désagréablement*, de *manière désagréable*) où *Adj* est l'adjectif associé à l'adverbe considéré.

Le graphe patron associé à la table PADV est représenté figure 5.1.

Les tables PC, PDETC

La table PC (voir table 5.2) est formée d'éléments de la forme *Prép C*, et la table PDETC (voir table 5.3) est formée d'éléments de la forme *Prép Dét C*.
Les propriétés que nous avons exploitées sont :

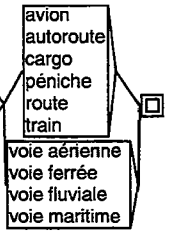
- *de façon C – a* (*par miracle=de manière miraculeuse*) ;
- *C – a ment* (*par miracle=miraculeusement*).

Dans PDETC, certains déterminants sont variables, par exemple :

- *espérer de tout $\underline{Poss_0}$ cœur* (*Il souhaite ta venue de tout son cœur.*) le 0 de $Poss_0$ indique la coréférence du déterminant possessif au sujet ;
- *tirer N en $\underline{Dnum}$ exemplaires* (*Il a tiré ce livre en trois exemplaires.*)
- *payer N en $\underline{Dadj}$ fois* (*Il paye sa voiture en plusieurs fois.*)
- *transporter N par $\underline{(E + LE)}$ Nvoie-de-communication* (*Max transporte son vin par autoroute.*, ou *Max transporte son vin par l'autoroute.*)

Nous avons traité ces différents cas, en introduisant des symboles grammaticaux ou des expressions régulières dans les entrées de la table :

- *espérer de* <*DET*:+Poss+REFPoss0+tout> *cœur* qui fait référence à la reconnaissance des déterminants définis. Le *tout* étant incorporé dans la reconnaissance du déterminant. L'information +REFPoss0 est mise à titre de vérification d'accord avec le sujet. Nous montrerons plus loin comment la vérification de telles contraintes peut-être faite ;
- *tirer N en* <*DNUM*> *exemplaires*, où <*DNUM*> représente toute séquence numérique ou littérale représentant un nombre entier ;
- *payer N en* <*DET*:+Dnum> *fois*, où les déterminants <*DET*:+Dnum> regroupent les déterminants numériques, mais aussi *plusieurs*, *quelques*, *au moins cinq*...
- *transporter N par* (<*le*>+<*E*>):NVoieDeCommunication. La substitution d'une expression complexe comme <*le*>+<*E*> dans le graphe patron suit le même modèle qu'une substitution d'une séquence simple. De la même façon, la séquence :NVoieDeCommunication correspond à un appel de sous-graphe dans un graphe. Dans ce cas là, il s'agit du graphe ci-contre.



Le graphe patron associé à la table PC est représenté figure 5.2, celui associé à la table PDETC est représenté figure 5.3.

V	*ADV*	*C – a*	$N_0 = N hum$	$N_0 = N - hum$	*de façon C – a*	*Adv = où ?*	*Adv = quand ?*	*Adv = comment ?*	*Adv = tout Adv*	*Adv = très Adv*
agir	:Dnumiemement		+	–	–	–	–	–	–	–
être content	doublement		+	–	–	–	–	–	–	–
être content	triplement		+	–	–	–	–	–	–	–
injurier N	abondamment		+	–	–	–	–	+	–	+
venir?	absolument!		+	–	–	–	–	–	–	–
se produire	actuellement		–	+	–	–	–	–	–	–
agir	additionnellement		+	–	–	–	–	–	–	–
vexer Nhum	affreusement	<*affreux.A*>	+	+	+	–	–	–	–	–
surprendre Nhum	agréablement	<*agréable.A*>	–	+	+	–	–	+	–	+

TAB. 5.1 – *Extrait de la table* PADV

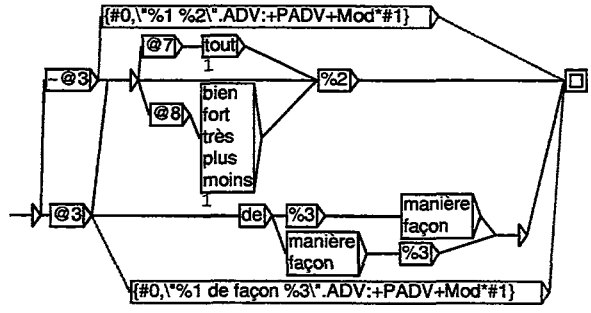


FIG. 5.1 – *Graphe patron pour la table* PADV

V	*Prép*	*C*	*C – a*	*C – a ment*	$N_0 = N hum$	$N_0 = N - hum$	*de façon C – a*	*C – a ment*	*Conjonction*
aimer N	en	:Dnumieme			+	–	–	–	–
perdre N	par	abandon			+	–	–	–	–
injurier N	de	abondance	<*abondant.A*>	abondamment	+	–	+	+	–
injurier N	en	abondance	<*abondant.A*>	abondamment	+	–	+	+	–
être bête	de	abord			+	+	–	–	+
se produire	de	abord			–	+	–	–	+
dire N à N	en	abrégé			+	–	–	–	–
parler de N	in	abstracto			+	–	–	–	–
se produire	en	apparence		apparemment	–	+	–	+	–

TAB. 5.2 – *Extrait de la table* PC

V	*Prép*	*Dét*	*C*	*C − a*	*C − a ment*	$N_0 =: Nhum$	$N_0 =: N - hum$	*de façon C − a*	*C − a ment*	*Conjonction*
transporter N	par	<*le*> + <*E*>	:NMoyenTransport			+	−	−	−	−
faire N	entre	deux	:NMoyenTransport			+	−	−	−	−
venir	à	<*le*>	:NNage			+	−	−	−	−
venir	à	la	brasse			+	−	−	−	−
venir	à	le	crawl			+	−	−	−	−
faire N	sous	<*DET*:+Dnum>	:Ntps			+	−	−	−	−
payer N	à	<*le*>	:Ntps			+	−	−	−	−
payer N	à	l'	année			+	−	−	−	−
transporter N	par	<*le*>+<*E*>	:NVoieCommunication			+	−	−	−	−
faire N	en	toute	abnégation			+	−	−	−	−
se produire Adv	dans	l'	absolu			−	+	−	−	−
démontrer N à N	par	l'	absurde			+	−	−	−	−
faire N	jusqu'à	la	absurde			+	−	−	−	−
coûter:Dnum N	à	l'	achat			−	+	−	−	−
dire N à N	en	toute	affection	<*affectueux.A*>		+	−	+	−	−
se produire Adv	dans	l'	affirmative			−	+	−	−	+
faire trop de N	pour	<*DET*:+Poss+REFPoss0>	<*âge.AN*>			+	−	−	−	−
dire N à N	en	toute	amitié	<*amical.A*>	amicalement	+	−	+	+	−

TAB. 5.3 – *Extrait de la table* PDETC

Les tables PAC et PCA

Ces deux tables sont respectivement formées des éléments de la forme *Prép Dét Adj C* (préposition-déterminant-adjectif-nom figé) et *Prép Dét C Adj*. Le déterminant pouvant être nul (*Dét* =<*E*>).
Dans les deux tables, il y a possibilité pour l'adjectif d'être omis dans certaines constructions *Max est parti à (tout+ϵ) jamais*, ou *Max a fêté ses cent ans dans la (plus stricte+ϵ) intimité*.
Cette possibilité est indiquée dans une colonne *Prép Dét C* que nous utilisons.
Pour les PCA, la possibilité de permutation de l'adjectif notée *Prép Dét Adj C* (*Il est intervenu pour une part importante dans cette histoire= Il est intervenu pour une importante part, dans cette histoire.*) est aussi utilisée.
Certaines entrées font référence au symbole P représentant une phrase entière: *comprendre N dans la mesure où P*, comme dans *Dans la mesure où il a la jambe cassée, je comprends qu'il n'a pas pu venir.* Pour reconnaître un tel schéma, il faudrait avoir décrit tous les schémas de phrases (élémentaires et complexes) possibles. Nous avons donc substitués d'une manière optimiste P par la catégorie <*P*>, mais cela sous-entend que l'analyse syntaxique complète soit effectuée.

Les tables PCDC et PCPC

Ces deux classes sont respectivement formées des éléments de la forme $Prép_1\ Dét_1\ C_1$ *de* $Dét_2\ C_2$ et $Prép_1\ Dét_1\ C_1\ Prép_2\ Dét_2\ C_2$, comme *sur un coup de poker*, et *du simple au double*.
La seule propriété de ces tables, mises à part, $N_0 =: Nhum$ et $N_0 =: N - hum$ est la possibilité d'effacement du $Prép_2\ Dét_2\ C_2$ comme dans *récompenser Nhum pour services rendus (à la patrie+ϵ)*. Le graphe patron de ces deux tables est donc très simple.
Certaines entrées contiennent des catégories générales <*ADV*> comme: *se produire plus* <*ADV*> *que prévu*:

Cette manifestation s'est terminée plus violemment que prévu.

Cette modification de l'adverbe *violemment* est du même ordre qu'une modification par *très*. La mise en œuvre en est cependant plus complexe. Pour les adverbes simples PADV: il faudrait pouvoir utiliser les deux tables à la fois pour reconnaître l'exemple précédent (si on veut vérifier que l'adverbe est modifiable).
Pour des adverbes composés comme *à bon prix*, la structure est modifiée puisque cela va donner la forme *à plus bon prix que prévu* dans laquelle les deux structures sont mélangées (on ne peut pas reconnaître l'une sans l'autre). De plus la réduction *plus bon= meilleur* s'opère également pour donner *à meilleur prix que prévu*[2]. Outre le fait que l'information indiquant la possibilité de modification par *très* n'est pas indiquée (**à très pleine main*); il serait nécessaire, ici, de pouvoir effectuer simultanément les reconnaissances des deux motifs, puisque la structure de chaque motif simple va changer. Ce problème n'a pas été résolu.

2. On peut supposer que l'analyse de cette séquence se fait en remplaçant d'abord systématiquement *meilleur* par *plus bon*.

Les tables PCDN et PCPN

Cette classe d'adverbes regroupe des adverbes *semi-figés*, c'est-à-dire, dont une partie est libre (le N). Cela revient à dire que nous avons affaire à des locutions prépositionnelles figées (par exemple *au nez et à la barbe de*). La reconnaissance de l'ensemble de la séquence adverbiale peut se faire ensuite avec le schéma *Prép GN*. Par exemple, la phrase (18) reçoit l'analyse (19).

(18) Certains se livrent à une course effrénée pour ravir une pièce au nez d'un concurrent.

(19) *Certains se livrent à une course effrénée pour ravir une pièce (au nez de)$_{PREP}$ un concurrent.*

Cette analyse minimale regroupe plusieurs mots, et lui attribue un étiquetage cohérent vis-à-vis de notre forme générale d'adverbe. Les tables PCDN et PCPN décrivent de plus certaines propriétés du complément libre N. En particulier, si :

- N est humain/non humain ($N_1 =: Nhum$, $N_1 =: N - hum$) ;
- N est facultatif (*de* $N_1 = E$ comme dans *réussir pour l'essentiel (de ce problème+<E>)*)
- N peut être possessivé (C_1 *de* $N_1 = Poss\ C_1$ comme dans *combattre sous l'étendard du roi→combattre sous son étendard*)
- et si N peut être remplacé par un adjectif générique (*de* $N_1 =$ *général* comme dans *agir dans l'intérêt de tous→agir dans l'intérêt général*).

Dans tous ces cas, il est possible de reconnaître l'adverbe complet. En fait, il est même obligatoire de le reconnaître à ce niveau, puisque si on se contentait de marquer la partie figée comme étant une préposition, il ne serait plus possible par la suite de «défaire» la préposition pour tenir compte de ces propriétés.
Cela nous conduit au problème suivant : nous reconnaissons *dans l'intérêt général*[3] comme adverbe figé, et pour la séquence *dans l'intérêt de tous* nous ne faisons qu'étiqueter *dans l'intérêt de* comme préposition figée, soit :

(20) De plus, (dans l'intérêt de)$_{PREP}$ tous, il demande que le prix adopté...

(21) (Dans notre intérêt commun)$_{ADV}$, cet accord doit aboutir.

(22) *(Dans l'intérêt général)$_{ADV}$, cet accord doit aboutir.*

Il est tentant de prolonger la reconnaissance à l'adverbe entier dans la phrase (20) pour être cohérent avec l'analyse de la phrase (22). Cet exemple montre bien que vouloir reconnaître les unités figées n'est pas une étape indépendante de l'analyse syntaxique générale : nous sommes en train de reconnaître les adverbes de la forme *dans l'intérêt de* N, nous avons donc accès aux contraintes de sélection sur le N. Il est clair que dans le cadre d'une analyse syntaxique générale, c'est à ce niveau qu'il faut reconnaître l'adverbe complet *dans l'intérêt de tous*. Sinon, il faut tenir compte de cette préposition composée qui entre dans la description régulière des adverbes, mais il faudra alors décrire ailleurs les propriétés de sélection de cette préposition. Cet exemple montre bien la limite de notre reconnaissance : *dans l'intérêt général* est figé puisque sa structure syntaxique n'est pas régulière (*e.g.* **dans l'intérêt très général*), alors que en étiquetant *dans l'intérêt de* comme une préposition, la séquence *dans l'intérêt de* N est «libre» ; la correspondance

3. On peut en fait substituer à *général* tout modifieur «collectif» comme *collectif, français, etc.*. Remarquons qu'il y aurait aussi l'adjectif *public* mais cet adjectif forme un nom composé avec *intérêt*. Le traitement d'un tel chevauchement d'expressions composées doit être décrit par ailleurs.

avec des unités sémantiques n'est donc pas assurée mais cela n'était pas le but initial. À *dans l'intérêt collectif* nous associerons la même valeur «sémantique» que celle calculée dans l'interprétation de *(dans l'intérêt de)$_{PREP}$ tous*. Dans notre optique de normalisation, nous pourrions aussi envisager le remplacement de *dans l'intérêt collectif* par *(dans l'intérêt de)$_{PREP}$ (tous+la collectivité)*. Cependant, comme nous cherchons dans cette phase à reconnaître le maximum d'adverbes pour pouvoir continuer l'analyse de la phrase, nous n'avons pas fait ce remplacement.
Les deux graphes des figures 5.5 5.6 montrent les deux graphes patrons qui correspondent à ces deux utilisations de la table PCDN.

La table PV

Cette classe contient des adverbes phrastiques de la forme :

$$Prép\ V\ W\ =:\ \textit{pour achever le tableau} + \textit{moyennant finance}$$

En voici quelques exemples tirés du corpus.

Pour conclure immanquablement: " J'espère que tu te convertiras et que tu...
"Tiens, comment vas-tu?" "Je vais devenir fou", répond Djeda, ...
* force le libre jeu de la démocratie, à savoir$_{Prép}$ la formation$_N$, au sein de l'Assemblée constituante, ...
Ce qui n'est pas, soit dit en passant, la moindre anomalie d'une dissolution anticipée...

Les deux colonnes VW et WV (*inversion*), indiquent l'ordre respectifs de V et W et sont utilisés.

5.1.1 Quelques chiffres

Nous donnons ici quelques chiffres concernant la reconnaissance des adverbes décrits précédemment. Nous avons aussi indiqué dans le tableau suivant les adverbes *les plus fréquents* de chaque catégorie.

PADV	25 904	même, si, où, bien, encore, alors, non, ainsi, donc, notamment, quand, toujours
PC	4 880	en effet, sans doute, en revanche, par exemple, en fait, de même, par ailleurs, en tout, généralement
PDETC	5 917	en toute solennité, au ralenti, en toute naïveté, en toute souveraineté, en toute tranquillité, en toute modestie, à la fois
PAC	5 357	ainsi donc, tout seul, un petit peu, la :Dnunième fois, autre part à court terme
PCA	4468	aujourd'hui même, là-même, ici même, le plus possible, mieux encore, le moins possible, pour la :Dnumième fois consécutive
PCPC	505	de plus en plus, tout à fait, peu à peu, de moins en moins, de temps en temps, de temps à autre, du jour au lendemain

PCDC	1 141	au cours du:Ntps, tout de même, une fois de plus, au fil de les:Ntps, tout au long de le:Ntps, en tout état de cause
PCDN(<*ADV*>)	4 519	à la fin, toute la vie, de sa part, à son égard, à son époque, au premier niveau, au contraire, à ses yeux
PCDN(<*PREP*>)	3 304	lors de, au cours de, le temps de, au sein de, à la fin de, faute de, en raison de, en faveur de, à propos de
PCPN(<*ADV*>)	493	à travers, en même temps, aussi bien, parallèlement, de préférence, en réaction, en hommage
PCPN(<*PREP*>)	2 600	rapport à, ainsi que, à travers, y compris, contrairement à, en même temps que, en plein, avant même, non sans, conformément à
PV	634	peut-être, en attendant, en passant, n'importe quoi, pour finir, sans rire
PPCO	16	comme par le passé, comme par magie, comme chez:LUI, comme de colin tampon, comme par enchantement, comme par hasard
PVCO	254	comme les autres, comme convenu, comme préalable, comme tout le monde, comme une ruche, comme un poisson, comme un roman

Il faut noter que ces chiffres représentent des occurrences potentielles d'adverbes, non pas les occurrences réelles. Ceci pour deux raisons : la possibilité d'ambiguïté des séquences, et la reconnaissance de parties d'unités plus longues. Par exemple, *en fait* peut être ambigu avec la construction *il en fait*, et *ainsi* apparaît aussi dans *ainsi donc*, il est donc compté deux fois. Nous avons vu que pour «lever ces ambiguïtés», il faut attendre d'avoir été le plus loin possible dans l'analyse de la phrase, et de faire ensuite une projection, ce qui serait prématuré à ce niveau de reconnaissance.

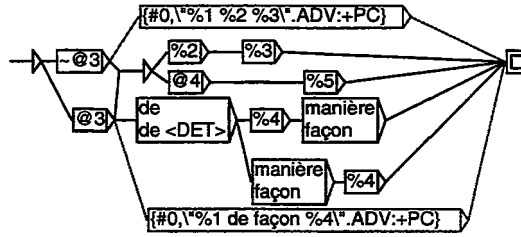


FIG. 5.2 – *Graphe patron pour la table* PC. *Les séquences de manière abondante et d'abondante manière sont aussi reconnues et lemmatisées de la même manière que en abondance ou abondamment.*

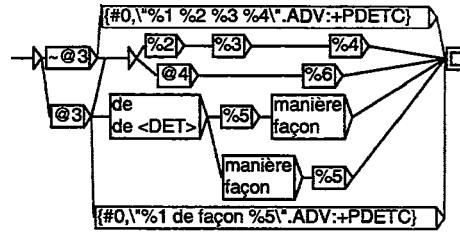
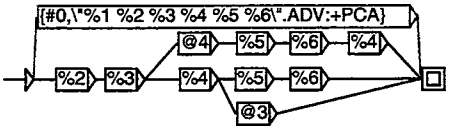
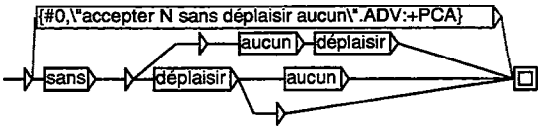
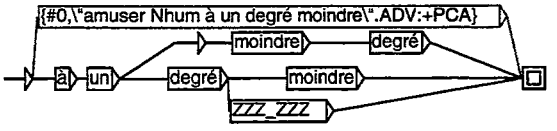


FIG. 5.3 – *Graphe patron pour la table* PDETC



V	*Prép* *Dét*	*C*	*A*	*N₀ =: Nhum*	*N₀ =: N – hum*	*Prép Dét C*	*Prép Dét Adj C*	*Conjonction*
accepter N	sans	déplaisir	aucun	+	–	+	+	–
amuser Nhum	à un	degré	moindre	+	+	–	+	–
être Dnum	à <*DET*>	<*AN*>	près	+	–	–	–	–





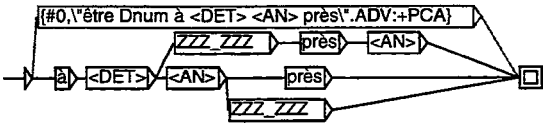


FIG. 5.4 – *Graphe patron, extrait de la table* PCA*, graphes dérivés de la table par le graphe patron. Dans le graphe patron %1 représente le verbe V, %2 la préposition Prép, %3 le déterminant Dét, %4 le nom figé C, et %5 %6 le modifieur Adj. 3 correspond à la colonne Prép Dét C, et 4 à Prép Dét Adj C*

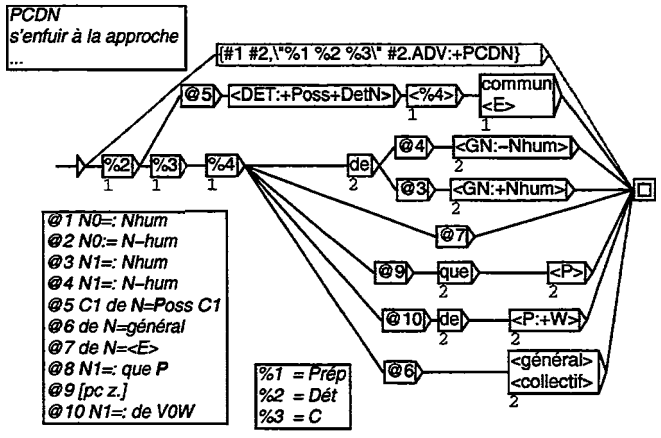


FIG. 5.5 – *Reconnaissance des adverbes décrits par la table* PCDN. *Cette reconnaissance se base sur la reconnaissance préalable des groupes nominaux* <*GN*>, *des phrases* <*P*>. <*P*:W> *représente une phrase à l'infinitif.*

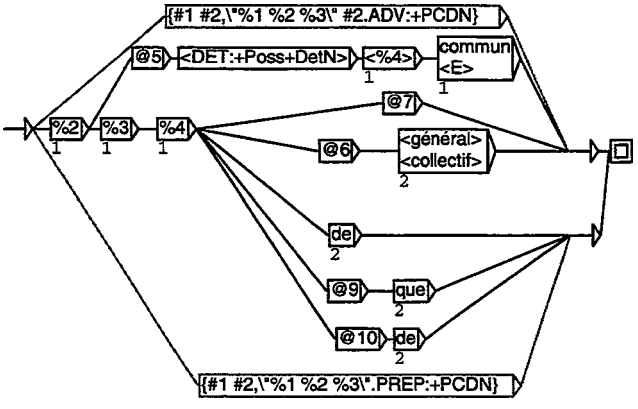


FIG. 5.6 – *Restriction de la reconnaissance aux expressions strictement figées décrites par la table* PCDN *: adverbes figés et prépositions intervenant dans des adverbes libres.*

5.2 Les autres adverbes

5.2.1 Les adverbes provenant des tables EPX

Les constructions décrites dans les tables EPX [13], sont de la forme N_0 *être Prép X*. D'une manière très générale, nous observons la transformation [**être z.**] qui conduit à la forme adverbiale *Prép X* :

(23) A son tour, le Rhône monte et les villes de Valence, Avignon et Arles sont en alerte.

(24) *En alerte, ces villes surveillent le Rhône.*

Nous avons donc recherché les adverbes (et les prépositions pour les tables en DN) provenant des tables :

EPC	en alerte, entre amis, sur bande
EPCA	dans le chaos le plus complet, à l'air libre, au beau fixe
EPCDC	à l'aube d'une année nouvelle, à l'autre bout du monde
EPCDN	à un certain moment de, dans la roue de, au chevet de
EPCPC	en accord avec les observations, en anglais dans le texte
EPCPN	au même point que, à niveau égal avec
EPCPQ	en attente de, dans la certitude de
EPDETC	d'une certaine aide, au chalumeau, à l'agonie

Pour beaucoup de ces entrées, il est possible de modifier le N principal : *entre amis qui se respectent*, *sur bande magnétique*, *en alerte rouge*... Nous n'avons pas traité cette modification pour la forme adverbiale de ces entrées. En fait, le processus général sera analysé dans la reconnaissance des structures verbales figées.
Nous avons laissé de côté la table EPA qui est plutôt une table d'adjectifs composés. Cette table contient *à peine croyable*, qui sera parfois accepté comme adverbe, mais surtout des adjectifs composés comme *afro-asiatique*. Pour des questions de proportion nous n'avons utilisé cette table que pour reconnaître des adjectifs composés.

5.2.2 Les adverbes provenant des tables A

Les constructions en *avoir* provenant des tables A12 (25) et (26), A1P2 (27), A1PC (28), A1PN (29) donnent presque systématiquement des adverbes en *avec*.

(25) *Il a tout faux.*$\longrightarrow$ *?* Avec tout faux, il commence mal l'année...*

(26) *Il a un air affairé.*$\longrightarrow$ *Avec un air affairé...*

(27) *Cet immeuble a trois mètres de haut.*$\longrightarrow$ *Avec trois mètres de haut...*

(28) *Elle a de beaux yeux.*$\longrightarrow$ *Avec (de+ses) beaux yeux...*

(29) *Il a des attentions pour N.*$\longrightarrow$ *Avec des attentions pour N...*

Remarquons que la construction négative associée en *sans* existe mais son déterminant ne se déduit pas trivialement :

(30) *Il n'a pas (?d'+son+l') air affairé.*$\longrightarrow$ *Sans (?ϵ+?un+son+?l') air affairé, il a l'air plus détendu...*

(31) *Il n'a pas le champ libre.*$\longrightarrow$ *Sans (?le+*son) champ libre...*

(32) *Cet immeuble n'a pas trois mètres de haut.*$\longrightarrow$ *Sans (*ϵ+ses) trois mètres de haut...*

(33) *Elle n'a pas de beaux yeux.*$\longrightarrow$ *Sans (de+ses) beaux yeux...*

(34) *Il n'a pas d'attentions pour Luc.*$\longrightarrow$ *Sans (ϵ+?*des+ses+aucune) attention(s) pour N...*

Enfin, il y a possibilité assez fréquente d'effacement de *avec* :

(35) *Il a tout faux.⟶ * Tout faux, il commence mal l'année...*
(36) *Il a la bouche en cœur.⟶ (Avec+ϵ) la bouche en cœur, il arrive.*
(37) *Cet immeuble a trois mètres de haut.⟶ ?* trois mètres de haut, cet immeuble est impressionnant.*

En absence d'une description plus détaillée, nous avons autorisé le possessif et le déterminant zéro d'une manière systématique dans la forme affirmative et négative, et *aucun* dans la forme négative. Nous avons aussi autorisé l'effacement de *avec* et recherché la structure adverbiale basée sur le participe présent : *en ayant*. Nous donnons ici quelques exemples[4] d'adverbes reconnus de cette manière :

(38) Des "conseillers associés" ne pouvaient pas "participer aux travaux du conseil municipal (avec voix consultative)$_{ADV+A12}$".
(39) Sans effort d'imagination et (sans accès à)$_{PREP+A1PN}$ les techniques de reproduction comme la lithographie, beaucoup de peintres se contentaient de reproduire...
(40) 80 milliards de francs avaient été transférés aux entreprises (sans effet sur)$_{PREP+A1PN}$ l'emploi.
(41) 80 milliards de francs ont été transférés aux entreprises (sans aucun effet sur)$_{PREP+A1PN}$ l'emploi (...)
(42) Rester, c'est accepter (la rage au cœur)$_{ADV+A1P2}$, des situations humiliantes pour sauver quand même quelques vies humaines.
(43) Il sera l'homme de la situation (avec une idée en tête)$_{ADV+A1P2}$: "..."

Le nombre d'adverbes reconnus par cette table est de l'ordre de quelques dizaines, mais ceux-ci n'auraient pas pu être localisés autrement. L'exemple (38) montre bien le caractère Zipfien (voir annexe B) de la distribution de ces adverbes. Il y a très peu d'adverbes comme *sans accès à* qui sont relativement répandus, mais par contre, il y a énormément d'adverbes comme *sans voix consultative* qui sont très peu répandus.

La table **ANP2** (*avoir N devant LUI, avoir N sur les bras*) est le premier exemple d'adverbe non connexe que nous devons localiser, puisque les adverbes associés sont de la forme *avec N devant LUI, avec N sur les bras*. La séquence libre intermédiaire *N* est bien décrite : il s'agit d'un groupe nominal, avec une possibilité d'insertion d'adverbe éventuelle : *avec mon frère (toute la journée)$_{ADV}$ sur les bras*. Par contre, il n'y a pas de possibilité d'inversion de *sur les bras* à gauche de la préposition *avec* : **sur les bras avec mon frère*, cela facilite la reconnaissance.
Nous donnons un extrait de la table ((table 5.5), ainsi que le graphe patron utilisé (figure 5.7) qui permettent de localiser ces adverbes, dont nous donnons quelques exemples :

(44) Il quête discrètement, (un morceau de carton en guise d'écriteau, timidement tenu à la main), sur lequel est inscrit ce cruel épitaphe : " J'ai honte, mais j'ai faim.
(45) Les guérilleros (armes automatiques au poing)$_{ADV:+ANP2}$, ont investis plusieurs édifices publics.
(46) ...(en ayant des recrues sous la main)$_{ADV:+ANP2}$...
(47)* Avec neuf heures d'avance sur nous, nos adversaires...
(48) (Avec la totalité des épreuves devant lui)$_{ADV:+ANP2}$, ...
(49) ...(laissant ainsi de côté son professeur d'économie)$_{ADV:+ANP2}$,...

Pour reconnaître ces adverbes, nous avons besoin d'une description des groupes nominaux. De la même façon que pour les **PCDN**, nous avons le choix du traitement de ces

4. Les deux exemples (40) et (41) ont bien été trouvés à deux endroits différents du texte. Et voici un exemple de redondance syntaxiquement non triviale...

V	*Prép*	*Dét*	*C*
avoir	devant		<*LUI*>
avoir	par devers		<*LUI*>
avoir	sur		<*LUI*>
avoir	en		abomination
avoir	en		aversion
avoir	pour	tout	bagage

TAB. 5.5 – *Extrait de la table* ANP2

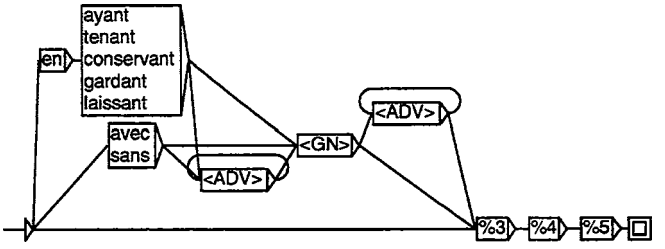


FIG. 5.7 – *Graphe patron pour la table* ANP2

adverbes. Soit nous regroupons la préposition et la partie P_2 pour former une préposition composée (50) et (51) (cf. la transformation du graphe 5.7), soit nous regroupons le tout pour former un adverbe composé, comme dans (44) à (49). Dans le deuxième cas, nous ne sommes plus dans le cadre strict de la reconnaissance d'expressions minimales, puisque ces adverbes sont composés de deux expressions minimales. Dans le premier cas, nous sortons les adverbes du groupe prépositionnel, car ils ne font pas partie de l'unité minimale, mais alors ils deviennent des adverbes de la phrase principale, ce qui est faux. Pour résoudre ce problème, il est indispensable ici de reconstituer toutes les phrases élémentaires .

(50) *...(en ayant sous la main)*$_{PREP:+\text{ANP2}}$ *des recrues...*

(51) *...(laissant de côté)*$_{PREP:+\text{ANP2}}$ *son professeur d'économie*$_{ADV:+ANP2}$ *ainsi,...*

La phrase (44) révèle un nouveau problème : en l'absence de la sous-séquence *timidement tenu*, la séquence *un morceau de carton à la main* serait analysée comme séquence adverbiale. Mais le verbe support est *tenir*, alors que le verbe support attendu est *avoir*. Ceci est un exemple de changement de verbe support, cette variante n'est pas généralisable à la table (voir (52) et (53)), par contre d'autres verbes supports comme *conserver* et *garder* conviennent dans ces exemples.

La phrase (47) ne correspond pas à l'entrée ANP2 de l'adverbe qui est *avec de l'argent sur lui.*

(52) *Il (a+*tient+conserve+garde) son frère sur les bras.*

(53) *Il (a+*tient+conserve+garde) un cigare à la bouche.*

5.2.3 Les adverbes de date, de durée, et de fréquence

Les adverbes de date que nous avons reconnus sont décrits par une bibliothèque comportant plus d'une centaine de graphes. Nous avons montré dans la partie précédente,

comment l'utilisation de valeurs numériques permettait de simplifier et de rendre plus précis une grande partie de ces graphes. Nous montrons ici le graphe de dépendance de ces différents graphes (figure 5.8), ainsi qu'un extrait de l'automate du texte (figure 5.9) sur lequel nous avons activé la représentation des différents sous-graphes, ce qui facilite la mise au point de la bibliothèque.

5.3 La mobilité adverbiale

Les différents adverbes que nous venons de reconnaître sont étiquetés par <*ADV*> et sont mis en fin de phrase[5] par le processus de reconnaissance du schéma de phrase élémentaire. On obtient la forme «canonique» <*P*> <*ADV*>. Nous allons présenter dans le prochain chapitre, des adverbes dont la reconnaissance exigera la vérification de la polarité négative de la phrase ou la présence d'un autre élément adverbial avec lequel ces adverbes se combinent pour former une unité minimale.
Nous présentons ici différents cas de figure, ainsi que la méthodologie de reconnaissance des unités minimales non connexes pour lesquelles il ne sera pas possible de décrire les séquences séparatrices avant la reconnaissance du schéma de phrase complet.

5.3.1 Vérification de contraintes

Considérons l'adverbe «pour autant». Cet adverbe s'emploie dans une phrase à polarité négative[6] :

(54)* *Tout est parfait pour autant.*
(55) Tout n'est pas parfait pour autant.
(56) Le ton, pour autant, ne doit pas ressembler à celui des FM jeunes.
(57) Malkin ne change pas de vie pour autant.
(58) Peut-on pour autant parler de grand retour du blanc.
(59) Pour autant, le marché ne sombre pas dans la tristesse.
(60) ...sans avoir été encadrées par la loi pour autant.
(61) *Pour autant que je sache, il a ensuite pris son train.*

Dans (55) à (60), *pour autant* signifie *malgré cela*. Dans (61), il s'agit d'un autre adverbe qui signifie *dans la mesure où*; on peut le trouver sous les formes réduites: *autant que je sache*, et *que je sache*. Il s'agit là d'un accident qui ne change en rien la contrainte entre *pour autant* et la négativité de la phrase.
La séquence *pour autant* ayant une mobilité complète dans la phrase, elle n'est étiquetable comme adverbe que si on a pu vérifier que la phrase était bien négative. Cette vérification ne se fait pas au moment de la reconnaissance, puisqu'il faudrait déjà avoir analysé le reste de la phrase, analyse qui peut être bloqué par la présence même de l'adverbe.
La stratégie que nous avons adoptée consiste à reconnaître provisoirement l'élément *pour autant*, à l'étiqueter <*ADV*> de manière à réprésenter sa mobilité et à lui ajouter une étiquette `+XI` indiquant que cet élément n'est pas autonome donc pas grammatical. Quand nous aurons effectué les différentes analyses, nous devrons rejeter tous les chemins contenant des éléments non grammaticaux. Examinons ce mécanisme sur les deux phrases suivantes:

5. Ceux qui sont inclus dans les groupes nominaux seront traités localement.
6. Voir la définition page 127.

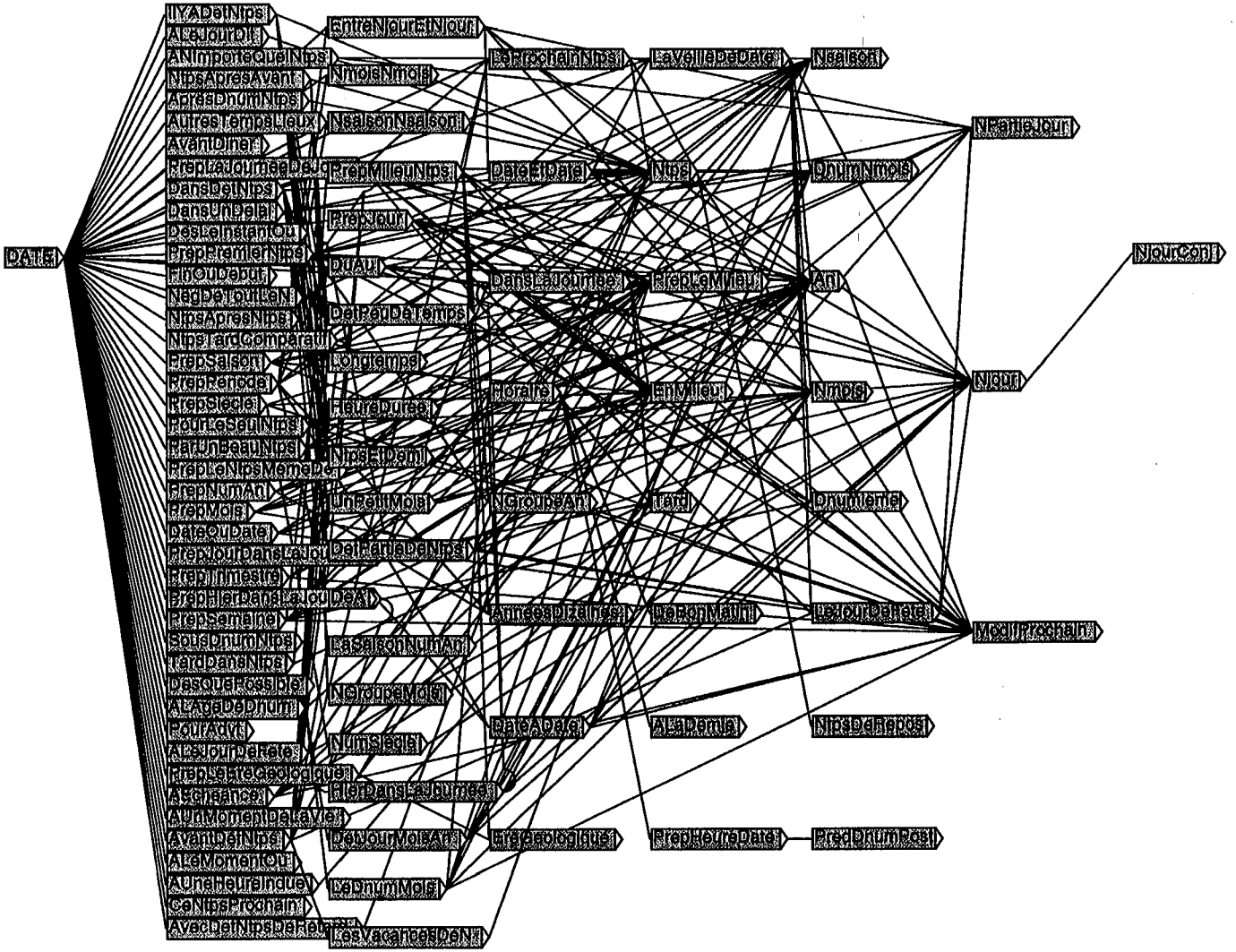
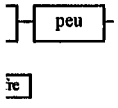


FIG. 5.8 – *Graphe de dépendance du graphe des dates.*



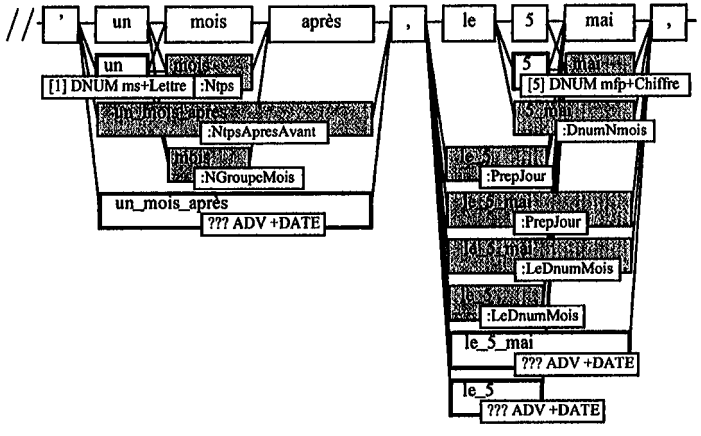


FIG. 5.9 – *Affichage des sous-graphes dans la reconnaissance des dates. Toutes les boîtes grises représentent des sous étapes du calcul. Leur affichage est facultatif.*

(62) *Pour autant que je sache, il prendra ensuite son train.*
(63) *Pour autant, il ne prendra pas son train à l'heure.*

Dans (62), *pour autant que je sache*, *pour autant*, *autant que je sache*, *que je sache* et *ensuite* sont reconnus comme des adverbes (automate du texte de la figure 5.10).

Dans (63) (automate du texte de la figure 5.11), la négation *ne...pas* est reconnue, ainsi que les deux adverbes *pour autant*, et *à l'heure* (qui est ambigu : *arriver à l'heure (ϵ+juste)* et *payer N à l'heure*).
Le schéma de phrase élémentaire N_0 *prendre un train* `C1I` est ensuite reconnu (cf. chapitre 7.2). L'ensemble des adverbes de ces deux phrases est rassemblé à droite des schémas de phrase respectifs. Il est alors possible, à l'aide du graphe de la figure 5.12, de vérifier que l'«adverbe» *pour autant* figure bien dans une phrase comportant une négation. Dans (62), ce n'est pas le cas : donc rien n'est changé, le *pour autant* reconnu porte la marque `+XI`, il ne sera donc pas pris en compte. Par contre, dans (63), le nœud *pour autant* est recréé sans sa marque `+XI`, ce qui valide sa reconnaissance.

5.3.2 Mobilité restreinte

Considérons maintenant le cas de *du tout*, séquence adverbiale associée à la négation *ne...(pas+plus+point). du tout* ne peut pas être interprété isolément, il faut qu'il soit rattaché à la négation associée (*pas du tout* est donc une unité minimale). Parallèlement à l'exemple précédent, il s'agit aussi d'une séquence avec une certaine mobilité qui est cependant restreinte à la droite de la négation comme on le voit sur (64) à (68).

(64) *Léa ne donne pas d'argent à Luc du tout.*
(65) *Léa ne donne pas d'argent du tout à Luc.*
(66) *Léa ne donne pas du tout d'argent à Luc.*
(67)* *Léa, du tout, ne donne pas d'argent à Luc.*
(68)* *Du tout, Léa, ne donne pas d'argent à Luc.*

En raison de cette mobilité, il est tentant de l'étiqueter aussi <*ADV*>, ce qui assurera sa mobilité, et de vérifier ensuite qu'il était bien placé par rapport à la négation. Cela se fait simplement à l'aide du graphe 5.13. Toutefois il n'est pas vraiment légitime d'étiqueter *du tout* par <*ADV*> puisqu'il ne s'agit pas d'une unité complète. En le faisant, nous normalisons néanmoins des phrases comme (65), et (66), dont l'analyse serait autrement bloquée au niveau de cet élément non étiquetable. Nous exploitons donc les propriétés de mobilité de l'adverbe, pour contourner ce problème.

5.3.3 Les adverbes portant sur des groupes nominaux

Considérons le problème posé par les prédéterminants (*Préd*) définis dans [11]. *en tout* en est un. Il s'agit d'une séquence adverbiale complétant un déterminant numérique. On a par exemple :

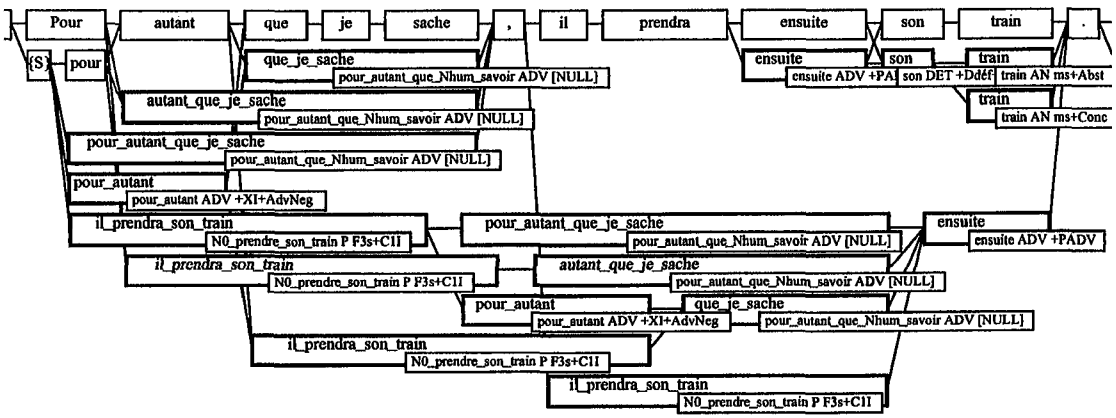


FIG. 5.10 – *Analyse de la phrase : «Pour autant que je sache, il prendra ensuite son train.» Les deux seuls chemins entre le début de la phrase et la fin de la phrase de la forme* <u><P> <ADV>*</u> *sont : «(il prendra son train)$_P$ (pour autant)$_{ADV+XI}$ (que je sache)$_{ADV}$ (ensuite)$_{ADV}$» et «(il prendra son train)$_P$ (pour autant que je sache)$_{ADV}$ (ensuite)$_{ADV}$». La première analyse est éliminée car (pour autant)$_{ADV}$ porte l'étiquette* `+XI`.

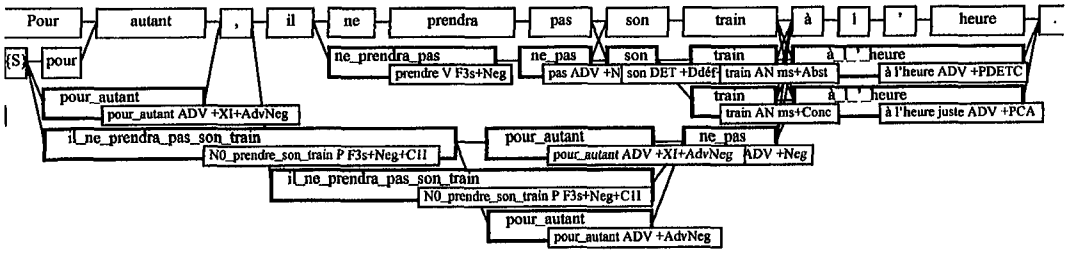


FIG. 5.11 – *Analyse de la phrase : «Pour autant, il ne prendra pas son train à l'heure. » pour autant est regroupé avec la négation ne...pas après reconnaissance de la phrase. La projection des chemins de la forme* <P> <ADV :-XI>* *est «(il prendra son train)*$_{P+Neg=ne...pas}$ *(pour autant)*$_{ADV+Neg}$ *(à l'heure)*$_{ADV}$ *».*

(69) L'ajout coûterait moins de 6 % l'an à l'emprunteur qui, en tout, pourrait lever entre 15 et 20 milliards de francs sur les marchés obligataires en 1994.
(70) Des petites communautés indiennes dans lesquelles vivent environ 5000 Indiens en tout.
(71) Il a en tout perdu plus de 9,8 % en deux jours.
(72) *En tout, il a perdu plus de 9,8 % en deux jours.*
(73) *En tout, j'ai donné lecture de ce poème à plus de 300 personnes.*
(74)* *En tout, j'ai donné lecture de ce poème à mes amis.*

Nous voyons que ce prédéterminant possède une mobilité adverbiale complète, et que cette mobilité ne dépend pas de la position syntaxique du groupe nominal associé : dans (69), ce groupe nominal est N_1, dans (70) il est N_0, dans (72) il est dans $Prép\ N_2$. De plus, *en tout* ne peut modifier qu'un déterminant numérique (comme le montre (74)). Comme *en tout* a une complète mobilité dans la phrase, le regroupement de *en tout* avec son déterminant associé ne peut être fait qu'après avoir reconnu les différents constituants de la phrase.
En supposant que la phrase est mise sous la forme :

$$N_0\ V\ ((Prép)\ N_1\ ((Prép)\ N_2))\ ADV^*$$

le graphe de la figure 5.14 associe les prédéterminants présents dans ADV^* aux déterminants numériques présents.

5.4 Conclusion

Nous avons montré dans ce chapitre comment nous utilisons des ressources existantes au LADL pour reconnaître des adverbes figés, en particulier l'utilisation des graphes patrons pour une application directe des tables du lexique-grammaire décrivant les structures de phrase dont dérivent des adverbes. Nous avons montré la complexité de la résolution de la portée des adverbes, et nous avons vu qu'il n'était pas possible dans l'état actuel de la description de la résoudre. La reconnaissance des adverbes est donc partielle, au sens où nous les avons seulement localisés : l'information qui leur est associée est insuffisante pour qu'un processus n'utilisant pas de ressource extérieure les analyse. Nous avons montré ensuite, qu'à cause de la mobilité adverbiale, plusieurs classes d'unités composées que nous décrivons dans les chapitres suivants sont non connexes. Nous avons indiqué la méthodologie que nous adoptons pour les regrouper. La méthode procède en deux temps : elle crée des éléments non grammaticaux +XI qu'elle étiquette <*ADV*>, ce qui permet une analyse de la phrase. Puis, parmi les <*ADV*> associés à la phrase, nous regroupons les éléments interdépendants en vérifiant les contraintes les concernant.

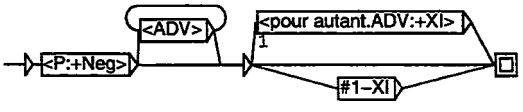


FIG. 5.12 – *Vérification que pour autant est bien dans une phrase négative. Dans ce cas, on rajoute le nœud (pour autant)$_{ADV}$ en parallèle avec le nœud grammatical (pour autant)$_{ADV+XI}$*

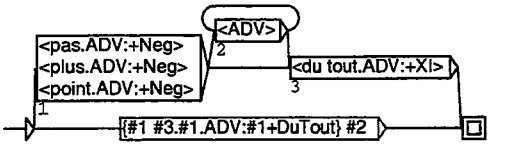


FIG. 5.13 – *Vérification qu'une négation du type ne...pas est associée à du tout : ces deux éléments sont alors regroupés dans la négation.*

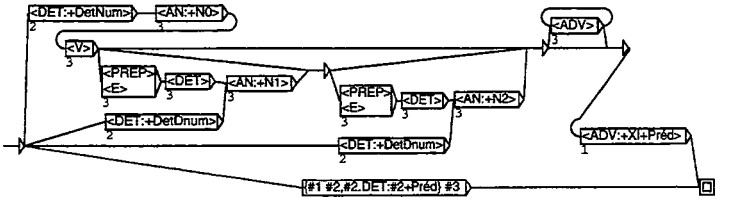


FIG. 5.14 – *Reconnaissance dans un schéma de phrase* N_0 V $((Prép)$ N_1 $((Prép)$ $N_2))$ ADV^* *de la présence simultanée d'un déterminant numérique et d'un prédéterminant, le prédéterminant et le déterminant concernés sont regroupés.*

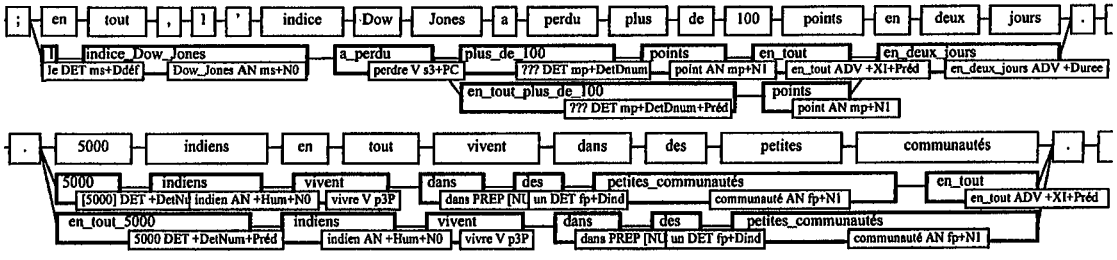


FIG. 5.15 – *Reconnaissance du prédéterminant en tout, dans deux phrases. Dans la première phrase, il s'applique au déterminant du N_1, dans la seconde, il s'applique au déterminant du N_0. La méthode est la même dans les deux cas : reconnaissance du schéma de phrase élémentaire, puis interprétation des adverbes. Nous voyons que dans la deuxième phrase, le prédéterminant était adjacent au groupe nominal, et il a été «repoussé» à la fin de la phrase, pour être ensuite analysé.*

Chapitre 6

Négation

LE TERME DE NÉGATION : «*mot, ou groupe de mot servant à rendre un énoncé négatif*[1] » est ambigu, il peut être vu d'un point de vue logique comme l'opération transformant la proposition P en non(P), d'un point de vue syntaxique comme la transformation qui combine la forme *ne...Nég* avec un verbe, ou d'un point de vue morphologique : comme une unité minimale binaire de la forme *ne...Nég*. Dans notre optique de regroupement d'unités minimales composées, cette dernière définition nous intéresse spécialement. Ces définitions sont loin d'être équivalentes. Par exemple, nous verrons que même pour une négation simplissime de la forme *ne...pas*, il n'est pas toujours évident d'arriver à établir une interprétation logique simple avec l'opérateur non. Avec d'autres négations comme *ne..guère*, la logique n'intervient plus puisque *ne...guère* peut presque toujours être paraphrasé par *peu* dans une phrase affirmative. Nous nous basons donc sur la définition morphologique : une phrase négative est une phrase comportant un *ne* préposé au verbe, en excluant provisoirement les constructions en *ne...que* (*Je ne veux décrire que la négation*) dites **restrictives**, et définitivement les phrases comportant un *ne* explétif (*J'ai peur qu'il ne vienne*) qui relèvent d'une toute autre analyse. Nous montrerons ensuite comment la **restriction** s'intègre naturellement dans ce cadre.

Les adverbes de négation dont nous serons amené à parler peuvent aussi intervenir dans des séquences non négatives, mais à **polarité négative** (par exemple : *Aimerait-il la bière pour autant ?*) que nous ne chercherons pas à décrire ici.

Avec la définition que nous venons d'en donner, la négation simple est naturellement un phénomène composé ; notre premier objectif est donc de rattacher le *ne* à l'élément associé dans la phrase. En plus, de cette «composition naturelle», il existe des adverbes de négations *Nég* composés décrits par M. GROSS dans [14]. Contrairement aux adverbes composés que nous avons décrits précédemment, ces unités composées ne sont pas toujours connexes, ce qui pourra nous donner des séquences en trois parties comme *ne...pas...du tout* qu'il nous faudra regrouper.

La négation a déjà été décrite à différents niveaux [3], [18]. En nous fixant comme objectif de reconnaître la négation indépendamment d'autres phénomènes, nous devons détailler d'une manière encore plus stricte la structure de la négation, sans avoir cependant besoin de considérer son interprétation qui serait une «sur-couche» dans la description. En particulier, nous verrons que les adverbes de négations héritent de certaines propriétés générales des adverbes, dont il faut nécessairement tenir compte pour compléter la re-

1. Négatif : «*Qui exprime une négation.* » (Dictionnaire du français Hachette)

connaissance.

Nous commençons par décrire la négation sous sa forme la plus simple *ne...pas*. Nous considérons ensuite les autres types de négation, et nous montrons les procédés mis en œuvre pour les reconnaître.

6.1 La négation *ne...pas*

6.1.1 Structure

(1) Les déportations ottomanes ne visaient pas les Arméniens.
(2) Les citoyens n'ont pas oublié la sanglante vague d'attentats de 1986.
(3) Il avait demandé aux jurés de ne pas le séparer de sa femme.
(4) Je n'en suis pas sûr.
(5) Cette image ne risque-t-elle pas de donner une fausse idée de la normalité?
(6) Ces questions n'ont, aujourd'hui, pas de solution.
(7) Cet intêret n'est pas très clair.
(8) La loi Falloux ne manquera pas d'intriguer les esprits cartésiens.
(9) ...dans le cas où l'OTAN déciderait de n'y pas participer.
(10) L'emphase de la presse traduit, à n'en pas douter, l'attente réelle...

Dans (1), la négation est sous sa forme la plus simple: elle est composée des deux séquences *ne* et *pas* situées de part et d'autre du verbe principal conjugué à un temps simple. Dans (2) et (3), on voit que la position de la négation dépend en fait de la conjugaison du verbe: la négation est entièrement située avant le verbe à l'infinitif (excepté pour les verbes *être* et *avoir*, voir (15) à (18), où les deux configurations sont permises) et de part et d'autre des formes verbales à un temps fini (i.e. ni participe passé, ni infinitif, ni temps composé). Les phrases (11) et (12) montrent que cette position n'est pas variable. Dans le cas de verbes modaux, la négation peut porter sur le modal comme dans (19) ou sur le verbe principal comme dans (20).

(11)* *Les déportations ottomanes ne pas visaient les Arméniens.*
(12)* *Il avait demandé aux jurés de ne le séparer pas de sa femme.*
(13) *Il remercie les jurés de ne pas l'avoir séparé de sa femme.*
(14) *Il remercie les jurés de ne l'avoir pas séparé de sa femme.*
(15) *Il regrette de n'avoir pas de temps à lui consacrer.*
(16) Il a horreur des piqûres au point de n'être pas à jour de ses vaccins.
(17) *Il regrette de ne pas avoir de temps à lui consacrer.*
(18) *Il a horreur des piqûres au point de ne pas être à jour de ses vaccins.*
(19) *Ils ne peuvent pas venir.*
(20) *Ils peuvent ne pas venir.*

Pour certains verbes *devoir*, *pouvoir*, et *oser*, le *pas* peut être omis[2] (21), (22) et (23). Il faut remarquer (26) que la négation «manquante» ne pourra pas se combiner avec les modifieurs *du tout*, *vraiment*, etc. que nous décrirons plus loin. Par contre dans (24), on voit que cette négation absente peut être factorisée à l'aide d'une coordination pour le verbe *vouloir* qui nécessite la négation (phrase (25)).

2. Uniquement en tant qu'auxiliaire de modalité: **Il ne doit 100 francs à Luc.* Cela fait une dépendance supplémentaire entre la reconnaissance de la négation et la reconnaissance des auxiliaires de modalité.

(21) On ne peut concevoir un espace de vie sans communication véritable.
(22) La valeur totale des titres possédés par l'ensemble des actionnaires ne doit excéder 30 % de ce capital.
(23) Un niveau de présence dont on n'ose croire qu'il se maintiendra durant les trois heures de la pièce.
(24) Les travaux qu'ils ne veulent ou ne peuvent effectuer : ...
(25)* *Les travaux qu'ils ne veulent effectuer : ...*
(26)?* *On ne peut du tout concevoir un espace de vie sans...*

Dans (4), le *ne* est séparé du verbe par un pronom préverbal. Dans le cas infinitif, la règle générale est de mettre les pronoms préverbaux entre le *ne...pas* et le verbe. On trouve cependant des exceptions (9) et (10). (10) est figé, mais il est facile d'obtenir la même permutation du *pas* et du *en* pour une forme libre (29). De même, la permutation du *pas* et du *y* semble générale (27). Le style est cependant plus soutenu. Aucun autre pronom préverbal n'accepte la permutation (28). L'ordre intrinsèque des pronoms préverbaux n'est pas modifié pour autant : les deux phrases (30) et (31) sont donc interdites[3].

(27)? *Il risque de n'y pas goûter.*
(28)* *Il risque de ne (le+les+leur) pas lire.*
(29) *Il risque de n'en pas profiter.*
(30)* *Il risque de ne l'y pas conduire.*
(31)* *Il risque de n'y pas le conduire.*

Enfin, dans (5), le *pas* est séparé du verbe par le pronom sujet du verbe, et dans (6) le *pas* est séparé du verbe par l'adverbe de temps *aujourd'hui*. Une insertion générale d'adverbe est possible :

(32) *Il n'a, sur ce tronc, pas pu graver son nom.*

Ces différents cas sont *structurellement* décrits par le graphe de la figure 6.1.

6.1.2 Portée de la négation, interprétation

L'intuition logique associée à la négation est qu'une phrase portant une négation peut être analysée comme la même phrase sans la négation à laquelle est appliquée l'opérateur logique non(). Par exemple, la phrase (34) s'analyse comme la négation logique de la phrase (33), c'est-à-dire avec une notation pseudo-fonctionnelle : (34)=non(33).

(33) *Luc dort.*
(34) *Luc ne dort pas.*

Cette interprétation de la négation, comme ayant une portée globale sur la phrase, sera toujours valable avec *ne...pas*, mais elle ne fait pas toujours ressortir le sens attendu. Par exemple, il est vrai que (36) implique non(35). Mais, il est peu vraisemblable[4] que (36) soit utilisée pour dire que *Luc n'a lu aucun livre*. Cette interprétation est néanmoins incluse dans non(35). Cela signifie que non(35) n'implique pas (36), c'est-à-dire que (36) n'est pas équivalent à non(35). Ici, on a plutôt l'équivalence entre la phrase (36) et la phrase (37), qui elle n'est pas une phrase négative :

3. Cette remarque montre que les reconnaissances des pronoms préverbaux et de la négation ne peuvent pas être faites indépendamment.

4. L'ambiguïté est toujours possible, la phrase (36) peut dans son contexte se prolonger par *En fait, il n'en a lu aucun.*, mais cela ressort plus d'un jeu de mots.

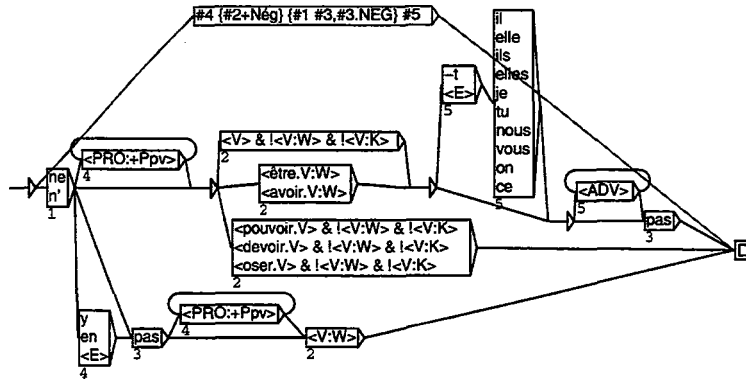


FIG. 6.1 – *Reconnaissance de la négation simple ne...pas. Les pronoms préverbaux sont reconnus dans une phase préalable et sont étiquetés* < *PRO*:+Ppv>. *Nous avons simplifié le lien entre le pronom sujet et le verbe : la présence du −t− est théoriquement liée à la terminaison du verbe, et seul le verbe être peut avoir le pronom ce.*

(35) *Luc a lu beaucoup de livres.*
(36) *Luc n'a pas lu beaucoup de livres.*
(37) *Luc a lu (?* pas beaucoup de livres+peu) de livres.*

On dit alors que la portée de la négation est sur le déterminant du groupe nominal, et (*pas beaucoup*)≡(*peu*)≠non(*beaucoup*).

En fait, la négation peut porter sur presque[5] tous les éléments de la phrase, et les différents sens peuvent être mis en évidence par adjonction d'un membre de phrase introduisant un effet contrastif.

(38) *Aujourd'hui, Luc ne lit pas ce livre sur Victor Hugo.*
(39) *Il le survole.*
(40) *Il le lira demain.*
(41) *Il lit celui de Victor Hugo.*
(42) *Il lit celui sur Charlie Chaplin.*
(43) *Il lit cet autre (sur Victor Hugo).*
(44) *Il lit cet article (sur Victor Hugo).*

Dans chacune des combinaisons possibles, il reste en commun que logiquement, nous avons non(45) :

(45) *Aujourd'hui, Luc lit ce livre sur Victor Hugo.*

Il s'agit de l'information minimale que nous pouvons conserver de la phrase (38).

Chacune de ces interprétations (39) à (44) ne peut être faite que par analyse «sémantique» du contexte. Il est en effet fort possible que la tournure contrastive ne soit pas aussi explicite que dans nos exemples, et donc que l'interprétation contrastive ne puisse pas être détectée syntaxiquement. Chacun des éléments soulignés dans les phrases (39) à (44) montre l'élément de la phrase (38) sur lequel porte la négation :

- Dans (39), *survole* fait contraste avec *lit*;
- Dans (40), *demain* fait contraste avec *aujourd'hui.*

5. Le sujet n'est pas entièrement concerné par ce processus.

- Dans (41), *de* fait contraste avec *sur*;
- Dans (42), *Charlie Chaplin* fait contraste avec *Victor Hugo*;
- Dans (43), *celui-là* fait contraste avec *ce*;
- Dans (44), *article* fait contraste avec *livre*;

On pourrait d'ailleurs reformuler les phrases (38) et (42) par (46) ou (47).

(46) **? Aujourd'hui, il lit ce livre sur # pas Victor Hugo, mais Charlie Chaplin.*

(47) *? Aujourd'hui, il lit ce livre pas sur Victor Hugo, mais sur Charlie Chaplin.*

Cette reformulation montre bien la signification logique d'une portée locale pour la négation.

Il est remarquable[6] que les éléments sur lesquels la négation porte, sont des mots de la phrase, et non pas des unités linguistiques (la préposition *de* (41) par exemple). Cela signifie, que la résolution de la portée ne peut être faite qu'au niveau de la structure de surface. Or nous venons de voir que l'analyse doit être faite à un niveau sémantique d'analyse du contexte, ce qui veut dire qu'une fois que l'analyse syntaxique complète est faite, il faudra revenir à une représentation de surface. La résolution de la portée de la négation est donc une procédure qui se situe à un niveau d'analyse très poussé.

Cette résolution ne concerne pas *a priori* l'étape de reconnaissance de la négation. Cependant, comme nous l'avons vu sur l'exemple (35), *pas beaucoup*, même s'il ne peut pas être utilisé directement comme déterminant, est proche d'un déterminant figé, ce qui signifie que l'étape de reconnaissance doit l'identifier. D'autre part, la phrase (48) (extraite de [11]) n'a pas d'interprétation contrastive:

(48) *Luc n'achète pas de lits.*

Aucune analyse ultérieure de portée de la négation ne doit donc lui être appliquée. Cette situation doit donc être marquée d'une manière ou d'une autre par le processus de reconnaissance.

Nous devrons donc tenir compte de négations dont la portée n'est pas globale, dans la mesure où leur analyse passe par la prise en compte d'éléments lexicaux, c'est-à-dire d'unités minimales figées. Nous verrons en particulier que certaines négations portant sur des adverbes nécessitent un tel traitement, il en est de même pour certains déterminants.

Dans les différents résultats qui seront produits par l'analyse de la négation, ceux pour lesquels la résolution de la portée n'aura pas été faite contiendront l'adverbe <*ADV*:`+Neg+XI`>. L'information `+XI` indiquant que le chemin ne peut pas correspondre à une analyse complète et qu'il faut que l'élément soit regroupé avec un autre.

Le verbe portera les différentes informations suivantes:

- `+Ne`, indiquant que le verbe est précédé d'un *ne*, mais qu'il n'est pas encore associé à une négation (*Il n'achète de lit à personne*). Un verbe comportant un *ne* non résolu est marqué `+XI`;
- `+PNeg`, indiquant que la phrase est négative, sans pour cela que la négation porte sur la phrase (*Il n'achète pas beaucoup de lits*);
- `+Neg`, indiquant que le verbe porte la négation (*Il ne dort pas*).

6. En fait, l'analyse contrastive se fait entre deux phrases; c'est donc un phénomène de *conjonction* au sens syntaxique. Il est bien connu que la conjonction opère sur la surface du texte.

La phrase (49) recevra, suivant ces conventions, les quatre analyses (50), (51), (52) et (53) :

(49) *Luc ne lit pas beaucoup de livres.*

(50) *Luc* {*lit, lire.V* :3Ps+Ne+XI} *pas beaucoup de livres.*

(51) *Luc* {*lit, lire.V* :3Ps+PNeg} {*pas, pas.ADV* :+Neg+XI} *beaucoup de livres.*

(52) *Luc* {*lit, lire.V* :3Ps+PNeg} {*pas beaucoup de, beaucoup.DET* :+Neg} *livres.*

(53) *Luc* {*lit, lire.V* :3Ps+Neg} *beaucoup de livres.*

(51) correspond à une analyse pour laquelle la portée de la négation n'a pas été reconnue. Seules les deux dernières analyses sont des analyses terminales (au sens, où il n'y subsite plus de mots +XI). Ces deux analyses correspondant aux deux interprétations: «il est faux que Luc lise beaucoup de livres» et «Luc lit peu de livres».

6.1.3 Négation et groupes nominaux

Nous présentons trois cas de figure différents d'«interaction» (il ne s'agit pas forcément de portée) de la négation avec des groupes nominaux (ou du moins, avec leur déterminant). Ceci nous permet de définir la notion de déterminant négatif, de déterminant acceptant la négation, et de groupe nominal négatif.

pas...de

Dans la phrase (6) : *Ces questions n'ont, aujourd'hui, pas de solution* le déterminant du groupe nominal en position d'objet direct est *de*. Ce déterminant n'existe pas dans les phrases sans négation ou seulement dans le cadre de quelques transformations.

(54) *Il y a (une+des) solution(s).* $\xrightarrow{\text{[Négation]}}$ *Il n'y a pas de solution(s).*

(55) *Il y a de (le pain+la bière).* $\xrightarrow{\text{[Négation]}}$ *Il n'y a pas de (pain+bière).*

(56) *Un homme lit.* $\xrightarrow{\text{[Négation]}}$ ** D'homme ne lit pas.*

(57) *Je parle à un enfant.* $\xrightarrow{\text{[Négation]}}$ ** Je ne parle pas à d'enfant.*

(58)? *De tel livre, je ne lis plus depuis longtemps.*

(59) *Je ne lis pas, à mon frère, de livre ennuyeux.*

Dans les phrases (54) et (55) le déterminant *de* provient des déterminants *un, une, des, du=de le,* et *de la*. La transformation s'opère obligatoirement sur le groupe nominal en position N_1 direct, comme nous le voyons dans (56) et (57). Les phrases (58) et (59) montrent que l'inacceptabilité de ces phrases n'est pas due à la position des compléments par rapport au verbe.

Remarquons que ce changement de déterminant peut être obligatoire:

(60) Une révolution ne connaissant pas de frontières.

(61)* *Une révolution ne connaissant pas des frontières.*

(62) Ainsi les deux tiers des rames de banlieue n'ont pas de portes automatiques.

(63)*? *Ainsi les deux tiers des rames de banlieue n'ont pas des portes automatiques.*

facultatif:

(64) 9 Français sur 10 ne lisent pas de quotidien national.

(65)? *9 Français sur 10 ne lisent pas un quotidien national.*

(66) Le premier ministre n'a-t-il pas pris un risque?

(67) *Le premier ministre n'a-t-il pas pris de risque?*

ou interdit :

(68) Ce n'est pas un hasard.

(69)* *Ce n'est pas de hasard.*

Nous excluons évidemment dans ces exemples l'interprétation de la négation dont la portée est le déterminant, ce qui est un autre problème. La différence de sens n'est pas perceptible pour (66) et (67), ni pour (64) et (65).

Le fait que certains exemples soient des constructions verbales figées ne change pas l'incertitude de l'application de la transformation. Le verbe a une influence dans l'application : la construction *ce n'est pas de N+s* semblant interdite pour les N libres (**ce n'est pas de livre*), par contre elle est autorisée pour certaines constructions figées (*ce n'est pas de chance* ↔ *c'est de la chance*). Dans nos exemples, la transformation semble plus facultative pour *un* que pour *des*. Nous ne chercherons pas à résoudre ce problème, ce qui nous conduira inévitablement à accepter un peu plus de phrases que les phrases réellement acceptables.

Comme M. GROSS le propose dans [11], nous avons choisi de transformer ce déterminant non usuel en un déterminant "régulier" à l'aide d'une transformation (plutôt que de considérer *pas de* comme un nouveau déterminant). Ce choix ne modifie pas la démarche à établir pour reconnaître le déterminant *de*, mais plutôt l'interprétation qu'on en fait (il faudra dans tous les cas faire le lien avec la présence de la négation *ne...pas*).

Ce changement de déterminant est décrit par la transformation :

$$[\mathbf{T}] : V\text{+}\mathbf{Neg}\ (\text{un+une+des+de le+de la})\ N_1 \rightarrow V\text{+}\mathbf{Neg}\ \text{de}\ N_1$$

Cette transformation s'applique au déterminant d'un groupe nominal objet direct d'un verbe. Nous ne pouvons donc pas appliquer l'inverse de cette transformation sans avoir analysé la phrase. Nous ne pouvons pas non plus faire une analyse locale, puisque le complément d'objet direct n'est pas nécessairement accolé à droite du verbe : il peut être permuté avant le verbe, ou être séparé par un complément quelconque[7].

On pourrait intégrer l'analyse de la négation à celle de l'analyse de la phrase élémentaire, mais cela impliquerait pour chaque graphe décrivant la structure d'une phrase élémentaire de contenir la description de la négation (et nous verrons qu'elle peut être assez complexe). Une autre solution consisterait à considérer *de N* comme un groupe nominal régulier. Cette solution engendrerait beaucoup d'erreurs. La solution que nous avons adoptée consiste à effectuer la transformation inverse $[\mathbf{T}]^{-1}$ sur toutes les séquences *de N* du texte, en ajoutant des informations sémantiques qui agiront comme des contraintes à vérifier ultérieurement. Le graphe de la figure 6.2 contient cette transformation dont nous montrons l'utilisation sur la phrase (70) (automate du texte de la figure 6.3).

(70) (...) et ne pas prononcer de paroles de sévérité, ...

Sur cet exemple, il y a trois groupes nominaux potentiels dérivant de la transformation [T] : *de paroles*, *de sévérité*, et *de paroles de sévérité* en supposant que *paroles*, *sévérité*, et *paroles de sévérité* sont reconnus comme des AN dans une analyse précédente. *ne pas prononcer* est analysé comme le verbe *prononcer* portant la négation *ne...pas* dont la portée est encore indéterminée. L'automate reconnaissant l'expression figée (voir page 162) *prononcer des paroles* reconnaît *de paroles* comme objet direct du verbe *prononcer*, ce qui résout le problème de la portée de la négation. Il est est de même pour *de paroles de*

7. Adverbe, ou groupe prépositionnel *Prép* N_2.

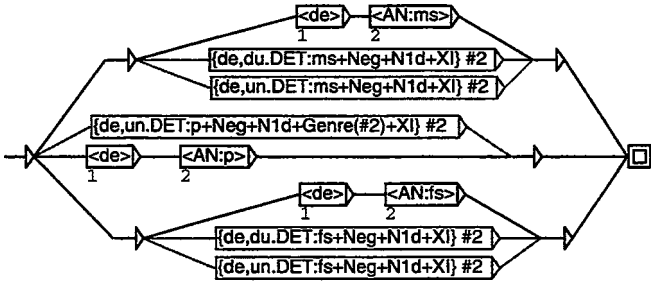


FIG. 6.2 – *Reconnaissance d'un groupe nominal potentiellement de la forme de N_1. La transformation $[\mathbf{T}]^{-1}$ est appliquée et les contraintes* `+N1d` *et* `+PNeg` *sur la nature du groupe nominal sont introduites sous forme d'information au déterminant. Les déterminants ainsi reconnus portent aussi l'information* `+XI` *indiquant que leur traitement est incomplet.*

sévérité. Par contre, *de sévérité* ne reçoit pas d'autre traitement, puisqu'aucune analyse[8] ne peut le faire apparaître comme complément d'objet direct d'un verbe portant une négation.

Les négations portant sur les déterminants

Dans le cas de *beaucoup* (35), nous avons montré que la négation portait sur le déterminant adverbial *beaucoup* pour former l'élément lexical *pas beaucoup* qui signifie *peu* et dont le sens ne se calcule pas directement à partir de celui de *pas* et de *beaucoup*. Dans cet exemple, *beaucoup de livres* est N_1. La séparation physique des deux éléments *pas* et *beaucoup* est possible :

(71) *Luc ne lit pas beaucoup de livres à son frère.*

(72) *Luc ne lit pas, à son frère, beaucoup de livres.*

(72) est, en effet, acceptable sans changement de sens. Nous sommes donc dans la situation d'une unité élémentaire non connexe que nous devons regrouper. Remarquons que le comportement est le même pour les compléments indirects mais pas pour le sujet :

(73) *Luc n'a pas lu cette lettre à beaucoup d'amis.*

(74) *Pas beaucoup d'entre eux ne l'ont donc comprise.*

Dans ce cas, le *pas* passe en début de phrase, mais peut revenir en position «normale» (75) si le déterminant adverbial (comme *tous*) est extrait du groupe nominal :

(75) *Ils n'ont pas tous compris cette lettre.*

Rappelons que la négation peut s'appliquer à n'importe quel déterminant mais que le résultat n'est pas forcément une nouvelle entrée lexicale. Par exemple, *pas un groupe* s'interprète comme la négation logique de «un groupe» dans :

(76) *Je n'ai pas lu cette lettre à un groupe d'étudiant.*

D'autres déterminants n'existent qu'en présence d'une négation comme *l'ombre de* (77) et (78) :

8. Cela pourrait être le cas si la phrase était *De sévérité de sa part, je ne voulais pas !*. Il n'y a donc pas de moyen *a priori* de supprimer cette analyse locale.

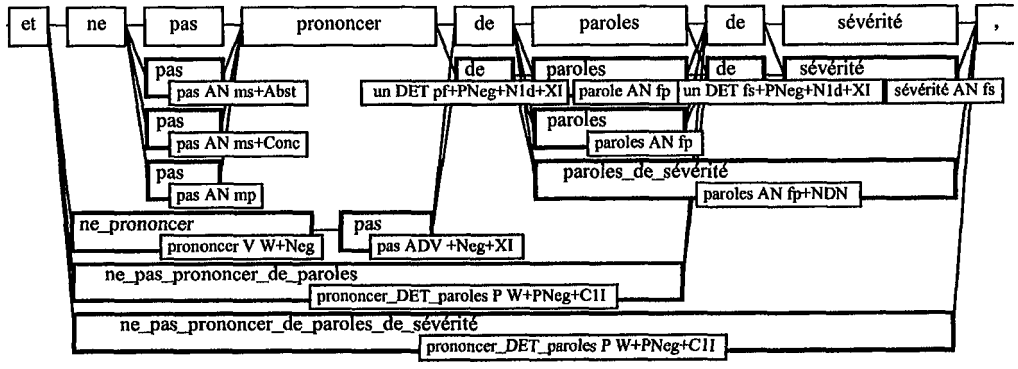


FIG. 6.3 – *Sur cet automate de texte, les séquences de N sont transformées en groupes nominaux de la forme DET AN. Au moment de la reconnaissance du schéma de phrase complet : l'analyse de de sévérité s'arrête puisqu'il n'y a pas de verbe portant une négation dont il est l'objet direct. On remarque aussi que la séquence prononcer de paroles n'est pas reconnue, puisque dans cette séquence le verbe ne porte pas la négation.*

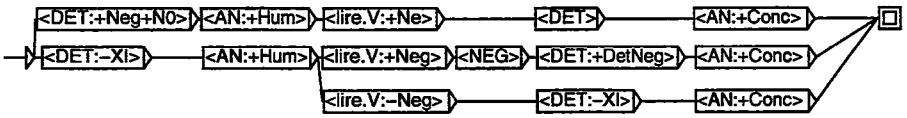


FIG. 6.4 – *Reconnaissance du schéma de phrase N_0 lire N_1. Il y a quatre chemins possibles : négation sur N_0 et verbe portant le ne, négation sur N_1, négation sur la phrase et aucune négation.*

(77) *Je n'ai pas vu l'ombre d'un cafard.*
(78)*?* *Pas l'ombre d'un cafard n'a été aperçue.*

Dans le processus de reconnaissance des déterminants, nous avons donc différencié les déterminants formant une entrée lexicale nouvelle avec la négation, comme *beaucoup*, *la moitié*, *trop*, par le biais d'une étiquette **+DetNeg** sur le déterminant signifiant qu'ils peuvent accepter une négation, et de l'étiquette **+Neg** signifiant qu'ils portent déjà la négation. L'étiquette **+N0** signifie que le groupe nominal concerné ne peut figurer qu'en N_0 (comme *pas beaucoup*).

Les graphes de reconnaissance des schémas de phrase élémentaire vérifient ensuite la cohérence entre les différentes étiquettes : en particulier, si un groupe nominal a été analysé comme <*DET*:+Neg+N0+XI> <*AN*>, alors il entre nécessairement dans un schéma de phrase élémentaire dans lequel le verbe porte «une demi-négation», c'est-à-dire seulement le *ne*, <*V*:+Ne>. Le graphe de la figure 6.4 est un extrait du graphe de reconnaissance du schéma de phrase de la construction N_0 *V* N_1 reconnaissant la négation sur son sujet, sur son complément d'objet direct, ou sur la phrase globale.

Les déterminants intrinsèquement négatifs

Les déterminants comme *aucun*, ou *nul*, ainsi que les «pronoms» *personne* et *rien* sont intrinsèquements négatifs, dans la mesure où ils sont accompagnés du *ne* porté par le verbe. En fait, *pas beaucoup* en N_0 est du même type. Le traitement des groupes nominaux comportant ces formes doit donc vérifier la présence du *ne*. Ces groupes nominaux mis sous la forme canonique <*DET*> <*AN*>, porteront sur leurs déterminants les informations **+Neg** et **+XI**.
Remarquons que le caractère négatif d'un groupe nominal peut aussi provenir d'un de ses modifieurs comme nous le voyons dans (79). La tournure semble cependant maladroite dans le cas général (80) :

(79) *Je ne vois (le gardien d'aucun de ses immeubles)*$_{GN+\texttt{Neg}}$.
(80)*?** *Un gardien d'aucun de ces immeubles (n+ϵ) est passé ce matin.*

Récapitulatif

Nous avons décrit trois types différents d'interaction entre la négation et les groupes nominaux. Nous pouvons distinguer les contraintes de position issues de cette interaction (**+N0**, **+N1d**...) des contraintes portant sur la négativité de la phrase. Les premières

contraintes ne sont pas spécifiques de la négation : certains groupes nominaux comme *à demi cette pomme* ne peuvent pas être en position N_0. Nous récapitulons les deuxièmes types de contraintes dans le tableau suivant, en explicitant l'opération à effectuer pour valider la contrainte.

aucun, pas beaucoup, nul	+Neg+XI	Le déterminant est négatif, le verbe ne doit porter que le *ne*. On vérifie <V:+Ne>
de	+PNeg+XI	Le déterminant ne porte pas la négation, mais n'est présent que dans une phrase négative. On doit donc vérifier <V:+Neg> ou <V:+PNeg>.
beaucoup	+DetNeg	Le déterminant *peut* porter la négation (c'est à dire, que si la négation porte sur celui-ci, il deviendra une autre entrée lexicale). Donc si on a <V:+Neg>, regroupement avec la négation.

6.2 Les autres négations

Nous avons développé la négation en *ne...pas*. Nous montrons maintenant les différences de comportement d'autres négations comme *ne...plus*, *ne...jamais*, *ne...point*, et *ne...aucunement* mais aussi de type nominal comme *rien*, *aucun* ou *personne*.

6.2.1 *ne...plus*

Toutes les propriétés que nous avons mises en évidence pour le moment sur *ne...pas* sont conservées pour *plus* (sauf l'effacement de *pas* dans les phrases à auxiliaire de modalité comme *il ne peut pas venir*, qui n'est pas permis pour *plus*). Cependant la combinaison de *plus* avec une autre négation peut avoir une interprétation différente. Considérons par exemple, les phrases (81) et (82) :

(81) *Aucun homme n'est plus à l'abri.*

(82) *?* Aucun homme n'est pas à l'abri.*

Dans (82), il y a deux négations et l'interprétation devrait être *tous les hommes sont à l'abri*, par contre la phrase (81) a comme interprétation première *plus aucun homme n'est à l'abri*, ce qui est le contraire. Nous retrouverons le même comportement pour *jamais* en combinaison avec toutes les autres négations (sauf *pas*).

Pour tenir compte de ce phénomène, il faut donc considérer que *plus aucun* est une unité minimale. Le regroupement se fait après analyse de la phrase, de la même manière que le regroupement de *pas* et *beaucoup* que nous avons détaillé précédemment. Nous ne traitons pas les doubles négations de type (82).

6.2.2 *ne...jamais*

Il faut d'abord différencier une ambiguïté possible sur le mot *jamais* non employé dans un contexte négatif, comme dans (83). Ce sens ne nous intéresse pas ici :

(83) L'information parvient toujours trop tard lorsqu'elle parvient jamais au simple citoyen.

jamais présente une variante essentielle par rapport à *ne...pas*: *jamais* a une certaine mobilité :

(84) Jamais il n'y fut répondu.

(85) Ils garderont le souvenir d'un journaliste fébrile qui jamais n'éprouvait la tranquilité d'être arrivé au bout de la route...

Cette mobilité est cependant réduite à une position fixée en début de phrase (86). Dans le cas d'une relative, la position est en début de relative, après le pronom relatif. En cas d'inversion du sujet (87)[9], la seule alternative est toujours le début de la phrase. On voit dans (88) et (89) qu'il y a cependant une possibilité de permutation avec des compléments circonstanciels (en début de phrase), avec éventuellement la même distinction de sens entre la négation sur la phrase, et la négation sur un complément introduisant un effet contrastif.

(86)?* *Il n'y fut répondu jamais.*

(87) *Jamais ne sont arrivées en Alsace les hirondelles...*

(88) *Par courrier, jamais il n'y fut répondu.*

(89) *Jamais, par courrier, il n'y fut répondu : il n'y fut répondu que par téléphone.*

6.2.3 *ne...point*

(90) Il demanda à n'en point parler.

(91) Richard Dreyfuss et Emilio Estevez reprennent donc leur numéro de duettistes, auquel point n'a été besoin de changer une seule réplique.

point possède une faculté de permutation (stylistique?) comme un adverbe normal ((92) et (93)). Cette faculté de permutation n'est pas celle de mobilité adverbiale décrite pour *jamais* comme nous le voyons dans la phrase (94) :

(92) *Les hirondelles sont arrivées en Alsace.→En alsace sont arrivées les hirondelles.*

(93) *Les hirondelles ne sont point arrivées en Alsace.→Point ne sont arrivées en Alsace les hirondelles.*

(94)* *Point, les hirondelles ne sont arrivées en Alsace.*

point semble porter systématiquement sur la phrase, puisque les deux phrases (95) et (96) ne sont acceptables qu'en marquant une pause avant *beaucoup* pour supprimer l'interprétation éventuelle *point beaucoup.*

(95) *Je n'ai point lu # beaucoup de livres.*

(96) *Je ne lis point # beaucoup de livres.*

Nous avons donc marqué systématiquement **+PNeg** les phrases contenant *ne...point.*

6.2.4 *ne...(aucunement+nullement)*

nullement et *aucunement* ont une mobilité aussi limitée que celle de *pas*, ils ne modifient pas le déterminant de l'objet direct :

9. Justement permise par *jamais.*

(97)* *Il ne lit aucunement de livres.*

Ils sont sémantiquement équivalents aux négations composés *en rien, en aucun cas, d'aucune manière*, etc. Ces derniers adverbes possèdent cependant, une mobilité complète dans la phrase :

(98) La conservation dans le bois ne convenant nullement à ces vins septentrionaux.

(99) En aucun cas, ce projet ne devra retarder leur entrée dans l'OTAN.

6.2.5 *ne...guère* et *ne...plus guère*

plus guère est un adverbe de négation composé signifiant *presque plus*. *guère* et *plus guère* sont donc deux adverbes différents. Nous avons considéré que *guère plus* avait les propriétés de *plus*.

Par contre, *guère* a un comportement proche de la forme *pas beaucoup* (et de plus signifie *pas beaucoup*). Cette «approximation» permet de rendre compte des phrases (100) à (103) dans lesquelles on retrouve le même comportement adverbial/déterminant que *beaucoup* dans (100) et (101). Cela explique, en plus, l'impossibilité que *guère* porte sur *ses amis* dans la phrase (103), la combinaison de *guère* avec *d'entre eux* est unique parmis les adverbes de négation, l'utilisation en N_0 dans (105), et la reprise par *pas beaucoup...non plus* dans (106).

(100) *Il n'a guère dormi.*

(101) *Il n'a guère mangé de soupe.*

(102) *On ne peut guère lui faire confiance.*

(103)* *Il n'a (guère+pas beaucoup) lu cette lettre à ses amis.*

(104) *Je ne vois (guère+*pas+*point+plus) d'entre eux.*

(105) *Guère de monde ne traîne à cette heure là.*

(106) *Il n'a guère lu de livres. Il n'a (pas beaucoup+*pas+guère) fait d'exercice non plus.*

Le traitement de ces deux adverbes s'est donc fait par substitution d'un adverbe par un autre par le graphe de la figure 6.5. Nous effectuons cette substitution au niveau même des formes.

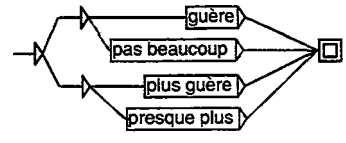


FIG. 6.5 – *Substitution des adverbes de négation contenant guère par des adverbes équivalents.*

6.2.6 *ne...que*

(107) Cette exigence ne concerne que les personnes qui auront dix-huit ans à compter du 1 janvier 1994.
(108) Cela a permis à Citroën de ne reculer que de 16 %.
(109) Ce gigantesque no man's land ne doit sa survie qu'à un projet de construction sur le site d'un hôpital, toujours repoussé, jamais réalisé, puis définitivement abandonné.
(110) Le nombre des personnes temporairement évacuées n'est jamais que l'équivalent de la population d'une petite ville.
(111) Il ne nous reste plus qu'à prier, pour que nos frères du Nord trouvent la paix dans leurs cœurs.
(112) *Luc n'est que (blessé+désagréable).*

Il était difficile de ne pas traiter *ne...que* dans le cadre de la négation. Logiquement, il ne s'agit pas moins d'une négation que *ne...guère*. La portée de la restriction peut être un groupe nominal ou un adverbe (107) à (111), mais aussi un adjectif ou un participe passé comme dans (112). La portée est plus difficilement que pour la négation un sous-constituant de la phrase. La position du *que* est nécessairement à droite du *ne*, ce qui fait que la phrase (114) n'est pas possible contrairement à la phrase (115). Syntaxiquement, la transformation [T] est acceptable (pour un complément autre que celui sur lequel porte la restriction). Nous rappelons que cette transformation était définie dans le cadre des négations. Par contre, la combinaison avec les adverbes de conjonction *pour autant* et *non plus* n'est pas possible (118) et (119).

(113) **? Luc ne lit un livre que de Victor Hugo.*
(114) **? Que cinq personnes ne sont venues.*
(115) *Ne sont venues que cinq personnes.*
(116) *Il n'a fait de voyage qu'en Europe.*
(117) *Il ne mange de soupe que rarement.*
(118) **? Il ne dit que des âneries pour autant.*
(119) * *Il ne reçoit que le mardi. Il ne fait des visites que le jeudi non plus.*

Dans un premier temps (voir figure 6.6), nous reconnaissons le *que* que nous marquons comme adverbe `+XI` (dans le seul but de permettre sa mobilité pour continuer l'analyse de la phrase). Et de la même façon que pour *pour autant*, nous vérifions, après analyse du schéma de phrase, qu'il est bien associé à un verbe portant le *ne* (<*V*:+Ne>, ou <*V*:+Neg>, ou <*V*:+PNeg>). Nous montrons ci-après le déroulement de l'analyse de *Il ne fait ses voyages qu'en Europe*. Les séquences soulignées dans chaque ligne indiquent les sous-séquences sur lesquelles vont porter les opérations à cette étape.

Il ne fait ses voyages qu'en Europe.
→ Il (fait)$_{V+3ps+Ne+XI}$ ses voyages (qu)$_{ADVC+XI+Que}$ en Europe.
→ (Il)$_{N_0}$ (fait)$_{V+3ps+Ne+XI}$ (ses voyages)$_{N_1}$ (qu)$_{ADVC+XI+Que}$ (en Europe)$_{ADV}$.
→ (Il)$_{N_0}$ (fait)$_{V+3ps}$ (ses voyages)$_{N_1}$ (ne que)$_{ADV+NeQue}$ (en Europe)$_{ADV}$.

Remarquons que cette analyse qui passe par un adverbe, n'est pas complètement satisfaisante d'un point de vue formel, mais elle conduit au résultat voulu: regrouper *ne* et *que*. De plus, elle perd l'information de portée de la restriction, en effet, *que* traité comme adverbe sera mis en fin de phrase par le processus de reconnaissance du schéma de phrase. Cependant, nous pouvons comparer *ne...que* à *seulement* [19] qui a la même fonction restrictive, et qui subira naturellement, dans l'état actuel de la description, cette délocalisation: nous avons vu en 5 que la résolution de la portée des adverbes était loin

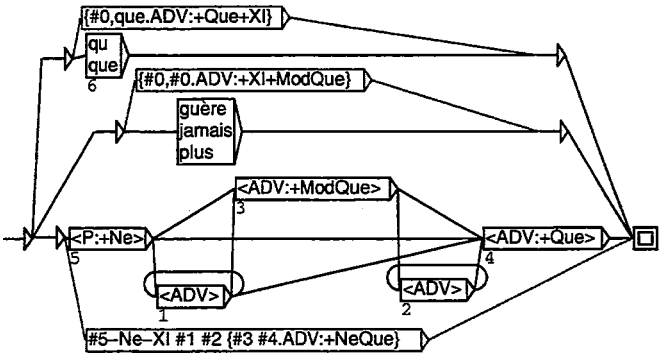


FIG. 6.6 – *Reconnaissance de la Restriction en deux étapes : reconnaissance du que que l'on note adverbe pour lui permettre une mobilité adverbiale, du ne sur le verbe, et du modificateur de Restriction potentiel noté* **+ModQue**. *Les trois parties séparément sont notées* **+XI**, *elles sont regroupées dans une deuxième étape, après analyse de la phrase.*

d'être réalisée.
La restriction forme de plus les composés suivants : *guère que*, *jamais que*, *plus que* (121), *rien que* (120), et *(rien+personne) d'autre que*. Le traitement de la composition avec *rien* et *personne* se fait dans la description de ces deux formes (d'autant que celles-ci demandent déjà le *ne* sur le verbe). Par contre le regroupement avec *jamais* doit être spécifique, puisque les combinaisons n'ont pas la mobilité des adverbes de négation décrits précédemment (122) et (123).

(120) *Il ne fait ses voyages (rien+ϵ) qu'en Europe.*
(121) *Il ne fait (jamais+guère+plus) ses voyages qu'en Europe.*
(122) *Il ne fait ses voyages (jamais+guère+plus) qu'en Europe.*
(123)* *Jamais, il ne fait ses voyages qu'en Europe.*

6.2.7 Négations composées

(124) Beaucoup n'imaginent même pas que cela soit possible.
(125) *Beaucoup imaginent même que cela (? soit+est) possible.*

Dans la phrase (124), l'interprétation ne peut pas être faite en dissociant *même* de *pas* : en effet, nous n'avons *même pas* l'implication (124)⇒non(125). Il n'y a aucun lien logique entre ces deux phrases. Ce qui prouve que *même pas* est une unité minimale : un adverbe de négation composé.

Dans cet exemple, aucune insertion n'est possible entre *même* et *pas*. Il existe d'autres séquences comme *pas du tout* présentant une caractéristique analogue de non-compositionnalité, mais dont la «fixité» syntaxique est moindre que celle de *du tout* ou de *pour autant*. Il faut cependant faire la différence entre le caractère figé d'une séquence et l'interaction possible entre plusieurs éléments d'une phrase. Considérons par exemple l'accord déterminant/nom. Un nom pluriel ne peut pas être employé sans déterminant pluriel. On ne dit pas que l'ensemble est figé. Il semble qu'il y ait le même phénomène pour la

négation et pour *pour autant*. Cette séquence ne peut apparaître que dans une phrase négative, mais semble posséder une autonomie syntaxique à la manière d'une conjonction. Quelques arguments non sémantiques pour l'affirmer existent. En particulier, *pour autant* est acceptable dans le contexte de n'importe quelle négation, contrairement à *du tout*:

(126) *Pour autant, (personne ne l'a deviné+il ne peut (ϵ+pas+jamais+vraiment pas+guère)) venir.*

(127) *?? personne ne l'a deviné du tout.*

(128) *?? Il ne peut du tout se déplacer.*

(129) **? Il n'a (guère+jamais vraiment) téléphoné du tout.*

La «portée» de l'adverbe *du tout* (on peut dire que *du tout* est un modifieur de la négation) est clairement la négation, celle de l'adverbe *pour autant* est la phrase.

Nous distinguerons donc *pour autant*, *non plus*, ou *de mémoire de N* comme étant des adverbes n'apparaissant que dans des phrases à polarité négative, mais indépendants de la négation, leur localisation et la vérification de la contrainte de «négativité» se fait de la manière décrite page 118.
La portée de la négation sur certains adverbes (par exemple *souvent* ↔ *pas souvent*), ou sur des comparatifs produit aussi des effets de «figement». Nous les étudierons plus tard, en les différenciant des «vraies» négations composées décrites ici.

Nous avons reporté dans le graphe de la figure 6.7, différentes négations composées répertoriées essentiellement dans [10]. Ces adverbes de négation peuvent être décomposées en trois zones : *Modifieur gauche* (*toujours+vraiment+absolument+même+presque+quasiment*), *Négation* (*pas+plus+jamais+ plus jamais+jamais plus+point+guère+plus guère+pas même+toujours pas+ pas encore*), et *Modifieur droit* (*du tout+le moins du monde+outre-mesure+des masses+trop+...*).
Les relations entre *Modifieurs droit* et *Modifieur gauche* sont indépendantes de la négation : **presque pas vraiment, quasiment pas du tout.* Les Modifieurs droits ne sont pas séparables de la négation, mais les Modifieurs gauches peuvent être mobiles: *il n'a pas mangé du tout.* Nous donnons dans la table 6.1 les modifieurs autorisés dans chaque cas, et nous indiquons également la possibilité de mobilité globale de la négation (suivant le modèle de *jamais*: *Plus jamais, je n'y retournerai.* et **Plus je ne veux en entendre parler.*).

6.2.8 Autres négations non compositionnelles

n'en V pas moins

La construction *n'en V pas moins* de la phrase (130) est décrite par le graphe de la figure 6.8.

(130) *Il n'en félicite pas moins le chef du gouvernement.*

Portée de la négation sur un adverbe

L'interprétation de la négation accompagnée de certains adverbes relève clairement d'unités minimales composées: nous avons vu l'exemple de *beaucoup* dans sa fonction de déterminant et de *pas beaucoup*. Il en va de même pour *beaucoup* dans sa fonction d'adverbe (131) et (132). Nous avons déjà vu que l'analyse de *beaucoup* passe par une

ADV	*toujours ADV NEG*	*vraiment ADV NEG*	*absolument ADV NEG*	*même ADV NEG*	*presque ADV NEG*	*quasiment ADV NEG*	*ADV NEG trop*	*ADV NEG vraiment+...*	*...le moins du monde*	*...du tout*	*...outre-mesure+des masses*	Mobilité *jamais*
pas	+	+	+	+	+	+	+	+	+	+	+	−
plus	−	+	+	+	+	+	+	+	−	+	+	−
jamais	−	+	+	−	+	+	+	+	−	+	+	+
plus jamais	−	+	+	+	+	+	+	+	−	+	+	+
jamais plus	−	−	−	+	+	+	+	+	−	+	+	+
point	+	−	−	−	−	−	−	−	−	−	+	+
pas même	−	−	−	−	−	−	−	+	+	−	−	−
pas toujours	−	−	−	+	−	−	+	+	−	−	+	+
pas encore	+	−	−	+	+	+	+	+	−	+	+	−

TAB. 6.1 – *Description des négations composées*

construction unique où il a office de déterminant (page 27).

(131) *Il n'a pas dormi cette nuit beaucoup.*

(132) *Il n'a pas beaucoup dormi cette nuit.*

Ce phénomène est le même pour de nombreux autres adverbes comme *souvent*:

(133) *Il n'est pas venu souvent = il est venu rarement.*

Nous n'avons cependant pas considéré l'adverbe *pas souvent* comme un adverbe de négation figé parce qu'il ne présentait aucune caractéristique syntaxique différente en présence de la négation ou de *pas*.

Tout comme les déterminants, il serait possible théoriquement de faire une liste des adverbes pour lesquels, l'application de la négation crée une nouvelle entrée lexicale. Nous ne l'avons pas fait, car nous avons trouvé ce phénomène très productif. Considérons par exemple la phrase (134). Son interprétation est très proche de celle de (135), et cette interprétation peut être forcée en modifiant *pas* par *vraiment* (136).

(134) *Il n'a pas terminé brillamment ses études.*

(135) *=Il a pitoyablement terminé ses études.*

(136) *Il n'a vraiment pas terminé brillamment ses études.*

Ce phénomène ne se limite pas aux adverbes simples (137), ni même aux adverbes strictement figés (138) :

(137) *Il n'a (vraiment+ϵ) pas fait cela dans les règles de l'art.* ≡ *Il a fait cela comme un cochon.*

(138) *Il ne vient pas fréquemment (d'une manière fréquente).* ≡ *Il vient rarement.*

Par contre, pour d'autres adverbes comme *régulièrement*, cette interprétation ne peut être entendue: *pas régulièrement*=non(*régulièrement*), même avec *vraiment*. Ce problème n'a donc pas été résolu et nécessiterait vraisemblablement une étude propre approfondie

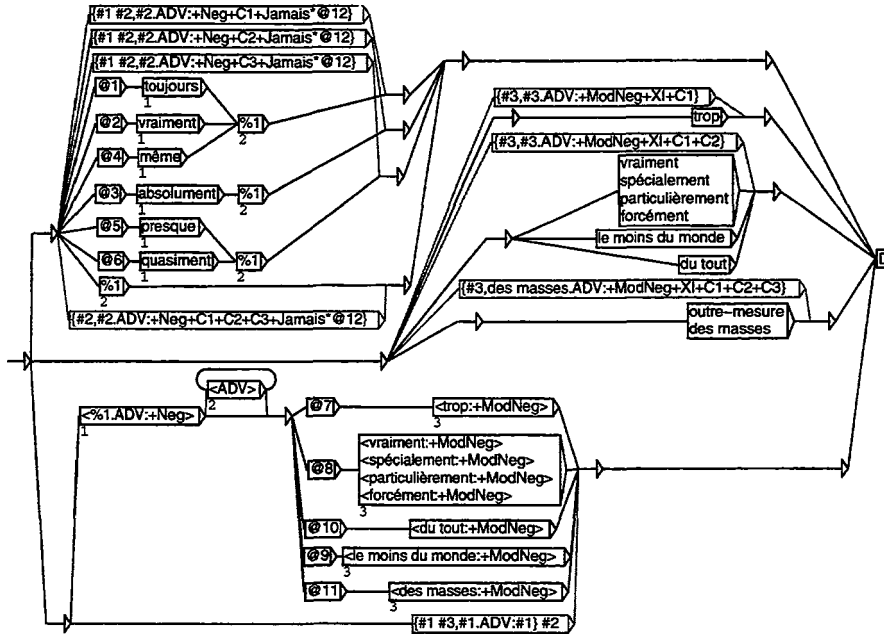


FIG. 6.7 – *Reconnaissance des négations composées décrites par la table 6.1. Cette reconnaissance se fait en trois phases, la première reconnaît la partie principale de la négation (en haut à gauche), la deuxième phase reconnaît la partie éventuellement détachée de la négation (à droite), et la troisième phase (en bas) reconstitue la négation en vérifiant les contraintes données par la table et les contraintes de combinaisons des pré-modifieurs et post-modifieurs de la négation.*

à la limite de la sémantique.

Portée de la négation sur un terme comparatif

L'interaction de la négation avec le comparatif dans (140) est nulle. On a bien la composition de sens : (140)=(141)=non(139). Il n'y a donc pas lieu de considérer comme élément lexical nouveau *ne...pas plus.* Par contre, d'une manière générale, il est possible que le référent dans la comparaison soit la forme générique *que ça* dans une phrase négative, alors que ce n'est pas possible dans une phrase affirmative (142)[10]. L'acceptabilité du

10. Nous ne parlons pas du démonstratif *ça*.

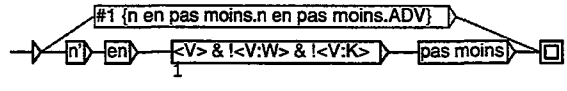


FIG. 6.8 – *Reconnaissance de l'unité minimale composée n'en pas moins.*

générique dépend d'ailleurs du comparatif (144). L'élément lexical nouveau est donc le *que çà*, la polarité négative de la phrase est exigée pour son emploi, mais cette polarité lexicale ne fait pas partie de l'élément lexical (sinon, cela reviendrait à distinguer deux types de comparatifs).

(139) *Il a plus mangé qu'hier.*
(140) *Il n'a pas mangé plus qu'hier.*
(141) Ce ne fut pas plus difficile que cela.
(142)* *Ce fut plus difficile que (cela+çà).*
(143) *Il n'a pas (plus+autant+tant) mangé que (çà+cela).*
(144) *Il n'a pas (?davantage+?* moins) mangé que (çà+cela).*
(145) ...sans oublier notre Chateaubriand [qui n'est] pas si évaporé que cela.

De plus, *si* peut être utilisé comme comparatif dans les phrases négatives et pas dans les phrases positives :

(146) *Il n'est pas si évaporé que ce que tu m'as dis.*
(147)* *Il est si évaporé que (çà+cela+ce qu'on m'avait raconté).*

Dans la reconnaissance des comparatifs, il faut donc reconnaître le terme générique *que (cela+çà)* et *si* comme comparatif, et vérifier par la suite que les phrases dans lesquels ces séquences apparaissent sont bien à polarité négative, de la même manière que pour *pour autant.*

6.3 Conclusion

Nous avons décrit dans le détail l'ensemble des procédures à mettre en œuvre pour reconnaître la négation simple et la négation composée dans un texte. Plusieurs caractéristiques ressortent de cette description :

- Il n'est pas possible de rendre l'analyse de la négation indépendante de l'analyse de la structure de la phrase, la dépendance des grammaires se manifestant à plusieurs niveaux.
- À chaque négation différente correspond une grammaire propre.
- La résolution de la portée de la négation est un problème complexe dont nous avons proposé le traitement dans des cas isolés. La résolution dans la plupart des cas doit se faire à un niveau sémantique : l'analyse syntaxique conduit donc à des ambiguïtés.

Chapitre 7

Groupe nominal et structure verbale figée

Dans ce chapitre, nous présentons parallèlement la reconnaissance des structures verbales figées qui sont les structures figées les plus «longues» ainsi que la reconnaissance des groupes nominaux «libres». Nous avons besoin de la description des groupes nominaux pour reconnaître les phrases complètes contenant les structures verbales figées, mais aussi pour reconnaître certaines structures verbales figées présentant des parties libres. Réciproquement, la description des groupes nominaux passe par l'utilisation de ressources de structures figées, et nous verrons sur un exemple que le cas «libre» est minoritaire. En présentant simultanément deux entités situées *a priori* aux deux extrêmes de l'échelle de figement [7], nous voulons montrer que la démarche que nous avons présentée pour la reconnaissance d'expressions figées va être la même que celle qui permet de reconnaître des expressions libres. Nous ne prétendons pas, en un chapitre, montrer la description complète de deux catégories syntaxiques aussi larges: chacune est un domaine que nous n'avons pas complètement couvert. La description des grammaires établies serait déjà trop volumineuse: nous nous situons donc au niveau méthodologique.

7.1 Le groupe nominal

Théoriquement, reconnaître les groupe nominal ne fait pas partie de notre démarche de reconnaissance des expressions figées. Nous en avons cependant besoin, puisque plusieurs types d'expressions figées intègrent des groupes nominaux «libres» dont les noms-tête sont figés avec le reste de la structure. En fait, la seule manière de distinguer la description d'une expression «libre» de celle d'une expression «figée» est la présence dans les règles d'éléments génériques comme les catégories <*N*> ou <*ADJ*>. Nous limiterons notre exposé à la description de quelques propriétés du modifieur (adjectival) associé à un nom dans le groupe nominal. Cette restriction importante nous permettra de focaliser l'exposé sur la démarche utilisée que nous montrerons être très proche de celle utilisée dans la reconnaissance d'expressions figées. Nous montrerons de plus que la reconnaissance de ces groupes est nécessairement hiérarchisée: c'est-à-dire que pour parler de reconnaissance et non de localisation, il faut organiser cette reconnaissance en différentes passes guidées par le lexique.

7.1.1 Introduction

Dans le texte suivant, nous avons parenthésé tous les groupes «identifiables», par leurs fonctions ou par leurs formes, à des groupes nominaux.

(Christian Ramussen, jeune Danois de 18 ans), n'a pas eu (de chance). Né dans (un milieu défavorisé), (il) n'a pas été à (le bout de (sa scolarité)). Après (un accident en rollers), qui lui a brisé (le genou droit), (il) a dû interrompre (sa formation de cuisinier). Mais (il) a (quelque chose) que (nombre de gens) peuvent (lui) envier en (Europe): (un emploi). Dans (une produktionsskole), (il) s'initie à (l'horticulture), construit (des étagères en bois). Certes, ce n'est pas (le job de (ses rêves)), mais (Christian) n'a pas (de autre choix). A (le Danemark), (les moins de 25 ans) qui ne trouvent pas (de place en formation) obtiennent au plus tard à (le bout de (six mois)), (un emploi). Il n'y a (pratiquement plus de sans-emploi) dans (ces tranches d'âge).

Nous avons classé ces groupes nominaux selon différentes catégories (voir table 7.1) comme illustration[1] de différents phénomènes linguistiques nécessitant une description. Dans cette table nous avons noté *le Danemark* et *Europe* comme deux «faux» groupes nominaux: ces deux noms interviennent dans deux groupes prépositionnels *au Danemark* et *en Europe*. On peut rétablir le déterminant pour *Danemark* par la règle *au→à le*. Il est plus difficile de faire la même chose pour *Europe*, car la règle *en→en la* n'est pas justifiée. Dans les deux cas, le problème est un faux problème, puisque l'introduction du déterminant n'apporte aucune information, le groupe élémentaire est le groupe adverbial locatif. De la même façon, il n'y a aucun intérêt à analyser *le bout des six mois* comme un groupe nominal, l'unité minimale ici est l'adverbe *au bout des six mois*.

Nous considérons maintenant la description du groupe nominal régulier et de l'extension de l'adjectif à un modifieur quelconque. À l'aide de quelques exemples, on montrera que la démarche est la même que pour les structures complètement figées.

Les groupes nominaux «réguliers»

On considère les groupes nominaux simples que l'on peut noter de la manière suivante:

$$GN \longrightarrow DET\ A^*\ N\ A^*$$

avec $DET \rightarrow le + la + les + son + sa + ses + ... + un + deux + trois + ... + une + ...$ L'accord en genre et nombre entre les DET, A et N est assuré par, soit la démultiplication de la règle en 4 cas, soit par l'utilisation de variables qui devront être unifiées:

$$\texttt{<}GN : \texttt{+N+G>} \longrightarrow \texttt{<}DET : \texttt{+N+G>}\ \texttt{<}A : \texttt{+N+G>}^*\ \texttt{<}N : \texttt{+N+G>}\ \texttt{<}A : \texttt{+N+G>}^*$$

avec $\texttt{N} \rightarrow \texttt{s} + \texttt{p}$, et $\texttt{G} \rightarrow \texttt{m} + \texttt{f}$.

L'origine de ces groupes nominaux peut être une phrase en *être*:

cet emploi est bien rémunéré $\longrightarrow$ *cet emploi qui est bien rémunéré*
$\longrightarrow$ *cet emploi bien rémunéré*

Remarquons que le déterminant a un rôle dans la tranformation, on ne peut pas dire que *un emploi bien rémunéré* provient de *un emploi est bien rémunéré*, mais plutôt de *un emploi,*

1. Ces catégories ne sont ni indépendantes, ni complètes, elles ne serviront qu'à titre d'exemples.

régulier	*l'horticulture* *un emploi* *six mois* *ses rêves*
pronom	*il, quelque-chose*
"Adjectif" figé	*des étagères en bois* (EPC) *le N de ses rêves* (ENPC) *les moins de 25 ans* (A1P2)
Nom figé	*Le genou droit* *un milieu défavorisé* *ces tranches dâges*
Nom propre	*Christian Ramussen* *Christian*
Prédicat nominal	*sa formation de cuisinier*
Verbe support	*un accident en rollers*
Faux GN	*le Danemark* *le bout de six mois* *Europe*
Déterminant composé	*nombre de gens*
Négation	*(ne...pas) de chance* *(ne...pas) de autre choix* *(ne...pas) de place en formation* *(ne...pratiquement plus) de sans-emploi*
Apposition	*Christian Ramussen, jeune Danois de 18 ans*

TAB. 7.1 – *Classification de quelques phénomènes intervenant dans les groupes nominaux*

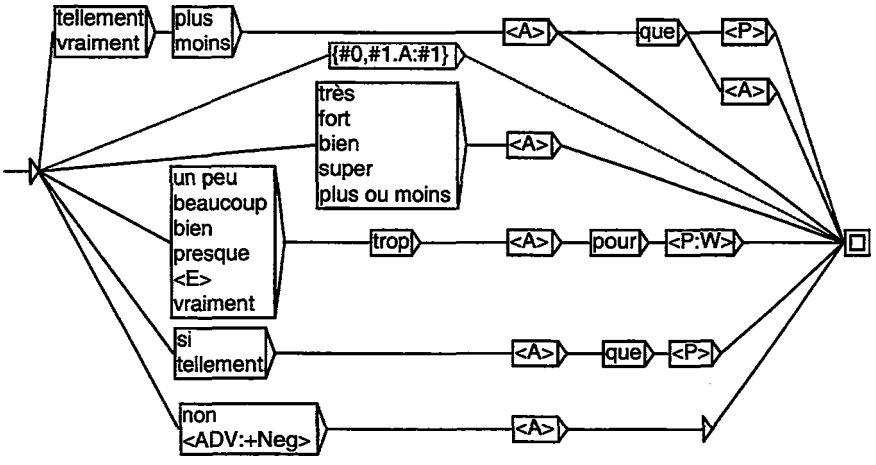


FIG. 7.1 – *Reconnaissance de la modification d'adjectif par un adverbe, ou un comparatif : la séquence reconnue est étiquetée <ADJ>.*

cet emploi est bien rémunéré. Certains adjectifs n'admettent pas cette transformation, *mon ancienne maison* ≠ *ma maison qui est ancienne.* La reconnaissance de ces groupes comporte ces analyses, ou du moins, chaque analyse devra faire l'objet d'un chemin propre dans l'automate du texte.

Modifications des adjectifs par des adverbes

Les adverbes qui peuvent modifier une phrase élémentaire source, peuvent modifier l'adjectif dérivé qui figure dans le groupe nominal (1) et (2).

(1) *Cet emploi est (souvent+en fin de semaine+en Afrique) bien rémunéré.*
(2) *Je cherche un emploi (souvent+en fin de semaine+en Afrique) bien rémunéré.*
(3) *Je mange un extrêmement bon gâteau.*
(4) *Je mange un gâteau qui est extrêmement bon.*
(5) *?* C'est extrêmement que ce gâteau est bon.*

Nous devons donc permettre l'insertion d'adverbe dans le groupe nominal. Cette insertion générale localise mais ne précise pas les constructions adverbiales spécifiques de la forme adjectivale. Par exemple, l'adverbe *très* ne peut modifier qu'un adjectif, jamais une phrase. La séparation entre ces deux catégories n'est pas facile, si nous décidons de ne conserver que les adverbes spécifique aux adjectifs, nous supprimons des modifieurs comme *extrêmement* dont l'analyse comme adverbe de phrase est moins courante, sans être toutefois interdite : (3) à (5). Nous avons donc adopté un critère intuitif de séparation. Les comparatifs entrent aussi dans cette catégorie. Nous reconnaissons ces adverbes par le graphe de la figure 7.1. Il faut aussi noter que la modification dépend de la combinaison adjectif/nom : *une enveloppe très blanche*, *?une enveloppe très ouverte*, **un afficheur très analogique*...
L'adverbe peut également porter sur le déterminant du groupe nominal, et dans ce cas, la reconnaissance par un schéma général n'est pas non plus suffisante. Cette classe d'ad-

verbes accolés à un adjectif suit le schéma de transformation suivant :

$$Det\ N_1\ Adv\ A \rightarrow Det\ N_1 - Det_{fraction}(ce\ N_1)\ est\ A -$$

(6) *la foule en partie d'origine nippone*
(7) *la foule d'origine nippone en partie*
(8)* *En partie, la foule d'origine nippone*
(9) *la foule qui est en partie d'origine nippone*
(10) *la foule – (une partie de) (cette foule) est d'origine nippone –*

Pour vérifier que ces adverbes sont liés à l'adjectif, on vérifie que leur mobilité est restreinte, à gauche et à droite de celui-ci. Par exemple, le groupe nominal (8) n'est pas accepté. Avec un contexte, *en partie* pourrait par contre être considéré comme un déterminant adverbial (c'est-à-dire l'adverbe modifie le déterminant du groupe nominal : *Les invités sont restés en partie* [11]).
Ces adverbes qui sélectionnent une partie du nom sur laquelle s'applique l'adjectif, peuvent être de trois types :

- *singulier non collectif*, que l'on note N: $\mathtt{s} - \mathtt{Coll}$, par exemple *une porte*, *un exercice*, *un repas*
- *singulier collectif*, que l'on note N: $\mathtt{s} + \mathtt{Coll}$, par exemple *une foule*, *une liste*, *un groupe*
- *pluriel*, que l'on note N: $\mathtt{p}$, par exemple *des galets*, *des gens*, ...

Suivant ces catégories, les adverbes que nous étudions n'auront pas les mêmes propriétés.
(11) *des clous tous tordus*
(12) *= des clous – (la totalité de) (ces clous) sont tordus –*
(13) *des clous en partie tordus*
(14) *≠ des clous – (une partie de) (ces clous) est tordue –*
Pour l'exemple (13), on comprend plutôt : *des clous qui sont chacuns en partie tordu.*
L'avantage de faire apparaître cette transformation, est que le traitement du déterminant est plus aisé, en tant que opérateur ou prédicat logique :
(15) *une porte entièrement verte*
(16) *= une porte – (l'intégralité de) (cette porte) est verte –*
(17) $\exists\ P \mid \mathtt{porte}(P)$ *et* $\mathtt{verte}(\mathtt{intégralité}(X))$

Certains de ces adverbes présentent des contraintes distributionnelles sur le nom-tête : *en minorité* s'applique essentiellement à des noms humains. Nous avons aussi trouvé certains adverbes, mais dont le "domaine d'application" est restreint à quelques adjectifs : par exemple *en tout point* ne s'appliquent qu'aux adjectifs *semblable*, *identique* ou *différent*, nous traiterons ces quelques cas comme des exceptions dans le graphe de la figure 7.3.

Toutes ces propriétés sont mises sous la forme d'une table (table 7.2). Nous retrouvons des entrées des tables des adverbes figés dont elles héritent de toutes les propriétés indiquées par ces tables que nous ne reprécisons pas. Comme certains de ces adverbes admettent des variations, nous avons regroupé toutes les variations dans des graphes. Nous avons indiqué en plus la modification (ou la complémentation de ces adverbes) par d'autres adverbes comme *quasiment*, *presque*, et *sans exception*. Enfin, certaines entrées, telle *en partie*, admettent plusieurs variantes, nous les représentons donc sous la forme

Adv	quasiment *Adv*	*Adv* sans exception	*N*: +Hum	*N*: −Hum	*N*: s − Coll	*N*: s + Coll	*N*: p	*Adv A*	*A Adv*
entièrement	+	+	+	+	+	+	−	+	−
en (quasi+ϵ) totalité	+	+	+	+	+	−	−	+	+
complètement	+	−	+	+	+	+	+	+	−
totalement	+	+	+	+	+	+	+	+	−
principalement	+	−	+	−	−	+	+	+	+
pleinement	+	−	+	−	−	+	−	+	+
<tout>	+	+	+	+	+	+	+	+	−
en plus <E> grande <E> partie	−	−	+	+	+	+	−	+	+
partiellement	−	−	+	+	+	+	−	+	−
en grande <E> majorité	+	−	+	−	−	+	+	+	+
dans Poss majorité	+	−	+	−	−	+	+	−	+

TAB. 7.2 – *Det* N_1 *Adv* $A \rightarrow$ *Det* N_1 $-Det_{fraction}$ (*ce* N_1) *est* A –

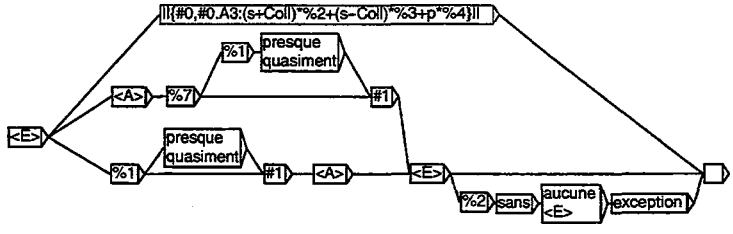


FIG. 7.2 – *Graphe patron pour les adverbes décrits par la table 7.2*

d'un graphe. Pratiquement, dans le fichier correspondant, cette entrée apparaît comme un appel à un sous-graphe, tout comme dans une «boîte» dans un graphe. Cette table est traitée par le graphe patron de la figure 7.2.

Voici quelques concordances trouvées dans le corpus :

(18) ... à l'Ukraine, peuplée de huit cent mille **habitants en majorité russophones**.

(19) ... ces deux cinéastes aux **biographies en partie semblables**.

(20) En apparence, la **gauche principalement socialiste**, [] a retrouvé ...

(21) **Inspection pleinement satisfaisante**, semble-t-il, pour ...

Adjectifs Gauches

Notre schéma général offrait la possibilité d'un nombre quelconque d'adjectifs à gauche du nom, nous étudions les restrictions de cette accumulation.

L'examen systématique de toutes les occurrences de <*A*> <*A*> <*N*> dans une année du journal Le Monde, nous donne environ 25 000 résultats. Examinés systématiquement,

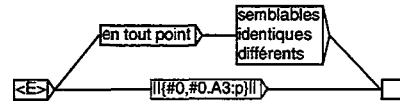


FIG. 7.3 – *Cas à part n'entrant pas dans le schéma général*

nous remarquons que la majorité des cas valides[2] consistent en la modification d'un mot composé *AN* (adjectif-nom, comme *haut fonctionnaire, premier ministre...*) par un adjectif préposé[3] :

un **super (haut-fonctionnaire)**$_{AN}$
Un **grand (jeune homme)**$_{AN}$ ...
L'Union européenne a un **gros (bon-sens)**$_{AN}$
... le **huitième (grand-prix)**$_{AN}$.
... le **meilleur (grand reporter)**$_{AN}$.
Le **prochain (joli mois de mai)**$_{ANDN}$ **sera-t-il chaud?**

Sur les 25 000, seule une centaine correspondent à de véritables *A A N* :

(22) Un **vrai joli larcin** de petite fille riche.
(23) ...qui abrite **l'unique grand golf** de l'île.
(24) ...de son **seul vrai revenu.**
(25) ...tient sans trop d'encombre dans un **seul petit** bateau.
(26) ...François Pinault, le **seul véritable** candidat en lice.
(27) ...le **probable futur président** sud-africain...
(28) [elles] doivent se contenter **d'un modeste quatrième prix.**
(29) [ils] disent nettement préférer **un bon vieil affrontement.**
(30) **Le deuxième grand journal** italien, ...
(31) **Le dernier gros roman** d'Umberto Eco, ...
(32) ... **quand un adorable petit phénomène** vous traîne devant les tribunaux...
(33) Ce n'est **qu'un gentil petit mammifère** qui volette dans la nuit !
(34) [ce]...est **un magnifique petit édicule** à corniche...

Nous voyons que la combinaison d'adjectifs est essentiellement due à la présence de quelques adjectifs, qui d'ailleurs se combinent même préférentiellement entre eux, ces adjectifs sont :

1. *seul, unique, autre* (23),(24),(27),(26) : ces trois adjectifs ne peuvent apparaître qu'en position extérieure, sans contraintes sur le deuxième adjectif (*le seul redoutable adversaire*). Ces deux adjectifs peuvent être traités comme des déterminants.

2. *premier, second, ... n*^{ème}. Ces adjectifs peuvent aussi être traités comme déterminants, mais on les voit aussi apparaître en seconde position dans *modeste quatrième prix* ou *Le devoir se terminait par un incroyable troisième exercice.* Cette position du déterminant après un modifieur est exceptionnelle. Dans ce premier exemple, on peut dire que *n*^{ème} *prix* est figé, c'est plus difficile dans le second. Ces adjectifs auront donc un traitement spécifique. Dans ce cas, comme dans le précédent, le second adjectif admet très facilement un modifieur.

3. *petit* (34), (32), (33) : un phénomène syntaxique particulier se produit dans les séquences *petit N* : *un beau petit bureau, une formidable petite université, un passionnant petit papillon*, dont le sens de l'adjectif *petit*, n'est pas petit en taille, mais plutôt affectif (aussi?), i.e. mignon, charmant. Cette règle est très productive. D'autres adjectifs ont à peu près le même comportement : *gros* ou *vieux* par exemple. Mais,

2. Les autres provenant d'ambiguïté.
3. Ce test de modification par un adjectif à gauche pourrait d'ailleurs être un critère supplémentaire de sélection des mots composés de type *AN*

il est plus difficile de trouver des contextes, où on entend ce sens : *(*?) un redoutable gros adversaire* mais *un remarquable gros établi.*) On remarque en plus que *bon* modifie ces adjectifs et uniquement dans leur sens affectif : *un bon vieil adversaire.* On retrouve le même phénonmène de non-compositionalité de sens et de composition possible avec un autre adjectif à gauche pour l'adjectif *grand* : *un pitoyable grand quincailler*, où *grand* signifie plutôt fameux. On remarque que *grand* admet le modifieur *très*, alors que *petit* ne l'admet pas :

* *un beau très petit bureau*

? *un remarquable très grand cuisinier*

Cette remarque montre par ailleurs qu'il est difficile d'intégrer *grand cuisinier, grand reporter* dans un dictionnaire de mots composés sans indiquer cette possibilité de variation.

4. *actuel, probable, ancien, précédent, nouveau, pontentiel* : ces adjectifs temporels se combinent facilement en première ou seconde position avec un ajectif quelconque. *mon futur redoutable adversaire, mon redoutable futur adversaire.*

5. *vrai, faux* et *véritable* se combinent avec un autre adjectif en première ou deuxième position.

Nous mettons à part les adjectifs qui seront interprétés comme des déterminants, nous représentons les autres par le graphe de la figure 7.4. Cette représentation n'est pas suffisante pour gérer des doubles combinaisons d'adjectifs à gauche du nom, et le graphe ne donne pas d'indication concernant le traitement ultérieur de ces adjectifs spéciaux. Pratiquement, il faudrait définir un graphe par adjectif, dans lequel on décrirait le traitement futur au moyen de grammaires d'utilisation. Par exemple, on peut associer à l'adjectif *grand* le groupe *devant l'Éternel.* Cette dernière expression est décrite dans la table ENPC, dans laquelle ne figurent pas les informations que nous venons de rajouter. Cela nous ramène au problème d'interaction entre les différents phénomènes linguistiques et à l'ajout d'exceptions ou de particularités dans une table du lexique-gramaire.

Adjectifs composés

Considérons la séquence *en métal*, on peut «attacher» les deux mots, et appeler le nouveau mot obtenu *en_métal* un adjectif. Cet adjectif, dont le sens est proche de *métallique*, présente les propriétés des adjectifs décrits précedemment comme on le voit dans (35) à (38) :

(35) *Luc a acheté une étagère (en métal+métallique).*

(36) *Cette étagère est (en métal+métallique).*

(37) *Cette étagère est trop (en métal+? métallique) pour cette utilisation.*

(38) [il] nous prouve, dans ce roman si en marge du reste de son oeuvre que ...

Nous appelons ces compléments des adjectifs composés. On trouve cette construction dans une table appelée EPC (être-Préposition-Complément figé) sous la forme :

$$N_0 \text{ <être> } Prép\ Complément\ figé$$

Cette classe n'est pas isolée, elle fait partie des tables du lexique-grammaire EPX. Il existe par ailleurs d'autres classes de constructions verbales donnant lieu à des groupes

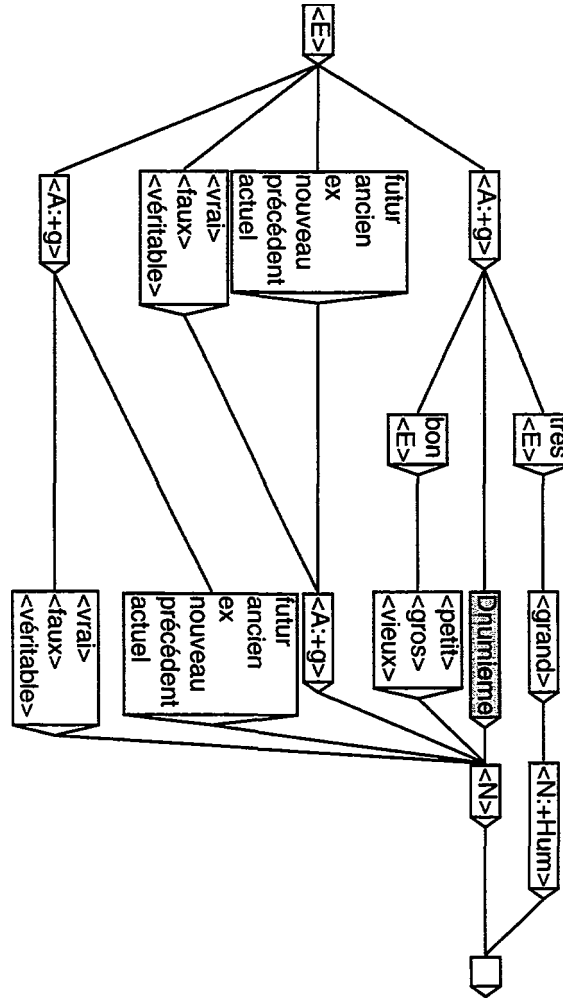


FIG. 7.4 – *Graphe reconnaissant les séquences A A N*

eapc	*être <A> en toutes choses*
epa	*être bien foutu*
epc	*être en bois*
epca	*être dans un abandon complet*
	être du bois dont on fait les flûtes
epcdc	*être à l'abri des poursuites*
epcdn	*être de l'entourage de N*
epcpc	*être en accord avec les observations*
epcpn	*être à égalité de forces avec N*
epcpq	*être à la condition de V: W*
epcdetc	*être dans l'administration*
enpc	*être l'homme à abattre*
edn	*être le meilleur ami de N*
a12	*avoir l'argent facile*
a1p2	*avoir affaire à la justice*
$a1p_c$	*avoir un joli visage*
a1pn	*avoir un gros avantage sur N*
anp2	*avoir un N en compte*
f2-1...	*faire l'examen de ce dossier*
fnpn...	*faire halte à un bistrot*

TAB. 7.4 – *Construction verbales donnant lieu à la production de groupes nominaux simples.*

nominaux simples, comme nous l'avons vu au début de chapitre: les constructions en *avoir*, les constructions en *faire*, et d'une manière plus générale, toutes les constructions à verbe support. Nous récapitulons celles que nous avons utilisées dans le tableau 7.4. L'utilisation de ces ressources se fait d'une manière analogue à celle décrite pour la reconnaissance des adverbes figés. Pour les constructions en *avoir*, les transformations suivantes s'appliquent:

$$N_0 \textit{ avoir } W \longrightarrow (N_0 \textit{ avec } W)_{GN}$$

et la forme parallèle pour les phrases comportant une négation:

$$N_0 \textit{ avoir} + NEG\ W \longrightarrow (N_0 \textit{ sans } W)_{GN}$$

(39) *des frères sans points communs*
(40) *une fille avec un joli visage*
(41) *un homme avec un bandeau sur les yeux*

On observe aussi la transformation N_0 *avoir* $X \longrightarrow N_0$ *de* X comme dans les phrases (42) et (43). Cette transformation s'applique uniquement aux constructions exprimant des mesures (le critère de décision est que le déterminant est numérique):

(42) *Ce mur a trois mètres de haut→Ce mur de trois mètres de haut.*
(43) *Cet homme a moins de 25 ans→Cet homme de moins de 25 ans.*
(44) *Cette famille d'une fille et quatre garçons.*

Les constructions en *faire* étudiée par J. GIRY [4, 5] regroupe des tables de nominalisations: N_0 *déménage son appartement→faire le déménagement de* (N_0+*son appartement)* et des tables de prédicats nominaux ayant le verbe support *faire*: *faire halte à ce bistro.* Ces deux types de construction forment des groupes nominaux, puisque le verbe *faire* peut être effacé (l'analyse de (45) se fait en considérant (47) dans laquelle le verbe *être* est effacé[4] (46) puis le verbe *faire*). La reconnaissance des structures en *faire* dans notre description est actuellement limitée aux deux tables FNPN et F2-1.

(45) Le ministre sud-coréen des affaires étrangères, Han Sung-joo, a déclaré lundi 3 qu'une percée dans le dossier nucléaire nord-coréen pourrait être suivie d'une reconnaissance de Pyongyang par les Etats-Unis et le Japon.

(46) *... a déclaré qu'une percée faite dans le ....*

(47) *Cette percée serait faite par N dans le dossier nucléaire.*

Les titres de profession

Considérons le modifieur *de la république.* Ce modifieur s'applique à quelques formes dont *président.* En raison de cette faible productivité, l'ensemble *président de la république* est enregistré comme nom composé dans le DELAC. Cependant, cet catégorisation ne tient pas en compte des séquences comme *président de la République Française* ou *président actuel de la république*, etc. Pour la première séquence, il faudrait l'adjectif *Française* modifie *république* et non l'ensemble. Cette modification n'est pas possible si on considère que *président de la république* est une unité minimale. Il n'est pas possible non plus d'espérer analyser librement *président de la (république Française)* puisque nous venons de dire que *président de la république* n'est justement pas analysable librement. Ce phénomène de modification de noms composés fait actuellement l'objet d'études [5]. Cependant dans le cadre des titres de professions (essentiellement politiques), nous avons considéré le problème autrement. Nous avons constitué une bibliothèque qui comporte actuellement plus de 300 graphes différents décrivant dans le détail, la grammaire de ces groupes nominaux.

Ces groupes nominaux sont de plus fortement liés à la description des noms propres, soit par un mécanisme d'apposition comme dans *Le président français, Jacques Chirac, a aujourd'hui...*, soit par un mécanisme de «titres courts» comme dans *le président Jacques Chirac.* La séparation entre les deux cas de figure est très faible (présence de la virgule, impossibilité dans un style soutenu de dire *le président, Chirac,* alors que *le président Chirac* passe mieux). En anglais, le nom propre peut même être intercalé entre deux éléments du titre (exemple (1) à (4) page 64. Ces phénomènes sont aussi décrits par nos graphes. Une particularité de ces groupes nominaux est qu'ils permettent en laissant des «trous» de constituer des dictionnaires de noms propres de manière presque totalement automatique [22], [22].

Ce micro-domaine relativement clos (en se limitant aux professions politiques) est un bon exemple de la nécessité d'une description détaillée de chaque entrée lexicale par un graphe indépendant, et d'un outil permettant une extraction d'information extrêmement fine grâce à une description linguistique.

4. Il s'agit de l'auxiliaire être de la forme passive.

7.1.2 Les différents niveaux de groupes nominaux

Nous considérons ici des groupes nominaux élémentaires de la forme :

$$<DET:+\text{Déf}> <ADJ> <N> <ADJ>$$

Nous montrons que si localiser de telles séquences est simple, la reconnaissance en est beaucoup plus complexe, et doit passer par la notion de «niveau» de groupe nominal introduite par HARRIS [15]. Considérons la phrase (48). Elle a une structure non ambiguë : N_0 *vouloir* N_1. Avec N_1 =*cette grosse balle rose.*

(48) *Je voudrai cette grosse balle rose.*

Nous avons vu que le groupe *Dét N A* pouvait être analysé *Dét N qui est A*. Nous avons donc deux possibilités d'analyse de la phrase précédente : (49) et (50).

(49) *Je voudrai (cette ((grosse balle) qui est rose)).*

(50) *Je voudrai (cette ((balle rose) qui est grosse)).*

Ces phrases ne sont pas équivalentes d'un point de vue logique, puisque l'adjectif *gros* à une valeur relative à l'ensemble modifié : si nous avons un ensemble de balles, et que les balles roses sont les plus grosses balles de toutes (mais elles n'ont pas toute la même taille). Alors, *une (grosse balle) rose* indiquera une des grosses balles et de couleur rose, l'adjectif *grosse* ici étant relatif à l'ensemble des balles. Alors que *une grosse (balle rose)* prendra comme référence les balles roses, et donc on cherchera parmi les balles roses les plus grosses, le mot *grosse* est donc ici relatif à l'ensemble des balles roses. Avec ce contexte à l'esprit, il est assez facile d'entendre les deux interprétations précédente que l'on peut d'ailleurs accentuer en introduisant des pauses dans la phrase. Il n'est pas possible de faire un choix *a priori* entre ces deux interprétations. D'ailleurs on peut forcer l'une ou l'autre des analyses avec des ajectifs bien choisis : (51) s'analyse par (52) et non pas par (53). Par contre (54) s'analyse par (56) et difficilement par (55).

(51) *notre ancienne maison blanche a été repeinte.*

(52) *notre ancienne (maison qui était blanche) a été repeinte.*

(53)* *notre (ancienne maison) qui (est+était) blanche a été repeinte.*

(54) *Je cherche cette ancienne édition introuvable.*

(55)?* *Je cherche cette ancienne (édition qui était introuvable).*

(56) *Je cherche cette (ancienne édition) qui est (maintenant+ϵ) introuvable.*

Cette question d'ordre d'analyse pour deux adjectifs portant sur un nom peut s'étendre à l'ordre d'analyse entre le déterminant et les adjectifs, entre les adjectifs et une proposition relative, etc. L'exemple précédent, montre clairement que cet ordre dépend des modificateurs considérés : nous ne chercherons donc pas à résoudre ce problème d'une manière générale. Ce problème montre bien qu'une grammaire du genre $<DET:+\text{Déf}>$ $<ADJ>$ $<N>$ $<ADJ>$ ne *reconnaît* pas la séquence mais ne fait que la *localiser*. Cela implique donc une organisation de la description du groupe nominal en «niveaux» correspondant à des passes sucessives de la procédure de reconnaissance. Tout comme HARRIS le propose, nous avons organisé nos différentes couches de groupe nominal par des indices croissants sur les groupes déjà reconnus. Nous pouvons par exemple, faire le choix de l'organisation donnée par le schéma de la figure 7.5. Ce schéma signifie que le premier niveau de reconnaissance que nous considérerons est celui des mots composés, ensuite la modification à droite, puis la modification à gauche, ensuite la modification par une relative, et enfin la modification par le déterminant.

Les phrases (57) et (59) sont parenthésées avec ce schéma en (58) et (60).

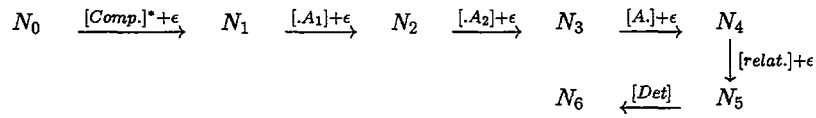


FIG. 7.5 – *Organisation possible de l'analyse d'un groupe nominal.*

(57) *J'aperçois cette immense étendue bleue que nous appelons communément la mer.*
(58) *J'aperçois* $(_6$*cette* $(_5(_4$*immense* $(_2$*étendue bleue*$)_2)_4$ *que nous appelons communément la mer*$)_5)_6$.
(59) *La prochaine visite de Paul que nous attendons est prévue fin mars.*
(60) $(_6$*La* $(_5(_4$*prochaine* $(_2$*visite de Paul*$)_2)_4$ *que nous attendons*$)_5)_6$ *est prévue fin mars.*
(61) *(...), estime un très haut fonctionnaire.*
(62)* *(...), estime un très* $(_1$*haut fonctionnaire*$)_1$.

Puisque nous avons vu que l'ordre d'analyse dépend des éléments lexicaux considérés, ce schéma par défaut introduira une analyse erronée dans certains cas: la phrase (61), analysée par (62) montre même que l'étape d'analyse des mots composés n'est pas nécessairement la première, puisque *très* modifie l'adjectif ***haut***[5]. En absence de données supplémentaires sur chaque entrée lexicale nous utiliserons quand même ce schéma élémentaire. Nous avons indiqué par ce schéma que la dernière étape de reconnaissance du groupe nominal est le rattachement du déterminant. Nous utiliserons cette caractéristique pour «normaliser» les groupes nominaux par <*DET*> <*AN*>. Nous avons déjà vu que le traitement de la négation avait besoin de cette forme canonique, nous verrons que la reconnaissance de structures verbales figées comme `C1I` nécessite aussi une telle normalisation.

7.2 Les structures verbales figées

La reconnaissance des expressions verbales figées se base sur la reconnaissance des autres expressions figées décrites auparavant et des groupes nominaux. La méthode est la même que pour les autres expressions figées, et nous ne la redécrirons pas. Nous nous concentrons plutôt sur les résultats obtenus et sur leurs améliorations potentielles. Nous discuterons essentiellement d'une seule table, la table `C1IPN` qui comporte les formes les moins figées de la série considérée. Nous faisons le bilan de toutes les tables appliquées, et nous donnons la couverture générale des unités reconnues sur le texte. La source de toutes les constructions verbales figées que nous reconnaissons est constituée des tables `C` du lexique-grammaire décrite dans [9] et [12].

7.2.1 La table `C1IPN`

La table `C1IPN` du lexique-grammaire regroupe les constructions verbales figées de type N_0 *V* C_1 *Prép* N_2, où C_1 est un groupe nominal dont le nom-tête est figé avec le verbe, mais admettant des modifieurs possibles. Les phrases (63) à (65), par exemple, correspondent à trois structures dérivées du schéma élémentaire N_0 *prendre des libertés avec* N_2. Dans cet exemple, *liberté* est nécessairement au pluriel (66). Le déterminant est aussi contraint car le groupe nominal doit être indéfini (67). Rappelons que l'indéfinition n'est

5. Même si cette séquence est analysée comme une sorte de jeu de mots, il se trouve que son analyse ne peut pas commencer par l'identification des mots composés.

pas nécessairement apportée par le déterminant (68).

(63) *Il prend des libertés avec l'Histoire.*
(64) *Il prend quelques libertés avec l'Histoire.*
(65) *Il prend des libertés incroyables avec l'Histoire.*
(66)* *Il prend une liberté avec l'Histoire.*
(67)* *Il prend les libertés avec l'Histoire.*
(68) *Il prend les libertés que l'on sait avec l'Histoire.*

La reconnaissance de ces expressions à l'aide de graphe patron est détaillée dans [21]. Un graphe patron permettant de reconnaître ces entrées est donné sur la figure 7.6, et les premières occurrences de notre texte sont données table 7.5.
Nous avons fait figurer en marge de chacune de ces entrées les marques *inc*, *prép*, *et*, *Loc*, *PRO* et le symbole *. Ces marques conduiront à faire des remarques beaucoup plus générales sur le processus global :

inc - La description des groupes nominaux est incomplète : pour les phrases 4 et 16, les «complèments» des noms *objectif* et *mesure* n'ont pas été identifiés. La première correspond à une structure *avoir un objectif de N* `A1PN?`, et la deuxième à la structure *être une mesure de N* `EDN?`. Ces deux structures ne sont pas présentes dans nos tables. Pour la phrase 10, le complément *de l'ONU* n'a pas été rattaché à *convention*. L'analyse de ce complément n'est pas claire : il peut s'agir d'un complèment du type *Ce* <*N*:+Conc> *être à* <*N*:+Hum>→*Le* <*N*:+Conc> *de* <*N*:+Hum> mais on a du mal à remplacer *l'ONU* par un autre groupe nominal **la convention de la France*, *?une convention internationale de la CEE*. Disons que cette construction est figée. Enfin, dans la phrase 10, la complétive n'a pas été reconnue, puisque nous nous sommes restreints aux groupes nominaux simples.

prép - La préposition attendue n'a pas été rencontrée et elle existe sous une forme différente. Par exemple, dans la phrase 19, la préposition associée à la construction *donner un coup de main* est *à*. Dans cette phrase, la préposition suivant la construction est *dans*. Nous pouvons faire deux analyses : soit *dans les fermes* est un locatif et le complément indirect *aux fermiers* est effacé, soit *dans les fermes* remplace le complément *aux fermiers*. Cette deuxième interprétation serait assez pertinente puisque *aux fermiers* est implicite dans la phrase 19. Dans la table (4.1) le complément *à* N_2 est marqué obligatoire pour cette construction, ce qui confirmerait aussi la deuxième hypothèse.

et - Dans la phrase 22, la structure verbale est *établir un lien entre N et N*. Le *et* fait partie de la structure figée dans la mesure où il est obligatoirement présent après *entre*. De plus il faut remarquer que les deux *N* peuvent (comme dans la phrase 22) ne pas avoir de déterminant. Cette propriété n'est pas indiquée dans la table.

* - Les phrases 5 et 17, identifiées comme des `C1IPN` correspondent en fait à d'autres structures. La construction de la phrase 17 *faire un pas dans cette direction* est répertoriée dans la table `C1P2` et ne correspond pas à *faire un pas (dans la rue)*. La phrase 5 provient de *les traits du visage tirés* `DELAC` et non pas de *tirer un trait sur* `C1IPN`.

Loc - Dans la phrase 1, la construction reconnue est *faire un pas Loc*, où *Loc* représente un adverbe circonstanciel de lieu. En absence d'une description, nous l'avons remplacé

1	Loc		plus proche de nous tous car nous **ne faisons (plus un pas) dans** la vie sans transport
2			qu'elle, il ne se trouvera pas pour **verser (des pleurs)** un seul bretonnant
3			l'immigration pour laquelle l'école publique **joue encore (un rôle décisif)**
4	*inc*		partir du moment où l'on **se fixe a priori (un objectif)** de rentabilité .
5		*	[elles] ne pouvaient suffire à expliquer **(les traits) tirés** et le manque d'entrain.
6			**(Ces travaux) étaient menés** sous le contrôle d'un expert.
7			entre deux mutations, prendra la décision d' **entreprendre (des travaux)** pour un risque à la
8			au SMIC métropolitain est l'illustration de **(ce "coup d'arrêt") donné à** la politique de rattrapage que
9			John Birt **tint (un tout autre langage)** .
10	*inc*		ils **laissent (des vides)** que les vivants ne reconnaissent pas
11			Lansari, cependant, **prend (quelques libertés) avec** l'Histoire
12			Si à Annecy, le Thiou, **jouait (un rôle esthétique et touristique)**
13			ceux et celles qui attendent leur tour **observent (un silence inquiet)** .
14	*prép*		Solide, il ne rechigne pas à **donner (un coup de main)** dans les fermes pendant ses congés.
15			la vente par correspondance **risque de porter (un coup fatal) à** la saison du blanc
16	*inc*		le gouvernement de M. Balladur **prendrait (des mesures)** de soutien notamment dans le secteur automobile
17		*	nous **allons faire (des pas importants) dans** cette direction
18			Albert Reynolds et (...), John Major, **lancent (un appel historique)** .
19	*prép*		**(Les renforts), qui seront envoyés** dans la zone démilitarisée au nord du pays, devraient venir du Ghana
20	*inc*		Ainsi a-t-il décidé de **ratifier (les trois conventions internationales)** de l'ONU relatives aux stupéfiants
21	PRO		il épouse une esclave indienne (...), qui **lui donna (deux enfants)** .
22	*et*		on se gardera d' **établir (un lien explicite) entre** partenariat et adhésion.
23			le Programme alimentaire mondial, qui, (...), **apporte (un soutien important) à** ces populations
24			a annoncé sa décision dans **(une lettre) adressée, mardi 4 janvier, à** M. Pasqua.
25			Mais l'UNAPEL **trouve (un écho) auprès de** le ministre François Bayrou
26			les chaînes **exercent (une pression à la baisse) sur** leur participation aux oeuvres commandées
27			Aux termes d' **(une convention) signée pour cinq ans avec** le Syndicat interhospitalier régional
28			Quand on **jette (un coup d'oeil) sur** le programme du spectacle
29			C'est définir les règles d' **(un jeu) que l'on ne jouera pas**
30			**(Ces travaux) sont principalement menés par** l'équipe du docteur Roger Gosden, de l'université d'Edimbourg.
31			**(Une expérience) sera menée dans les mois qui viennent sur** les TGV Atlantique et Nord-Europe

TAB. 7.5 – *Premières occurrences des structures verbales de la table* `C1IPN` *dans le texte.*

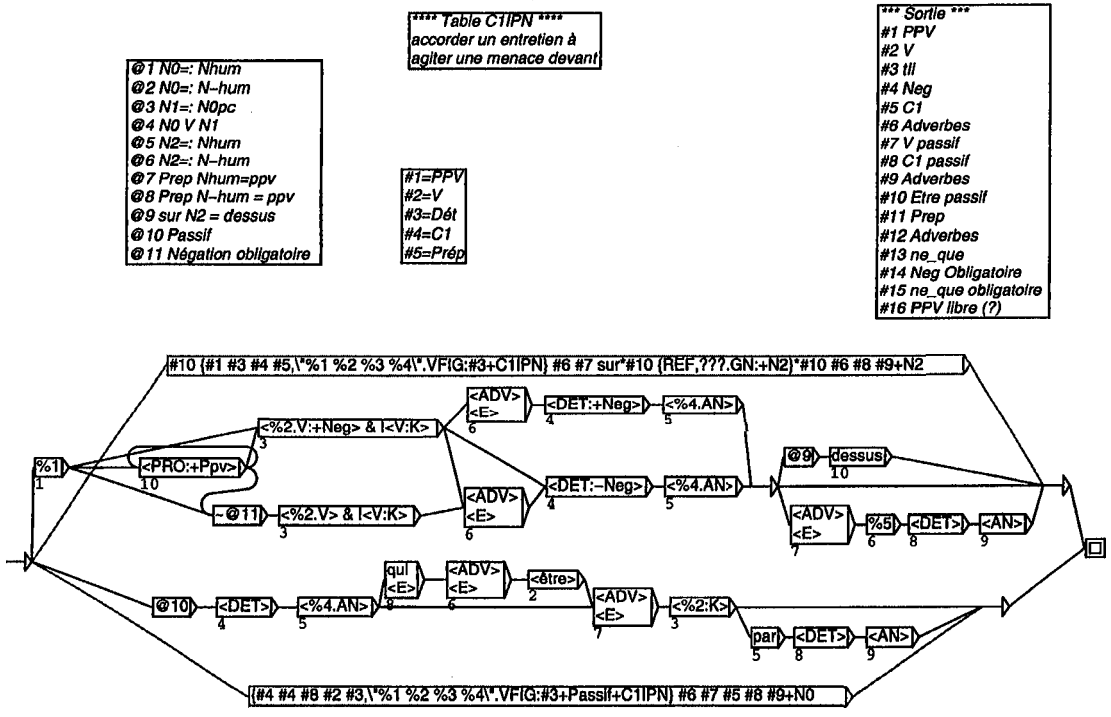


FIG. 7.6 – *Graphe patron de la table* C1IPN

par (*dans* + *sur* + *sous* + ... + *vers*) N. Nous identifions alors *dans la vie* comme un locatif. Le problème ici, n'est pas très important, puisque *dans la vie* est décrit comme adverbe dans la table PCA et donc le longest-match fonctionnera encore.

PRO - Dans la phrase 21, le complément indirect de la structure reconnue N_0 *donner un enfant à* N_1 est pronominalisée en *lui* comme un complément indirect standard. Le pronom *lui* est «reconnu» par <*PRO*:+Ppv> dans le graphe 7.6.

Nous développons maintenant ces différentes remarques dans le cadre plus général de la reconnaissance des expressions figées.

7.2.2 Limites de la reconnaissance

Ressources incomplètes, et conflits entre ressources

Le point *inc* présenté précédemment signale que les ressources existantes ne sont pas complètes, et qu'il ne sera pas possible de tout reconnaître sans les compléter. Cette remarque ne doit pas être l'occasion de repousser le problème ailleurs. La version du DELAC que nous avons utilisée contient les noms composés *mesure de rétorsion*, *mesure d'intimidation*, *mesure d'urgence*, *mesure de protection*, etc. mais pas *mesure de soutien*. Le processus semble assez productif: *ceci est une mesure de (économie+prévision+radinerie+...)* et l'adverbe *par mesure de N* est d'ailleurs décrit dans PCDN. Il s'agit de la même structure:

C'est par mesure de radinerie que les feuilles sont utilisées deux fois
vs. Que les feuilles soient utilisées deux fois est une mesure de radinerie.

Nous avons donc une relation propre au mot *mesure* entre ces deux constructions. Cela nous conduit au problème beaucoup plus général: tous les phénomènes linguistiques (figés pour ce qui nous intéresse ici) ont une grammaire propre. Pour répertorier les structures correspondant à un phénomène donné, des classes de structures figées ont été créées regroupant ces structures suivant leur forme syntaxique. Ce regroupement permet d'effectuer des factorisations des propriétés syntaxiques de ces différentes structures. Cependant, chacune des entrées reste particulière, et possède des propriétés «uniques» non factorisables. L'incomplétude des ressources n'est donc pas uniquement due au fait de l'absence d'entrée dans certaines tables.

Compléments implicites

L'exemple de la phrase 19 montre une limite importante dans l'utilisation des colonnes des tables du lexique-grammaire: certaines propriétés syntaxiques, comme ici la présence obligatoire d'un complément *à* N, sont mises en défaut dans certaines phrases dont l'interprétation se fait au niveau sémantique (ou pragmatique?). Nous savons qu'il y a des habitants dans les fermes effectuant des travaux l'été, et donc nous comprenons sans autre contexte que *donner un coup de main dans les fermes* signifie *donner un coup de main à ces gens travaillant dans les fermes*. Le complèment est obligatoire, mais il est rendu implicite par un autre complément (qui est ici un vrai complément locatif), ou même par le contexte. Si ce complément est rendu implicite par le contexte, il faudra donc avoir analysé la phrase complètement ainsi que son contexte pour pouvoir espérer le retrouver. Nous ne pouvons donc pas bloquer l'analyse d'une phrase parce qu'un complément marqué obligatoire est présent.

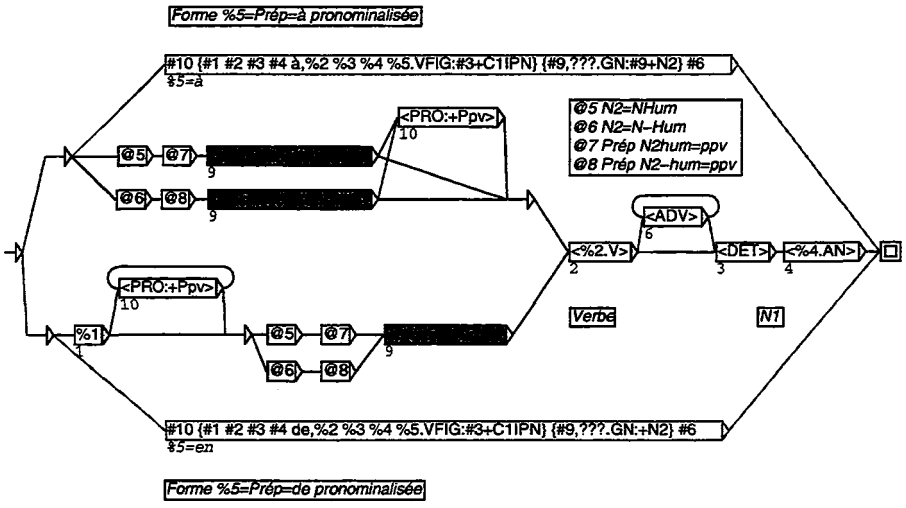


FIG. 7.7 – *Reconnaissance des pronoms (en grisés) en et <lui.PRO> pour les entrées de la table* **C1IPN**. *Le groupe nominal est reconstitué en introduisant une référence marquée par???.*

Interférence avec la grammaire libre

Nous avons dit que le pronom de la phrase 21 a été «reconnu» par le graphe patron de la table **C1IPN**. En fait, il a simplement été localisé, avec notre définition de la reconnaissance. la phrase incorrecte **L'école publique lui joue encore un rôle décisif* serait acceptée par le même graphe. Nous pourrions différencier dans ce graphe les constructions à complément indirect *à* N_2 qui autorisent le pronom pré-verbal *lui* des autres. Une vérification de chacune des entrées de **C1IPN** montre que cette analyse serait correcte. Cependant, il faudrait alors aussi rattacher[6] le pronom *en* à un complément en *de* comme dans la phrase (69). Cependant, cela ne «marche» pas pour la phrase (70) et cette propriété est indiquée dans les colonnes de la table: *Prép Nhum = ppv*.

(69) *Luc va bien, j'en reçois des nouvelles quotidiennes.*

(70) *Max a reçu un affront de Luc, *?Il en a reçu un affront.*

La reconnaissance du pronom doit donc se faire de la manière suivante: «Si une *PPV* est présente, alors on ne recherche pas le groupe *Prép* N_2 correspondant. La propriété *Prép Nhum* = *ppv* doit alors valoir + et dans ce cas, *PPV* = *en* si *Prép* = *de*, *PPV* = *<lui>* si *Prép* = à, *PPV* = *y* si *Prép* = *Loc.*» Cette propriété est une propriété de pronominalisation «libre». Le graphe de la figure 7.7 traduit cette reconnaissance pour les pronoms *en* et *lui+leur+te+me+nous+vous*.

De la même façon, nous avons indiqué que la description de la phrase 22 nécessitait de tenir compte du *et* relié à *entre*: cette propriété est aussi une propriété libre d'un groupe prépositionnel en *entre*.

6. En vérifiant que cela est correct pour chaque verbe.

Enfin, les constructions pour lesquelles *Prép = avec* comme (71), peuvent être symétrisées comme dans (72). Cette opération n'est cependant pas systématique (73), (74).

(71) *Luc règle un conflit avec Léa.*
(72) *Luc et Léa règlent un conflit.*
(73) *Jean fait un parallèle avec les réseaux hydrauliques.*
(74)* *Jean et les réseaux hydrauliques font un parallèle.*

Cette propriété est encore une propriété qui n'est pas propre aux constructions figées puisqu'elle s'observe aussi dans les constructions libres. Nous pouvons encore décrire ce phénomène pour la table `C1IPN` à l'aide d'un graphe patron.
Nous pouvons continuer à faire l'inventaire des transformations s'appliquant à nos constructions figées (s'appliquant aussi aux constructions libres) : passif, insertion d'adverbes, extraction de compléments, pronominalisation de compléments, négation, etc. Toutes ces transformations doivent figurer dans notre graphe patron, elles permettent de reconnaître les formes différentes prises par une structure[7]. Nous pouvons représenter chacune de ces transformations sur notre graphe patron. Mais quand nous les représentons simultanément, nous atteignons des limites de lisibilité du graphe, et la démarche n'est pas *pratique* comme nous voulions qu'elle le soit (voir page 56). Nous devons représenter tous les chemins correspondant à tous les cas de figure, ce qui entraîne des répétitions dans la description. De plus, les phénomènes que nous décrivons ici ne sont pas propres à ces constructions verbales figées.
Pour simplifier cette description, il faudrait pouvoir générer le graphe patron contenant toutes les structures possibles automatiquement à partir de la description de chacun des phénomènes.

Les ambiguïtés

Les phrases 5 et 17 montrent des cas d'ambiguïté locale contenant des structures verbales figées. Une grande partie de ces ambiguïtés provient d'une autre structure verbale figée (comme 17), et certaines disparaissent lors de la projection qui ne conserve que la forme la plus longue. Sur un extrait de 1 000 occurrences de structures verbales figées, après intersection, nous n'avons trouvé que 3 ambiguïtés résiduelles (par exemple (75), et (76)). Ce qui donne un taux de reconnaissance d'environ 99,7%. Cependant, il faut bien voir que ces ambiguïtés ne représentent pas un problème formel à notre procédure : tant que nous travaillons sur l'automate du texte, la notion d'ambiguïté n'existe que dans le cas où il existe deux chemins différents reconnus partant d'un début de phrase, et terminant sur une fin de phrase.

(75) En **fait, de plus en plus, (les affaires sérieures)**)$_{C1I}$ se traitent...
(76) [elles] n'expliquent pas **(les traits) tirés**$_{C1IPN}$ et le manque d'entrain.

7.2.3 Les différentes tables appliquées

Nous avons appliqué l'ensemble des grammaires dont nous avons mentionné l'existence dans cette partie, ainsi que toutes les tables d'adverbes figés, les dictionnaires de noms composés, les tables décrivant des prépositions et des adjectifs figés, et les tables de constructions verbales figées dont nous donnons la liste et la taille ainsi que des exemples d'occurrences dans un texte de style journalistique comportant un million de mots. Les

7. C'est le point de départ des graphes patrons.

résultats numériques que nous donnons ici correspondent à la première passe de la reconnaissance sur le texte (c'est la raison pour laquelle ils ne correspondent pas avec ceux donnés dans la partie 11). Des résultats plus généraux figurent dans cette partie.

Taille	Table	Nb Occ.	Premières occurrences dans le texte.
1 092	C0C	87	çà va pas ! çà y est, le soleil brille, qui fait quoi ? le vent tourne
703	C0E	3 411[8]	honneur aux dames, circulez ! À la trappe !
518	C0PC	2	Le champagne coulait à flots, le sourire étaient sur toutes les lèvres
858	C0PN	115	Le temps presse, Il y va de, la chance sourit à, les charges pesant sur
430	C0Q	298	On peut penser que, il est temps pour, on dirait que, Dieu sait pourtant que
2 004	C1D	196	arrêter les frais, cessez le feu[9], les faits qui lui sont reprochés, expédier les affaires courantes
623	C1DPN	207	imposant la présence de, donne la réplique, relance le débat sur, donne le jour à
806	C1G	109	atteindre cet objectif, marquer son territoire, rédige ses mémoires
479	C1GPN	98	fait son entrée, apporte son soutien à, Présente aussi mes félicitations à, conserver ses distances à l'égard de
2 280	C1I	307	tenait une conférence, prend d'autres proportions, ferait un calcul dangereux, essuyer de nombreux revers
1 310	C1IPN	499	cette prière dite, faire de nouveaux sacrifices, Ces attaques lancées par un mouvement, prendre éventuellement des mesures coercitives contre
1 581	C1P2	187	laisser le temps au temps, font flèche de tout bois, tire le mieux son épingle du jeu, laissent un goût amer
768	C1R	210	prend corps, reprendre pied, faire un point, devenue femme
600	C1RPN	590	lui tiennent compagnie, porte atteinte à, coûtent chaque jour plus cher, dit sobrement adieu aux
252	C5	107	prêterait plutôt à sourire, répondre à leurs demandes, tombe à pic, échapper des décennies durant au regard
392	C6	350	est montée de toutes pièces, mettent en cause, mettant en avant, mener à bien
194	C7	217	prend discrètement plaisir à, met un terme à, la préférence est donnée à, faisant allusion au
417	C8	475	a envie de, avoir la volonté de, avait conscience de, en tenir compte
487	CADV	233	iront un peu plus loin, passent mal, en arriver là, dire vrai

8. Cette valeur élevée est due à la présence de phrases figées de type : *allons !*, ou *à nous !* dans la table. Ces entrées sont recherchées sans vérification que l'ensemble forme une phrase complète, ce qui engendre des interprétations fausses. Ces interprétations ne seront cependant pas conservées grâce à l'heuristique de la longest-match.

9. Il s'agit ici de la forme nominale : *un cessez le feu.*

Taille	Table	Nb Occ.	Premières occurrences dans le texte.
992	CAN	78	ferment les yeux, sauver la face, lui complique la tâche, déblayer le terrain
1 182	CDN	444	recevoir la visite de, prendre le contrepied de, sonner le glas de, a essentiellement valeur de
523	CPN	88	tomber dans les bras de, va bien au delà, tombe aux mains des, venir à bout de
316	CPP	381	arrivée à échéance, obéissent à, en prend à son aise, leur venir en aide
520	CPPN	1 081	mettent en garde contre, mettre au point, Condamné en 1989 à trente ans de prison, faire usage de la force
505	CPPQ	495	rendre compte de, tient compte de, soient mises en oeuvre pour, est seulement tenu
648	CPQ	499	veut dire, prendre le temps de, aient lieu, laisser passer
885	CNP2D	119	montré du doigt, prendra au passage, sont présentés à la signature, avait été chassé du pouvoir
679	CNP2R	48	gagner en un jour, prenait sous son aile, est suspendu de ses fonctions, traduit en trente trois langues
826	CNP2Z	388	est pris pour cible, écrite en braille, mettre en scène, remettre en état
1 045	CP1D	87	revienne à la raison, partent rapidement avec la caisse, veille au grain
762	CP1R	48	sorti de son lit, est allé dans le bon sens, sont restés un peu sur leur faim, arrive à son terme
874	CP1Z	106	tourner en rond, reviendra désormais à eux, demeurent sans réponse, meurent de froid
572	CV	135	entendant beaucoup parler, avait mieux à faire, se laissait aller, se faire entendre
132	A12	0	
604	A1P2	17	a aussi son mot à dire, avait d'autres chats à fouetter, ont du pain sur la planche, a aujourd'hui pignon sur rue
75	A1PC	0	
208	A1PN	237	a recours à, donner confiance, mettaient leur nez jusque dans
103	ANP2	0	
357	E1	59	est aussi un fait, est une bonne chose, être bien plus important, seraient également la règle
1319	EAPC	43	est âgé de quarante neuf ans, est toujours le même, était fait pour, est prévu pour
782	EDN	163	est un acte de, est le dernier de, est la chance de, sont aussi ses amis
418	ENPC	0	
699	EPA	295	est plus fort que, Etre mal aimé, est trop beau pour être vrai
1 368	EPC	0	
724	EPCA	26	sont sous contrôle judiciaire, est au rendez vous, est sur la corde raide, est au sens strict

Taille	Table	Nb Occ.	Premières occurrences dans le texte.
957	EPCDC	14	sont au coeur du problème, est même dans la nature des choses, être dans le peloton de tête, mettre au goût du jour
935	EPCDN	306	sera aussi au menu de, serait de la responsabilité de, est donc à l autre bout de, met en forme
205	EPCPC	5	était en avance sur son temps, est à feu et à sang, sont en noir et blanc, étaient à prendre ou à laisser
911	EPCPN	357	est en relation également avec, est en vogue, est en contact avec, serait elle en mesure de, met en évidence
184	EPCPQ	170	est loin de, est en train de, est surtout à cause de, sommes là pour, est le seul à
1 161	EPDETC	176	sont au chômage, était une heure du matin, sommes ici au théâtre, était alors sur la fin
102	F2-1	47	faire une démonstration de sérieux, faisait des déclarations, font encore de larges concessions à leurs origines protestantes
414	FNPN	49	font un détour par, fait néanmoins une percée impressionnante dans, entraîner un mouvement naturel vers
351	YA	22	il y a pire, il y a de fortes chances pour, il n y aura aucun problème, il y a lieu de
38 131		11 110[10]	

Ces 11 110 occurrences couvrent une surface d'environ 35 000 mots. Cela signifie qu'un sur 30 environ fait partie d'une structure verbale figée, c'est-à-dire que près d'une phrase sur trois en contient une. Pour reconnaître ces différentes entrées, nous avons utilisé 69 tables du lexique-grammaire correspondant à 45 124 graphes (engendrés pour chaque entrée) et 238 graphes. Ces chiffres tiennent compte de la reconnaissance des autres unités figées préalables, ainsi que des groupes nominaux simples. Le temps de compilation total de ces quelques 50 000 graphes a été de 11 min et 15 secondes, soit environ 1,3 centième de seconde par graphe.

7.2.4 Extrait de texte transformé

Nous donnons ici un extrait de texte dans lequel les unités figées ont été reconnues. Nous n'avons pas indiqué les groupes nominaux reconnus puisque ceux-ci sont «libres», et que leur reconnaissance n'a pour but que de reconnaître les structures verbales figées. Nous avons souligné quelques séquences auxquelles la reconnaissance devrait être élargie.

(**Neuf mois après**)$_{\texttt{Prép+Date}}$ sa nomination à l'(**hôtel Matignon**)$_{\texttt{N+Loc}}$, (**Edouard Balladur**)$_{\texttt{N+NPP}}$ triomphe dans les baromètres des principaux instituts chargés de mesurer l'état de l'(**opinion publique**)$_{\texttt{N+NA}}$. Personnage politique (**de l'année**)$_{\texttt{N+ENPC}}$, il (**est éclipsé par**)$_{\texttt{V+Passif}}$ dans la (**société civile**)$_{\texttt{N+NA}}$ (**ne guère que**)$_{\texttt{ne...que}}$ (**l'abbé Pierre**)$_{\texttt{N+NP}}$, selon un sondage de CSA pour (**le Parisien**)$_{\texttt{GN+NJournal}}$, ce qui donne (**une autre**)$_{\texttt{Dét}}$ mesure de sa performance. (**A deux reprises**)$_{\texttt{ADV+PDETC}}$ pourtant, la faveur dont le (**premier ministre**)$_{\texttt{N+Titre}}$ bénéficie (**auprès de**)$_{\texttt{Prép}}$ l'opinion (**a semblé sur le point**

10. Cette valeur n'est pas égale à la somme du nombre d'occurrences de chaque table, puisqu'il peut y avoir des ambiguïtés entre différentes entrées: seule la plus longue est conservée dans ce compte.

de se dissiper)V+EPCPQ : (en juin)ADV+Date, (tout d'abord)ADV+PCDC, après l'annonce des (mesures de rigueur)N+NDN, parmi lesquelles la hausse de la CSG ; (en octobre)ADV+Date, ensuite, (du fait de)CONJ les (tensions sociales)N+NA qui se (allaient cristalliser)V+Mod dans le conflit (Air France)N+Société.Dans les deux cas, (la tendance a été inversée)V+C1I+Pas. .

(En août)ADV+Date, explique (Hugues Portelli)N+NP, qui veille sur les (courbes d'opinion)N+NDN (au sein de)Prép+PCDN le (cabinet du premier ministre)N+Ministere, (il y a eu)IlYA le consensus monétaire après la crise de juillet et, pour cette (fin d'année)N+NDN, le (premier ministre)N+AN (a profité)V+PC , (à la fois de)Prép+PCDN la gestion du conflit d'(Air France)N+Société, (des actions menées contre par)V+C1IPN+Passif (Charles Pasqua)N+NP, notamment (le FIS)N+Parti, et, enfin, (du résultat obtenu)V+C1I+Passif sur le GATT."

Cette forte popularité du (premier ministre)N+AN s'accompagne d'un brouillage des repères politiques traditionnels. Seuls les (sympathisants communistes)N+Parti restent réfractaires, depuis sa nomination, à l'attraction de (M. Balladur)N+NP. (A gauche)ADV+PC, (une forte minorité)Dét+Dnom ou (une très faible majorité de)Dét+Dnom (sympathisants socialistes)N+Parti et écologistes, selon les circonstances, y succombe. (A droite)ADV+PC, l'adhésion (sans réserve)??? suscitée tant à l'UDF qu'au RPR (celle-là)PRO se montrant parfois plus " balladurienne " que (celui-ci)PRO s'accompagne d'un résultat très honorable au (Front national)N+Parti, où les avis positifs, (dans le pire des cas)ADV+PCDC, équilibrent les avis négatifs.

(A gauche)ADV+PC, explique (Stéphane Rozès)N+NP, de l'(institut CSA)N+Institut, (E. Balladur)N+NP profite, (d'une part)ADV+PDETC, de la continuité avec les grands choix économiques de ses prédécesseurs socialistes et offre, (d'autre part)ADV+PDETC, une (ligne de résistance)N+NDN au libéralisme effréné de 1986 et de ses (mises en cause)N+NPN sociales. (A droite)ADV+PC, il procure (d'autres)Dét (lignes de résistance)N+NDN : contre le chômage ; contre le libéralisme, pour certains (sympathisants du Front national)N+Parti ; contre le combat Chirac-Giscard, qui (a alimenté)V+PC , (à sa manière)ADV+PDETC, (une partie du)DET+Dnom vote Front national. " " (Edouard Balladur)N+NP est moins attendu sur ses résultats que sur l'assurance d'une (cohésion sociale)N+NA maintenue (contre vents et marées)ADV+PCONJ ", ajoute (Jérôme Sainte-Marie)N+NP, de Louis-Harris. " Ce qui prime, c'est l'idée selon laquelle (l'effort sera réparti)V+C1I+Passif (de façon équitable)ADV+DeFaçon ", conclut (Pierre Giacometti)N+NP de BVA. Ce "positionnement" prive, (par la même occasion)ADV+PAC, le (président de la République)N+NDN, déjà usé par douze années de pouvoir, de tout espace politique.

La structure socioprofessionnelle de la popularité du (premier ministre)N+AN confirme cette analyse. Comme on l'imagine, chez les agriculteurs, les commerçants et artisans, les (chefs d'entreprise)N+NDN et les retraités, (Edouard Balladur)N+NP (obtient d'excellents résultats)V+C1I, mais (il en va de même avec)V+COPC les professions intermédiaires. Les soldes de popularité obtenus chez les ouvriers et chez les employés, (les deux)Dét catégories les moins " balladuriennes ", sont, eux aussi, souvent positifs. " Pour (ces deux)Dét catégories, indique (M. Portelli)N+NP, les avis défavorables (ont doublé)V+PC (pendant ces neuf mois)ADV+DUREE, mais nous (avons conservé)V+PC le même compte d'avis favorables. " " Ses soutiens les plus affirmés les agriculteurs, les commerçants artisans et (chefs d'entreprise)N+NDN (ont décliné)V+PC (pendant l'automne)ADV+Date, (alors que)CONJ, (dans le même temps)ADV+PDETC, le (premier ministre)N+AN (a retrouvé)V+PC un solde positif chez les ouvriers et chez les employés avec le conflit d'(Air France)N+Société ", complète (M. Rozès)N+NP.

Les (milieux financiers)N+NA espagnols maintenant (attendent de savoir)V+Mod si Banesto (sera absorbé par)V+Passif (une autre)Dét banque. Répondant à (des informations parues dans la presse)V+COPC, la (Banque Bilbao Vizcaya (BBV))N+Société (a annoncé)V+PC , (jeudi 30 décembre)ADV+Date, qu'elle (prendrait le contrôle de)V+CDN (ne pas)NEG Banesto. (Du côté français)ADV+PCDN, les (Assurances générales de France (AGF))N+Société se (sont déclarées)V+PC très sereines sur l'issue (des négociations engagées avec)V+C1IPN+Pass. le (groupe bancaire)N+NA espagnol Banesto (quant à)CONJ l'avenir de leur

(**filiale commune**)$_{\texttt{N+NA}}$, l'assureur (**Union y el Fenix (UFE)**)$_{\texttt{N+NP}}$. Les AGF et le Banesto s'(**étaient mis d'accord pour**)$_{\texttt{V+CPPQ}}$, (**le 7 décembre**)$_{\texttt{ADV+Date}}$ une fusion d'UFE avec AGF Seguros.

7.3 Conclusion

Nous avons montré comment nous pouvions reconnaître plusieurs types d'unités figées à partir des ressources linguistiques disponibles: essentiellement les tables du lexique-grammaire et de bibliothèques de graphes. Cette reconnaissance est faite à partir des outils décrits dans le chapitre 4. Nous avons vu que, au fur et à mesure de la description des différents phénomènes, la reconnaissance ne pouvait pas se limiter à l'application d'outils informatiques à des «tables», mais que l'analyse linguistique des phénomènes est profondément liée à la construction de ces grammaires. En particulier, l'examen approfondi de toute unité lexicale conduit à la particularisation de cette entrée. À terme, il semble que chaque entrée lexicale différente aura une grammaire propre construite semi-automatiquement: un graphe sera associé à chaque entrée par un mécanisme analogue à celui des graphes patrons, et des retouches seront ensuite faites manuellement à chacun de ces graphes. Nous avons montré que la description des structures figées utilise des propriétés propres aux structures «libres», et que réciproquement la description de séquences «libres» doit utiliser la même démarche que la description des séquences figées. Nous avons montré l'importance numérique des constructions figées dans les textes (des chiffres plus détaillés sont donnés dans la partie 11). Enfin, le nombre élevé de graphes différents devant être appliqués en passes successives au texte met en évidence un besoin d'algorithmes adaptés.

Bibliographie

[1] L. Danlos. Les expressions figées construite avec le verbe support *être prép. Langages - Larousse*, 1988.

[2] A. Fotopoulou and J. Senellart. Transformation sur les mots composés. Communication au 17e colloque du lexique-grammaire des langues romanes, 1998.

[3] D. Gaatone. *Négation.* Duculot, 1998. Collection Champs Linguistiques.

[4] J. Giry-Schneider. *Les nominalisations en français.* Droz, Genève, 1978.

[5] J. Giry-Schneider. *Les prédicats nominaux en français. Les phrases simples à verbe support.* Droz, Genève, 1987.

[6] J. Giry-Schneider et A. Balibar-Mrabti. Classes de noms construits avec avoir. Rapport Technique, Institut Blaise Pascale, 1993.

[7] G. Gross. Degré de figement dans les noms composés. *Langages - Larousse*, 90:57–72, 1988.

[8] G. Gross. *Les expressions figées en français : noms composés et autres locution.* Ophrys, 1995.

[9] M. Gross. Une classification des phrases figées du français. *Revue Québécoise de linguistique*, 11(2):299–311, 1982.

[10] M. Gross. *Grammaire transformationnelle du Français – Syntaxe de l'adverbe*, volume 3. Cantilène, 1986.

[11] M. Gross. *Grammaire transformationnelle du Français – Syntaxe du nom.* Cantilène, 1986.

[12] M. Gross. Les phrases figées en français. In *L'information grammaticale*, volume 59, pages 36–41. UQAM, Montréal, 1988.

[13] M. Gross. Les formes être prép x du français. *Linsguiticæ Investigationes*, XX:217–270, 1996.

[14] M. Gross. Les structures figées - la négation composée. Rapport Technique, LADL, 1998.

[15] Z.S. Harris. *Mathematical Structure of Language.* Wiley - Interscience, New-York, 1968.

[16] J. Labelle. Verbes supports et opérateurs dans la construction en avoir à un ou deux compléments. *Lingvisticæ Investigationes*, 7(2):237–260, 19832.

[17] D. Maurel. *Reconnaissance des séquences de mots par automates. Adverbes de date du français.* Thèse de doctorat, Université Paris 7, 1989.

[18] C. Muller. *La négation en français. Syntaxe, sémantique et éléments de comparaison avec les autres langues romanes.* Droz, Genève, 1991.

[19] M. Piot. La restriction *seul(e)(s)* et *ne...que* (aperçu à partir de leur distribution). *Recherche Linguistique - Université Paris 8 - Vincennes*, 3:226–264, 1975.

[20] J. Senellart. Locating noun phrases with finite state transducers. In *Proc. of ACL-COLING'98*, 1998.

[21] J. Senellart. Reconnaissance automatique des entrées du lexique-grammaire des phrase figées. Rapport Technique 57, LADL, 1998. en parution dans Travaux de Linguistique.

[22] J. Senellart. Tools for locating noun phrases with finite state transducers. In *Workshop on nominals, ACL-COLING'98*, 1998.

Troisième partie

Indexation

Chapitre 8

Introduction

LA RECONNAISSANCE DES EXPRESSIONS FIGÉES que nous avons décrite dans la partie précédente est basée sur l'application d'un nombre important[1] de transducteurs à un texte. Ces transducteurs proviennent de bibliothèques de graphes (par exemple les dates, les titres de personnalité) et de la transformation des entrées du lexique-grammaire pour les structures verbales figées. Cette collection de transducteurs continuera à s'accroître avec la description d'expressions linguistiques figées et libres. Nous avons vu, que ces transducteurs contiennent des appels à des «sous-transducteurs».

La première étape de l'application de ces transducteurs au texte est la phase de reconnaissance des expressions qui y sont décrites. La seconde étape est la phase de transformation des unités reconnues. Nous discutons dans cette partie exclusivement de la première phase : nous ne considérerons donc que la partie «entrée» de ces transducteurs, c'est-à-dire uniquement des automates finis. Une possibilité pour effectuer cette reconnaissance est de faire l'union de tous ces automates, de les mettre sur un seul niveau[2] et de déterminiser l'automate résultant. Plusieurs arguments s'opposent à cette démarche. Certains sont d'ordre purement pratique, d'autres un peu plus «théoriques» :

- L'algorithme de déterminisation d'automate peut avoir une complexité exponentielle en fonction du nombre de noeuds [12], [19]. Cela serait ici catastrophique. De plus, cette déterminisation doit être re-effectuée après une modification sur une seule de ces grammaires.
- Imaginons que l'automate résultant représente à terme, l'ensemble de la grammaire du français. Il y aura, alors, autant de transitions partant du noeud initial que de mots différents pouvant commencer une phrase en français, c'est-à-dire de l'ordre de 100,000 (un mot sur deux du vocabulaire environ ?). L'algorithme utilisant un tel automate n'est vraisemblablement pas le même que celui qui met en œuvre des automates sur un alphabet de quelques centaines de mots.
- La structure linéaire de texte est une structure très provisoire puisque nous avons montré que les opérations réelles sont faites sur la structure d'automate de texte.

1. Le nombre des expressions figées est ici de l'ordre de 50 000, chaque transducteur ayant une centaine d'états environ.

2. C'est-à-dire de remplacer chaque appel à un sous-graphe, par le sous-graphe lui-même. Cela nous empêcherait de définir des graphes récursifs dont nous avons montré l'existence au chapitre 4.2.3, la récursivité devrait être remplacée par un mécanisme fonctionnant en plusieurs passes.

L'algorithme de reconnaissance des expressions décrites par l'automate dans le texte doit donc pouvoir s'adapter à la reconnaissance d'expressions dans un automate acyclique. De plus, cette structure sera dynamique, puisque l'application des transformations modifie justement la représentation du texte.

- La distribution naturelle des mots sur le texte ne serait pas utilisée, alors que cette distribution est remarquable (la loi de Zipf, que nous décrivons en annexe B).

- Toute la factorisation naturelle faite par le concepteur des grammaires est perdue lors du processus d'«étalement de l'automate». Cela a deux implications: il devient très difficile de corriger la grammaire «étalée», puisque l'information concernant l'origine de chaque état est perdue. Ensuite, d'un point de vue informatique, l'information qu'un motif se répète à différents endroits est aussi perdue, ce qui supprime des factorisations potentielles. Cette information devrait pourtant être utilisée pour optimiser l'algorithme de reconnaissance.

- Enfin, la recherche isolée d'un seul graphe est une extraction d'informations du texte. Chaque graphe représentant un phénomène particulier, il doit être possible de les appliquer indépendamment. Une complexité linéaire en fonction de la taille du texte ne peut alors être admissible pour l'application d'un seul graphe.

Pour ces différentes raisons, nous ne regrouperons pas tous ces automates et nous conserverons leur hiérarchie naturelle. Nous devrons donc disposer d'algorithmes permettant d'appliquer un graphe seul très rapidement, vu le nombre total de graphes développés.

Une autre de nos préoccupations est d'être capable de gérer des textes de taille importante, cette préoccupation est relativement récente dans le domaine du langage naturel. En effet, des textes de taille considérable sont désormais accessibles (CD-ROM, Web) et la gestion de ces textes nécessite des algorithmes adaptés. Cette préoccupation est liée à la notion d'index. Dans [27] et [18], des algorithmes permettant d'indexer des textes de grande taille et des algorithmes permettant de les compresser sont décrits. Nous avons étudié dans [17], la compression d'index en tenant compte de données linguistiques.
La notion d'index dans la littérature est cependant ambiguë. Il peut s'agir d'une structure (*partial inverted text*) qui indique la présence ou l'absence d'un *token*[3] dans un texte ($I(a) = \delta(a \in \mathcal{T})$) ou dans le cas d'une collection de textes[4], elle indique dans quel texte se trouve un mot donné ($a \in \mathcal{T} \Rightarrow \mathcal{T} \in I_a$.). Contrairement à ces derniers types d'index (équivalents aux index de livres), un autre type d'index (dont la forme de base est appelée en anglais *full inverted text*) comporte toute l'information du texte: à chaque mot est associé l'ensemble des positions de ces mots dans le texte initial. Enfin, on trouve aussi le terme d'index pour désigner toute structure précalculée sur le texte (par exemple, les *Patricia tree* [11]) dont la manipulation donne des adresses absolues dans le texte initial.

Pour pouvoir combiner la préoccupation de gestion d'un grand nombre d'automates à la préoccupation de gestion d'un grand texte, nous proposons une structure d'index (basée sur l'index des mots) plus riche, dans laquelle les automates eux-mêmes auront

3. Suivant l'utilisation, le *token* peut être un motif comme par exemple dans une base de donnée d'ADN [25], un caractère, un mot, etc.

4. Par exemple, les index des moteurs de recherche sur le Web.

une entrée. Nous montrerons comment cette notion d'index de texte, s'étend naturellement à la notion d'index d'automate du texte, ce qui est notre objectif. Nous montrerons également que la structure d'index peut-être maintenue dynamiquement et qu'elle offre un moyen d'accéder à une structure de texte plus riche (la distinction entre structure d'index et structure de texte ne pouvant plus être si facilement faite).

Nous présentons donc cette structure en omettant la construction de l'index dont le coût est linéaire en fonction de la taille du texte ([27], [3]) et sa compression. La taille d'un index sans algorithme de compression est approximativement le tiers de la taille du texte. La compression n'est donc pas une préoccupation majeure: nous ne possédons pas de textes qui ne tiennent pas sur support magnétique avec leurs index.
Nous montrons ensuite comment cette structure d'index peut être utilisée pour la reconnaissance d'expressions complexes. Nous serons amené à considérer essentiellement deux types d'algorithmes permettant de localiser des expressions complexes à l'aide d'index:

- Les algorithmes *Index-Index*, qui effectuent la totalité du travail de recherche sur l'index.
- Les algorithmes *Index-Texte*, qui utilisent l'index pour donner un point de départ à la recherche mais qui terminent la recherche sur la structure du texte.

Nous comparons notre approche à d'autres approches de reconnaissance dans des textes d'expressions décrites par des expressions régulières (*pattern-matching*). Cette comparaison sera néanmoins théorique. En effet, nous manipulons une structure basée sur celle du texte linéaire, mais enrichie à l'aide de dictionnaires par exemple. Nous montrons que la comparaison est alors limitée à des sous-domaines très restreints des domaines d'applications de nos algorithmes. De plus, nous ne connaissons pas d'autres applications manipulant aussi intensément la localisation d'expressions décrites par un si grand nombre d'automates avec lesquelles il serait possible d'établir des comparaisons directes. Enfin, dans notre application, certaines considérations sur les automates finis manipulés ne sont pas pertinentes (par exemple, l'existence de motifs de longueur non bornée). Nous utilisons le fait que la distribution des mots (et des unités plus compliquées) sur un texte suit la loi de distribution de Zipf, ce qui limite encore la possibilité de comparaison avec des algorithmes plus généraux. Remarquons que cette distribution de Zipf est la seule caractéristique des textes en langage naturel que nous exploitons. Aussi nous montrerons des tests sur des textes non pas en langage naturel, mais généré aléatoirement suivant une loi de Zipf. Cette loi étant observée dans diverses distributions naturelles, nos algorithmes ne sont pas *a priori* dédiés au langage naturel.

Chapitre 9

L'indexation de textes

9.1 Définitions

Nous allons considérer un texte comme une séquence constituée de mots du lexique et de différents signes de ponctuation (*tokens*). Les caractères «espaces» sont donc par défaut ignorés (on suppose qu'il y en a un entre deux mots ou signes de ponctuations). La suppression des espaces en tant que caractères pertinents supprime éventuellement des informations de mise en page (comme l'alignement) ou de typographie. Par exemple, avant la virgule il y a pas d'espace: la présence d'un espace à cet endroit pourrait donc être significatif d'un effet de mise en page, de l'utilisation de la virgule dans un autre but, ou d'une erreur. De telles analyses relèvent d'un niveau tellement fin qu'elles ne peuvent pas être utilisées dans l'état actuel des analyses automatiques.

Définition 1 *Soit $\mathcal{T}$ un texte. On appelle* **token** *toute sous-séquence t de caractères alphanumériques de $\mathcal{T}$ vérifiant les conditions suivantes:*

1. *t est exclusivement composé de lettres, de chiffres, ou de caractères de ponctuation.*
2. *Il n'est pas possible de prolonger t (ni vers la droite ni vers la gauche) en conservant la condition précédente.*

On appelle l'ensemble $\mathcal{A} = \{a_1, ..., a_n\}$ de tous les tokens a_i de $\mathcal{T}$: l'**alphabet du texte** $\mathcal{T}$, les a_i sont les différents **caractères** de l'alphabet.
Le découpage d'un texte en tokens, est traditionnellement appelé la segmentation du texte. En fait, la notion d'index que nous allons présenter, ainsi que les opérations que nous effectuerons, ne dépendent pas de cette segmentation. Nous verrons que l'efficacité de certains de nos algorithmes repose sur une distribution spéciale de l'alphabet dans le texte et que cette distribution n'existe que si on segmente un texte en mots, aussi elle n'existe plus si on segmente le même texte en lettres. De plus, pour les raisons déjà évoquées et discutées en 3.2, cette définition du mot est un choix arbitraire, mais elle est à la base des grammaires que nous avons développées.
Dorénavant, nous considérons qu'un texte est la concaténation de lettres de l'alphabet:

$$\mathcal{T} = a_{i_1} a_{i_2} ... a_{i_{|\mathcal{T}|}}$$

où $|\mathcal{T}|$ est la taille du texte. Un mot (sauf mention spéciale) est une suite de lettres $a_{i'_1} a_{i'_2} ... a_{i'_k}$ qu'il ne faudra pas confondre avec le mot «linguistique» qui est une lettre de l'alphabet $\mathcal{A}$.

Tout mot $m \in \mathcal{A}^*$ de $\mathcal{T}$ est repéré par deux paramètres : sa position (i.e. si $\mathcal{T} = umv$, la position de m est $|u|$) et sa longueur $|m|$.

Définition 2 *Soit m une sous-séquence de $\mathcal{T} = umv$, on appelle le couple $(|u|, |m|)$: le* **descripteur** *de m.*

Définition 3 *Pour tout $a_i \in \mathcal{A}$, on désigne par* **entrée de l'index correspondant à** *a_i (ou quand il n'y a pas d'ambiguïté, l'***index de** *a_i), l'ensemble I_{a_i} des descripteurs de chaque occurrence de a_i dans le texte.*

Définition 4 *L'***index simple** *$\mathcal{I}$ d'un texte $\mathcal{T}$ de vocabulaire $\mathcal{A}$ est l'ensemble de toutes les entrées d'index des lettres de $\mathcal{T}$:*

$$\mathcal{I} = \{I_{a_i} \mid a_i \in \mathcal{A}\}$$

Exemple : Considérons le texte $\mathcal{T}$ suivant :
abaabcabbdcabacabaababbdabadacbaabaca
Le vocabulaire de ce texte est $\mathcal{A} = \{a, b, c, d\}$.
L'index de ce texte est $\mathcal{I} = I_a \cup I_b \cup I_c \cup I_d$ avec [a] :
$I_a = \{1, 3, 4, 7, 12, 14, 16, 18, 19, 21, 25, 27, 29, 32, 33, 35, 37\}$
$I_b = \{2, 5, 8, 9, 13, 17, 20, 22, 23, 26, 31, 34\}$
$I_c = \{6, 11, 15, 30, 36\}$
$I_d = \{10, 24, 28\}$

[a] Chaque entrée d'index de lettres est composée de couple de la forme $(p, 1)$ par définition des lettres. Pour simplifier l'écriture, nous omettons systématiquement la longueur.

Il est clair, que toute l'information [1] du texte $\mathcal{T}$ est contenue dans l'index $\mathcal{I}$. Réciproquement, l'index ne contient pas plus d'information que le texte. Nous parlerons désormais de texte sous sa forme linéaire $\mathcal{T}$ et sous sa forme indexée $\mathcal{I}$.

Dans l'index du texte comme il vient d'être défini, seuls les caractères de l'alphabet sont indexés. Nous parlerons d'**enrichissement de l'index**, quand nous y introduirons les index de séquences complexes que nous définirons.

Nous étendons facilement la notion d'entrée d'index aux mots sur $\mathcal{A}$ et à tout sous-ensemble rationnel de $\mathcal{A}^*$:

Définition 5 *Soit $E \in rat(\mathcal{A}^*)$, l'entrée d'index de E est :*

$$I_E = \{(||u||, |\mathcal{T}| - |u| - |v|) \mid u, v \in \mathcal{A}^* \text{ tels que } \mathcal{T} = uEv\}$$

Il s'agit simplement de l'entrée d'index correspondant à toutes les occurrences de l'expression rationnelle dans le texte.

Exemple : Dans le texte de l'exemple précédent :
L'index I_{aa} de aa vaut $\{(3, 2), (18, 2), (32, 2)\}$.
L'index I_{ba^*b} de ba^*b vaut $\{(2, 4), (8, 2), (17, 4), (20, 3), (23, 2), (31, 3)\}$.

1. Mises à part, d'éventuelles subtilités autour de l'usage de l'espace. Mais même quand nous parlerons de compression, nous comparerons la compression à une compression du texte linéaire complet, les anamolies d'utilisation du caractère espace étant marginales.

Plus généralement, nous appellerons tout ensemble de la forme $\{(p,l)\,|\,p \leq |\mathcal{T}|$ et $1 \leq l \leq |\mathcal{T}| - p\}$ une entrée d'index. On appelera **index complet** du texte l'ensemble $\mathcal{I}$ des entrées d'index différentes :

$$\mathcal{I} = \{I\,|\,I = \{(p,l)\,|\,1 \leq p \leq |\mathcal{T}|,\ \text{et}\ 1 \leq l \leq |\mathcal{T}| - p\}\}$$

Exemple : Dans nos exemples précédents :
L'entrée d'index des sous-séquences comprises entre deux lettres **c** et adjacentes à ces lettres est $I' = \{(7,4),(12,3),(16,14),(31,5)\}$:
abaabcabbdcabacabaababbdabadacbaabaca
I' ne correspond pas à l'index d'une expression rationnelle, puisque la séquence **aba** par exemple, possède d'autres occurrences dans le texte qui ne sont pas indexées.

Définition 6 *Soit I une entrée d'index, on dit que I est de* **longueur fixe** $l' = ||I||$, *si* $\forall\,(p,l) \in I,\ l = l'$. *Sinon on dit que I est de* **longueur variable** *et $||I||$ n'est pas défini.*

Exemple : $||I_a|| = ||I_b|| = ||I_c|| = ||I_d|| = 1$. $||I_{aa}|| = 2$ et I_{ba^*b} ainsi que I' ne sont pas de longueurs fixes : $||I_{ba^*b}||$ et $||I'||$ ne sont pas définis.

Nous verrons que le comportement des index est différent suivant qu'ils sont de longueur fixe ou de longueur variable. Les premiers (essentiellement les index des lettres de $\mathcal{A}$) ont une très grande maniabilité.
En mémoire, les possibilités de structure pour l'index sont vastes et nous en discuterons plus loin. Dans tous les cas, les structures feront explicitement apparaître le nombre d'occurrences de chaque entrée : nous montrerons en annexe B que cette information pourrait facilement être approximée en temps constant pour des distributions «naturelles», mais pour la simplicité de la discussion, nous supposerons que dans les structures utilisées, ce nombre d'occurrences est une donnée. Nous notons $|I_{a_k}|$ le nombre d'occurrences du mot a_k dans le texte (ou encore la fréquence de a_k), c'est à dire le cardinal de I_{a_k}.

9.1.1 Désambiguïsation terminologique

Parce que le terme d'*index* désigne plusieurs notions, nous récapitulons les différentes acceptions sous lesquelles il sera utilisé. Deux critères permettront de ne pas confondre ces sens : la notation en lettre ronde $\mathcal{I}$ (vs. la notation I) et le contexte. Nous donnons également la structure figée par laquelle chaque sens est exprimé, mais chacune des ces structures pourra être utilisée sous son abbréviation : **index**.

- $\mathcal{I}$, l'**index (simple)** du texte est l'ensemble de toutes les entrées d'index des lettres du texte.
- $\mathcal{I}$, un **index (enrichi)** du texte, est un ensemble d'entrées d'index contenant au minimum l'index simple.
- $\mathcal{I}$, l'**index (complet)** du texte, est l'ensemble de toutes les entrées d'index du texte.
- I, l'**(entrée d')index** d'une lettre ou d'une expression rationnelle *dans* le texte est la liste de tous ses descripteurs.

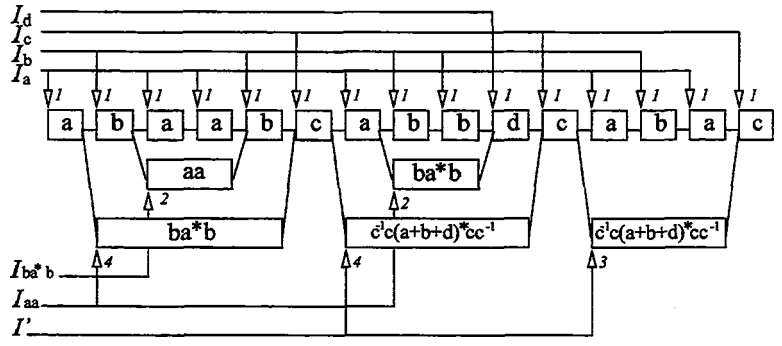


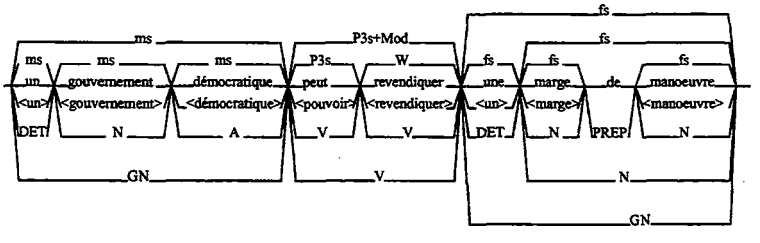
FIG. 9.1 – *Index de texte linéaire. La «structure induite» décrite par* $\mathcal{I} = I_a \cup I_b \cup I_c \cup I_d \cup I_{aa} \cup I_{ba^*b} \cup I'$ *est un graphe direct acyclique dont tous les nœuds partent et arrivent sur un chemin donné : le texte linéaire.*

9.2 Structure induite par l'index

Un texte linéaire sous sa forme indexée consiste en l'ensemble des entrées d'index de ses caractères (ces entrées d'index ont toutes par définition, une longueur fixe égale à 1). Un index quelconque I (représentant ou non une expression rationnelle) «ajoute» des nœuds parallèles au texte linéaire. Les arcs d'entrée de ces nœuds partent du texte linéaire et les arcs de sortie arrivent au texte linéaire (cf figure 9.1). Il y a donc plus d'information dans l'index que la forme linéaire du texte. La structure induite du texte correspondante est une sorte d'«**automate plat**».

Exemple : Pratiquement, la structure ainsi créée permet l'enrichissement du texte à l'aide d'étiquettes et d'informations sémantiques. Le graphe de la figure suivanteest celui de la séquence *un gouvernement démocratique peut revendiquer une marge de manœuvre* enrichie d'étiquettes et d'informations linguistiques.
L'index correspondant à cette figure contient par exemple les entrées suivantes :

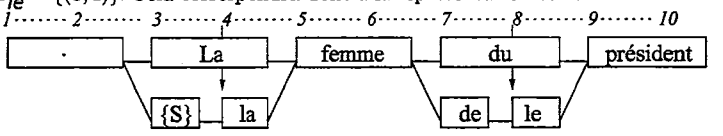
- l'index de la forme *un* : $I_{un} = \{(1,1)\}$;
- l'index du lemme <un> : $I_{<un>} = \{(1,1),(6,1)\}$;
- l'index de l'information +ms : $I_{+ms} = \{(1,1),(2,1),(3,1),(1,3)\}$;
- l'index de l'étiquette <N> : $I_{<N>} = \{(2,1),(7,1),(9,1),(7,3)\}$.



Sous cette forme, les sous-chemins entre deux nœuds du texte initial ont une longueur

maximale d'un nœud. Il ne sera donc pas possible de «défaire» par exemple des contractions comme *du*⟶ *de le*. Cependant, on peut facilement contourner cette définition en introduisant dans la définition de la lettre la propriété qu'elle a une longueur $n > 1$ dans le texte (ce qui revient à considérer que le texte a une longueur $n \times |\mathcal{T}|$.

Exemple : Considérons le texte: . *La femme du président* ..., en choisissant $n = 2$, on a les entrées d'index suivantes: $I_{\text{«.»}} = \{(1,2)\}$, $I_{La} = \{(3,2)\}$, $I_{femme} = \{(5,2)\}$, $I_{du} = \{(7,2)\}$, $I_{président} = \{(9,2)\}$.
La procédure du reconnaissance des phrases construira $I_{\{S\}} = \{(3,1)\}$ et $I_{la} = \{(4,1)\}$ et la procédure de traitement de la contraction produira $I_{de} = \{(7,1)\}$ et $I_{le} = \{(8,1)\}$. Cela correspondra donc à la représentation suivante:



Avec un telle représentation, nous pouvons obtenir des structures de graphes directs acycliques complexes représentées par un index sur un texte linéaire en choisissant n grand. Cependant, à part pour pouvoir défaire les contractions comme nous l'avons montré dans l'exemple précédent, cette méthode est lourde à gérer puisque la longueur de chaque «lettre» ne correspond plus vraiment à une longueur réelle dans le texte, il faut pouvoir gérer les «positions intermédiaires» libres et le nombre de ces positions sera de toute façon borné par n. Nous montrerons page 223 comment pouvoir réellement manipuler une structure de graphe complète pour gérer en particulier l'automate du texte. Nous avons néanmoins implémenté cette structure que nous appelons «pseudo-linéaire» (avec $n = 2$, en tenant compte du traitement des contractions et de la séparation en phrase décrite plus haut) pour effectuer des opérations de reconnaissance simples ne nécessitant pas la construction de l'automate du texte. L'avantage de cette structure est sa simplicité, son enrichissement possible qui la différencie d'une vraie structure linéaire et l'efficacité des algorithmes que nous décrirons.

9.3 Opérations de base

Soit I, J deux entrées d'index. Nous effectuerons sur ces entrées les opérations suivantes:

1. la lecture séquentielle de tous les éléments de l'ensemble (cette lecture peut suivre ou non un ordre sur les positions);
2. l'opérateur booléen δ qui indique si une position donnée p appartient à I. On note $\delta_I(p) = (p \in I)$;
3. l'insertion et la suppression d'éléments;
4. le tri;
5. la construction à partir d'une liste de positions non ordonnée;
6. l'union: $I \cup J = \{(p,l) \mid (p,l) \in I, \text{ ou } (p,l) \in J\}$;
7. l'intersection: $I \cap J = \{(p,l) \mid (p,l) \in I, \text{ et } (p,l) \in J\}$;

8. la composition : $I \circ J = \{(p,l) \mid \exists\, l' < l,\ (p,l') \in I,\ (p+l', l-l') \in J\}$;
9. la translation : $\alpha_k(I) = \{(p,l) \mid (p-k,l) \in I\}$.

Pour formaliser la notion d'index, nous considérons essentiellement trois types de structures pour les entrées d'index. La *liste* (notée L), la *liste ordonnée* (notée LO) et *l'arbre binaire équilibré* (noté ABE). Toutes ces structures sont incrémentales, ce qui est une nécessité pour la démarche. Nous n'entrerons pas dans les détails d'implémentation ; les complexités (en temps et en mémoire) que nous donnons sont des complexités moyennes à un coefficient multiplicatif près. Néanmoins, nous indiquons parfois des coefficients multiplicatifs, qui ne devront donc être interprétés qu'en comparaison avec les autres complexités.
Nous allons établir les différentes complexités en temps et en mémoire de ces trois types de structure :

- $t_0^{\geq}$: le temps d'accès au plus petit élément de l'ensemble ;
- $t_{+1}^{\geq}$: le temps d'accès à l'élément immédiatement supérieur à celui en cours ;
- t_0 : le temps d'accès au premier élément de la liste (suivant un ordre défini par la structure) ;
- t_{+1} : le temps d'accès à l'élément suivant, dans cet ordre-là ;
- t_δ : le temps de recherche d'un élément donné dans l'ensemble ;
- t_{ins} : le temps nécessaire pour ajouter un élément à l'ensemble ;
- t_{cons} : le temps nécessaire pour construire la structure à partir d'une liste non ordonnée ;
- $t_\cap$: le temps nécessaire à l'intersection de deux entrées d'index ;
- $t_\cup$: le temps nécessaire à l'union de deux entrées d'index ;
- $t_\circ$: le temps nécessaire à la composition de deux entrées d'index ;
- t_α : le temps nécessaire à la translation d'une entrée d'index ;
- *mem* : la taille en mémoire d'une structure contenant n éléments.

Nous donnons dans la table suivante, les valeurs des complexités précédentes. Leur calcul se déduit trivialement des définitions des structures (pour des raisons de lisibilité, nous donnons les complexités sous la forme $f(n)$ et non pas $O(f(n))$) :

	$t_0^{\geq}$	$t_{+1}^{\geq}$	t_0	t_{+1}	t_δ	t_{ins}	t_{cons}	*mem*
L	n	n	1	1	n	1	n	$2n$
LO	1	1	1	1	$\frac{n}{2}$	$\frac{n}{2}$	$n.\log n$	$2n$
ABE	$\log n$	1	1	1	$\log n$	$\log n$	$n.\log n$	$4n$

Remarque : Pour la structure ABE, $t_{+1}^{\geq}$ se calcule de la manière suivante (cf figure 9.2) : la distance parcourue pour aller de la feuille la plus à gauche à la feuille la plus à droite est deux fois le nombre total de transition : $2n_t$. n_t vaut exactement $n-1$ (le nombre de nœuds dans l'arbre), donc le temps total de parcours est $O(n)$. Le temps de passage d'un

élément à un autre est donc $O(1)$ en moyenne (en tout on passe deux fois par chaque nœud non terminal) ; au pire, il faut remonter tout l'arbre, ce qui donne $\log n$ opérations.

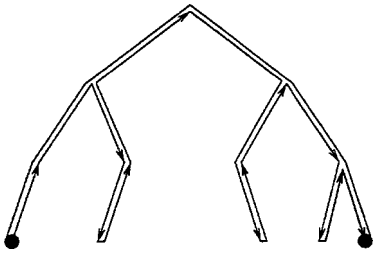


FIG. 9.2 – *Arbre binaire équilibré, parcours dans l'ordre de tous les nœuds*

Pour l'intersection et la composition, nous avons trois possibilités de construction :

1. Lecture séquentielle de I : pour chaque élément, on vérifie l'existence dans J d'un élément pouvant correspondre à l'intersection ou la composition (figure 9.3). Le coût d'une telle méthode est $t_0 + t_{+1}|I|.t_\delta|J|$. Donc, en tenant compte du coût de construction de $I \cap J$, on a :

$$\boxed{t_\cap = t_0 + t_{+1}|I|.t_\delta|J| + t_{cons}(I \cap J)}$$

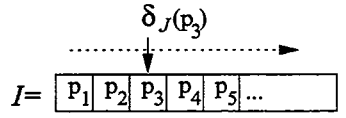


FIG. 9.3 – *Calcul de $I \cap J$ avec une lecture séquentielle de I et une vérification pour chaque élément de l'appartenance à J.*

2. Solution symétrique de la précédente :

$$\boxed{t_\cap = t_0 + t_{+1}|J|.t_\delta|I| + t_{cons}(I \cap J)}$$

3. Lecture séquentielle ordonnée (cf figure 9.4) de I et J, le coût est donc :

$$\boxed{t_\cap = 2t_0^{\geq} + t_{+1}^{\geq}|I| + t_{+1}^{\geq}|J| + t_{cons}(I \cap J)}$$

Nous pourrons choisir dynamiquement quelle méthode adopter, mais *grosso modo*, la première méthode devra s'appliquer dans le cas d'une intersection d'un «gros» index avec un petit, et la troisième, dans le cas de l'intersection de deux index de tailles comparables. Nous avons donc :

$$\boxed{t_\cap = \min\left(t_0 + t_{+1}|I|.t_\delta|J|, t_0 + t_{+1}|J|.t_\delta|I|, 2t_0^{\geq} + t_{+1}^{\geq}|I| + t_{+1}^{\geq}|J|\right) + t_{cons}(I \cap J)}$$

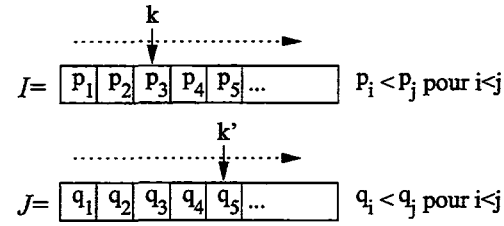


FIG. 9.4 – *Calcul de $I \cap J$ avec une lecture séquentielle ordonnée simultanée de I et J : à tout moment $q_{k'} \leq p_k \leq q_{k'+1}$.*

$$\boxed{t_\circ = \min(t_0 + t_{+1}|I|.t_\delta|J|, t_0 + t_{+1}|J|.t_\delta|I|, 2t_0^{\geq} + t_{+1}^{\geq}|I| + t_{+1}^{\geq}|J|) + t_{cons}(I \circ J)}$$

Remarquons que la complexité de l'opération est liée à la taille de l'ensemble résultant. Comme notre but sera d'ordonner les opérations sur les index de manière à minimiser le coût global, nous verrons qu'il faudra estimer[2] cette quantité.
Pour l'union, seule la troisième méthode proposée ci-dessus peut fonctionner, nous avons donc :

$$\boxed{t_\cup = 2t_0^{\geq} + t_{+1}^{\geq}|I| + t_{+1}^{\geq}|J| + t_{cons}(I \cup J)}$$

et ici, $\max(|I|, |J|) \leq |I \cup J| \leq |I| + |J|$.
Pour α, le nombre d'opérations est exactement le nombre d'éléments de l'index. Donc $t_\alpha = |I|.log|I|$. Cependant, vu la simplicité de l'opérateur translation qui ne modifie pas l'ordre dans la stucture, l'opération n'a pas vraiment besoin d'être effectuée sur l'ensemble de la structure : il suffit qu'un champ de la structure indique de combien tous ses éléments sont translatés. Nous considérerons donc maintenant que :

$$\boxed{t_\alpha = 1}$$

9.4 Propriétés des index et des opérateurs

Pour comprendre la sémantique des différentes opérations définies sur les index, nous devons auparavant en établir quelques propriétés remarquables. Soit $\mathcal{T}$ un texte et $\mathcal{I}$ l'*index complet* de ce texte :

$$\mathcal{I} = \{I \mid I = \{(p,l) \mid 1 \leq p \leq |\mathcal{T}|, \text{ et } 1 \leq l \leq |\mathcal{T}| - p\}\}$$

Lemme 1 Soit $I, J, K \in \mathcal{I}$ des entrées de l'index et $k, l \in I\!N$:

1. La composition $\circ$ est associative : $I \circ J \circ K = (I \circ J) \circ K = I \circ (J \circ K)$.
2. La composition $\circ$ est distributive par rapport à l'union $\cup$: $I \circ (J \cup K) = (I \circ J) \cup (I \circ K)$.
3. La translation α est distributive par rapport à la composition $\circ$: $\alpha_k(J \circ K) = \alpha_k(J) \circ \alpha_k(K)$.

2. Sur des très gros index, des possibilités d'échantillonage pourraient être envisagées. Sur les index que nous manipulerons cette technique est trop coûteuse.

4. $\alpha_k(\alpha_l(I)) = \alpha_{k+l}(I)$.
5. $I \circ J = \alpha_{-||I||}(J) \circ \alpha_{||J||}(I)$.

Preuve : *Les preuves de 1, 2, 3 et 4 sont immédiates. Pour 5 :*

$$\begin{aligned} \text{Soit } (p,l) \in I \circ J &\iff \exists p',\ l'\ (p,l') \in I \text{ et } (p+l', l-l') \in J \\ &\iff (p+l-l', l') \in \alpha_{l-l'}(I) \text{ et } (p, l-l') \in \alpha_{-l'}(J) \\ &\iff (p,l) \in \alpha_{-l'}(J) \circ \alpha_{l-l'}(I) \\ &\iff (p,l) \in \alpha_{-||I||}(J) \circ \alpha_{||J||}(J) \end{aligned}$$

□

Lemme 2 Si I est l'index de $r \in Rat(\mathcal{A}^*)$ dans $\mathcal{T}$ et J l'index de $q \in Rat(\mathcal{A}^*)$ alors :

1. $I \cup J$ est l'index de $(r+q)$.
2. $I \circ J$ est l'index de rq.

Exemple : L'index de l'expression rationnelle sans étoile : **d(a+cd+c)** est $I_d \circ (I_a \cup (I_c \circ I_d) \cup I_c) = I_d \circ I_a \cup I_d \circ I_c \circ I_d \cup I_d \circ I_c$.

Pour pouvoir établir une correspondance complète entre $Rat(\mathcal{A}^*)$ et l'ensemble des expressions rationnelles sur $\mathcal{I}$, il nous faut aller un peu plus loin. On note $I_\epsilon = \{(p,0) \mid 1 \leq p \leq |\mathcal{T}|\}$ et $I_\emptyset = \emptyset$. Sans étendre la définition de $\mathcal{I}$, on a $I_\emptyset \in \mathcal{I}$. Par contre, $I_\epsilon \notin \mathcal{I}$. On note aussi $I^{\circ n} = \underbrace{I \circ I \circ ... \circ I}_{\text{n fois}}$.

Proposition 1 On a les différentes propriétés suivantes sur $\mathcal{I}$:

1. $(\mathcal{I} \cup I_\epsilon, \circ, I_\epsilon)$ et $(\mathcal{I}, \cup, I_\emptyset)$ sont des monoïdes.
2. L'application $\phi : \mathcal{A}^* \mapsto (\mathcal{I} \cup I_\epsilon, \circ, I_\epsilon)$ définie par $\phi(r) = I_r$ $\phi(\epsilon) = I_\epsilon$ est un morphisme de monoïde.
3. L'application $\Phi : (Rat(\mathcal{A}^*), +, \emptyset) \mapsto (\mathcal{I}, \cup, I_\emptyset)$ définie par $\Phi(r) = I_r$ et $\Phi(\emptyset) = I_\emptyset$ est aussi un morphisme de monoïde.
4. $(\mathcal{I} \cup I_\epsilon, \circ, I_\epsilon)$ est nilpotent d'indice $n \leq |\mathcal{T}| + 1$.

Preuve :

1. *On a bien pour tout $I \in \mathcal{I}$: $I \circ I_\epsilon = I_\epsilon \circ I = I$ et $\circ$ est associative. De même $I \cup I_\emptyset = I_\emptyset \cup I = I$. L'union est clairement aussi associative (et commutative).*
2. *Il faut montrer $\phi(xy) = \phi(x) \circ \phi(y)$.*
   Or $m = (p,l) \in I_{xy} \Leftrightarrow \mathcal{T} = uxyv$ avec $|u| = p$ et $|xy| = l \Leftrightarrow (p, |x|) \in I_x$ et $(p+|x|, l-|x|) \in I_y \Leftrightarrow m \in I_x \circ I_y$.
   On vérifie de plus facilement que la définition de $\phi(\epsilon)$ est bien compatible.
3. *La démonstration est analogue à la précédente.*
4. *Pour montrer le principe, considérons I un index de longueur fixe. Il est clair qu'on a $||I \circ I|| = 2||I||$ ou bien $I \circ J = I_\emptyset$. On a donc $||I^{\circ n}|| = n||I||$ ou $I^{\circ n} = I_\emptyset$. Or la longueur maximal d'un index est $|\mathcal{T}|$ (i.e. $l \leq |\mathcal{T}|$ par définition). Donc $||I^{\circ|\mathcal{T}|+1}|| > \mathcal{T}$ ou $I^{\circ|\mathcal{T}|+1} = I_\emptyset$. Le premier terme est faux, donc le second est vrai. Ce qui prouve la proposition. Si I n'est pas de longueur fixe, la démonstration est analogue mais il faut définir la notion de longueur minimale d'un index.*

□

Ces différentes propositions permettent d'établir une équivalence entre l'écriture d'une expression rationnelle et le calcul de son index. Remarquons que l'image par Φ de $Rat(\mathcal{A}^*)$ est identique au monoïde libre généré par les index des lettres de $\mathcal{A}$ et de I_ϵ, mais Φ n'est pas un isomorphisme. En effet, l'index d'une expression rationnelle est vide si cette expression rationnelle n'est pas dans le texte. Cela explique aussi que la composition n'est pas inversible (ce que nous considérons ci-dessous) et que $\mathcal{I}$ soit nilpotent. En fait, cette dernière propriété est intéressante : une étoile dans une expression rationnelle aura donc un équivalent «borné» dans le calcul de son index. Cette propriété est triviale, puisque cela consiste à dire qu'il ne peut pas y avoir de mots de longueur aussi grande que l'on veut dans un texte fini, mais elle est très proche de l'intuition linguistique disant que tous les phénomènes considérés ont une étoile bornée. On gardera cependant la notation étoile :

$$I^* = I_\epsilon \cup \bigcup_{1 \leq n} I^{\circ n} = I_\epsilon \cup \bigcup_{1 \leq n \leq |\mathcal{T}|} I^{\circ n}$$

On notera maintenant sans ambiguïté $I^{\circ n} = I^n$ provenant de $I_r^{\circ n} = \phi(r^n) = \phi^n(r) = I_r^n$ et cette notation s'étend naturellement aux puissances d'entrées d'index quelconque.
Une propriété d'inversibilité faible existe par ailleurs :

Proposition 2 *Propriété d'inversibilité faible.* Pour tout $I \in \mathcal{I}$ de longueur fixe, on peut définir I^{-1} tel que $I \circ I^{-1} = I^{-1} \circ I \subset I_\epsilon$. On note $I \circ I^{-1} : \epsilon_I$. Dans le cas général, $\epsilon_I \neq I_\epsilon$. Par contre $\forall I' \in \mathcal{I} : I' \circ \epsilon_I \subset I'$, $\epsilon_I \circ I' \subset I'$ et si $I' \subset I$:

$$\epsilon_I \circ I' = I' \circ \epsilon_I = I'$$

Preuve : *Considérons* $I^{-1} = \{(p+l,-l) \mid (p,l) \in I\}$. $I^{-1} \notin \mathcal{I}$, *en conservant la définition de* $\circ$, *on obtient :* $(p,l) \in I \circ I^{-1} \Rightarrow \exists l_1 \leq 0$ *tel que* $(p, l - l_1) \in I \Rightarrow l - l_1 = ||I||$. *Or* $l_1 = ||I||$ *par définition. Donc* $l = 0$. *Donc* $(p,l) \in I_\epsilon$ *et* $(p,l) \in I^{-1} \circ I$. *De la même façon, on montre* $I^{-1} \circ I \subset I \circ I^{-1}$. *Donc* $\epsilon_I = I \circ I^{-1} = I^{-1} \circ I$ *et* $\epsilon_I \subset I_\epsilon$.
On considère maintenant (p,l) *in* ϵ_I. *On a clairement* $(p',l') \in I' \circ \epsilon_I \Rightarrow (p',l') \in I'$. *Donc* $\epsilon_I \circ I' \subset I'$. *Enfin, si* $I' \subset I \Rightarrow \forall (p',l') \in I', (p',l') \in I$ *donc* $(p'+l', 0) \in I^{-1}$, *donc* $(p', 0) \in \epsilon_I$ *Ce qui entraîne* $I' \subset \epsilon_I' \circ I'$. □

> **Exemple :** Illustrons cette propriété d'inversibilité faible sur l'exemple de la page 181. Nous avions considéré l'index I' des séquences comprises entre deux **c** :
> **abaabcabbdcabacabaabababbdabadacbaabaca**
> On a clairement $I' = I_c^{-1} \circ I_{c(a+b+d)^*c} \circ I_c^{-}1$ et I' est différent de $I_{(a+b+d)^*}$. On peut dire que I' est l'index de la séquence $c^{-1}c(a+b+d)^*cc^{-1}$.

L'exemple précédent montre assez clairement qu'il sera possible d'opérer des opérations sur des grammaires dépendant du contexte en appliquant les compositions avec des index de séquences de la forme : $A^{-1}A$.

Enfin, nous montrons le rôle de l'intersection d'index à l'aide de la proposition suivante :

Proposition 3 Soit $A, B \in Rat(\mathcal{A}^*)$: on a $I_A \cap I_B = I_{A \cap B}$.

Preuve : $Rat(\mathcal{A}^*)$ *est fermée pour l'intersection,* $A \cap B \in Rat(\mathcal{A}^*)$. *Donc* $I_{A \cap B}$ *est défini comme l'ensemble des occurrences de* $A \cap B$ *dans* $\mathcal{T}$, *cet ensemble est aussi l'intersection de l'ensemble des occurrences de* A *et de l'ensemble des occurrences de* B, *c'est à dire* $I_A \cap I_B$. □

Exemple : Pratiquement, l'intersection sera utilisée pour faire des tests sur une séquence à l'aide de plusieurs critères. Par exemple, si nous avons indexé les formes, les lemmes et les catégories, nous pouvons calculer l'index de $I_{<être.V>}$ en prenant l'intersection de $I_{<être>}$ et $I_{<V>}$:

$$I_{<être.V>} = I_{<être>} \cap I_{<V>}$$

Nous venons de voir qu'il est possible de calculer l'index de toute expression rationnelle (donc de tout automate fini) uniquement au moyen d'opérations sur l'index. Pour tout expression rationnelle A, nous avons montré quel index correspond à A^n et A^*. Nous devons maintenant examiner la complexité du calcul de ces deux quantités.

9.4.1 Le calcul de $I_r^* = I_{r^*}$ et de I^n

Nous avons montré :

$$I_{r^*} = I_\epsilon \cup I_r^+ = I_\epsilon \cup \bigcup_{1 \leq k \leq |\mathcal{T}|} \bigcirc_{1 \leq j \leq k} I_j$$

Nous pouvons limiter le calcul de I_r^* à I_r^+. I_ϵ n'ayant pas besoin d'être construit explicitement : en effet, de la même façon que le nombre de translations d'un index est indiqué dans sa structure, nous pouvons indiquer si l'index contient I_ϵ, puisque les opérations avec I_ϵ n'ont pas besoin d'être effectuées (on a directement $I \circ I_\epsilon = I$). La formule précédente n'est cependant pas satisfaisante, puisque la complexité de l'algorithme associé est de $\sum_{1 \leq k \leq |\mathcal{T}|} k$ opérations, c'est à dire $O(|\mathcal{T}|^2)$ opérations ! Cette complexité est atteinte dans le pire des cas (pour un texte $\mathcal{T} = a^n$ et $r = a$) et dans le cas général, on peut arrêter le calcul quand $\bigcirc_{1 \leq j \leq k} I_k = \emptyset$. On ne peut espérer faire mieux pour le cas le plus défavorable puisque dans l'exemple précédent $|I_{r^*}| = O(|\mathcal{T}|^2)$. Cependant, il est possible d'effectuer le calcul de $|I_{r^*}|$ en une seule passe sur I avec une complexité minimale égale à $O(|I_{r^*}|)$ grâce à l'algorithme[3] de la figure 9.5. Cet algorithme parcourt I dans l'ordre croissant[4] et maintient à tout moment une liste[5] des séquences de I_{r^+} potentiellement «rallongeables» (c'est-à-dire pour lesquelles la position du dernier caractère est supérieure à la position de l'entrée d'index courante p). Toutes les séquences déjà construites de I_{r^+} dont la position du dernier caractère est strictement inférieure à la position de l'index courant p sont stockées dans *Ires* qui contient le résultat.
Quand I a été entièrement parcouru, on ajoute tous les éléments de A à *Ires*.
La complexité de cet algorithme peut être calculé comme le nombre d'ajouts effectués dans *Ires* (ligne 6 et 8), ou dans A (ligne 10). Ces deux complexités sont équivalentes et égales au nombre d'éléments de $Ires = I^+$, ce qui est la complexité minimale pour construire I^+.

Sur ce modèle, on peut calculer I^n pour n quelconque en une seule passe, en ajoutant un champ à la structure A qui contient l'exposant de chaque élément. La complexité est cependant maintenant liée au nombre d'états intermédiaires construits : c'est à dire $\sum_{1 \leq k \leq n} |I^k|$.

3. L'indentation sert de délimiteur de blocs comme dans tous les algorithmes présentés.
4. Ce qui suppose donc que I est ordonné dans l'ordre croissant.
5. Cette liste est stockée dans A qui est un arbre binaire équilibré permettant d'ordonner les couples (p_i, p_f) suivant l'ordre des p_f.

```
1.  Fonction ConstruitIPlus(Index I):Index
2.  Index Ires := ∅;
3.  ABE A := ∅;
4.  Pour tout p diffrents tel que ∃l, (p,l) ∈ I
5.      Tant que !(A → Vide()) Et (p_f := A → PlusPetit(&p_i) < p)
6.          A → RetirePlusPetit(); Ires → Ajoute(p_i, p_f − p_i);
7.      Tant que !(A → Vide()) Et (p_f := A → PlusPetit(&p_i) = p)
8.          Ires → Ajoute(p_i, p_f − p_i);
9.          Pour tout (p,l) tel que p' = p dans I
10.             A → Ajoute(p_i, p_f + l);
11. Tant que !(A → Vide())
12.     p_f := A → PlusPetit(&p_i);
13.     A → RetirePlusPetit();
14.     Ires → Ajoute(p_i, p_f − p_i);
15. Retourne Ires;
16. Fin
```

FIG. 9.5 – *Algorithme de calcul de* I^+.

9.4.2 Quelques index spéciaux

Nous avons déjà vu que l'entrée spéciale d'index I_ϵ n'aura jamais besoin d'être explicitement codée, mais que ce sera plutôt une donnée contenue dans la structure des entrées d'index.
On considérera aussi dans nos applications pratiques, l'index $I_{<MOT>}$ défini ainsi:

$$\bigcup_{I \in \mathcal{I}} I = \{(p,1) \mid 1 \leq p \leq |\mathcal{T}| - 1\}$$

où $\mathcal{I}$ est l'index simple du texte. $<MOT>$ est donc l'équivalent du caractère «.» dans *grep*: il correspond à n'importe quel caractère. Nous n'aurons pas besoin de manipuler un tel index dont la taille est celle de la taille du texte. En effet, quelque soit l'index I, nous avons une correspondance[6] directe entre les éléments de I et les éléments de $I \circ I_{<MOT>}$ et en particulier $|I \circ I_{<MOT>}| = |I|$.
Nous avons ici aussi ajouté un champ spécial dans la description des entrées d'index indiquant de combien les descripteurs de chaque occurrence doivent être étendus à droite et à gauche. On tient compte de ces champs au moment d'une composition ou d'intersection réelle avec un autre index. Le mécanisme est donc le même que celui décrivant la translation. Cela nous permet d'obtenir un coût constant pour l'intersection d'un index quelconque avec l'index $I_{<MOT>}$.
Chaque entrée d'index est donc définie par $(L, \alpha, U_\epsilon, Mgauche, Mdroite)$ avec:

- L: liste de descripteurs;

6. Il y a juste un problème au niveau de la borne supérieure: il faut exclure de $I \circ I_{<MOT>}$ tous les descripteurs de I dont la position est $|\mathcal{T}|$. Nous ne le ferons pas explicitement, ce qui introduira quelques approximations locales dans le comptage des éléments de $I \circ I_{<MOT>}$.

- α : entier représentant la translation commune à tous les descripteurs ;
- U_ϵ : booléen indiquant l'union avec I_ϵ ;
- *Mgauche* et *Mdroite* : entiers représentant l'extension à gauche et à droite commune à tous les descripteurs.

Exemple : Considérons $I = (L, 0, faux, 0, 0)$. On veut calculer $I_1 = \alpha_{-2}(I^*) \circ I_{<MOT>}$. On note L^+ la liste de descripteur de I^+ dont nous avons montré le calcul. $I_2 = (L^+, -2, vrai, 0, 0)$ est l'index de $\alpha_{-2}(I^*)$ et donc $I_1 = (L^+, -2, vrai, 1, 0)$. Nous avons effectué le seul calcul de L^+. Les différents opérations correspondant aux champs ne seront effectivement calculées que lors d'une opération ultérieure sur I_1. Cela signifie que le calcul de l'index de $(\mathbf{a}^*)..\mathbf{b}$ aura la même complexité en temps que le calcul de l'index de $\mathbf{a}^+\mathbf{b}$.

9.5 Complexité réelle et complexité théorique

La complexité des différentes opérations sur les index est exprimée en nombre d'opérations élémentaires. Pour de grands textes et sur des ordinateurs personnels courants aujourd'hui (300MHz), la complexité observée de nos algorithmes sera très fortement liée au nombre d'accès au support magnétique. Les calculs eux-mêmes demandant peu de puissance du processeur. Il faut évidemment tenir compte du fait que la mémoire des ordinateurs est de plus en plus grande et qu'il devient difficile de savoir avec exactitude quand ont lieu les accès au support magnétique, du fait du rôle de la mémoire cache[7]. Nous pouvons toujours invoquer le fait qu'il y aura toujours des accès pour les textes qui ne pouvent pas être stockés en mémoire centrale. Pour notre application linguistique, cette situation est vraisemblable, cependant notre but n'est pas d'avoir accès à un texte de taille maximale, mais d'effectuer en temps optimal les opérations que nous avons décrites sur un texte donné. En minimisant le rôle de la mémoire cache, nous pouvons faire quelques suppositions :

- La taille des graphes recherchés sera toujours suffisamment petite pour tenir en mémoire (cela provient directement des contraintes pratiques imposées à nos graphes).
- Tous les index de petite taille concernant un graphe donné peuvent être chargés en mémoire (cette notion de petite taille est reliée à la distribution des mots sur le texte que nous considérerons). Les opérations faites sur ces index sont donc des opérations faites directement en mémoire vive.
- Par contre, les index de grande taille ne sont pas chargés en mémoire et leur accès se fait directement sur le support magnétique (de fait, les index correspondant aux caractères les plus fréquents ont une taille comparable à celle du texte).
- Les accès au texte ou à une structure comme la table des suffixes se font aussi directement sur le support magnétique.

7. Cette mémoire n'étant pas seulement la mémoire de bas niveau présente actuellement dans tous les ordinateurs, mais aussi la mémoire vive, qui est utilisée par de nombreux systèmes comme mémoire cache.

Nous tiendrons compte de ces hypothèses minimales. En particulier, à tout moment de la reconnaissance, nous aurons deux stratégies concurrentes : soit continuer le calcul sur l'index courant J_i, soit utiliser cet index pour vérifier dans le texte chacune des occurrences possibles (ce que nous appelons **la stratégie d'échappement**). Il faudra évidemment choisir la solution la moins coûteuse en temps. Nous montrerons que le choix se fait au moyen d'heuristiques et qu'il n'est pas possible d'atteindre un minimum sans une connaissance et une maîtrise complètes de la gestion du support magnétique.

9.6 Pattern matching à l'aide des index

Nous avons montré que le problème de localisation des structures figées consistait en l'application successive d'un grand nombre de graphes (de l'ordre de 50 000). La première étape de cette application est la reconnaissance de l'expression décrite dans le texte. Ce problème très étudié sous le nom anglais de *pattern-matching* recouvre plusieurs types d'algorithmes [6]. Le *pattern-matching* a d'abord été étudié [14] comme problème de localisation de mots dans un texte (*string matching*). Ce problème s'étend à la reconnaissance d'un grand nombre de mots : *multi string matching* [5] dans un texte et à la reconnaissance de séquences décrites par des expressions régulières [1] (*regular expression matching*). Plus récemment, des algorithmes de recherche de motifs avec erreurs sont aussi apparus [10]. Sous UNIX, plusieurs programmes de recherche constituent la série des **grep**. Il faut distinguer parmis ces algorithmes, ceux qui effectue un calcul préalable sur le texte et ceux qui effectue un calcul préalable sur le motif recherché mais pas sur le texte.

Pour les raisons évoquées en début de chapitre, nous avons choisi une structure d'index qui, d'une certaine manière, est une structure linéaire sur laquelle un calcul préalable à été effectué. Cependant, nous avons montré que cette structure est complètement substituable au texte lui-même. Elle peut naturellement être enrichie pour représenter une structure de données beaucoup plus complexe que le texte lui-même, et nous montrerons qu'elle reste quasi-identique pour représenter des automates de texte. Elle est de plus complètement dynamique (c'est-à-dire qu'elle permet de gérer toutes les transformations que nous avons décrites).

Nous allons commencer par présenter la recherche de mots simples à l'aide de cette structure. Cette présentation est un peu éloignée de nos besoins pratiques, puisque nous chercherons à reconnaître des séquences décrites par des automates finis. Elle nous permettra de comparer cette démarche à d'autres algorithmes opérant sur une structure précalculée. De plus, nous mettrons en évidence un certain nombre de propriétés liées à la reconnaissance de mots à l'aide d'index, propriétés qui seront toujours présentes dans la recherche d'expressions régulières. Pour cette étude, nous nous baserons sur une propriété de la distribution des mots en langue naturelle appelée loi de Zipf (que nous détaillons en annexe B). Cependant, pour ne pas se restreindre pas aux textes en langue naturelle (la distribution de Zipf pouvant être rencontrée dans de nombreuses distributions), nous travaillerons dans cette partie sur un texte aléatoire de distribution zipfienne. Cela nous permettra en plus, de ne pas confondre les propriétés de notre algorithme propre à la distribution considérée et celles qui sont propres à la langue naturelle. Par contre, nous étudierons ensuite la reconnaissance d'expressions régulières sur des textes en langue naturelle. Il n'y a pas tellement de sens en effet à générer des expres-

sions régulières aléatoirement. Les complexités que nous donnerons dans cette partie sont donc des complexités observées.

9.6.1 Algorithmes existant de *pattern-matching* sur un texte utilisant une structure précalculée.

Nous différencions ici les algorithmes utilisant des index de texte, comme ceux dont nous parlerons, et les algorithmes manipulant les suffixes d'un texte (cette séparation est arbitraire, certaines structures manipulant les suffixes d'un texte étant appelées des index par leurs auteurs). Nous discutons ensuite de la pertinence du choix de travailler sur les mots d'un texte (et non pas sur les caractères) éliminant la plupart de ces algorithmes.

Arbre et table des suffixes

L'idée principale de ces deux structures est de considérer l'ensemble des suffixes d'un texte. Cet ensemble comporte N éléments et la somme des longueurs de tous ces suffixes est d'une taille en N^2. L'arbre des suffixes factorise l'ensemble de ces suffixes et permet la recherche de toutes les occurrences d'un mot de taille n en temps $O(n)$. La construction de cette structure peut se faire en temps linéaire $O(n.\log|\mathcal{A}|)$ à l'aide des algorithmes de WEINER [24], MCCREIGHT [17] et UKKONEN [23]. Un tel arbre n'est pratiquement pas utilisable dans notre application à cause de la présence du coefficient $|\mathcal{A}|$ dans le temps de construction : en travaillant sur les mots d'un texte, nous avons un alphabet de taille sous-linéaire en fonction de la taille du texte (voir annexe B). Pour des requêtes composées exclusivement de formes, il est possible de considérer le texte comme une suite de caractères alphanumériques et d'en construire l'arbre des suffixes sur lequel le test d'appartenance d'un mot m au texte est effectué en temps $O(|m|)$. Une structure «compactant» différemment l'arbre des suffixes est la structure appelée *Patricia tree*[8] décrite dans [11] permet la recherche de l'ensemble des occurrences d'un mot en temps $O(\log n)$ en moyenne. Cette structure permet [4] d'autre part, la recherche d'expressions régulières quelconques en temps sous-linéaire. Cet algorithme implémenté pour la consultation de l'*Oxford English Dictionnary* présente une complexité observée en $O(\sqrt{(n)})$ sur les demandes réelles des utilisateurs.
Une structure alternative, proposée par MANBER et MYERS [15] est la table des suffixes. Cette table peut-être vue comme la liste ordonnée de toutes les positions du texte suivant le suffixe présent à cette position. Une deuxième table contenant la longueur des préfixes communs à une sous classe de suffixe permet une recherche effective dans cette table d'un motif en temps $O(n + \log N)$. La construction de la table s'effectue indépendamment de la taille de l'alphabet et en temps $O(N.\log N)$. La valeur $O(n + \log N)$ nous servira de référence quand nous considérerons la complexité de nos algorithmes. Cependant, il faut bien voir que ces algorithmes ne seront pas vraiment comparables dans le cas général, puisque le *suffix-array* se limite à la recherche sur un texte et on ne peut pas envisager d'enrichir ce texte. Supposons par exemple que nous enrichissions au minimum ce texte de la manière suivante : à chaque mot a_i du texte, nous associons son lemme noté $<a_i>$. Le texte est toujours représentable linéairement en alternant forme et lemme de la manière

8. Ce nom est un acronyme de «Practical Algorithm To Retrieve Information Coded In Alphanumerical».

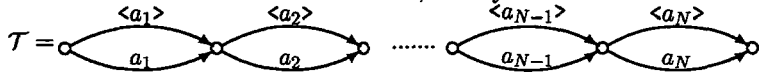
suivante :

$$\mathcal{T} = a_1 <a_1> a_2 <a_2> \ldots a_N <a_N>$$

Faire une recherche en mélangeant les deux notions par exemple : $b_1 <b_2>$ revient à effectuer la recherche de :

$$b_1 \ldots <b_2>$$

c'est-à-dire, une recherche d'un motif contenant deux symboles joker «.». Ce type de recherche n'est pas possible sur la table de suffixe. En effet, cela revient à rechercher le mot b_1 seul (en temps $\log N$) et à vérifier qu'il est bien suivi de $\ldots <b_2>$. La table des suffixes est alors exactement utilisée comme un index très faible. D'autre part, si on considère que le texte a la structure suivante, alors nous ne pouvons plus considérer l'ensemble des suffixes de la structure suivante, car il y en a $2^{(N+1)}$:



Utilisation d'index

Les index sont déjà utilisés dans plusieurs algorithmes et de plusieurs manières différentes. Nous citons essentiellement *glimpse* et *Igrep* qui font une utilisation non triviale de cette structure. Il est certain que cette structure «simple» est aussi utilisée par les moteurs de recherche sur le Web et par des bases de données, mais cela d'une manière plus triviale. Dans *glimpse*, le texte est découpé en blocs. Un index permet de localiser les zones de texte dans lesquelles une occurrence est possible. Dans [3], un algorithme permettant de localiser des séquences avec erreur est décrit, il a conduit à l'implémentation de *Igrep*. Cet algorithme utilise exactement les index que nous avons décrits. Pour l'algorithme sans erreur, la recherche se fait en localisant le plus petit index de la séquence recherchée afin d'en faire une intersection avec les autres index.

MANBER et WU proposent d'utiliser un index partiel du texte en combinaison avec une recherche séquentielle (implémenté dans *Agrep* [28]). Dans leur système *Glimpse* [16], les textes sont approximativement divisés en 256 blocs de tailles équivalentes. L'index contient, pour chaque mot, la liste des blocs dans lesquels ces mots apparaissent. Toute recherche se fait en deux étapes : recherche sur le vocabulaire du texte des mots de la requête, suivi d'une recherche dans les blocs concernés par la requête. La taille de l'index est très réduite, puisqu'à chaque mot est associé (*grosso-modo*) un masque de 256 bits sur lequel les opérations d'intersection et d'union sont triviales. Un tel système est de plus capable de gérer des recherches de chaîne avec erreur.

Dans [3], ARAÚJO, NAVARRO et ZIVIANI proposent un système *Igrep* de recherche de chaîne avec erreurs, pouvant aussi traiter les recherches exactes. Leur système est basé sur un index complet du texte. La recherche d'une séquence s'effectue au moyen de l'intersection du plus petit index du mot de la séquence recherchée avec les index des autres mots. Ils proposent de plus la recherche d'expressions régulières quelconques *sur les lettres*[9] *de la séquence*. Le texte n'est pas utilisé pour la recherche contrairement à l'algorithme précédent. La complexité de la recherche est en $O(n^\beta)$ avec $\beta \approx 0.5$ pour des séquences de deux mots et en $O(\log n)$ pour des séquences plus longues.

9. Les lettres ici correspondent aux caractères des différents mots. L'expression régulière permet donc de sélectionner les mots correspondants dans le vocabulaire.

Dans ces deux algorithmes le point le plus important est l'opération «d'intersection d'index» qui permet, pour le premier, de vérifier que deux mots sont simultanément présents dans le même bloc et pour le deuxième que deux mots sont bien adjacents. Dans le premier cas, l'index n'est utilisé que pour amorcer la reconnaissance, alors que dans le second, toute la reconnaissance se fait dans un calcul sur le texte. Nous désignerons ces deux stratégies par la suite *Index-Texte*[10] et *Index-Index*. Nous étudions cette opération d'intersection et l'extension de la recherche de séquences simples à la recherche d'expressions régulières complexes.
Nous désignerons par **Index-Texte d'ordre** k les algorithmes effectuant k opérations sur l'index avant de passer au texte et par **Index-Index approximatif** les algorithmes effectuant un calcul complet sur l'index mais dont le résultat est un ensemble contenant l'ensemble des occurrences recherchées. Les deux algorithmes présentés plus haut sont donc respectivement *Index-Texte* d'ordre 1, et *Index-Index*.

Le travail sur les mots

Nous venons de décrire des algorithmes offrant des temps de recherche pour des mots en $O(\log n)$ et proposant des algorithmes de recherche d'expressions régulières sur les lettres (c'est-à-dire sur un ensemble de tokens dont la taille est petite). Nous pourrions transformer une grande majorité de nos automates sur des mots en des automates sur des lettres en résolvant en même temps l'accès au dictionnaire. Par exemple, il serait facile de transformer l'expression régulière *(dès+aussitot) <que> (<ce> <être>+ϵ)possible* en une expression régulière sur *a...z*. Pour cela, il suffirait de chercher dans le dictionnaire toutes les formes du verbe *être* et de les subsituer au symbole *<être>*, etc. Cet automate pourrait alors être recherché à l'aide des algorithmes précédent (les tables de suffixes pourraient même être ici utilisées en utilisation *multi-pattern matching*). Pour d'autres graphes contenant des références à des catégories comme *<N>*, la taille de l'automate construit, et le temps de construction serait prohibitif.
Nous n'avons pas essayé d'implémenter cette solution, même pour les graphes «simples». Nous avons déjà donné des arguments concernant le fait que la structure linéaire du texte est une seule des applications possibles que nous voulons développer et qu'il faudrait donc plutôt des structures permettant directement de représenter texte linéaire et automate acyclique. Un autre argument qui se justifie par les résultats que nous obtenons, et que en travaillant sur les lettres, nous ignorons la loi de distribution des mots sur les textes qui est suffisamment remarquable pour ne pas être utilisée.

9.6.2 Recherche de séquences formées de lettres

Soit $b_1...b_n$ un mot de $\mathcal{A}^*$ que l'on recherche dans $\mathcal{T}$. On veut donc calculer:

$$I_{b_1...b_n} = I_{b_1} \circ I_{b_2} \circ ... \circ I_{b_n}$$

On suppose que $||I_{b_i}|| = 1$ pour tout i.

10. Même quand la structure considérée ne sera plus un texte mais un automate.

Exemple : Rechercher **abaaba** dans un texte revient à calcul son index I_{abaaba}.
On peut effectuer ce calcul de plusieurs manières :
$I_{abaaba} = \underbrace{I_a \circ I_b \circ I_a}_{I_1} \circ \underbrace{I_a \circ I_b \circ I_a}_{I_1}$
$I_1 = I_a \circ I_b \circ I_a$ localise le mot **aba** et donc $I = I_1 \circ I_1$.
ou $I_{abaaba} = \underbrace{\underbrace{(I_a \circ \alpha_{-1}(I_a))}_{I_2} \circ \alpha_{-1}(\underbrace{(I_a \circ \alpha - 1(I_a))}_{I_2})}_{I_3} \circ \alpha_1(\underbrace{\alpha_2(I_b) \circ I_b}_{I_4})$
$I_2 = I_a \circ \alpha_{-1}(I_a)$ localise **a.a** $I_3 = I_2 \circ \alpha_{-1}(I_2)$ localise **a.aa.a**
$I_4 = \alpha_2(I_b) \circ I_b$ localise **b..b**
et donc $I = I_3 \circ \alpha_1(I_4)$.

L'optimisation d'un tel calcul ressemble au problème de programmation dynamique de parenthésage d'opérations matricielles. Les différences sont :

- Il est possible[11] d'effectuer toutes les permutations voulues dans l'expression grâce à l'opérateur α.
- Il est possible de factoriser des sous-motifs.
- La taille d'une composition de deux index n'est pas connue avant son calcul.

Nous détaillons dans la suite ces premières différences : permutations dans l'ordre du calcul et factorisation de sous-motifs. La taille d'une composition de deux index sera estimée en fonction de la taille des index. Il est important de souligner que ce que nous cherchons à optimiser n'est pas le nombre d'opérations, mais le coût cumulé de ces opérations. Nous avons vu précédemment que le nombre d'opérations nécessaires pour calculer $I' = I \circ J$ était de la forme $c(|I|, |J|) + |I'|.\log_2 |I'|$. La valeur $c(|I|, |J|)$ correspond au nombre d'opérations nécessaires pour calculer $I \circ J$ et le nombre $|I'|.\log_2 |I'|$ correspond au nombre d'opérations nécessaires pour construire l'index résultat sous la forme d'un ABE. Nous désignerons ces deux quantités respectivement par T_{comp} et T_{cons}.

Permutations dans l'ordre du calcul

Nous montrons qu'il est possible d'effectuer, grâce à l'opérateur α, les compositions d'index dans un ordre quelconque.

Proposition 4 Soit $I_1 \circ I_2 \circ ... \circ I_n$ un mot de $\mathcal{I}^*$ et soit $\sigma : [1, n] \mapsto [1, n]$ une permutation de $[1, n]$, alors :

$$I_1 \circ I_2 ... \circ I_n = \alpha_{(1-\sigma(1))}(I_{\sigma(1)}) \circ \alpha_{(2-\sigma(2))}(I_{\sigma(2)}) \circ ... \circ \alpha_{(n-\sigma(n))}(I_{\sigma(n)})$$

Preuve : *Par récurrence sur n :*
Si $n = 1$, la seule permutation de $\{1\}$ est l'identité, donc $\alpha_{(1-1)}(I_1) = \alpha_0(I_1) = I_1$.
Supposons que cela soit vrai à l'ordre $n-1$. Soit σ une permutation de $[1, n]$ et soit i tel que $\sigma(i) = n$.
Soit $I' = I_{\sigma(1)} \circ ... \circ I_{\sigma(i)} \circ \overbrace{I_{\sigma(i+1)} \circ ... \circ I_{\sigma(n)}}^{I''}$. On note $I'' = I_{\sigma(i+1)} \circ ... \circ I_{\sigma(n)}$ et on effectue dans I' la permutation de $I_{\sigma(i)}$ et de I'' à l'aide du lemme 1.

$$I' = \underbrace{I_{\sigma(1)} \circ ... \circ I_{\sigma(i-1)} \circ \alpha_{-1}(I'')}_{I'''} \circ \alpha_{||I''||}(I_n)$$

11. Dans la mesure où les index sont de longueur fixe, ce qui est vrai ici puisque les b_i sont des caractères.

On note $I''' = I_{\sigma(1)} \circ ... \circ I_{\sigma(i-1)} \circ \alpha_{-1}(I'')$.
En snbsituant I'' *par sa valeur et par distributivité de* α, $I''' = I_{\sigma(1)} \circ ... \circ I_{\sigma(i-1)} \circ \alpha_{-1}(I_{\sigma(i+1)}) \circ ...\alpha_{-1}(I_{\sigma(n)})$.
Considérons $\sigma' : [1, n-1] \mapsto [1, n-1]$, *définie par:*

$$\sigma'(j) = \begin{cases} \sigma(i) & si\ j \leq i-1, \\ \sigma(i+1) & si\ j \geq i. \end{cases}$$

σ' *est une permutation de* $[1, n-1]$. *Donc, par hypothèse de récurrence,* $I''' = \alpha_{(1-\sigma'(1))} \circ ... \circ \alpha_{i-1-\sigma'(i-1)} \circ \alpha_{(i-\sigma'(i))-1}(I_{\sigma'(i)}) \circ ...\alpha_{(n-1-\sigma'(n-1))-1}(I_{\sigma'(n-1)})$. *En permutant à nouveau dans* I, $I_{\sigma(i)} = I_n$

$$avec\ I' = \alpha_{(i-\sigma'(i))-1}(I_{\sigma'(i)}) \circ ...\alpha_{(n-1-\sigma'(n-1))-1}(I_{\sigma'(n-1)})$$

on obtient le résultat. □

Nous avons donc une formule permettant de calculer les coefficients de translation de tous les caractères d'un anagramme d'un mot. Il est alors possible d'effectuer le calcul de I' dans un ordre quelconque en translatant correctement les différentes étapes du calcul.

Soit C le coût du calcul de $I_1 \circ ... \circ I_n$ en effectuant l'opération dans l'ordre d'écriture, tel qu'il est parenthésé dans la formule:

$$I = (...((I_1 \circ I_2) \circ I_3 \circ ...) \circ I_n)$$

On note les index intermédiaires de calcul J_k avec pour k, $2 \leq k \leq n$, $J_k = I_1 \circ I_2 \circ ... \circ I_k$ et $J_1 = I_1$. On a:

$$C = \sum_{2 \leq i \leq n} c(|J_{i-1}|, I_i) + |J_i|.\log_2 |J_i|$$

Avec une structure ABE, $c(|I|, |J|)$ est donné par:

$$c(|I|, |J|) = \min(|I|.\log_2 |J|, |J|.\log_2 |I|) \text{ voir (page 185)}$$

Nous cherchons donc la permutation de I qui minimise C. Nous allons trouver un minimum local en minimisant indépendamment les deux termes de la somme. Nous supposons pour le moment que nous pouvons estimer $|I \circ J|$ par $|I \circ J| \approx \beta \min(|I|, |J|)$, avec $0 < \beta \leq 1$. On a donc $|J_i| = \beta \min(|I_i|, |J_{i-1}|)$.

Proposition 5 Soit $I = I_1 \circ I_2 \circ ... \circ I_n$. Pour toute permutation σ de $[1, n]$, $I = \alpha_{(1-\sigma(1))}(I_{\sigma(1)}) \circ ... \circ \alpha_{(n-\sigma(n))}(I_{\sigma(n)})$. Le minimum de $\sum |J_i|.\log_2 |J_i|$ et le minimum de $\sum_i c(|J_{i-1}|, I_i)$ sont atteints pour σ tel que $|I_{\sigma(1)}| \leq |I_{\sigma(2)}| \leq ... \leq |I_{\sigma(n)}|$

Preuve : *Si on suppose que* $|I_{\sigma(1)}| \leq |I_{\sigma(k)}|$ *pour tout* k, *on obtient* $|J_i| = \beta^{i-1}|I_1|$ *(par récurrence simple sur* i*). Il est clair que cela minimise la seconde somme. Sachant cela, le terme de la première somme vaut* $c(\beta^{i-2}|I_1|, I_i)$. *En subsituant la valeur de c, on obtient le résultat.* □

Cela nous permet de déduire qu'en faisant le calcul des compositions d'index ordonnés par taille croissante, nous aurons un coût des plus faibles. Nous ne pouvons pas affirmer qu'il s'agit du coût minimum, d'abord du fait des approximations faites et ensuite parce qu'on a minimisé les deux termes de la somme et non pas la somme elle-même.
Pratiquement, l'approximation de $|I \circ J|$ en fonction de $|I|$ et $|J|$ est plus complexe: en tirant I et J au hasard, β est une fonction décroissante de $\max(|I|, |J|)$ (ce qui justifie

d'autant plus la proposition précédente). Mais dans des situations concrètes, on ne peut pas faire d'estimation plus fine. Les lettres qui apparaissent ensemble dans les mots recherchés présentent une certaine affinité, justement puisqu'on les recherche ensemble[12]. Nous ne chercherons donc pas affiner cette approximation.
La localisation du mot $B = b_1...b_n$ se fait donc à l'aide de l'algorithme de la figure 9.6. Dans cet algorithme, CalculSigma(mot B) correspond à la fonction triant les différentes lettres de B suivant leur fréquence croissante et exprimant ce tri par une permutation (c'est-à-dire un tableau entier de taille $|B|$). Nous n'avons pas fait figurer dans cet algorithme le traitement des champs spéciaux de l'index décrits page 190. Leur implémentation est directe, mais alourdit l'algorithme.

```
1.  Fonction RechercheMotSansFactorisation(mot B):Index
2.  int permutation[|B|]:=CalculSigma(B);
3.  Index I:=I_ε;
4.  Pour i := 1  i = |B|
5.      I=Composition(I,index(B[permutation[i]]));
6.      I=alpha(I,i - permutation[i]);
7.  Retourne I;
```

FIG. 9.6 – *Algorithme construisant l'index d'un mot B, uniquement à l'aide d'opérations sur les index.*

Factorisation de sous-séquences

Considérons maintenant la possibilité de factoriser le calcul des sous-séquences du mot. Il est clair que si on pouvait éviter de refaire plusieurs fois les mêmes calculs, le coût global serait inférieur.
Cependant, ce que nous cherchons à factoriser n'est pas clair. Nous avons représenté sur la figure 9.7, cinq cas de figure différents qui peuvent se présenter simultanément pour le même mot à factoriser et qui correspondent aux cinq stratégies de factorisation suivantes :

(a) rechercher les plus longues sous-séquences répétées sans chevauchement :

$$\max_{m \in A^*} |m| \text{ tel que } \exists u, v, w \text{ et } M = umvmw$$

12. Par exemple, supposons que nous recherchons le terme *ancien président de la république*: *république* et *président* sont les deux mots de l'expression les plus rares dans le texte. Nous allons donc commencer par calculer $I_{république} \circ I_{président}$. Cependant la composition de $I_{président}$ et $I_{république}$ va donner un index très proche de $I_{république}$ puisque ce mot n'est guère utilisé que dans ce contexte. Ici il faudrait commencer par faire la composition $I_{république} \circ I_{ancien}$, ce qui donnerait presque tout de suite l'index résultat minimal sur lequel on pourrait faire les autres opérations. L'ordre des opérations devrait donc aussi tenir compte des coefficients de corrélation entre les différents mots (en composant les unités ayant les plus faibles corrélations) de manière à faire dimininuer au plus vite la taille de l'index temporaire sur lequel les calculs sont effectués. Cependant le calcul de ce coefficient nécessiterait une lecture des deux index simultanée, ce que nous cherchons à éviter.

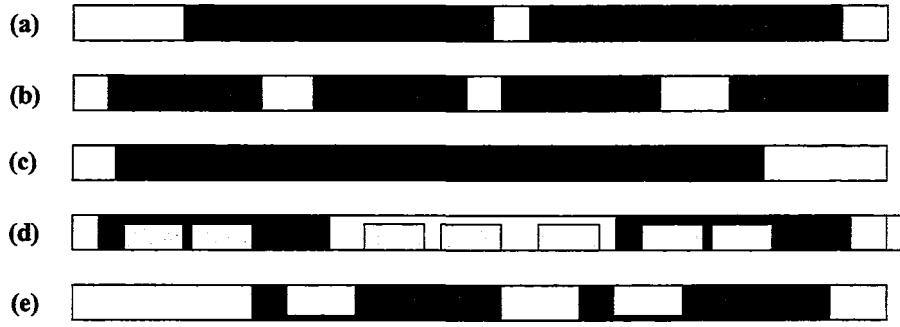


FIG. 9.7 – *Cinq factorisations potentielles différentes.*

(b) rechercher la sous-séquence répétée dont la couverture est maximale:

$$\max_{(m,k)\in\mathcal{A}^*\times I\!N} k.|m| \text{ tel que } \exists\, u, v_1, ..., v_k \text{ et } M = umv_1...mv_k$$

(c) rechercher les plus longues sous-séquences répétées avec chevauchement: en effet comme l'opération $\circ$ est inversible (dans le cadre de la proposition 2): il peut être intéressant d'avoir factorisé un long sous-motif en corrigeant les chevauchements comme dans l'exemple suivant:

$$I_{abcd(abcda)^3} = (I_{abcda} \circ I_a^{-1}) \circ (I_{abcda})^3$$

Ce que nous cherchons à optimiser est donc:

$$\max_{m\in\mathcal{A}^*} |m| \text{ tel que } \exists\, u, u', v, v', w \text{ et } m = wv' = u'w,\ M = uu'wv'v$$

(d) rechercher le motif répété qui lui-même offrira ensuite la meilleure factorisation.

(e) rechercher le plus long motif répété non connexe. Cela nous est permis grâce à l'opérateur α. Par exemple, le calcul de $I_2 = I_{adbacbaabacb}$ peut se faire en calculant $I_1 = I_{a.bacb} = \alpha_1(I_a) \circ I_{bacb}$. On obtient ensuite I_2 par:

$$I_2 = \alpha_{-1}(I_d) \circ I_1 \circ \alpha_{-1}(I_a) \circ I_1$$

Ce que nous cherchons à optimiser est donc:

$$\max_{m\in\mathcal{A}^*} |m| \text{ tel que } \begin{cases} \exists\, k, m_1, ..., m_k, u, v, u', v', i_1, ..., i_{k-1}, i'_1, ..., i'_{k-1} \\ M = um_1i_1m_2...i_{k-1}m_kv \\ M = u'm_1i'_1m_2...i'_{k-1}m_kv' \\ \text{et } \forall j : |i_j| = |i'_j| \end{cases}$$

UNIVERSITE de MARNE-la-VALLEE * BIBLIOTHEQUE *

Théoriquement, la meilleure factorisation est celle qui donne un temps de calcul pour I minimal. La complexité des différentes opérations présentées ci-dessus est très importante, car tous les cas de figure peuvent être concurrents dans la factorisation. Pratiquement, la question ne se pose guère: si on prend aléatoirement une phrase de 10 mots

dans un texte, au pire la factorisation que l'on peut s'attendre à trouver sera du type (a) $umvmw$ avec $|m| \leq 3$. Il n'est pas complètement évident qu'il y ait intérêt à la factoriser. Considérons par exemple la phrase suivante:

Les suites de la polémique née de l'abrogation de la loi Falloux

Le calcul de son index en permutant les termes est (sans les translations):

$$I_{polémique} \circ I_{Falloux} \circ I_{née} \circ ... \circ I_{la} \circ I_{la} \circ I_{de} \circ I_{de}$$

Si nous voulions factoriser *de la*, nous ferions alors explicitement le calcul de $I_{la} \circ I_{de}$ qui serait des plus coûteux puisque les deux index sont importants. Ce calcul est évité en faisant la composition avec des index de tailles minimales (ceux construit par intersection des index croissants).

Nous avons donc laissé de côté une possibilité de factorisation dans les mots qui permettrait dans certains cas d'optimiser le calcul, mais qui, dans le cas général, nécessiterait des calculs trop importants, uniquement en vue de décider d'une factorisation éventuelle. Ce choix est complètement lié à la distribution des mots dans nos textes. Nous verrons cependant qu'il n'en est pas de même dans les automates de motifs à rechercher et que les différents cas de figure présentés pourraient se rencontrer dans des automates réels.

Complexité théorique

Pour calculer formellement le coût de la recherche d'un mot avec l'algorithme précédent, cherchons quelle est la fréquence moyenne de la lettre la plus rare d'un motif de k lettres. Soit $\mathcal{T}$ un texte et $\mathcal{A}$ son alphabet. On ordonne $\mathcal{A}$ suivant les fréquences décroissantes de chaque lettre: $\mathcal{A} = \{\alpha_1, ..., \alpha_n\}$. On note $p(\alpha_i)$ est la fréquence de α_i donnée par la loi de Zipf: $p(\alpha_i) = \frac{A}{i}$.

$\sum_{1 \leq i \leq n} p(i) = |\mathcal{T}|$, ce qui nous donne $P(\alpha_i) = \frac{p(i)}{|\mathcal{T}|}$ la probabilité d'occurrence de la lettre α_i à chaque position de $\mathcal{T}$.

On recherche les occurrences d'un mot $b_1...b_k$ de longueur n. Le nombre d'occurrences de la lettre la plus rare du motif est $\min_{i=1...k} p(b_i)$. Nous voulons connaître la moyenne de cette valeur sur l'espace de tous les mots de longueur k respectant la distribution du texte:

$$\boxed{p_{min}(k) = \frac{1}{N} \sum_{a_1...a_k \in \mathcal{T}} \min_{i=1...k} p(a_i)}$$

Pour un texte suffisamment grand, la valeur moyenne de la fréquence minimale d'une sous-séquence de longueur k dans le texte est aussi:

$$\begin{aligned}
p_{min}(k) &= \frac{A}{n^k} \sum_{c_1...c_k \in A^k} P(c_1...c_k).\min_{i=1...k} p(c_i) \\
&= \frac{A.k}{n^k} \sum_{r_1...r_k \in [1,n]^k} P(\alpha_{r_1}...\alpha_{r_k}) p(\max_i r_i) \\
&= \frac{A.k}{n^k} \sum_{1 \leq r_k \leq n} \sum_{r_1...r_{k-1} \in [1,r_k]^k} P(\alpha_{r_1}...\alpha_{r_{k-1}}) P(\alpha_{r_k}) p(r_k) \\
&= \frac{A'.k}{n^k.N^{k-1}} \sum_{1 \leq r \leq n} P(\alpha_r) p(r) (\sum_{1 \leq i \leq r} \frac{1}{i})^r \\
&= \frac{A''.k}{(n.N)^k} \sum_{1 \leq r \leq n} \frac{1}{r^2} (\sum_{1 \leq i \leq r} \frac{1}{i})^{k-1} \leq \frac{A''.k}{(n.N)^k} \sum_{1 \leq r \leq n} \frac{1}{r^2} (\log r))^{k-1} \\
&\leq \frac{A''.k}{(n.N)^k} \sum_{1 \leq r \leq n} r^{k-3} \leq \frac{B.k.n^{k-2}}{(n.N)^k} = \frac{B.k}{n^2.N^k}
\end{aligned}$$

On trouve donc:

$$p_{min}(k) \leq B \frac{k}{n^2.N^k}$$

Ce qui signifie que la fréquence moyenne de la lettre la plus rare d'une séquence de k lettres tirée aléatoirement en respectant une distribution zipfienne est de la forme

$p_{min}(k) = O(k.N^{-k})$. Remarquons que la «constante» $B.n^2$ est dépendante de N (voir annexe B). La décroissance est exponentielle.
Nous avons calculé numériquement la somme précédente pour un texte de $N = 20.10^6$ lettres[13], avec $k \leq 5$, $p_{min}(k) \ll \log N$, c'est-à-dire le nombre de recherches par un algorithme de *suffix array*. Pour un texte de cette taille et pour des motifs de taille supérieure à 5, Un algorithme *Index-Texte* fera donc moins d'accès aléatoires que le nombre d'accès à une structure triée comme le *suffix-array*.
$p_{min}(k)$ nous donne de plus la taille maximale des index intermédiaires du calcul de $I_{b_1 \ldots b_n}$ pour l'algorithme *Index-Index*. Afin d'évaluer la complexité du calcul complet, il nous faudrait rechercher la valeur moyenne de la fréquence de la deuxième lettre la plus rare du motif, puis de la troisième, etc. Nous ne présentons pas ces calculs compliqués et sans intérêt majeur, puisque nous ne connaîtrons pas pour autant la taille de la composition des p premiers index. Ces valeurs concordent parfaitement avec la complexité observée pour un texte aléatoire que nous présentons plus loin.

Notre hypothèse de distribution zipfienne est primordiale. L'idée est que «quand on prend un mot aléatoirement suffisamment grand, il y aura une lettre rare». Supposons que nous ayons une distribution aléatoire où chaque lettre a la même probabilité d'occurrence. Le calcul précédent se simplifie en :

$$p_{min}(k) = \frac{k}{n^k} \sum_{i=1 \ldots n} i.(n-i)^{k-1}$$

Le calcul exact de cette valeur est fastidieux et nous trouvons pour $k \ll n$:

$$p_{min}(k) \approx \frac{n}{k+1}$$

Dans ce cas, la fréquence moyenne de la lettre la plus rare d'un mot de k lettres varie d'une manière inversement proportionnelle à la taille du motif : pour un vocabulaire de 100 000 lettres, la fréquence moyenne de la lettre la plus rare d'un mot de 10 lettres est de l'ordre de 10 000. Pour commencer la recherche avec un index d'une taille inférieure à 10, il faudrait rechercher un mot d'une taille de 10 000 lettres !

Complexité observée

Les résultats que nous donnons ici ont été obtenus sur des textes aléatoires de distribution zipfienne que nous avons généré avec l'algorithme décrit à l'annexe B. Ces textes ont une longueur de 10 000 caractères. Cette distribution a été obtenue à partir d'une loi $p(r) = 1\,000.r^{-1}$, ce qui correspond à des coefficients de la loi de Zipf proches de ceux d'une distribution naturelle. Avec une telle distribution, le vocabulaire est d'environ 5 000 mots. Cette taille a été choisie délibéremment petite par rapport à des situations concrètes de textes en langue naturelle pour mieux faire ressortir certains problèmes que nous préciserons.

Le graphe de la figure 9.8 donne l'évaluation, sur un tel texte, de la quantité $p_{min}(k)$ définie précédemment, de la fonction qui donne la taille moyenne du p^e plus petit index

13. Une année d'un quotidien comporte 20 millions de mots.

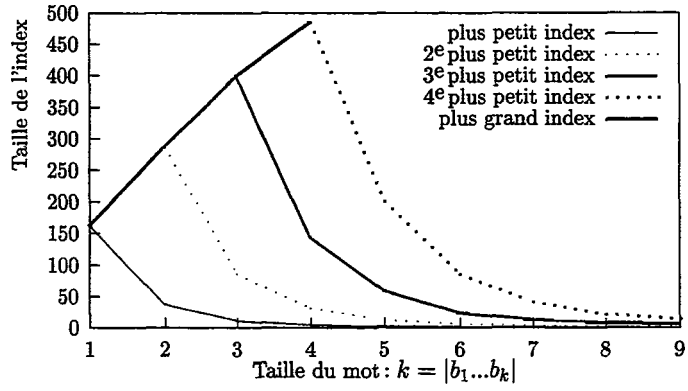


FIG. 9.8 – *Fréquence moyenne du plus rare caractère b_i d'un mot de k lettres.*

d'un mot de longueur k en fonction de k, ainsi que la fonction $p_{max}(k)$. Par exemple, pour des mots de longueur 1, $p_{min}(1)$ vaut[14] environ 160. Pour des mots de longueur 2, la fréquence moyenne du caractère le plus rare est environ 40 et la fréquence moyenne du second caractère est environ 280. On voit que la fréquence maximale $p_{max}(k)$ croît avec k (ce qui est normal, puisque qu'on a plus de chance de rencontrer un caractère plus fréquent), mais cette croissance est lente[15].

Le graphe de la figure 9.9 représente les différentes quantités relatives à la recherche d'un sous-séquence d'un texte dans le texte lui-même: N_{occ}, T_{cons} et T_{comp}. L'abscisse de ce graphe correspond à la taille k du mot recherché. N_{occ} représente (échelle logarithmique de droite) le nombre d'occurrences moyen des mots de taille k. Par exemple, le nombre d'occurrences moyen de mots de taille 1 est 162, donnée que nous avions déjà sur le graphe précédent. Le nombre d'occurrences moyen des mots de taille 2 est environ 4. Ensuite, le nombre d'occurrences moyen des mots de taille 3 est inférieur à 1. T_{comp} et T_{cons} représente (échelle de gauche) les deux parties du coût du calcul de l'index du mot recherché. On remarque que T_{cons} (le nombre d'opérations nécessaires pour construire les index intermédiaires) est négligeable devant T_{comp}. Nous dirons donc que la complexité est proportionnelle à cette dernière quantité. T_{comp} est nul pour $k = 1$ puisque la recherche d'un mot de longueur 1 à l'aide de l'index est instantanée (par définition de l'index).

Nous observons un pic de complexité de 140 opérations pour $k = 2$. Ce pic correspond au nombre moyen d'opérations nécessaires pour calculer la composition des index des deux lettres. Essayons de comprendre cette «très grande» valeur. Dans le graphe de la figure 9.8, on voit que la valeur moyenne du caractère le moins fréquent dans un mot de deux lettres est 40 et la valeur moyenne du caractère le plus fréquent est 280. Ces deux valeurs sont relativement importantes, puisque 40 correspond à la 25e lettre la plus fréquente de la distribution et 280 correspond à la quatrième sur 5 000. Ce phénomène provient donc directement de la distribution: en choisissant aléatoirement un mot de 2 lettres,

14. $p_{min}(1)$ vaut aussi la fréquence moyenne des mots de longueur 1, c'est-à-dire des lettres.

15. En fait, le nombre maximal d'occurrences d'un caractère dans notre texte exemple est de 1 000 et $p_{max}(k)$ tend naturellement vers cette limite

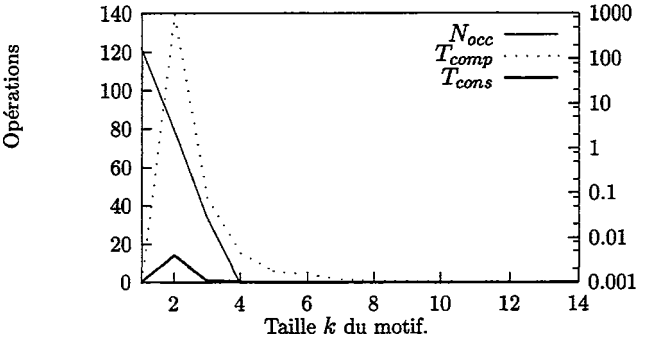


FIG. 9.9 – *Recherche d'une séquence aléatoire dans un texte par des calculs sur l'index.*

nous avons beaucoup de chance d'avoir au moins une lettre fréquente et une chance non négligeable d'en avoir deux. Leur poids est très fort fait que ces mots fréquents vont avoir une moyenne très élevée. Ce phénomène a aussi été observé dans *Igrep* [3]. Pratiquement, il faut cependant voir que les motifs recherchés ne sont pas aléatoires dans notre application, et que la probabilité qu'une grammaire soit constitué de deux mots grammaticaux est très faible[16].
Pour $k = 3$ le nombre total d'opérations passe à 40, puis 15 pour $k = 4$ et 5 pour $k = 5$. *Pour des motifs d'une taille supérieure à* 5 *lettres, la recherche d'un motif aléatoire nécessitera en moyenne sur ce texte moins de* 5 *opérations.* Cette valeur dépend évidemment de la taille du texte, mais dans tous les cas, la borne supérieure que nous avons calculée pour $p_{min}(k)$ montre que pour n'importe quel texte, il existe k petit, tel que le nombre de recherches d'un mot de longueur k sur ce texte coûte moins de 5 opérations en tout sur l'index. Il faut comparer cette valeur à la complexité du *suffix-array* par exemple, elle est en $O(\log N + N_{occ})$ et indépendante de k.

Il faut néanmoins moduler ce résultat: nous avons recherché, dans le cas de figure précédent, un motif n'appartenant pas *a priori* au texte. Nous examinons le comportement de l'algorithme qui recherche un motif extrait du texte. Considérons le graphe de la figure 9.10 qui représente les mêmes quantités, mais lors d'une recherche d'un motif aléatoirement choisi dans le texte. Pour $k \leq 4$ les trois courbes sont sensiblement les mêmes. Par contre, pour $k > 5$, il y a une différence dans T_{comp} qui est croissant et dans le nombre d'occurrences qui tend vers 1. La croissance de T_{comp} est linéaire sur la portion visible et un test pour k très grand (voisin de 1 000) montre que la croissance se poursuit. Cela signifie, que le temps de recherche dans ce texte est en $O(k)$, pour k supérieur à 5. Nous démontrons que ce résultat est plus général.
Soit un texte de taille N, on y recherche un motif de taille k_1 (extrait du texte), suffisamment grande pour que le motif soit unique dans le texte. On a donc un index intermédiaire J_{k_1} qui ne contient plus qu'une seule entrée. Si on augmente maintenant

16. Une telle grammaire existe néanmoins, par exemple la grammaire défaisant la contraction *s'il*⟶*si il* recherche les deux mots grammaticaux fréquents *s* et *il*.

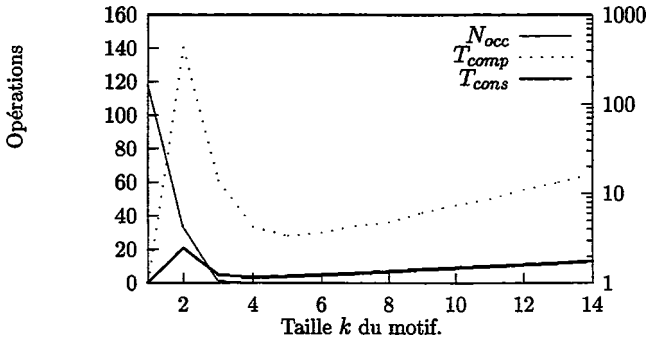


FIG. 9.10 – *Recherche d'une sous-séquence de $\mathcal{T}$ dans $\mathcal{T}$.*

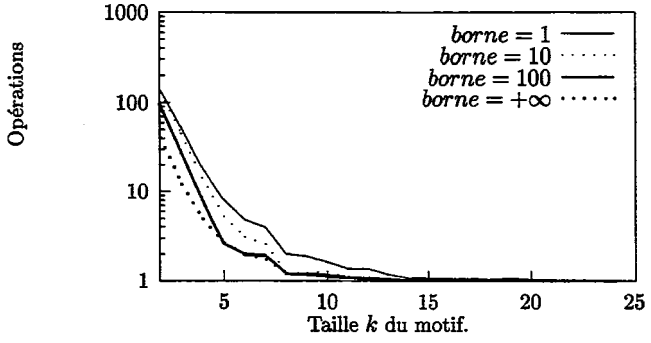


FIG. 9.11 – *Valeur $T_{comp} + T_{texte}$ pour la recherche d'une sous-séquence de $\mathcal{T}$ dans $\mathcal{T}$ avec interruption du calcul dès que $J_k \leq borne$.*

la taille de ce motif, pour tout $k \geq k_1$, on a $|J_k| = 1$. Chaque fois qu'on ajoute une nouvelle lettre, on ajoute aussi une composition. Il faut remarquer qu'avec une distribution zipfienne telle que nous l'avons générée, environ 10 mots couvrent un tiers de la surface totale du texte. La fréquence du 10^e mot est de $\frac{N}{10 \times 10}$. Toutes les trois augmentations de motifs en moyenne, on ajoute donc la composition d'un index de taille 1 avec un index de taille $A.N$. Cette composition nécessite $\log_2 A.N$ opérations. La croissance de T_{comp} va donc être linéaire (il est facile de voir qu'au pire, on devra effectuer cette composition pour chaque augmentation). On trouve donc $T_{comp} = O(k.\log_2 N)$. D'un point de vue théorique, cela diminue toute la puissance de l'index qui, en très peu d'opérations, aura localisé une occurrence potentielle, mais qui devra effectuer un grand nombre d'opérations coûteuses pour valider l'occurrence. Dans une application comme la recherche bibliographique, pour laquelle les requêtes existent dans la base documentaire, cela n'est pas satisfaisant. Pour cette raison, nous avons couplé la recherche par index avec une recherche basée sur le texte (algorithme *Index-Texte*).

Pour cela, nous avons stoppé les calculs sur les index dès que l'index intermédiaire avait

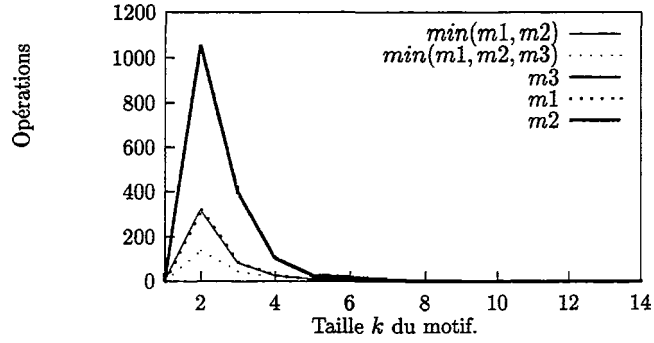


FIG. 9.12 – *Comparaison des différents algorithmes de calcul de* $|I \circ J|$.

une taille inférieure à une constante *borne*. En effectuant la recherche des mêmes motifs avec cet algorithme, nous obtenons le graphe de la figure 9.11. Sur ce graphe, nous avons représenté le temps de calcul de l'index qui se décompose maintenant en T_{comp}, le temps de calcul sur les index et T_{texte}, le nombre d'opérations pour vérifier chaque occurrence dans le texte[17]. Le temps de recherche est à nouveau décroissant avec k. On remarque aussi que la complexité diminue quand la valeur de la borne augmente. La meilleure complexité est pour *borne* $= +\infty$, c'est-à-dire pour un calcul qui se contente de vérifier les occurrences autour de la lettre la plus rare ! Il faut cependant être prudent dans l'interprétation de ce résultat. Nous avons sommé le nombre d'opérations nécessaires pour calculer l'index et le nombre d'accès directs au disque. Ces derniers sont évidemment beaucoup plus coûteux (pour des textes ne pouvant être chargés en mémoire centrale) que des recherches dans des structures chargeables en mémoire pour la plupart en un accès au disque (voir page 191). Nous montrerons l'influence réelle de la borne sur un texte suffisamment important dans le cadre de la recherche d'expressions régulières. Il faut cependant retenir que l'heuristique prenant la lettre la plus rare du mot et vérifiant sur le texte autour de cette lettre s'il y a une occurrence, est en elle-même meilleure pour des motifs suffisamment grands qu'un algorithme ayant une complexité constante en $\log N$.

Avant de conclure sur la complexité de la recherche d'un mot dans un texte, regardons comment les différentes méthodes que nous avions proposées (page 185) pour calculer $I \circ J$ sont utilisées. On suppose $|I| \leq |J|$. On appelle m_1 la méthode consistant à parcourir I séquentiellement et à vérifier par l'opérateur δ la correspondance dans J. On appelle m_2 la méthode symétrique et m_3 la méthode lisant séquentiellement d'une manière ordonnée les deux index. Nous avons représenté T_{comp} pour chacune de ces méthodes dans le cas de la recherche dans le texte d'un mot tiré aléatoirement (figure 9.12). On observe qu'aucune des trois méthodes n'est une bonne approximation de la méthode consistant à choisir dynamiquement la méthode de composition. Cependant, seule la méthode m_1, assure une complexité raisonnable. Il est net que m_2 ne sert que très rarement (en fait m_2 est

17. La vérification est très simple, puisque la position exacte de l'occurrence est déduite des calculs déjà faits (ce qui nous donne donc une complexité linéaire avec la taille de l'index, mais maintenant l'opération se déroule en mémoire centrale : il n'y a en tout qu'un seul accès direct).

plus efficace que m_1 dans le cas $y.\log x \leq x \log y$ avec $x \leq y \Rightarrow x \leq y \leq x+1$!).

Recherche de mots dans un texte : conclusion

Nous avons présenté un algorithme de recherche de mots dans un texte. Cet algorithme appelé *Index-Index* utilise uniquement l'index du texte et ordonne des opérations de manière à minimiser le coût de la recherche. Nous avons montré que dans le cadre d'une distribution zipfienne, nous pouvons résumer le comportement de cet algorithme ainsi : mis à part pour des motifs de taille 2, il selectionne un sous-ensemble potentiel d'occurrences en un nombre très petit d'opérations (comparé à la complexité d'un algorithme comme le *suffix-array*[18]). Cependant, deux caractéristiques négatives peuvent jouer :

- comme l'algorithme recherche un élément rare dans le motif, pour un motif de petite taille (en fait pour $k = 2$), l'algorithme devra parfois faire la composition de deux gros index. Le nombre d'opérations qu'il effectuera alors sera au pire en $\frac{N}{A}$ où A est le coefficient propre à la loi de Zipf (entre 10 et 20 pour des textes en français). Par exemple, cette mauvaise situation sera atteinte si on recherche la séquence *de de* dans un texte en français. Pour ce problème, nous n'avons pas apporté de solution. Il se trouve qu'aucune des grammaires que nous avons écrites ne présente de telles séquences et il est vraisemblable que sur un système d'information quelconque, ce genre de requête n'intéresse personne[19].

- Après avoir localisé, en un petit nombre d'opérations, la ou les occurrences possibles d'un mot, pour des grands motifs, la complexité augmente linéairement avec la taille du motif et ce, avec un coefficient assez important. Pour résoudre ce problème, lié encore une fois à la manipulation des index de mots grammaticaux, nous proposons une stratégie d'échappement. Cette stratégie, variante *Index-Texte* de notre algorithme, arrête le calcul dès que le nombre de candidats possible est petit[20] et continue la recherche en vérifiant directement dans le texte chaque occurrence potentielle.

Nous avons aussi montré qu'il serait possible d'accélérer la recherche en effectuant des factorisations dans le calcul. Pour une séquence en langue naturel, ces factorisations sont extrêment rares et celles qui se produisent ne sont pas intéressantes. Nous reviendrons sur cette possibilité de factorisation intrinsèque à la notion d'index pour la recherche d'expressions régulières.

9.6.3 Recherches de mots décrits par des index de longueur variable

Nous avons considéré jusqu'à maintenant, la recherche de mots formés de lettres ayant des index de taille fixe. Ce cas n'est pas le cas général. Supposons, par exemple, que nous ayons indéxé les adverbes de dates noté <*DATE*>. On recherche les occurrences dans le texte de l'expression *réunion de* <*DATE*>. L'index de ce «mot» est $I = I_{\textit{réunion}} \circ I_{\textit{de}} \circ$

18. Nous rappelons que si cette comparaison permet de situer les deux algorithmes, notre algorithme est utilisable (et utilisé) sur une structure non linéaire (structure enrichie) sur laquelle le suffix-array ne peut fonctionner.

19. D'ailleurs, certains moteurs de recherche sur le Web n'ont pas ce problème puisqu'ils n'indexent pas du tout ces mots, appelés mots grammaticaux.

20. Ce qui est toujours le cas pour un grand motif. Pour un petit motif, ce problème n'existe pas.

$I_{<DATE>}$. $I_{<DATE>}$ n'est vraisemblablement pas un index de taille fixe, puisque *le 3 janvier* ou *demain* en sont deux occurrences possibles. Supposons que $|I_{réunion}| = 200$, $|I_{de}| = 200\,000$ et $|I_{<DATE>}| = 4\,000$. Pour optimiser le calcul de I, nous aimerions pouvoir effectuer $I_{réunion} \circ \alpha_{-1}(I_{<DATE>}) \circ \alpha_{||I_{<DATE>}||}(I_{de})$. Or $||I_{<DATE>}||$ n'est pas défini, puisque $I_{<DATE>}$ est de longueur variable.
Ici la solution consiste à permuter *réunion* avec *de*. On obtient :

$$I = \alpha_{-1}(I_{de}) \circ \alpha_1(réunion) \circ I_{<DATE>}$$

calcul qui s'effectue en deux étapes : calcul de $I_1 = \alpha_1(réunion) \circ I_{<DATE>}$, et puis celui de $I = \alpha_{-1}(I_{de}) \circ I_1$.

D'une manière plus générale, si $b_1....b_k...b_n$ est un mot, avec b_k qui n'est pas de longueur fixe, alors on ne pourra pas effectuer de permutation des éléments de $b_{k+1}...b_n$ avec les éléments de $b_1...b_{k-1}$. Il n'y aura pas de permutation dans l'ordre du calcul qui franchira les éléments de longueur variable. La stratégie que nous avons adoptée dans ce cas est une réorganisation locale dans les zones «libres» où l'on choisit dynamiquement les compositions les moins coûteuses. Pour cela, tout motif est organisé comme indiqué sur le schéma de la figure 9.13 : les lettres de longueur variable dans les cases en pointillées alternent avec des zones libres qui sont des ensembles de lettres de longueur fixe stockées dans des listes ordonnées (dans les colonnes). Ces zones libres sont numérotées. À chaque itération, le programme peut effectuer sur chaque colonne i les quatre compositions notées iG, iD, iS et iV, soit :

- iG : composition du premier élément de la colonne i avec l'élément de longueur variable à gauche ;
- iD : composition du premier élément de la colonne i avec l'élément de longueur variable à droite ;
- iS : composition des deux premiers éléments de la colonne i ;
- iV : dans le cas où la colonne est vide, composition des deux éléments de longueur variable à gauche et à droite de la colonne i.

Pour chacune des opérations de taille fixe, une translation devra être appliquée avant composition dans le cas général, puisque l'ordre sera modifié : ce coefficient se calcule de la même maniére que dans la partie précédente.

Sur la structure initiale, l'ensemble de ces opérations sont énumérées et classées par coût dans une file de priorité. À chaque itération, le programme effectue la première opération de la file de priorité, il met à jour cette file et la structure complète. Notons que quand le programme effectue l'opération $1G$ par exemple, il doit supprimer de la file de priorité les entrées correspondant à $1D$ et $1S$. Pour cela, un pointeur est conservé sur chacune des entrées de la file. Le nombre d'opérations faites sur la file de priorité est en $O(k)$, où k est la taille du mot.

Il est facile de voir que cet algorithme est équivalent au précédent dans le cas d'un mot ne contenant aucune lettre de longueur variable. Par contre, l'utilisation d'une borne

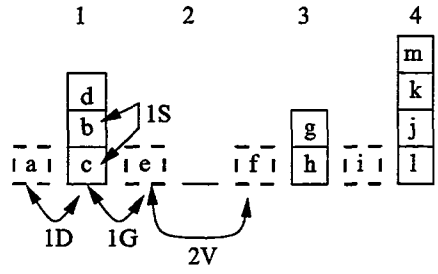


FIG. 9.13 – *Rangement d'un mot abcde...m contenant des lettres de longueur variable: a,e,f et i. Les lettres de longueur fixe sont des listes ordonnées par fréquence croissante représentées en colonne. La colonne 2 est vide.*

pour appliquer *Index-Texte* ne fonctionne plus, puisque à toute étape du calcul, nous n'avons pas un index intermédiaire mais autant d'index que de colonnes. Pour cela, nous supprimons avant la première itération l'ensemble des index de longueur fixe dont la taille est supérieure à une valeur fixée. Cette suppression suit le principe du lemme suivant:

Lemme 3 Soit $I = I_1 \circ I_2 \circ I_3$. Si I_2 est de longueur fixe, alors tout descripteur (p,l) de I dans le texte, contient un descripteur (p',l') de $I_1 \circ \alpha_{-||I_2||}(I_3)$ (c'est à dire $p' \geq p$ et $p'+l' \leq p+l$).

Preuve : *Soit* $(p,l) \in I$, *on a* $l = l_1 + l_2 + l_3$ *avec* $l_2 = ||I_2||$ *et* $(p,l_1) \in I_1$, $(p+l_1,l_2) \in I_2$ *et* $(p+l_1+l_2,l_3) \in I_3$. *Donc* $(p+l_1,l_3) \in \alpha - ||I_2||(I_3)$, *ce qui entraîne* $(p,l-l_2) \in I_1 \circ \alpha_{-||I_2||}(I_3)$ □

C'est en fait ce lemme qui était utilisé dans la partie précédente quand nous arrêtions le calcul sur l'index pour vérifier les occurrences sur le texte. En effet, nous avions ordonné la composition $I = I_1 \circ \alpha_{-||I_2||}(I_3) \circ \alpha_{||I_3||}(I_2)$ et le calcul s'arrêtait après avoir effectué $I_1 \circ \alpha_{-||I_2||}(I_3)$. On vérifiait alors dans le texte si la séquence indexée correspondait à une occurrence réelle. Nous allons donc limiter le calcul aux index suffisamment petits. L'algorithme décrit précédemment permet donc d'étendre l'algorithme *Index-Index* et l'algorithme *Index-Texte* dérivé, pour des mots quelconques.
Nous considérons maintenant la recherche d'expressions régulières quelconques.

9.6.4 Recherche d'expression régulière

Nous avons vu dans le lemme 2 qu'il y avait équivalence directe entre les opérations d'addition «+» et de concaténation «.» dans les expressions rationnelles avec les opérations «∪» et «∘» dans les index associés à ces expressions rationnelles. Nous savons faire correspondre à tout automate fini une opération sur les index du texte. De la même manière que précédemment, il va être nécessaire de réordonner cette opération de manière à être le plus rapide possible. Nous commençons par montrer les nouveaux problèmes se posant pour les expressions régulières, lorsqu'on essaye de faire correspondre l'algorithme le plus simple utilisé jusqu'à maintenant: l'algorithme *Index-Texte* avec l'utilisation de l'index uniquement pour sélectionner le point de départ de la recherche.
Rappelons le fonctionnement de cet algorithme pour un mot sur un exemple simple:

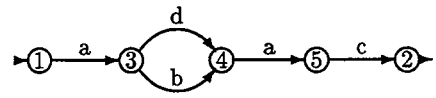


FIG. 9.14 – *Confrontation de l'intuition de «l'élément le plus rare» à un automate trivial.*

Exemple : Nous recherchons toutes les occurrences de *abac* dans le texte suivant :

abaabcabbdcabacabaabab bdabadacbaabaca

L'index de ce texte est :
$I_a = \{1, 3, 4, 7, 12, 14, 16, 18, 19, 21, 25, 27, 29, 32, 33, 35, 37\}$
$I_b = \{2, 5, 8, 9, 13, 17, 20, 22, 23, 26, 31, 34\}$
$I_c = \{6, 11, 15, 30, 36\}$
$I_d = \{10, 24, 28\}$

Comme la lettre **c** du motif recherché a la fréquence la plus faible dans le texte, celui-ci est choisi comme point de départ :

abaabcabbdcabacabaabab bdabadacbaabaca

avec
- '↓' Points de démarrage du pattern-matching local.
- '←' Test réussi sur le caractère courant.
- 'x' Test raté sur le caractère courant.
- '.' Le motif a été trouvé.

Pour localiser le motif, 5 accès directs au texte sont nécessaires.

Pour une expression représentée par un automate fini, l'idée consistant à rechercher l'élément de l'automate le plus rare dans le texte ne donne pas le meilleur résultat possible. Considérons par exemple l'automate de la figure 9.14. La lettre la plus rare de notre texte qui figure dans cet automate est **d**. Si nous commençons la recherche autour de cette lettre, nous aurons besoin de 3 tests pour vérifier le chemin **adac**. Nous devons encore rechercher le chemin **abac**, ce qui demandera avec la même méthode 5 accès directs au texte. Le coût total de la recherche est donc de 8.
Si nous avions commencé la recherche directement autour de la lettre **c**, le coût global aurait été de 5, puisque tous les chemins de cet automate passent par cette lettre. La méthode de la lettre la plus rare ne convient donc pas pour des recherches d'expressions régulières.

L'algorithme *Index-Text* pour les automates finis

Soit $\mathcal{G} = (\mathrm{A}, \mathcal{N}, T, s, t)$ un automate fini, où $\mathcal{A}$ est l'alphabet, $\mathcal{N}$ l'ensemble des états, $T \subset \mathcal{N} \times \mathcal{N} \times A$ l'ensemble des transitions, s l'état initial et t l'état final. On notera $\bar{t}$ l'étiquette d'une transition $t = (n_1, n_2, \bar{t})$.
Soit $C \subset T$ un sous-ensemble des transitions de $\mathcal{G}$. C est appelée une **coupe** de l'automate $\mathcal{G}$, si pour tout chemin de l'automate, il existe au moins une transition du chemin qui est un élément de C.
On note $p(a) = |I_a|$ le nombre d'occurrences de a dans T.
Pour tout couple (x, y) de $\mathcal{N} \times \mathcal{N}$, on note $\mathcal{SG}(x, y)$, le sous-automate de $\mathcal{G}$ défini par $\mathcal{SG} = (\mathcal{A}, \mathcal{N}, T, x, y)$. *Texte-Index* est défini, pour les expressions régulières, par l'algorithme de la figure 9.15. L'algorithme de «pattern-matching» utilisé localement peut être n'importe lequel des algorithmes fonctionnant sur le texte linéaire.

Pour cet algorithme, le nombre d'accès directs au texte est $\sum_{(a,b,l)\in C} p(l)$, il s'agit de la

```
1.  Fonction Index-Texte(G,C,I,T):Index
2.  Index position := ∅;
3.  Pour tout (a,b,l) ∈ C
4.      Pour tout (p,l) ∈ I_l
5.          Si SG(s,a) matche T en (p-k,p-1)
6.              Et SG(a,t) matche T sur (p+l,p+n)
7.          Alors position → Ajoute(p-k,n+k+1);
8.  Retourne position;
```

FIG. 9.15 – *Algorithme Index-Texte pour rechercher des expressions régulières dans un texte. Cet algorithme utilise une coupe C de l'automate.*

quantité que nous voulons minimiser :

$$\min_{C:cut} \sum_{(a,b,l)\in C} p(l) \qquad (min_1)$$

On appelle $\sum_{(a,b,l)\in C} p(l)$ le poids de la coupe.

Exemple : Dans l'automate de la figure 9.14, en conservant les fréquences de l'exemple précédent, on considère les sous-ensembles de $\mathcal{T}$: $C_1 = \{(1,2,\mathbf{a})\}$, $C_2 = \{(1,2,\mathbf{a}),(3,4,\mathbf{a})\}$, $C_3 = \{(2,3,\mathbf{d})\}$, $C_4 = \{(4,5,\mathbf{c})\}$. Il est clair que C_1, C_2 et C_4 sont des coupes de poids respectifs 18, 36 et 5. C_2 n'est pas une coupe car le chemin passant par $(2,3,\mathbf{b})$ ne passe par aucun élément de cet ensemble. (min_1) vaut 5 et est atteint pour C_4.

En supposant que seuls les accès directs au texte sont coûteux, nous voyons qu'il est possible de factoriser les accès au texte autour des transitions de même étiquette. En effet, dans l'algorithme précédent, pour chaque transition différente de la coupe C, nous comptons un nouvel accès au texte, indépendamment du fait qu'on a déjà fait appel à l'index de la même lettre.
Nous appelons $C_{\mathcal{A}}$ la projection de C sur $\mathcal{A}$. L'algorithme précédent devient alors :

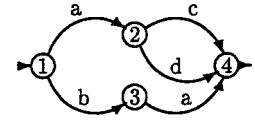
```
1.  Fonction Index-Texte2(G,C,I,T):Index
2.  Index position := ∅;
3.  Pour tout l ∈ C_A
4.      Pour tout (p,l) ∈ I_l
5.          Pour tout (a,b,l) ∈ C
6.              Si SG(s,a) matche T en (p-k,p-1)
7.                  Et SG(a,t) matche T sur (p+l,p+n)
8.              Alors position → Ajoute(p-k,n+k+1);
9.  Retourne position;
```

Grâce à cette factorisation, l'index de chaque lettre de C est utilisée une seule fois.
Nous voulons maintenant factoriser :

$$\min_{C:cut} \sum_{l \in C_{\mathcal{A}}} p(l) \qquad (min_2)$$

Nous appelons toute solution C de (min_2) : l'ensemble de **transitions pivots** de $\mathcal{G}$ et la projection associée $C_{\mathcal{A}}$: l'ensemble des **lettres pivots** de $\mathcal{A}$.

Exemple : Considérons l'automate ci-contre, qui doit s'appliquer à un texte, dans lequel $|I_a| = 18$, $|I_b| = 12$, $|I_c| = 5$ et $|I_d| = 3$.



Soit $C_1 = \{(2,4,c),(2,4,d),(1,3,b)\}$. C_1 est une coupe de poids 20 minimisant (min_1). Par contre C_1 ne minimise pas la somme (min_2) puisque (min_2) vaut toujours 20 alors que cette somme vaut 18 pour $C_2 = \{(1,2,a),(3,4,a)\}$, le poids pour (min_1) de C_2 vaut 36.

Nous montrons dans le chapitre suivant que (min_1) est le problème dit de la *coupe minimale* dans le contexte *flot maximal/coupe minimale* (*max flow/min cut*. Ce problème qui est résoluble par les algorithmes des chemins améliorants (*augmenting path*). Nous montrerons aussi comment la notion de flot et de chemin améliorant s'adaptent pour résoudre (min_2).

Remarquons que pour accéder au texte à l'aide d'une coupe, nous n'avons pas du tout tenu compte du fait que l'automate était acyclique ou non. Dans le chapitre suivant, nous aurons besoin de considérer que l'automate est acyclique pour la résolution de (min_2),. Nous pouvons donc restreindre la recherche de la coupe minimale aux automates acycliques, quitte à rendre notre automate cyclique en «coupant les boucles», ce que nous avons réalisé en faisant un parcours en profondeur d'abord de nos arcs. La complexité théorique d'une telle approche peut-être exponentielle. Sur nos graphes de petite taille et faiblement cycliques, le comportement est excellent.

L'algorithme décrit précédemment utilise une «coupe» de l'automate comme point de départ de l'analyse (nous décrivons l'algorithme permettant de déterminer cette coupe dans le chapitre suivant). L'index est utilisé d'une manière simple, sans opération aucune. Nous montrons que nous pouvons réutiliser les stratégies présentées pour la localisation de mots pour un algorithme *Index-Index*.

Stratégie naïve

Considérons une transition $t = (n_k, n_l, e)$ de la coupe. En explorant en profondeur tous les chemins entre n_l et t, on peut calculer l'index du sous-graphe «à droite de la transition t» (algorithme 9.18), comme le montre la figure 9.16. On fait de même (algorithme 9.19) en inversant le sous-graphe gauche, on obtient l'index de tous les chemins passant par t (algorithme 9.17). Cet algorithme donne le résultat attendu, puisque tous les chemins seront explorés par définition de la coupe. Le coût principal du calcul est celui premières compositions autour de la transition (que nous avons choisi à droite par défaut[21] :

21. En fait, il faut choisir le minimum des coûts «vers la droite» et «vers la gauche», nous supposons dans la suite que c'est ce qui est fait, de manière à ne pas doubler la taille des algorithmes présentés.

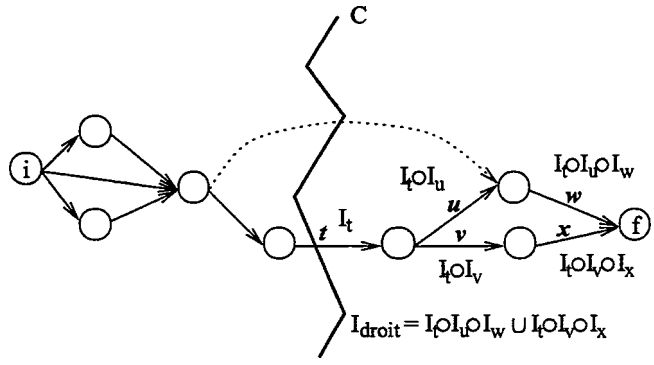


FIG. 9.16 – *Calcul de l'index des sous-chemins droits partant de la transition t de la coupe.*

$$C(\mathcal{C}) = \sum_{(n_1,n_2,e)\in\mathcal{C}} \sum_{(n_2,n_3,f)\in\mathcal{T}} c(|I_e|, |I_f|)$$

En attribuant à chaque transition (n_1, n_2, e) la valeur $\sum_{(n_2,n_3,f)\in\mathcal{T}} c(|I_e|, |I_f|)$, il est donc possible de trouver $\mathcal{C}$ à l'aide de l'algorithme de résolution de (min_1), cela est réalisée par la fonction PondereTransition(Graphe) (figure 9.20).

Cette stratégie naïve est assez peu optimisée et ne tient presque pas compte de toutes les propriétés des index définis auparavant. Nous montrons maintenant comment optimiser cette stratégie.

Deux transitions de la coupe dans le chemin

Avec l'algorithme précédent, si deux éléments d'une coupe se trouvent sur le même chemin, ce chemin sera examiné lors des deux calculs des chemins passant par chacune des transitions. L'ajout d'un simple test dans les fonctions IndexCheminsDroits et IndexCheminsGauches permet de vérifier qu'on ne considère pas une transition appartenant à la coupe et déjà analysée, auquel cas l'index du sous-graphe en cours doit être retourné vide, puisque tous les chemins considérés ont déjà étés indexés.
Remarquons que dans ce cas là le coût est «théoriquement» donné par la formule suivante (en supposant que $\mathcal{C}$ est ordonné ainsi $\{t_1, t_2, ..., t_k\}$) :

$$C(\mathcal{C}) = \sum_{1\le i\le k : t_i=(n_1,n_2,e)} \quad \sum_{t'=(n_2,n_3,f)\in\mathcal{T} \text{ et } t'\neq t_j \text{ pour } j<i} c(|I_e|, |I_f|)$$

La minimisation de $C(\mathcal{C})$ par les méthodes utilisées pour (min_1) et (min_2) ne sera pas possible, car on ne peut plus attribuer à chaque transition une valeur indépendamment de la coupe, comme nous l'avions fait auparavant : en effet, le terme $\sum_{...} c(|I_e|, |I_f|)$ dépend de la coupe elle-même. Cependant, il est simple de voir qu'on ne peut pas avoir deux transitions «adjacentes» dans la coupe minimale. Supposons qu'il existe $n_1 \in \mathcal{N}$ et

```
1.  Fonction IndexAutomate(coupe C):Index
2.  Index I =: ∅ ;
3.  Pourtoute transition t = (n_k, n_l, e)
4.      Index Itemp :=index(e) ;
5.      Itemp :=IndexCheminsDroits(I,n_l) ;
6.      Itemp :=IndexCheminsGauches(I,n_k) ;
7.      I := I ∪ Itemp;
8.  Retourne I;
9.  Fin
```

FIG. 9.17 – *Algorithme de calcul de l'index des chemins passant par la coupe.*

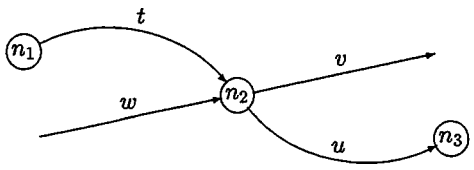
```
1.  Fonction IndexCheminsDroits(Index I,Noeud n_l)
2.  Si n_l = t Alors Retourne I ;
3.  Index I' := ∅ ;
4.  Pour toute transition t' = (n_l, n_m, e')
5.      I' := I'∪ IndexCheminsDroits(Circ(I',index(e'),n_m) ;
6.  Retourne I' ;
7.  Fin;
```

FIG. 9.18 – *Algorithme de calcul de l'index des sous-chemins droits basés sur I et partant du nœud n_l.*

```
1.  Fonction IndexCheminsGauches(Index I,Noeud n_k)
2.  Si n_l = s Alors Retourne I ;
3.  Index I' := ∅ ;
4.  Pour toute transition t' = (n_m, n_k, e)
5.      I' := I'∪ IndexCheminsDroits(Circ(index(e'),I'),n_m) ;
6.  Retourne I' ;
7.  Fin;
```

FIG. 9.19 – *Algorithme de calcul de l'index des sous-chemins gauche basés sur I et partant du nœud n_k.*

$u, t \in C$ comme dans le schéma suivant :



Lemme 4 Ce cas de figure est impossible si C est minimal.

Preuve : *Il existe nécessairement une deuxième transition v partant de n_1, sinon $C/\{t\}$ serait aussi une coupe, ce qui exclu le caractère minimal de C. La transition v ne peut appartenir à la coupe minimale, sinon :*

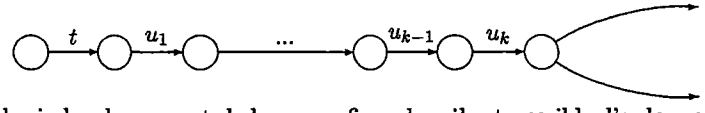
- *soit il existe une troisième transition partant de n_1 pour les mêmes raisons (ce que nous supposons faux, le raisonnement serait le même avec cette transition supplémentaire)*
- *soit t n'appartient pas à la coupe, ce qui faux par hypothèse. De même, il existe aussi une deuxième transition w qui n'est pas dans la coupe arrivant en n_2. Les chemins passant par les transitions w et v passent nécessairement aussi par une transition de C. Cette (ou ces transitions) sont soit en amont de w soit en aval de v.*

Dans chacun de ces cas, cela entraîne nécessairement qu'il existe une coupe complète à droite de n_1 (ou à gauche n_1), ce qui fait que t (ou u) est de trop dans la coupe, ce qui est contradictoire avec la minimalité de la coupe. □

Cela prouve que C correspond au schéma de minimalisation de l'algorithme naïf que nous avons montré être du type (min_1). Donc on obtient C par l'algorithme de résolution de (min_1).

Permutations locales

La caractéristique des index la plus intéressante que nous exploitée pour la reconnaissance de mots dans un texte, était la possibilité de permutation des termes. Cela permettait en particulier de reporter au plus tard la composition avec un «gros» index. Nous montrons que dans une certaine mesure, nous pouvons toujours faire cette permutation. Le premier cas simple est la permutation sur un chemin «sans bifurcation» (ce que nous notons pour tout $u_i : t \rightarrow^* u_i$) :



Si tous les index des u_i sont de longueur fixe, alors il est possible d'ordonner la composition $I_t \circ I_{u_1} \circ ... \circ I_k$ suivant les tailles croissantes des index de la même manière que pour les mots. De plus, le coût des premières compositions est alors donné par :

$$C(C) = \sum_{t \in C} \sum_{t' \text{ tel que } t \rightarrow^* t' \text{ et } |I_{t'}| \text{ minimal}} c(|I_{\overleftarrow{t}}|, |I_{\overrightarrow{t'}}|)$$

C est encore obtenu par la résolution de (min_1) en attribuant comme poids à chaque transition t la valeur $\sum_{t'}$ tel que $t \rightarrow^* t'$ et $|I_{t'}|$ minimal $c(|I_t|, |I_{t'}|)$.

Considérons maintenant la permutation sur des chemins «bifurquants» comme dans l'exemple de la figure 9.21(a). Dans ce cas et avec les valeurs indiquées dans la figure, le coût de la première composition autour de cette transition est $c(50, 10\,000)$. Pour chacun des sous-chemins, (t, u, v) et (t, u, w) on aimerait pouvoir effectuer les permutations : $I_t \circ I_v \circ I_u$ et $I_t \circ I_w \circ I_u$, ce qui donnerait le coût cumulé plus intéressant : $c(50, 30) + c(50, 30)$. Une telle démarche effectue une «défactorisation» sur l'automate, puisque cela revient à appliquer la méthode décrite juste précédemment à l'extrait d'automate de la figure 9.21(b). Il n'est pas envisageable de défactoriser ainsi tous les chemins entre la transition t et l'état final, car ce nombre peut croître d'une manière exponentielle et cette défactorisation n'est pas réalisable tout le temps puisque nous travaillons sur des automates cycliques. Cette stratégie sera uniquement utilisée pour permuter localement une composition d'une transition de la coupe avec un trop gros index. Ici aussi, il est possible d'exprimer le coût en fonction de la coupe sous la forme d'un problème (min_1). Cela nous définit une nouvelle fonction PondereTransition(Graphe).

Déterminisation locale

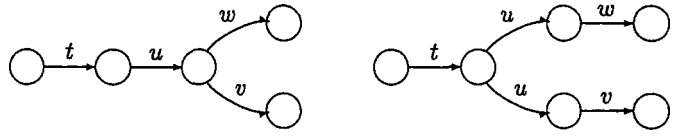
D'une manière symétrique à la «défactorisation» précédente pour repousser plus loin la composition avec un «gros index», il est possible de refactoriser localement des chemins après permutation comme nous le voyons sur la figure 9.22.

Composition approximative

Les algorithmes que nous avons présentés jusque là dans le cadre de *Index-Index* permettent une réorganisation locale de l'automate autour des transitions de la coupe pour optimiser le temps de recherche. Nous avions vu dans la recherche des mots, que le calcul jusqu'au bout sur l'index n'était pas intéressant même en reportant au dernier moment les compositions avec les gros index. Il en est de même *a fortiori* pour la recherche d'automate. Nous avons présenté au début de la partie sur la recherche d'expression régulière, un algorithme *Index-Texte* d'ordre 1. Nous pouvons envisager d'interrompre l'ensemble des algorithmes précédents pour continuer la recherche sur le texte, mais cela ne sera pas optimal puisque les algorithmes précédents considèrent uniquement les index locaux. Nous montrons ici qu'il est possible en faisant des compositions approximatives, de simuler des algorithmes *Index-Texte* d'ordre quelconque (pratiquement cependant, vu la taille des automates considérés, cette stratégie sera exclusivement utilisée à l'ordre 2). Considérons par exemple l'automate de la figure 9.23. C est la coupe minimale. Comme tous les chemins passant par y, si $|I_y|$ est petit, on pourrait envisager comme première composition $I_y \circ (I_t \cup I_u \cup I_v \cup I_w)$. Le problème est que la distance entre la transition y et chacune des transitions t, u, v, w et x n'est vraisemblablement pas fixe. Si on peut la majorer par une constante B, on peut alors considérer l'ensemble des descripteurs de l'index I_y dont la «distance» à ceux de l'index $I_t \cup I_u \cup I_v \cup I_w$ est inférieure à B : cela nous donnera un sous-ensemble réduit de points de démarrage potentiels dans le texte. Plus formellement, le schéma de l'algorithme *Index-Texte* d'ordre 2 sur les mots (voir algorithme de la figure 9.24), consiste à faire la composition des deux index les plus petits du mot. Puisque nous considérons des automates et non des mots, nous ne travaillons plus

```
1.  Fonction PondereTransition(Graphe G = (A, N, T, i, f))
2.  Pour toute transition t = (., m, t̄) ∈ T
3.      Int poids := 0 ;
4.      Pour TOUTE transition t' = (m, ., t̄') ∈ T
5.          poids := poids+c(|index(t̄)|,|index(t̄')|) ;
6.      poids(t) := poids ;
7.  Fin
```

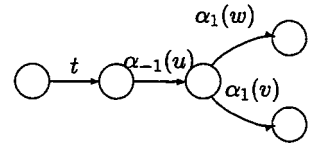
FIG. 9.20 – *Première fonction de pondération des transitions de $\mathcal{G}$.*



(a) Situation initiale. (b) «Défactorisation» locale.

FIG. 9.21 – *Exemple de permutation possible le long de chemins factorisés :* $|I_t| = 50$, $|I_u| = 10\,000$, $|I_v| = |I_w| = 30$

(a) Situation initiale. (b) Permutation.



(c) Factorisation des transitions permutées.

FIG. 9.22 – *Exemple de factorisation de transitions permutées. On suppose ici que* $|u| \ll \min(|v|, |w|)$.

sur des index simples mais sur les index des coupes. La coupe *minimale* est l'équivalente de la lettre la plus rare du mot. En gardant ce parallèle, la seconde lettre la plus rare devrait donc être donnée par une seconde coupe qui s'obtient en considérant les deux sous-graphes $(\mathcal{A}, \mathcal{N}, \mathcal{T}, i, \{n \in \mathcal{N} \mid \exists (n, m, \mathbf{k}) \in \mathcal{C}\})$[22] et $(\mathcal{A}, \mathcal{N}, \mathcal{T}, \{n \in \mathcal{N} \mid \exists (m, n, \mathbf{k}) \in \mathcal{C}\}, f)$[23]. Une seconde coupe minimale est calculée sur chacun de ces deux automates et la plus petite des deux est prise en compte. Ce schéma peut-être itéré plusieurs fois. Pour pouvoir faire la «composition» entre les index de ces coupes, il faut ensuite calculer la distance maximale entre les deux. Cela n'est réalisable que si il n'y a pas de cycle entre les deux. Dans le cas contraire, nous faisons ici une approximation valable dans notre contexte linguistique: chaque cycle ne peut être emprunté que C_{max} fois pour un même chemin. La fonction DistanceMax(C,C') calcule alors cette distance maximale.
Dans ce cas, les deux coupes sont obtenues à l'aide de la minimisation (min_2), puisque la complexité sera liée au nombre d'accès au texte.

Recherche de sous-motifs dans l'automate

La deuxième propriété remarquable du travail avec les index que nous avions mise en évidence sur les mots est la possibilité de factoriser des sous-parties de l'expression. Nous avions montré que pour les mots, cette possibilité était très réduite (parce qu'improbable avec une distribution zipfienne) et guère intéressante quand elle se produisait. Sur des automates décrivant des grammaires en langage naturel, ces factorisations sont par contre beaucoup plus fréquentes (voir exemple sur la figure 9.25). Nous n'avons pas plus proposé d'algorithme de localisation de motifs répétés dans un automate. Cependant, le choix fait dès le départ de ne pas «ouvrir» les sous-graphes permet d'utiliser la factorisation naturelle faite par le concepteur de la grammaire. Chaque sous-graphe est indexé indépendamment et l'indexation d'un graphe fait directement appel à l'indexation de ses sous-graphes. Cette démarche présente en plus une particularité primordiale sur le plan du développement d'une bibliothèque de graphes: la modification d'un graphe ne vas entraîner la recompilation des index que des graphes appelant ce graphe.

22. Il s'agit de l'automate de départ dans lequel les états finaux sont les nœuds de départ des transitions de la coupe.
23. Il s'agit de l'automate de départ dans lequel les états initiaux sont les nœuds d'arrivée des transitions de la coupe

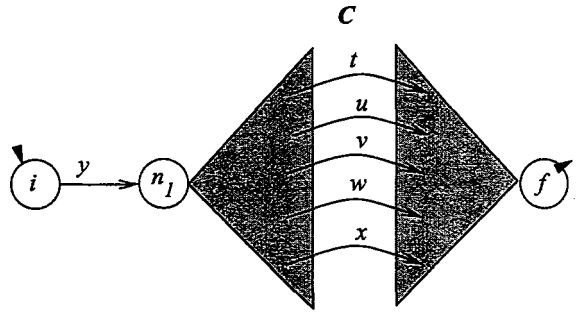


FIG. 9.23 – *Première coupe minimale* $\mathcal{C} = \{t, u, v, w, x\}$ *sur un automate.*

```
1.  Fonction IndexTexteOrdre2(Graphe G):Index
2.  PondereTransition(G);
3.  EnsembleTransition C :=CoupeMinimale(G);
4.  Graphe G' := G;
5.  Pour toute transition t = (n1, n2, t̄) ∈ C
6.      CoupeTransition(G',t);
7.      AjouteTransition(G',n1,f,ε);
8.  EnsembleTransition C' :=CoupeMinimale(G');
9.  Graphe G'' := G;
10. Pour toute transition t = (n1, n2, t̄) ∈ C
11.     CoupeTransition(G'',t);
12.     AjouteTransition(G'',i,n2,ε);
13. EnsembleTransition C'' :=CoupeMinimale(G'');
14. Si p(C') < p(C'')
15.     Alors Retourne ApproxCirc(C,C',DistanceMax(C',C));
16.     Sinon Retourne ApproxCirc(C,C'',DistanceMax(C',C));
17. Fin
```

FIG. 9.24 – *Algorithme de calcul de l'index d'ordre 2 pour Index-Texte.*

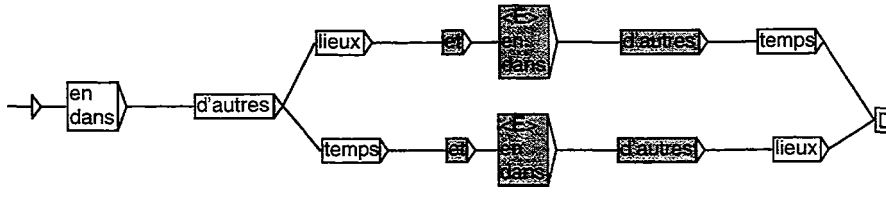


FIG. 9.25 – *Exemple de graphes présentant une factorisation potentielle intéressante (en grisé)*

Performance

Pour évaluer la performance de ces algorithmes nous avons utilisé des bibliothèques de graphes existantes et nous les avons appliquées avec plusieurs algorithmes différents sur un texte en français d'un million de mots.
Nous présentons ici trois séries de tests. La première a été effectuée sur des graphes de date qui représentent 107 graphes comportant en tout environ 3 500 nœuds. Le nombre d'expressions reconnues dans notre texte par cette grammaire est d'environ 13 000. La deuxième série de test a été faite sur la grammaire de *ministre* (titre de personalité contenant le mot *ministre*). Cette grammaire comporte 30 graphes et le nombre de nœuds est d'environ 300. Le nombre d'occurrence de cette grammaire est de l'ordre de 1 200. Enfin, la troisième série de test porte sur l'ensemble des 100 premières expressions différentes extraites des concordances des deux grammaires précédentes. Cette série de test est équivalente à un *multi-word matching* avec 100 mots différents. Le nombre d'expression reconnue est de l'ordre de 1 000.

Les différents algorithmes testés sont :

- **IT1 global** : Algorithme *Index-Texte* d'ordre 1 travaillant uniquement sur le graphe principal dans lequel tous les appels aux sous-graphes ont été substitutié par les sous-graphes.
- **IT1** : Algorithme *Index-Texte* d'ordre 1 compilant exactement l'index associé à chaque sous-graphe (donc pour chaque sous-graphe il utilise l'index et le texte).
- **IT1 approx** : Algorithme *Index-Texte* d'ordre 1 ne vérifiant les occurrences proposées par l'index que pour le graphe principal. Pour tous les sous-graphes, il construit des index approximatifs basés sur la consultation d'index d'ordre 1.
- **IT2** : Algorithme *Index-Texte* d'ordre 2 compilant exactement l'index associé à chaque sous-graphe (donc pour chaque sous-graphe il utilise l'index et le texte).
- **II naïf** : Algorithme *Index-Index* naïf de la page 211.
- **II** : Algorithme *Index-Index* avec les possibilités de permutation locale (page 215).
- **II approx** : Algorithme *Index-Index* approximatif ne faisant pas l'intersection avec les index de taille supérieure à 1 000. Une vérification de chaque occurrence sur le texte est alors ensuite nécessaire.

Chacun de ces algorithmes excepté **IT1 global** construit les index de chaque sous-graphe pendant le calcul de l'index de la grammaire principale. Les temps de compilation sont donnés dans le tableau 9.1. Ils sont indiqués en seconde. Nous indiquerons le temps de recompilation pour une modification dans le graphe principal dans le tableau 9.2. Les temps sont donnés en seconde pour des séries de test effectuées sur un *Pentium II 300Mhz*. Nous avons distingué dans ce «temps de compilation», la partie propre à la manipulation de l'index et la partie propre à la consultation du texte. Seuls les algorithmes **II naïf** et **II** ont un temps de consultation du texte nul puisque tout le calcul se fait sur l'index.

L'algorithme **II approx** n'est intéressant dans aucune des trois séries de tests : pour la grammaire de date, le temps consacré aux calculs sur l'index est réduit, mais la réduction est négligeable par rapport au coût apporté par la vérification de chaque occurrence dans le texte. Dans les deux autres séries ce temps de calcul sur l'index n'est même pas réduit (cela s'explique par le fait que les index manipulés sont plus importants). L'algorithme **II** est nettement[24] plus intéressant que **II naïf** : c'est normal puisqu'il s'agit du même algorithme optimisé : le coefficient 2 entre ces deux algorithmes justifie les raffinements présentés. Pour les algorithmes *Index-Texte*, **IT2** est le plus intéressant pour les grammaires de date (à peu près deux fois plus rapide que le suivant). Par contre pour les grammaire décrivant les titres de *ministre* **IT1 approx** est le plus efficace mais les quatre algorithmes sont cependant comparables. Pour les phrases, les deux algorithmes **IT1 global** et **IT1 approx** n'ont pas été appliqués puisque ces deux se particularisent par leur traitement des sous-graphes : le premier les supprime en mettant tous les nœuds au même niveau et le second les traite approximativement. Pour la série phrase, **IT1** est légérement plus efficace que **IT2**.
Si on compare les algorithmes *Index-Texte* et *Index-Index* : la première stratégie est légèrement plus efficace pour la grammaire *ministre* et pour les phrases. Cependant cette stratégie est au minimum trois fois plus lente pour la grammaire de date. De plus, les temps de recompilation sont sensiblement équivalents au temps de compilation pour les algorithmes *Index-Texte*, alors qu'ils sont huit fois plus petits que les temps de compilation pour les algorithmes *Index-Index*. La différence d'efficacité relative pour la grammaire de *ministre* par rapport à la grammaire de date entre les algorithmes *Index-Index* et *Index-Texte* s'explique de la manière suivante. Dans la grammaire de *ministre*, quasiment toutes les expressions contenant le mot ministre vont être localisée par notre grammaire[25] puisque la construction de la grammaire avait justement pour objectif de recenser tous les contextes possibles du mot ministre (en politique). Donc, si on prend l'index de tous les mots ministres : on a presque exactement toutes les occurrences recherchées. Les calculs sur les index faits par les algorithmes *Index-Index* vont donc faire un grand nombre de vérication surperflues. Pour la grammaire de date, cela ne va pas du tout être la même situation : il n'y a pas un seul mot commun et caractéristique à toutes les dates possibles. Les compositions d'index sont alors peu coûteuses et réduisent fortement le nombre de candidats possibles.
D'une manière globale l'algorithme **II** est le plus efficace et en particulier si on veut pouvoir recompiler la grammaire après certaines modifications.

Il faut remarquer enfin que le temps de compilation obtenu par graphe est de l'ordre d'un tiers de seconde pour les deux séries *ministre* et *DATE*. Nous avons vu page 168 que le temps de compilation par graphe issu du lexique-grammaire des expressions figées est de quelques centièmes seulement. Le nombre d'occurrence des expressions décrites par chacune de ces grammaires en est la raison : pour tous les algorithmes présentés la complexité est fortement liée au nombre d'occurrence trouvée. Pour les expressions figées, un grand nombre d'entrées n'a aucune occurrence dans le texte et ces entrées sont éliminée juste par la première consultation de l'index.

24. D'un coefficient deux pour la grammaire de date.
25. Le mot *ministre* n'ayant qu'un sens non politique : *ministre du culte* qui n'apparaît que deux fois dans notre corpus de référence.

	Index-Texte				*Index-Index*		
	IT1 global	**IT1**	**IT1 approx**	**IT2**	**II naïf**	**II**	**II approx**
DATE	10+140	5+195	5+117	15+63	43+0	27+0	20+32
MINISTRE	0+7	0+7	0+6	4+6	11+0	8+0	14+3
Phrase×100	–	0+5	–	6+2	15+0	9+0	10+2

TAB. 9.1 – *Temps de compilation en secondes pour chacun des algorithmes et pour les trois séries de tests. Le temps de compilation est présenté comme une somme. Le premier terme de cette somme est le temps total de travail sur la structure de l'index et le second est le temps de travail sur la structure du texte.*

	Index-Texte				*Index-Index*		
	IT1 global	**IT1**	**IT1 approx**	**IT2**	**II naïf**	**II**	**II approx**
DATE	10+140	1+93	1+117	2+63	4+0	4+0	3+32
MINISTRE	0+7	0+3	0+6	1+6	2+0	1+0	4+3

TAB. 9.2 – *Temps de recompilation en secondes après modification du graphe principal.*

9.7 Indexation d'automate.

Nous avons réalisé deux moteurs utilisant les techniques décrites dans ce chapitre pour reconnaître des expressions décrites par des graphes dans un texte. Le premier (appelé I_2) travaille exclusivement sur une structure de texte enrichie comme nous l'avons montrée page 182. Ce moteur utilise un algorithme *Index-Index* pur. C'est de ce moteur que nous avons montré la performance dans la partie précédente. La structure induite (voir figure 9.26) est un automatique acyclique très simple puisque la longueur des sous-chemins quittant le chemin principal est bornée. Cette structure n'est pas suffisante pour pouvoir appliquer les transformations sur notre texte et manipuler un «vrai» automate de texte. Cette structure est néanmoins largement suffisante pour un extraire de l'information à l'aide de graphes (voir par exemple [22]). Le deuxième (appelé I_5) travaille sur un automate de texte complet. Nous montrons que l'indexation d'un automate est analogue à l'indexation d'un texte linéaire, mais que nous perdons la notion de «longueur» d'un index et avec cette notion, la propriété forte de permutation des opérations dans le calcul de l'index.

Soit A l'automate du texte et $\mathcal{N}$ l'ensemble de ces nœuds. Indexer une forme dans l'automate du texte consiste à considérer un sous-ensemble de $\mathcal{N}$. Indexer une expression complexe, par contre, nécessite de mémoriser le sous-chemin complet : la notion de longueur d'un descripteur de l'index n'est plus suffisante, puisque partant d'un nœud donné plusieurs chemins peuvent avoir cette longueur (même si le chemin est unique au moment de l'indexation, puisque la structure est dynamique). Cela signifie que pour indexer l'équivalent d'un couple (p, l), il faut une position n et l'ensemble des nœuds formant un chemin : c'est ce que nous avons représenté sur la figure 9.27. La gestion d'un tel index serait très lourde. De plus, même en indexant les chemins, la longueur du chemin n'a pas de sens en elle-même puisque chaque nœud peut regrouper plusieurs mots de la phrase initiale. Enfin, cela ne serait pas cohérent avec les transformations que nous avons présentées qui regroupent les séquences reconnues.

Nous avons donc choisi de n'indexer dans I_5 que des nœuds simples comme sur la figure 9.28. Une entrée d'index sur cette structure est donc dans le cas général une sous-ensemble de $\mathcal{N}$. L'opération de composition n'est donc plus définissable de la même manière pour la raison simple que la composition de deux index n'est pas un index, mais un chemin dans l'automate, ce qui nous ramène à la structure théorique. La composition de deux index est donc définie ainsi :

$$I_a b = I_a \circ I_b = \{(n_1, n_2) \,|\, n_1 \in I_a \text{ et } n_2 \in I_b \text{ et } \underline{\exists t \; n_1 \rightarrow^t n_2 \in A}\}$$

La condition soulignée implique forcément que l'automate du texte est accessible pour effectuer la composition : pour chaque entrée de I_a nous devrons faire un accès direct à l'automate du texte. Donc les algorithmes *Index-Index* n'existent plus. Nous montrons que nous pouvons cependant les adapter (pour obtenir des *Index-Index* approximatifs) pour les utiliser.

Pour cela, nous avons numéroté de 1 à N les nœuds du texte linéaire et assuré une numérotation réelle croissante localement régulière (pas nécessairement entière) de chacun des nœuds rajoutés à l'automate du texte. On note $pos(n)$ la valeur associé au nœud n. L'algorithme est simple : soit $(n_1, n_2, n_3, ..., n_k)$ un chemin créé entre les nœuds n et m dans l'automate du texte par une transformation. On définit $pos(n_i) = pos(n) + \frac{pos(m) - pos(n)}{k} \times i$. La «longueur» l_{max} maximale d'un nœud est maintenue à tout mo-

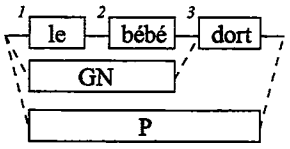


FIG. 9.26 – *Structure induite par l'index de texte linéaire. Les enrichissements sont représentés à l'aide des arcs en pointillé.*

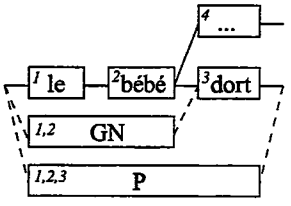


FIG. 9.27 – *Structure induite par l'index théorique d'un automate du texte. Chaque nœud de l'automate est numéroté et chaque enrichissement est repéré par son état initial et final et par le chemin suivi.*

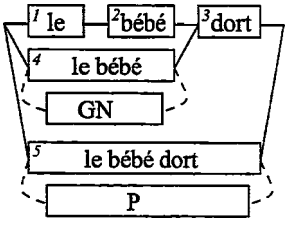


FIG. 9.28 – *Structure induite par l'index d'un automate du texte dans lequel toutes les séquences reconnues ont été dupliquées. Les enrichissements peuvent être vus comme des chemins de longueur* 1 *de l'automate dans un plan perpendiculaire.*

ment égale au maximum des $\frac{pos(m) - pos(n)}{k}$. Les index sont ensuite ordonnés à l'aide de cette valeur et la composition approximative $I_a \circ I_b$ est définie par :

$$I_a \circ I_b = \{(n_1, n_2) \mid (n_1, p_1) \in I_a \text{ et } (n_2, p_2) \in I_b \text{ et } \underline{p_2 - p_1 \leq l_{max}}$$

On voit facilement que cela permet de conserver les algorithmes établi dans la partie précédente, en donnant cependant un résultat approximatif (c'est-à-dire trop large) qu'il faut ensuite vérifier.

Une conséquence positive de cette «extension», est que la notion d'index de taille variable n'existe plus non plus, et qu'il est donc possible d'opérer toutes les permutations voulues dans les motifs recherchés, ce qui permet une reconnaissance plus performante.

9.8 Limites de l'indexation

Dans l'exemple de la page 9.2, nous avons choisi d'indexer la forme, le lemme, l'ensemble des informations sémantiques et flexionnelles, et la catégorie grammaticale. La taille de l'index du texte linéaire seul est donc quatre fois la taille de l'index des formes. La recherche d'une forme particulière, d'un lemme particulier, d'une catégorie particulière est instantané, puisque déjà indexée telle quelle. La recherche d'un motif comme <*être.V*> se fait simplement par intersection de $I_{\hat{e}tre}$ et $I_{<V>}$: $I_{\hat{e}tre} \cap I_{<V>}$. Cependant, qu'avec un tel choix des objets indexés, il n'est pas par contre direct d'obtenir <*MOT*> (n'importe quel mot) ni <*MOT*:s> (n'importe quel mot au singulier). Puisque l'index du premier se construit en calculant l'union des index de toutes les formes, et l'index du second est l'union des index de toutes les informations sémantico/morphologique contenant +s. Nous montrerons qu'il est possible naturellement de contourner le problème posé par <*MOT*>, par contre le calcul de <*MOT*:s> sera très coûteux, puisqu'il faut pouvoir extraire la marque +s de l'ensemble des marques. D'une manière très générale, l'accès aux informations sémantico/flexionnelles est lourd. Si nous voulions indexer indépendamment chacune de ces informations, la taille de l'index de chacune des marques aurait une taille de l'ordre de $O(N)$. Nous avons donc supposé qu'il n'y avait pas de sens à rechercher des objets de la forme <*MOT*:s>.

9.9 Conclusion

Nous avons présenté dans cette partie une notion d'index plus fine que celles existant par ailleurs. Nous en avons étudié les propriétés et leurs applications pour le *pattern-matching*. En particulier, nous avons démontré que l'ordre naturel des lettres avait une implication faible dans le calcul de la composition d'index. Nous avons par ailleurs montré qu'un calcul portant uniquement sur l'index est avantageux si les conditions suivantes sont vérifiées : les motifs ne sont pas trop longs et les textes sont suffisamment gros pour que les accès directs sur celui-ci soient coûteux. Nous avons établi des stratégies d'échappement du calcul sur l'index ; la reconnaissance se termine alors sur le texte lui-même. Nous avons étendu cette notion d'index à l'index d'une structure non linéaire : l'automate du texte. Les différentes opérations sur les index de texte linéaire sont conservées mais le résultat est forcément approximatif, ceci impliquant une vérification sur la structure du texte d'un ensemble d'occurrences potentielles. Nous avons aussi

montré le gain qu'il pourrait y avoir à factoriser des sous-motifs identiques dans des automates. Nous n'avons pas directement traité cette question mais nous avons simplement conservé la factorisation naturelle présente dans la présentation en niveaux séparés des automates. Une grande nouveauté dans nos algorithmes est la capacité de réindexation rapide d'une grammaire légèrement modifiée : seuls les graphes dérivant du graphe modifié sont recompilé, ce qui donne une orientation plus «programmation» au développement et à la mise au point de grammaires. Nous avons montré que grâce à des propriétés non triviales inversibilité, les indexs permettait de décrire des ensembles décrit par des grammaires dépendant du contexte. Enfin, la structure d'index est une structure excellente pour une parallélisation du traitement. Il est de même pour le processus d'application continue de graphes à un texte : plusieurs graphes peuvent être appliqués simultanément ou un unique graphe peut être appliqué par plusieurs processus parallèle en restreignant la portée de chaque processus à un bloc donné du texte.

Chapitre 10

Flot

10.1 Introduction

Dans le chapitre précédent, nous avons montré qu'il était simple de choisir des «points de départ» du *pattern-matching avec index* sur des mots dans un texte. Parallèlement, pour pouvoir choisir des «points de départ» du *pattern-matching avec index* dans un automate fini quelconque, nous devons choisir une coupe de l'automate vérifiant, suivant les algorithmes, une des deux équations de minimalisation suivante :

$$\min_{C:coupe} \sum_{(a,b,l) \in C} p(l) \qquad (min_1)$$

$$\min_{C:coupe} \sum_{l \in C_A} p(l) \qquad (min_2)$$

La différence essentielle entre ces deux problèmes de minimalisation est la différence de traitement des transitions présentant une même étiquette. Nous montrons dans ce chapitre que (min_1) est un problème classique de minimalisation de flot dans un réseau. La notion de flot dans un réseau a été introduite et formalisée par FORD et FULKERSON en 1956 [8]. Une synthèse récente du sujet est donnée dans [20]. Nous proposons un algorithme de résolution de (min_2) qui est aussi un problème de minimalisation de flot, mais avec des contraintes supplémentaires. Un flot respectant ces contraintes sera appelé un **flux**.

La complexité de l'algorithme de résolution de (min_1) sur lequel nous nous baserons [7] pour résoudre (min_2) est en $O(n^2m)$ où n est le nombre de nœud de l'automate, et m est le nombre d'arcs. Des algorithmes «plus efficaces» comme l'algorithme des excès échelonnés [2] existent et ont une complexité en $O(nm + n^2 \log B)$ où B est la capacité maximale d'un arc. Dans notre application, B est proportionnel à la taille du texte, ces algorithmes ne présentent donc pas d'intérêt dans ce cas. De plus, comme nous avons choisi d'indexer chaque sous-graphe, la taille d'un graphe (c'est-à-dire n et m) sont limités (voir page 56) par la taille de l'écran, ils sont bornés. Pratiquement, le temps de calcul de la coupe minimale est négligeable pour toutes nos grammaires considérées quelle que soit la taille du texte.

10.2 Rappels, définitions

Soit $\mathcal{G} = (\mathrm{A}, \mathcal{N}, T, s, t)$ un automate fini; dans le contexte de «flots», $\mathcal{G}$ est aussi appelé **réseau**, et les transitions sont appelées les arêtes. Dans un réseau, une valeur numérique appelée **capacité** (souvent entière et positive) est associée à chaque arête u. Nous la noterons $c(u)$.

Soit $f : T \mapsto I\!N$ qui associe un entier positif à chaque arête du réseau. f est appelé un **flot** sur $\mathcal{G}$.

On dit que f est un **flot admissible** sur $\mathcal{G}$ ssi $\forall n \in \mathcal{N}$ tel que $n \neq s$ et $n \neq t$:

$$\sum_{m,a:(n,m,a)\in T} f(n,m,a) - \sum_{m,a:(m,n,a)\in T} f(n,m,a) = 0 \qquad (a)$$

$$\text{et } \forall u = (n,m,a) \in T \ : \ f(u) \leq c(u) \qquad (b)$$

(a) est la loi de conservation des flots qui exige que la somme des flots rentrants dans chaque noeud est égale à la somme des flots sortants pour tout les noeuds à part s (la source) et t (un puit).
(b) est la contrainte du réseau: le flot dans chaque arête est inférieur à la capacité de celle-ci.

À cette notion de flot est associée (par les théorèmes que nous rappellerons) la notion de coupe définie ainsi: Une **s-t coupe** de $\mathcal{G}$ (ou **coupe** de $\mathcal{G}$ simplement, quand il n'y a pas d'ambiguïté) est un sous-ensemble $\mathcal{C}$ de T, tel que tout chemin entre s et t (c'est-à-dire, tout chemin de l'automate) a au moins une transition dans $\mathcal{C}$. On appelle $c(\mathcal{C})$, la capacité de la coupe, définie par $c(\mathcal{C}) = \sum_{(n_1,n_2,a)\in\mathcal{C}} c(a)$.

Le problème dit du flot maximal est le suivant: «Trouver f, un flot admissible de $\mathcal{G}$, qui maximise le flot entrant dans le réseau: $\sum_{m,a:(s,m,a)\in T} f(s,m,a)$.»

Ce problème est généralement associé au problème de coupe minimale: «Trouver $\mathcal{C}$ une coupe de $\mathcal{G}$ qui minimise $c(\mathcal{C})$».

Ces deux problèmes bien connus ont été étudiés en premier par FORD et FULKERSON [9]. Nous avons les résultats suivants [2]:

Théoreme 1 *Théorème d'existence* – Le flot maximal existe.

Théoreme 2 *Max-Flow, Min-Cut Theorem (Ford and Fulkerson (1956))* – La valeur maximale des flots sur un graphe est égale à la capacité minimale des coupes.

Ce théorème donne l'équivalence entre le problème du flot maximal et le problème de la coupe minimale. La résolution de ce problème utilise la notion de chemin améliorant. Un **chemin améliorant** d'un flot admissible dans un réseau $\mathcal{G}$ est un chemin sur les arêtes non orientées de $\mathcal{G}$ entre s et t. Ce chemin vérifie les contraintes suivantes: pour toute arête (n_1, n_2) du chemin améliorant, nous avons:

- Soit $\exists\, u = (n_1, n_2, \mathrm{a}) \in T$, dans ce cas $f(u) < c(u)$

- Soit $\exists u = (n_2, n_1, \mathbf{a}) \in T$ et dans ce cas $f(u) > 0$.

Ce chemin améliorant va faire varier la valeur du flot en cours en conservant son caractère admissible de la manière suivante :

- sur les arêtes orientées positivement du chemin, $f(u)$ augmente d'une unité ;
- sur les arêtes orientées négativement du chemin, $f(u)$ diminue d'une unité.

Exemple : Considérons par exemple, le réseau de la figure 10.1(a) sur lequel nous avons fait figurer les capacités et la valeur du flot pour chaque arête. Il est clair que ce flot est admissible. Le chemin améliorant représenté figure 10.1(b) vérifie les contraintes précédentes et accroît le flot global du réseau pour donner celui de la figure 10.1(c).

Cette définition des chemins améliorants utilise des augmentations d'une unité du flot entrant. On peut étendre la définition à des chemins améliorants valant k unités en vérifiant $f(u) + k \leq c(u)$ et $f(u) \leq k$.
Avec cette notion de chemin améliorant, FORD et FULKERSON proposent un algorithme de résolution du flot maximal (et donc de la coupe minimale) basée sur le théorème suivant.

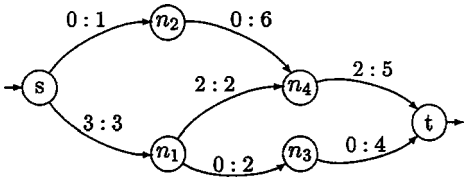
Théoreme 3 *Ford and Fulkerson (1956))* – f est un flot admissible maximal, si et seulement si, il n'y a aucun chemin améliorant pour f.

La recherche d'un chemin améliorant est une opération récursive simple sur un réseau. Partant d'un flot nul sur tout le réseau (le flot nul est admissible), on augmente le flot le long de chemins améliorants jusqu'à ce qu'on ne puisse plus en trouver. Dans ce cas, on a trouvé le flot maximal. Divers algorithmes [7], [13] se basent sur ce principe pour établir le flot maximal. Nous ne les décrirons pas en détail. Leur principe seul seul[1] permettant la compréhension de notre analyse.
Cet algorithme de construction permet de plus d'établir le théorème suivant qui nous permettra de travailler sur $I\!N^N$:

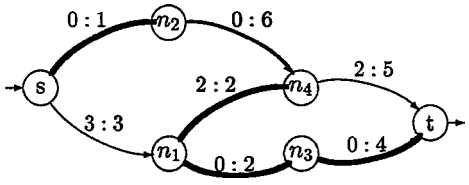
Théoreme 4 Il existe une solution du flot maximal qui est entière.

Nous supposerons que ce théorème est aussi vrai pour (min_2). La démonstration de la validité dans notre problème nécessite la présentation de l'algorithme, ce que nous repoussons après l'analyse du problème, par souci de clarté.
L'analogie hydraulique de la notion de flot est très intuitive. Le réseau représente un réseau hydraulique composé de tubes de tailles proportionnelles à la *capacité* de ces tubes. Le point d'entrée de ce réseau est s, le point de sortie est t. La valeur du flot dans ces tubes est le débit hydraulique existant. Le flot maximal va être le débit maximal de liquide que ce réseau sera capable de transporter en sachant que la vitesse du liquide en entrée est constante, mais que le débit est potentiellement infini (donc la largeur du tube d'entrée est supérieure à la somme des largeurs de tous les tubes du réseau). Une coupe sur un réseau correspond à une coupe effective d'un ensemble de tubes de manière

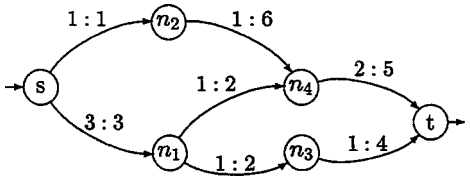
1. Ce principe seul n'est cependant pas optimal du point de vue de la vitesse de convergence.



(a) Flot admissible. Le flot entrant est 3.



(b) Chemin améliorant compatible avec le flot existant.



(c) Flot résultant de l'application du chemin améliorant. Le flot entrant est maintenant 4.

FIG. 10.1 – *Représentation du flot dans un réseau (a). Les couples a : b représentent le flot a dans chaque arête et la capacité b de chaque arête. L'application du chemin améliorant (b) augmente d'une unité la valeur valeur du flux entrant dans le réseau (c).*

séparer totalement le reseau connexe à s et celui connexe à t. La capacité de la coupe est la quantité de liquide qui coule de ces tubes coupés (instanténement, car le débit global est interrompu). L'équivalence entre le flot maximal et la coupe minimale consiste à voir que quand le débit est maximal, on peut trouver une coupe pour laquelle tous les tuyaux sont à leur plus forte charge. Un chemin améliorant est un supplément de flot envoyé dans le réseau. Ce supplément de flot peut aller dans les tubes non saturés, mais il peut aussi aller dans les tubes saturés en «repoussant» une quantité d'eau équivalente dans ce tube. L'analogie hydraulique ne doit cependant pas être poussée trop loin car les tubes sont orientés ce qui n'est pas vrai à un réseau hydraulique simple.

10.3 Coupe minimale et ensemble pivot

Le premier lien entre le problème du flot maximal et notre problème de pattern-matching est donné par la proposition suivante:

Proposition 6 Si pour toute arête u de l'automate: $p(u) = c(u)$ alors:

1. La solution de (min_1) est donnée par la solution du problème de la coupe minimale.
2. Si toutes les étiquettes des transitions de $\mathcal{G}$ sont différentes, alors l'ensemble des lettres pivots (*i.e.* la solution de (min_2)) est donnée par la solution de (min_1).

Preuve : *La définition de la coupe que nous venons de donner est équivalente à la définition de la coupe que nous avions donnée en 9.6.4. Cela prouve* 1. *Nous voyons facilement d'autre part, que la différence entre* (min_1) *et* (min_2) *réside uniquement dans la manière de traiter les transitions possédant des étiquettes identiques. Donc, s'il n'y a pas deux transitions avec la même étiquette,* (min_1) *et* (min_2) *sont équivalents. Cela prouve* 2. □

Nous avons donc une équivalence entre (min_1) et le problème de la coupe minimale, problème pour lequel il existe un algorithme de résolution que nous préciserons. Pour (min_2), nous avons une solution uniquement dans un cas très précis: le cas où toutes les transitions sont étiquetées différemment. Nous allons voir qu'il est possible d'étendre les algorithmes existant pour (min_2), en présentant d'une manière un peu différente le problème de la coupe minimale.

10.3.1 La matrice chemin-transition

Soit $\mathcal{G}$ un automate et $\mathcal{L}$ l'ensemble[2] de tous les chemins de $\mathcal{G}$. On associe à $\mathcal{G}$ une matrice binaire B appelée **matrice chemin-transition** dont les lignes représentent les chemins de $\mathcal{G}$ et les colonnes représentent les transitions. La taille de B est $(\#T \times \#\mathcal{L})$ et $B_{(c,u)} = 1$ si et seulement si le chemin correspondant à la colonne c passe par la transition correspondant à la colonne u. Par souci de clarté, nous parlerons de transition u et de chemin c à la place de «transition correspondant à la colonne u» et «chemin correspondant à la ligne c». De même, nous pourrons désigner chaque transition, et chaque chemin par leur indice. On note $N = \#T$.

2. Pour construire la matrice, il est indispensable que cet ensemble soit fini et donc que l'automate soit acylique.

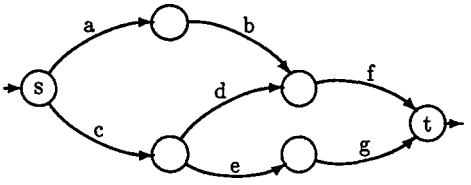


FIG. 10.2 – *Automate exemple*

Exemple : À l'automate de la figure 10.2, nous associons la[a] matrice chemin-transition suivante :

$$A = \begin{array}{c} \begin{array}{ccccccc} \downarrow a & \downarrow b & \downarrow c & \downarrow d & \downarrow e & \downarrow f & \downarrow g \end{array} \\ \left[\begin{array}{ccccccc} 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}\right] \end{array} \begin{array}{l} \leftarrow \text{abf} \\ \leftarrow \text{cdf} \\ \leftarrow \text{ceg} \end{array}$$

[a] En fait, vu que nous n'ordonnons ni les transitions ni les chemins, la matrice est une des nombreuses matrices associées à l'automate

Tout sous-ensemble C de T peut être représenté par un vecteur $x = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$ où $x_i = 1$ si la transition associée à i est dans C, et 0 sinon.

On peut maintenant définir le vecteur poids associé $P = [p_1\, p_2 \ldots p_N]$, où p_i est la capacité de l'arête i. Dans notre application, il s'agit de la fréquence dans le texte de l'étiquette associée à cette transition. Il est clair que $P.x$ vaut ce que nous avions défini page 209 comme étant la capacité de la coupe.

Proposition 7 La coupe minimale de $\mathcal{G}$ est la solution de :

$$(1) \begin{cases} \min_{x \in \mathbb{N}^N} P.x & \text{avec les contraintes :} \\ \forall c \in \mathcal{L} \ : \ (Ax)_c \geq b_c = 1 \\ \text{et } \forall u \in T \ : \ x_u \geq 0 \end{cases}$$

Preuve : *x représente une coupe C si dans chaque chemin c de $\mathcal{G}$, il y a au moins une transition u telle que $x_u = 1$ (par définition d'une coupe).*
$\Leftrightarrow$ il existe u tel que $x_u = 1$ et $A_{c,u} = 1 \Leftrightarrow \sum_{u \in T} A_{c,u} x_u \geq 1 \Leftrightarrow (Ax)_c \geq 1$.
De plus, par définition de la représentation des sous-ensembles par un vecteur x : $\forall u, x_u \geq 0$.
Nous voulons minimiser $\sum_{u \in C} p(u) = \sum_{u \in T} p(u) x_u = P.x$.
Remarquons que si y est une solution, alors $\forall u$, $y_u \geq 0$ par contrainte et $y_u \leq 1$ par minimalisation, cela valide la représentation binaire d'un sous-ensemble de C. □

(1) est un programme linéaire dont le programme linéaire dual (voir [26]) est :

$$(2)\begin{cases} \max \sum_{c\in\mathcal{L}} \pi(c) & \text{avec les contraintes :} \\ \forall u \in T \ : \ \sum_{c\in\mathcal{L}} A_{c,u}\pi(c) \leq p(u) \text{ et } \pi(c) \geq 0 \end{cases}$$

Proposition 8 Les solutions du programme linéaire (2) sont les solutions du problème du flot maximal sur $\mathcal{G}$.

Preuve : *Regardons à quoi correspond la variable π. Appelons $\pi(c)$ le* **flot propre** *du chemin c. Considérons le flot f construit de la manière suivante : le flot $f(u)$ à travers la transition u est égal à la somme des flots propres de tous les chemins passant par cette transition : $f(u) = \sum_{c\in\mathcal{L}} A_{c,u}\pi(c)$.*
Nous voulons montrer que f est admissible. L'équation (2) nous donne $\forall c,\ \pi(c) \geq 0$, donc nous avons $\forall u,\ f(u) \geq 0$. Par ailleurs, $\forall u \in T \ : \ \sum_{c\in\mathcal{L}} A_{c,u}\pi(c) \leq p(u)$, De plus $\forall u, f(t) \leq p(u)$. Cela signifie que f respecte les contraintes du réseau.
Par ailleurs, comme tous les chemins qui entrent dans un noeud en sortent (à part s et t), le flot entrant dans un noeud est égal au flot en sortant. Donc f est un flot admissible.
Enfin, $f(s) = \sum_{c\in\mathcal{L}} \pi(c)$ est maximal par définition de π, donc f est le flot maximal.
Reciproquement, si nous avons le flot maximal, nous pouvons construire le flot propre dans chaque chemin et vérifier que la somme de ces flots propres est bien maximale. La méthode pour obtenir les flots propres à partir du flot maximal n'est pas triviale ; elle utilise les chemins améliorants. Nous reviendrons sur cette construction page 237. □

Le théorème de dualité fort (*strong duality theorem*) [26] dit que si (1) a une solution, alors (2) a aussi une solution, et ces deux solutions sont associées. Cela prouve d'une autre manière le théorème du flot maximal/coupe minimale.

Remarquons que la matrice chemin-transition a été introduite pour la démonstration, mais elle n'est pas utilisable en pratique : le nombre de chemins dans un automate peut être une fonction exponentielle du nombre de noeuds. Cependant, ayant rendu clair la relation de dualité entre le flot maximal et la coupe minimale, nous avons fait ressortir la notion nouvelle de **flot propre** à chaque chemin. Cette notion est fortement liée à la notion de chemin améliorant, et cela va nous permettre en adaptant la notion de flot propre à notre problème de généraliser l'algorithme des chemins améliorants.

10.3.2 Les nouvelles contraintes du problème (min_2).

Soit $\mathcal{G}$ un automate fini comportant n étiquettes différentes. Nous ordonnons T de la manière suivante :

$$T = \{u_1^1, u_2^1, ..., u_{k_1}^1, u_1^2, u_2^2, ..., u_{k_2}^2, ..., u_1^m, u_2^m, ..., u_{k_m}^m, u^{m+1}, ..., u^n\}$$

où les transitions sont groupées par étiquettes : deux transitions ont la même étiquette si et seulement si leurs indices sont égaux. Dorénavant, nous appelons **transition à étiquette unique** une transition dont l'étiquette n'est pas utilisée par une autre transition. Les $(u^i)_{i\geq m+1}$ sont donc toutes des transitions à étiquette unique.
On considère de nouveau la matrice chemin-transition A associée à notre automate en prenant comme ordre des colonnes, l'ordre défini précédemment sur T. On décompose cette matrice en sous-matrices qui correspondent aux transitions de même étiquette de la manière suivante :

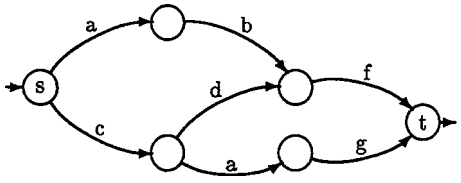


FIG. 10.3 – *Automate exemple : deux transitions portent la même étiquette* **a**.

$$A = \left[\begin{array}{|c|c|c|c|c|}\hline A^1 & \cdots & A^m & A^{m+1} & \cdots & A^n \\ \hline\end{array}\right]$$

Considérons maintenant la matrice A' suivante :

$$A' = \left[\begin{array}{|c|c|c|c|c|}\hline C^1 & \cdots & C^m & A^{m+1} & \cdots & A^n \\ \hline\end{array}\right]$$

dans laquelle les colonnes C^i sont égales à la somme booléenne des colonnes associées aux transitions étiquetées i : $(C^i)_j = \bigoplus_{1 \le k \le k_i} A^i_{j,k_i}$.

Examinons un exemple. Soit l'automate de la figure 10.3, nous lui associons la matrice A suivante :

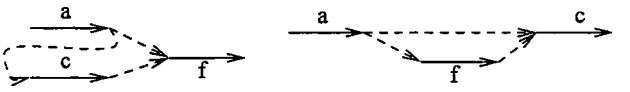
$$A = \begin{array}{c} \begin{array}{ccccccc} \text{a} & \text{a} & \text{b} & \text{c} & \text{d} & \text{f} & \text{g} \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array} \\ \left[\begin{array}{ccccccc} 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right] \end{array} \begin{array}{l} \longleftarrow \text{abf} \\ \longleftarrow \text{cdf} \\ \longleftarrow \text{cag} \end{array}$$

La matrice A' associée est donc :

$$A' = \begin{array}{c} \begin{array}{ccccccc} \text{a} & \text{b} & \text{c} & \text{d} & \text{f} & \text{g} \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array} \\ \left[\begin{array}{ccccccc} 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right] \end{array} \begin{array}{l} \longleftarrow \text{abf} \\ \longleftarrow \text{cdf} \\ \longleftarrow \text{cag} \end{array}$$

Remarquons que la matrice A' obtenue n'est pas nécessairement la matrice chemin-transition d'un automate : dans ce cas précis, l'automate correspondant devrait avoir uniquement trois chemins, le premier passe par **a**, **b** et **f** ; le second par **c**, **d**, et **f**, et le troisième par **a**, **c**, et **g** (l'ordre peut être quelconque dans les trois cas). En particulier, il faudrait pouvoir construire un automate contenant trois transitions **a**, **c** et **f** et acceptant les trois chemins (non ordonnés) **af**, **cf** et **ac**, mais pas le chemin **afc** (ou **acf**, etc.), ni le chemin **f** seul. Un simple croquis (figure 10.4) montre que ce n'est pas possible, donc cet automate n'existe pas.

Nous allons utiliser cette matrice représentant un automate théorique pour montrer que le programme linéaire précédent est maintenant équivalent au programme (min_2).
On représente de la même façon un sous-ensemble C de T par un vecteur colonne binaire,



(a) a et c du même côté par rapport à f.

(b) a et c de part et d'autre de f.

FIG. 10.4 – *Deux combinaisons différentes des transitions étiquetées* **a**, **f** *et* **c** *pour représenter les chemins* **af**, **cf** *et* **ac**. *Dans les deux cas, le chemin* **afc** *est aussi accepté.* **a** *et* **c** *ayant un rôle symétrique dans le problème, ces schémas prouvent qu'il n'est pas possible de construire un automate acceptant les trois premiers chemins mais pas* **afc**.

et x' sa projection dans l'espace dans lequel les transitions de même étiquette sont mises en commun. Le vecteur poids est défini de la même façon que précédemment.

Proposition 9 (min_2) est équivalent au programme linéaire suivant :

$$\min_{x' \in \mathbb{N}^n} \sum_u P'.x' \text{ avec } \forall c \in \mathcal{L}\ (A'x')_c \geq 1 \text{ et } \forall u\ x'_u \geq 0 \tag{3}$$

Preuve : *Montrons d'abord qu'un x' respectant les contraintes précédentes est équivalent à une projection d'une coupe x. $\forall c \in \mathcal{L}\ (A'x')_c \geq 1 \Leftrightarrow \exists u$ tel que $A'_{c,u} = 1$ et $x'_u = 1 \Rightarrow$ le chemin passe par au moins une transition d'étiquette u, or si $x'_u = 1 \Rightarrow$ toutes les transitions étiquetées u sont dans $\mathcal{C} \Leftrightarrow \mathcal{C}$ est une coupe. La démonstration est la même que pour la proposition précédente, il faut juste voir que l'espace des transitions est projeté sur un espace où les transitions ont toutes une étiquette différente. Il s'agit exactement du problème (min_2) qui est donné par $\min_{\mathcal{C}:cut} \sum_{l \in \mathcal{C}_\mathcal{A}} p(l)$ où $\mathcal{C}_\mathcal{A}$ est la projection d'une coupe dans cet espace.* □

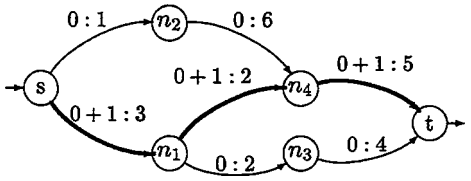
Nous avons donc l'équivalence entre (min_2) et un programme linéaire obtenu de manière directe à partir de l'automate. Nous allons maintenant regarder ce qu'il est advenu des notions de flot propre et de chemin améliorant.

Proposition 10 Le programme linéaire dual de (3) est :

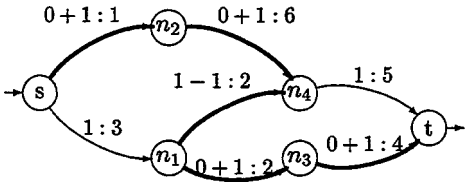
$$\begin{cases} \max \sum_{c \in \mathcal{L}} \pi(c) \qquad \text{avec les contraintes :} \\ \forall u \in T\ :\ \sum_{c \in \mathcal{L}} A_{c,u} \pi(c) \leq p(u) \text{ et } \pi(\mathbf{c}) \geq 0 \\ \boxed{\forall \mathbf{a} \in \mathcal{A} \text{: } \sum_{c \in \mathcal{L}: c \text{ passe par une transition } \mathbf{a}} \pi(c) \leq p(\mathbf{a})} \end{cases} \tag{4}$$

Preuve : *La preuve est directe : on écrit le programme linéaire dual de* (3), *et on simplifie en substituant leurs valeurs à toutes les occurrences de p' ou de A'.* □

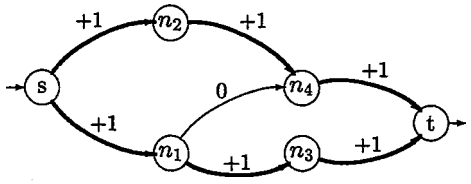
(4) est le problème du flot maximal (2) avec une contrainte supplémentaire : la somme des flots propres passant par les transitions de même étiquette est inférieure à la capacité commune de ces transitions. Nous allons maintenant clarifier la relation entre les chemins améliorants et les flots propres. Cela nous permettra d'introduire d'une manière naturelle cette contrainte supplémentaire dans l'algorithme de calcul du flot maximal.



(a) Première augmentation.



(b) Deuxième augmentation.



(c) Autre décomposition en somme de deux chemins améliorants respectant le sens des transitions de l'automate

FIG. 10.5 – *Les deux chemins améliorants (a) et (b) sont redécomposés dans (c) en la somme des deux flots propres* (s, n_2, n_4, t) *et* (s, n_1, n_3, t). *Le flot qui venait de* (s, n_1) *et qui allait dans* (n_1, n_4, t) *a été «détourné» pour aller dans* (n_1, n_3, t), *alors que le flot qui allait de* n_4 *à* t *vient maintenant de* (s, n_2).

10.3.3 Chemins améliorants et flots propres

Soit $\mathcal{G} = (\mathcal{A}, \mathcal{N}, T, s, t)$ un automate auquel est associée une fonction de poids à chaque transition. $p : T \mapsto I\!N$. Soit f un flot admissible sur $\mathcal{G}$.
Les algorithmes des chemins améliorant construisent un flot sur un automate par des incrémentations successives le long de chemins améliorants (définis page 229) jusqu'à ce qu'aucun chemin améliorant ne puisse plus être trouvé.
Remarquons qu'incrémenter le flot le long d'un chemin améliorant contenant une transition de T inversée consiste en fait à supprimer une précédente augmentation, et à la remplacer par une autre. Considérons par exemple, les automates des figures 10.5(a) et 10.5(b). Ces deux automates montrent deux étapes successives de la construction du flot, nous y avons indiqué en gras les chemins améliorants permettant de passer d'une étape à l'étape suivante. L'automate de la figure 10.5(c) donne une autre décomposition de ces deux augmentations en n'utilisant que des chemins améliorants compatibles avec l'orientation des transitions de l'automate.
Formellement, les chemins améliorants $c_1 \overrightarrow{\mathbf{a}} c_2$ et $c'_1 \overleftarrow{\mathbf{a}} c'_2$ s'additionnent pour donner $c_1 c'_2$ et $c'_1 c_2$.

$$\boxed{c_1 \overrightarrow{\mathbf{a}} c_2 + c'_1 \overleftarrow{\mathbf{a}} c'_2 \equiv c_1 c'_2 + c'_1 c_2}$$

Cette simple transformation prouve sur cet exemple que chaque flot peut être décomposé en une somme de *flots propres*: en effet, en appliquant plusieurs fois la transformation précédente, nous pouvons changer toute somme de chemins améliorants en une somme de chemins améliorants ne contenant que des transitions orientées positivement par rapport à l'automate. La notion de flot propre que nous avions vu apparaître naturellement dans les programmes linéaires (2) et (4) représentent le flot transporté par chaque chemin d'une manière spécifique.

10.3.4 Nouvelle définition d'un flot admissible

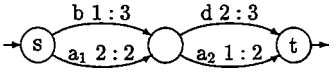


FIG. 10.6 – *Sur cet automate, deux transitions portent l'étiquette a. Nous identifions le nom de la transition au nom de l'étiquette en ajoutant éventuellement des indices.*

Nous avons vu que le programme linéaire associé à (min_2) pouvait être exprimé de manière analogue au problème des flots maximum, en ajoutant néanmoins une contrainte. Nous appelerons un flot admissible respectant cette contrainte un **flux**. La définition du flux impose cette contrainte en plus des contraintes d'admissibilité normales des flots. Dans cette partie nous prouvons que, contrairement à l'admissibilité des flots, on ne peut pas décider si un flot est un flux uniquement à l'aide de la valeur du flot dans chaque transition.
La démonstration se base sur l'exemple de l'automate de la figure 10.6. Pour cet automate et son flux f, les contraintes (4) sont :

$$\begin{cases} (i) & \pi(bd) + \pi(ba_2) \leq p(b) \\ (ii) & \pi(a_1d) + \pi(bd) \leq p(d) \\ (iii) & \pi(a_1d) + \pi(a_1a_2) \leq p(a) \\ (iv) & \pi(ba_2) + \pi(a_1a_2) \leq p(a) \\ (v) & \pi(a_1d) + \pi(ba_2) + \pi(a_1a_2) \leq p(a) \end{cases}$$

(i), (ii), (iii) et (iv) sont les contraintes normales pour un flot, par exemple:
(i) $\Leftrightarrow$ le flot à travers la transition b est inférieur au poids de b: $f(b) \leq p(b)$.
Par contre, (v) ne peut pas être exprimé uniquement à l'aide des flots, mais nécessite les *flots propres*. En effet, dans cet automate, le flot peut être décomposé en chemins améliorants, au moins de deux manières:

- $f = 1 \times a_1a_2 + 1 \times ad + 1 \times bd$, nous avons alors:
  $\pi(a_1a_2) + \pi(a_1d) + \pi(ba_2) = 2 \leq p(a) = 2$, et dans ce cas (v) est bien respecté.

- $f = 1 \times ba_2 + 2 \times a_1d$, nous avons alors:
  $\pi(a_1a_2) + \pi(a_1d) + \pi(ba_2) = 3 > p(a) = 2$, ce qui signifie que (v) n'est pas respectée.

Cela prouve que nous ne pouvons pas décider de l'admissibilité d'un flux défini par (4) en considérant seulement la valeur du flot dans chaque transition. Nous devons avoir mémorisé le flot propre associé à chaque chemin. Nous devons donc stocker les différentes étapes de la construction du flux, et pas seulement les dernières valeurs qu'il prend dans chaque transition. Dans la partie suivante, nous présentons un nouvel algorithme qui construit un flux admissible à l'aide d'augmentations successives le long d'un nouveau type de chemin améliorant.

10.3.5 L'algorithme des lettres pivots

L'idée principale est de décomposer tout chemin améliorant en flots propres. En fait, nous avons seulement besoin de considérer les flots propres des chemins contenant des transitions dont l'étiquette n'est pas unique. En même temps que nous construisons le chemin améliorant, nous vérifions si sa décomposition en flots propres est compatible avec la nouvelle contrainte.
Pour cela, nous définissons d'abord une fonction vérifiant si le début d'un chemin améliorant $c_1' \overleftarrow{\mathbf{t}}$ peut être transformé en une somme de flots propres $1 \times c_1'c_2$ and $1 \times c_1c_?$. Ceci est possible, si, parmi les différents flots passant par la transition $\mathbf{t}$, il en existe une: $c_1 \overrightarrow{\mathbf{t}} c_2$ telle que $c_1'c_2$ est admissible.
À chaque étape de l'algorithme, nous maintenons une decomposition du flux en flots propres. Cette décomposition est stockée dans un tableau[3] $t[1]...t[n_{path}]$. La valeur du flux propre associé au chemin numéroté i est $t[i]$. Pour chaque transition, nous maintenons également la liste $L(t)$ de tous les flots propres passant par la transition. De cette manière, nous pouvons calculer directement la valeur du flot au travers de chaque transition (réelle ou fictive), et donc, nous pouvons vérifier facilement s'il respecte effectivement

3. En fait une table de hachage contient des références aux seuls flots propres construits.

les conditions d'admissibilité du flux.

```
1.  Fonction TraverseTransitionInv(c'1:path,t:transition):boolean
2.  Pour tout flot propre c1 -t-> c2 ∈ L(t)
3.      Supprime(c1 -t-> c2);
4.      Si EstAdmissible(c'1 c2) Alors
5.      Ajoute(c'1 c2);
6.          Si CheminAmeliorant(c1,c'2) Alors
7.              result:=c2;
8.              Retourne true;
9.          Sinon Supprime(c'1 c2);
10.     Ajoute(c1 -t-> c2);
11. Retourne false;
```

Cette fonction utilise les fonctions de bases suivantes:

- *Ajoute*(c : *chemin*), ajoute le flot propre $1 \times c$ au tableau $t[\,]$;
- *Supprime*(c : *chemin*), supprime le flot propre $1 \times c$ au tableau $t[\,]$;
- *EstAdmissible*(c : *chemin*), vérifie si on l'ajout du flot propre $1 \times c$ au tableau $t[\,]$ respecte la nouvelle contrainte d'admissibilité du flux.
- *CheminAugmentation*(c : *chemin*, var c' : *chemin*) : *boolean* est une fonction classique (cf. [1]) de calcul de chemin améliorant, qui, étant donné un début de chemin c, construit le chemin améliorant cc'. Cette fonction fait évidemment appel à *TraverseTransitionInv* quand elle tente de passer par une transition inversée.

Un chemin améliorant $\mathbf{x}$ du réseau à chaque étape de l'algorithme est donnée par: *CheminAmeliorant*$(\epsilon, \mathbf{x})$.

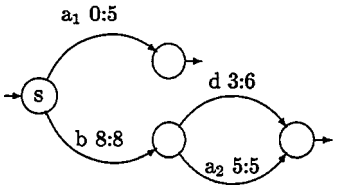
10.3.6 Flux maximal et équilibrage.

L'algorithme précédent construit des chemins améliorants qui vont accroître le flux total dans le réseau. Notre but est de trouver le flux maximal, puisque nous avons montré que pour ce flux, nous aurons obtenu le minimum de (min_2). Nous prouvons maintenant, que l'algorithme précédent s'arrête sur un maximum local, et qu'un nouveau processus d'équilibrage doit alors être utilisé pour faire passer le flux au delà de ce maximum local.

Proposition 11 Le flux à travers une coupe est inférieur à la capacité de la coupe.

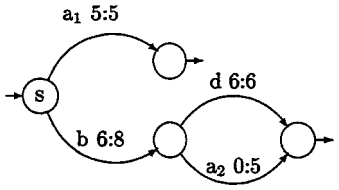
Preuve : *Le flux f_C à travers une coupe C est égal à :*
$\sum_{a \in C_A} \sum_{u \text{ transition étiqueté } a} f(u)$.
La contrainte d'admissibilité du flux nous donne $\sum_{u \text{ transition étiqueté } a} f(u) \leq p(a)$. *Donc* $f_C \leq \sum_{a \in calC_a} p(a) = p(C)$. □

Nous avons dit que le flux avait atteint un maximum local, s'il n'y avait pas de chemin améliorant.



$\mathcal{C} = \{\mathbf{a_1}, \mathbf{b}\}$, $C_A = \{\mathbf{a}, \mathbf{b}\}$.
Les flots propres sont :
$5 \times \mathbf{ba_2} + 3 \times \mathbf{bd}$
Le flux entrant est $= 8$

(a) Réseau présentant un flux localement maximal



$\mathcal{C} = \{\mathbf{a_1}, \mathbf{a_2}, \mathbf{d}\}$, $C_A = \{\mathbf{a}, \mathbf{d}\}$.
Les flots propres sont :
$5 \times \mathbf{a_1} + 6 \times \mathbf{bd}$
Le flux entrant est $= 11$

(b) Le même réseau avec un plus grand flux.

FIG. 10.7 – *Preuve de la non-équivalence entre flux localement maximal, et flux maximal.*

Contrairement au problème du flot maximal, la maximalité locale n'est pas équivalente à la maximalité globale, comme nous pouvons le voir sur les figures 10.7(a) et 10.7(b). Pour le même réseau, le flux entrant dans le premier cas est localement maximal, puisque nous ne pouvons pas trouver de chemin améliorant (respectant la contrainte (4)), mais il n'est pas globalement maximal, puisque le flux d'entrée dans le second cas est supérieur.

La proposition suivante établit que l'existence des maximums locaux est causée par les transitions ayant même étiquette :

Proposition 12 Si une coupe associée à un flux localement maximal est composée uniquement de transitions dont les étiquettes sont uniques, alors le flux est un flux maximal, et la coupe est une coupe minimale.

Preuve : *Comme toutes les transitions de la coupe ont des étiquettes qui ne sont pas utilisées pas d'autres transitions, la capacité de la coupe est exactement la somme des capacités de chaque transition. Donc, la capacité de la coupe est égale à la valeur du flux, ce qui implique que le flux est minimal (Max Flow/Min Cut Theorem).* □

Pour pouvoir dépasser ce maximum local, nous proposons la fonction *EquilibreCA* (figure 10.8) qui sera appelée par la fonction du chemin améliorant, quand celle-ci bloquera sur une transition, pour laquelle le flux ne pourra pas être augmentée (donc pour une transition de la coupe), et dont l'étiquette est utilisée par une autre transition. La fonction essaye de rééquilibrer le flux dans les deux transitions de manière à augmenter le flux global.
Si nous considérons par exemple l'exemple de la figure 10.7(a) : le chemin améliorant bloque sur la transition a_1. Nous appelons donc la fonction EquilibreCA avec les argu-

```
1.  Fonction EquilibreCA(c1:chemin,u = (n1, n2, a)):boolean
2.  Si p(u) = p(a) Alors Retourne false;
3.  Pour toute transition u' = (n1', n2', a)
4.      Si p(u') > 0 Alors
5.          Pour tout flot propre c1' u' c2'  ∈  t[]
6.              Supprime(c1' u' c2');
7.              Si CheminAmeliorant(c1 u,c2) Alors
8.                  Ajoute(c1 u c2);
9.                  Si CheminAmeliorant(c1',c2'') Alors
10.                     Ajoute(c1' c2'');
11.                     Retourne true;
12.                 Sinon Supprime(c1 u c2);
13.             Ajoute(c1' u' c2');
14. Retourne false;
```

FIG. 10.8 – *Fonction EquilibreCA*

ments ($c_1 = \epsilon$,$u = a_1$).

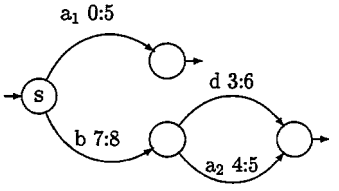
ligne 2. $p(u = a_1) = 0 < 5$

ligne 3. $u' = a_2$

ligne 4. $p(u') = 5 > 0$

ligne 5. Le seul flot propre passant par a_2 est $1 \times \mathbf{ba_2}$. Donc $c_1' = b$, $c_2 = \epsilon$

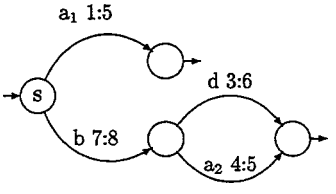
ligne 6. $1 \times \mathbf{ba_2}$ est supprimé de $t[]$, le réseau est donc dans l'état suivant :



Le flux n'est pas localement maximal.
Les flots propres sont :
$5 \times \mathbf{ba_2} + 3 \times \mathbf{bd}$
Le flux entrant est $= 7$

lign 7. Nous recherchons un chemin améliorant passant par $t = a_1$, nous trouvons $1 \times \mathbf{a_1}$.

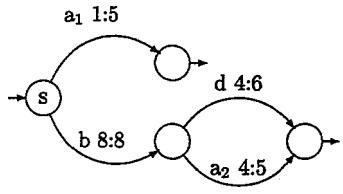
lign 8. $1 \times \mathbf{a_1}$ est ajouté à $t[]$, le réseau est dans l'état suivant :



Le flux n'est pas localement maximal.
Proper flows are:
$5 \times \mathbf{ba_2} + 3 \times \mathbf{bd} + 1 \times \mathbf{a_1}$
Le flux entrant est $= 8$

lign 9. Nous trouvons le chemin améliorant $1 \times \mathbf{bd}$

lign 10. $1 \times \mathbf{bd}$ est ajouté à $t[]$, le réseau est dans l'état suivant :



Le flux est localement maximal.
$\mathcal{C} = \{\mathbf{a_1}, \mathbf{b}\}, \mathcal{C}_{\mathcal{A}} = \{\mathbf{a}, \mathbf{b}\}$

Les flots propres sont :
$4 \times \mathbf{ba_2} + 4 \times \mathbf{bd} + 1 \times \mathbf{a_1}$
Le flux entrant est $= 9$

Le flux entrant a été augmenté d'une unité. Nous voyons que nous pouvons appliquer cette fonction 5 fois de suite, incrémentant à chaque fois le flux entrant d'une unité. Le flux résultant dans le réseau est alors celui de la figure 10.7(b).
Il est clair que la fonction *EquilibreCA* ne peut plus s'appliquer à ce réseau. En effet, le flux trouvé est le maximum global. Nous prouvons que cet algorithme donne bien ce résultat dans le cas général.

Proposition 13 Un équilibrage sur une coupe associée à un flux localement maximal augmente le flux.

Preuve : *Si la fonction d'équilibrage return* **true**, *alors elle doit avoir incrémenté* $t[]$ *deux fois et effectué une seule suppression. Donc le flux global a augmenté.* □

Proposition 14 Si le flux est localement maximal, et si il n'y a pas de possibilité d'équilibrage, alors le flux est globalement maximal.

Preuve : *Supposons que cela ne soit pas vrai : alors nous avons un flux localement maximal* f *sur le réseau et la coupe associée* $\mathcal{C}$. *Soit* f_1 *un flux globalement maximal. Le flux* f *traversant la coupe est strictement inférieur au flux* f_1 *traversant la coupe puisque si le flux entrant augmente, alors le flux traversant n'importe quelle coupe augmente. Puisque le flux* f *traversant les transitions à étiquette unique est maximal, il doit y avoir une transition* $u = (n_1, n_2, \mathbf{a_1})$ *ayant une étiquette partagée dans laquelle le flux a augmenté entre* f *et* f_1. *Si il n'y a pas d'équilibrage possible sur cette transition :*
- Soit, quand nous supprimons un flot propre passant par une transition de même étiquette u' *nous ne pouvons pas trouver ensuite un chemin améliorant passant par* u *(échec à la ligne 7). En supprimant un flot propre traversant* u', *nous supprimons la contrainte caractéristique du flux sur cette étiquette, donc nous sommes dans une situation classique et s'il n'y a pas de chemin améliorant passant par* u, *cela signifie que le flux passant par* u *est maximal, ce qui est contradictoire avec les hypothèses.*
- Soit, quand nous augmentons le flux passant par u, *nous ne pouvons pas trouver un autre chemin améliorant (échec ligne 9), ce qui signfie que dans un «cas classique» (nous pouvons considérer que* u *et* u' *sont indépendants et saturés) le flux traversant la coupe ne peut pas être augmenté, ce qui est contraire au fait qu'il existe un flux avec une valeur supérieure traversant la coupe.*
Donc, il n'est pas possible qu'il n'y ait pas d'équilibrage sur cette transition u. □

Nous avons donc prouvé que quand le flux était localement maximal, nous pouvions appliquer la fonction d'équilibrage et qu'alors le flux augmente. Donc, quand nous ne pouvons plus appliquer la fonction d'équilibrage, le flux est globalement maximal. L'algorithme précédent construit donc le flux maximal.

10.4 Conclusion

Dans le chapitre précédent, nous avons montré l'utilité de calculer un ensemble minimisant les deux équations (min_1) et (min_2). Nous avons montré que le problème (min_1) est exactement le problème du flot maximal, pour lequel nous possédons plusieurs algorithmes de résolution. Le problème de minimalisation (min_2) peut être vu comme le problème (min_1) sur des automates «théoriques». La modification des algorithmes de résolution de (min_1) pour résoudre (min_2) fait ressortir la notion de flot propre non directement visible et les relations entre cette notion et celle de chemin améliorant. L'algorithme de résolution de (min_2) construit des chemins améliorants vérifiant la nouvelle contrainte. Cependant, quand il n'existe plus de chemin améliorant, le flux n'est pas nécessairement maximal. Une fonction d'équilibrage permet d'augmenter le flux global et de dépasser ainsi ce maximum local. Chaque itération de l'algorithme augmente donc le flux jusqu'au maximum global. La solution exacte de nos deux problèmes de minimalisation est donc construite.

Bibliographie

[1] A.V. Aho. *Handbook of Theorical Computer Science - Algorithms and complexity (J. Van Leeuwen)*, chapitre Algorithms for finding patterns in strings, pages 255–300. Elsevier-Amsterdam, 1990.

[2] R.K. Ahuja, J.B. Orlin, et R.E. Tarjan. Improved time bounds for the maximum flow problem. *SIAM Journal on Computing*, 18:939–954, 1989.

[3] M. Araújo, G. Navarro, et N. Ziviani. Large text searching allowing errors. In *Proc. of WSP'97*, pages 2–20. Carleton University Press, 1997.

[4] R.A. Baeza-Yates et H.G. Gonnet. Fast text searching for regular expressions or automaton simulation over tries. *JACM*, 43(6):915–936, October 1997.

[5] M. Crochemore, A. Czumaj, W. Rytter, et al. Fast multi-pattern matching. Rapport Technique, IGM, Université de Marne la Vallée, 1993.

[6] M. Crochemore et W. Rytter. *Text Algorithms*. Oxford University Press, 1994.

[7] E.A. Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. *Soviet Mathematics Doklady*, 11:1277–1280, 1970.

[8] L.R. Ford et D.R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 3:399–404, 1956.

[9] L.R. Ford et D.R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[10] Z. Galil et R. Giancarlo. Data structure and algorithms for approximate string matching. *J. Complexity*, 4:33–72, 1988.

[11] G.H. Gonnet et R. Baesa-Yates. *Handbook of Algorithms and Data Structures*. Addison-Wesley, Workingham,UK, 1991.

[12] J.E. Hopcroft et J.D. Ullman. *Introduction to Automata, Languages and Computations*. Addisson-Wesley, Reading, Mass., 1979.

[13] A.V. Karzanov. Determining the maximal flow in a network by the method preflows. *Soviet Mathematics Doklady*, 15:434–437, 1974.

[14] D.E. Knuth, J.H. Morris Jr., et V.R. Pratt. Fast pattern matching in strings. *SIAM J. Comput.*, 1977.

[15] U. Manber et E.W. Myers. Suffix arrays: a new method for on-line string searches. *SIAM J. Comput.*, 22(5):935–948, 1993.

[16] U. Manber et S. Wu. Glimpse: A tool to search through entire file systems. In *USENIX'94*, pages 23–32, 1994.

[17] E.M. McCreight. A space-economical suffix tree construction algorithm. *J. ACM*, 23(2):262–272, 1976.

[18] A. Moffat et J. Zobel. Compression and fast indexing multi-gigabytes text databases. *Australian Computer Journal*, 26(1):1–9, 1994.

[19] D. Perrin. *Handbook of Theorical Computer Science - Formal Models and Semantics (J. Van Leeuwen)*, chapitre Algorithms for finding patterns in strings, pages 1–57. Elsevier-Amsterdam, 1990.

[20] J.B. Orlin R.K. Ahuja, R.K. Magnanti. *Network Flows – Theory, Algorithms, and Applications.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[21] J. Senellart. Indexation de grands corpus en tenant compte de données linguistiques. Rapport de Stage d'Option, Ecole Polytechnique, 7 1995.

[22] J. Senellart. Locating noun phrases with finite state transducers. In *Proc. of ACL-COLING'98*, 1998.

[23] E. Ukkonen. Constructing suffix trees on-line in linear time. In *IFIP'92*, pages 484–492, 1992.

[24] P. Weiner. Linear pattern matching algorithms. In *Proc. 14th IEEE Annual Symposium on Switching and Automata Theory*, pages 1–11, 1973.

[25] H. Williams et J. Zobel. Indexing nucleotide databases for fast query evaluation. In *Proceedings of Advances in Database Technology (EDBT'96)*, Avignon, France, march 1996.

[26] W.L. Winston. *Operations Research: Applications and Algorithms.* PWS-Kent, 1991.

[27] Witten, A. Moffat, et Bell Van Nostrand. *Managing Gigabytes.* Reinhold, 1994.

[28] S. Wu et U. Manber. Fast text searching allowing errors. *Communications of the ACM*, 35(10):83–91, 1992.

Quatrième partie

Expressions figées et Statistique

Chapitre 11

Expressions figées et Statistique

11.1 Introduction

Les outils que nous avons présentés nous permettent de reconnaître dans un texte, toutes les séquences figées décrites à ce jour. Nous avons effectué un essai de reconnaissance sur un texte d'un million de mots et nous avons effectué la projection de l'automate de ce texte. Le résultat présenté dans [7] est surprenant. Nous le résumons dans le tableau 11.1 qui fait le décompte des réductions effectuées automatiquement sur ce texte de référence. Nous pouvons y voir que 40 % des mots du texte initial sont concernés par une réduction d'expressions figées en unités sémantiques. En fait, dans [7], nous prouvons que ce pourcentage est porté à 50 % par des réductions manuelles qui depuis, ont été en partie intégrées à la reconnaissance automatique. Cela signifie qu'un mot sur deux dans un texte fait partie d'une expression figée, et donc qu'*un mot sur deux n'est pas pertinent syntaxiquement, et donc à fortiori sémantiquement*[1]. Donc le phénomène des «constructions figées» est très répandu. De plus, si un mot sur deux ne correspond pas à une unité syntaxique minimale[2], il devient alors difficile d'effectuer la moindre opération non strictement linguistique sur un texte brut.

Cette dernière affirmation est assez intuitive et semble relever du bon sens. Nous ne demanderions pas à un humain ne connaissant pas une langue donnée de faire un résumé d'un texte dans cette langue, ni même d'en extraire des mots clefs. Cependant, à l'heure actuelle, la majorité des systèmes d'extraction automatique d'information dans des textes sont des systèmes statistiques, dont la composante linguistique est extrêmement réduite. Des approches sémantiques commencent à voir le jour avec Wordnet [12] par exemple, mais les systèmes utilisant de telles données sont encore essentiellement basés sur des outils statistiques.

Que la notion d'information soit intimement liée à la notion de probabilité n'a rien de surprenant, comme nous le verrons en rappelant certaines bases de la théorie de l'infor-

1. En effet, si un mot ne s'intègre pas dans un schéma syntaxique régulier, on ne peut pas lui associer de schéma régulier de calcul de sens (pour autant que un tel schéma existe). Il faudra une procédure de calcul de sens propre.

2. Il est déjà bien connu qu'environ un mot sur trois dans un texte n'est porteur d'aucun sens (mot vide), les mots qui interviennent dans nos regroupements ne sont pas ces mots là puisque il s'agit des articles simples et de la préposition *de* qui n'est rattachée à une unité figée que si elle sert de lien entre deux mots pleins. La suppression des mots vides va donc réduire encore d'autant le pourcentage de mots sémantiquement pertinents.

Bilan de la réduction		Nombre de réductions effectuées	Mots affectés : Nombre	Mots affectés : %	Nombre de mots par forme réduite	Nombre de mots intégrés aux composés	Nombre de mots restants
Négation : ne ...	rien/personne	341	933	0,09 %	2,7	542	999 408
	pas	5 613	11 949	1,19 %	2,12	6 336	993 072
Adverbes...	de dates	8 490	79 114	7,96 %	2,14	42 149	975 221
	de durées	2 615					
	de fréquences	371					
	de lieu	2 950					
	:(être) Prép C	1 500					
	figés	21 039					
Temps composés		14 151	29 009	2,90 %	2,05	14 858	960 363
Verbes aspectuels/causatifs/modaux		6 584	15 802	1,62 %	2,4	9 218	951 145
Structures verbales figées		14 447	43 207	4,55 %	2,99	28 760	922 385
Noms composés (DELAC)		21 944	213 800	23,1 %	3,35	150 061	772 324
Noms propres (géographiques)		3 706					
Noms propres (institutions)		2 554					
Titres (portant sur des personnes)		9 989					
Noms composés (grammaires locales)		1 637					
Déterminants composés		11 359					
Adjectifs composés		650					
Prépositions composées		9 499					
Conjonctions composées		2 401					
Toutes réductions :		138 813	393 814	**39,4 %**	—	—	772,324

TAB. 11.1 – *Décompte des unités figées dans un texte d'un million de mots. 40% des mots du texte initial ont été modifiés par une transformation.*

mation. Mais que la contribution d'analyses linguistiques fines[3] ne soit ni nécessaire, ni utile, ni même utilisable est plus paradoxal. Cette attitude semble être assez répandue, comme nous pouvons nous en rendre compte par plusieurs communications dont les titres posent eux-même le problème: «*Beyond N-Grams: Can Linguistic Sophistication Improve Language Modeling?*» [4], «*Do we need linguistics, when we have statistics? A comparative analysis of the contributions of linguistic cues to a statistical word grouping system.*» [8]. Quoique ces communications montrent qu'il pourrait être intéressant d'apporter des outils linguistiques aucune méthode concrète n'est proposée, et cette conclusion est présentée comme une manière de dépasser une limite statistique déjà atteinte. Remarquons que la présentation de la problèmatique de cette manière provocante «*A-t-on besoin de linguistique, quand on a la statistique?*» est relativement récente. La statistique textuelle a dû s'imposer d'un point de vue méthodologique. C'est avec l'arrivée de corpus de plus en plus grand que cette démarche se popularise: pour pouvoir extraire de l'information d'une quantité de textes non humainement lisible, il faut en extraire de l'information pertinente ou des tendances générales, ce pourquoi les statistiques sont faites. Une caractéristique de l'analyse statistique d'un texte est sa capacité de donner des résultats concrets, ce qui est facile, tant l'information présente dans un texte abonde sous toutes les formes. Au problème: extraire d'un texte un mot-clef, quelle que soit la méthode statistique choisie (fréquence, spécificité, etc.), une réponse sera donnée (cette réponse pourra même souvent reliée à un indicateur de pertinence). D'un autre côté, aucune réponse à ce problème ne peut actuellement être apportée par un système purement linguistique.
Nous considérons ici uniquement les systèmes d'extraction d'information[4]. Nous ne considérerons donc pas les systèmes automatiques d'étiquetages statistique, dont nous avons déjà discutés dans 2. En effet, ceux-ci conduisent à une reformulation non naturelle de textes et cette reformulation n'extrait pas clairement de l'information du texte (donc ne peut pas être considéré comme un système d'extraction d'information). L'idée sous-jacente à ces systèmes est qu'il est possible, en partant de presque rien, de construire une grammaire complète du langage, et donc d'obtenir par des outils statistiques la description du langage, ce qui justifierait le fait qu'une analyse linguistique manuelle ne soit pas utile. Le résultat présenté qui montre qu'un mot sur deux ne suit pas une règle linguistique régulière montre les limites de tels systèmes. Enfin, ces systèmes de catégorisation automatique ont un but clair, la lemmatisation de formes. Nous ne discuterons pas non plus ici de l'influence que peut avoir la lemmatisation sur une analyse statistique.

Notre but est de montrer qu'il existe de l'information (statistiquement utilisable) dans la reconnaissance des expressions figées, et que cette information n'est pas extractible par des méthodes statistiques.
Pour comprendre l'interaction possible entre des opérations statistiques sur des textes et l'analyse linguistique, nous commencerons par décrire quelques traitements statistiques de texte qui extraient de l'information d'un texte. Nous verrons ensuite que l'information contenue dans un message est fortement liée à la surprise que chaque sous-unité de la séquence apporte. Pour que cette «surprise» soit analysable, il faut qu'il y ait un modèle de référence implicite: ce modèle de référence pouvant être la grammaire elle-même.

3. C'est-à-dire dépassant des réductions à des schémas simplistes du genre: une phrase est de la forme <*GN*> *V* <*GN*>, et un groupe nominal est de la forme <*DET*> <*N*> <*ADJ*>.

4. Le meilleur exemple étant l'extraction de mots clefs pour une indexation de corpus électroniques.

Nous baserons cette étude sur quelques exemples concrets calculés sur des extraits de corpus de quotidien, mais nous ne chercherons pas à prouver la validité de nos résultats sur des textes. Il est en effet trop facile d'obtenir tout résultat souhaité sur des textes bien choisis. Nous donnerons plutôt des justifications théoriques.

11.2 Statistique textuelle, outils et utilisation.

Certaines données numériques sur des textes sont obtenues en considérant d'une part le vocabulaire utilisé, sa répartition sur le texte et d'autre part, les liens statistiques pouvant exister entre la présence de deux formes. Ces liens sont définis par rapport à une distribution de référence des formes sur les textes, cette distribution est souvent aléatoire.

11.2.1 Les outils : du texte aux valeurs numériques

Calculs directs sur le vocabulaire

Soit $\mathcal{T}$ un texte, on note $\mathcal{A} = \{a_1, ..., a_n\}$ son vocabulaire (l'ensemble des formes différentes). La première mesure qu'il est possible d'effectuer sur le texte, est la taille du vocabulaire, soit $|\mathcal{A}|$. Cette mesure est aussi appelée *richesse du vocabulaire.*
En prenant des textes de même longueur, il est donc possible de mesurer la richesse relative du vocabulaire de différents textes.
Pour un texte donné, on peut aussi tracer une courbe donnant la taille du vocabulaire en fonction de la taille du texte (figure 11.1). Cette courbe représente l'accroissement du vocabulaire rencontré en fonction de la taille du texte. Cette courbe permet, par exemple, d'interpoler la taille d'un texte qu'il faudra parcourir rencontrer tous les formes d'un lexique donné (voir [12]).

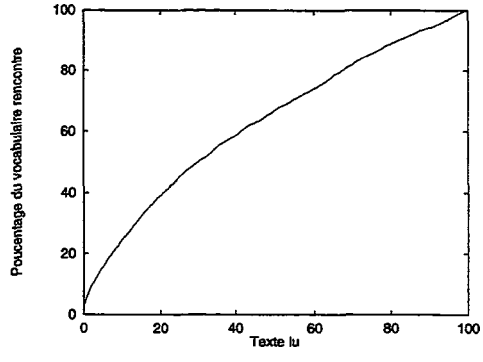


FIG. 11.1 – *Evolution du vocabulaire en fonction de la taille du texte*

Une deuxième mesure consiste à étudier la distribution en fréquence de chaque a_i : on associe à chacune des formes différentes a_i d'un texte le nombre d'occurrences f_i dans le texte. Si on ordonne les a_i suivant leur f_i décroissant et qu'on note r_i leur rang r_i, on obtient une loi *fréquence/rang* dont la forme caractéristique est $f(x) = \frac{A}{x^\alpha}$ avec $\alpha \approx 1$ et qui s'appelle la loi de Zipf (figure 11.2). Nous discutons de cette loi plus en détail en

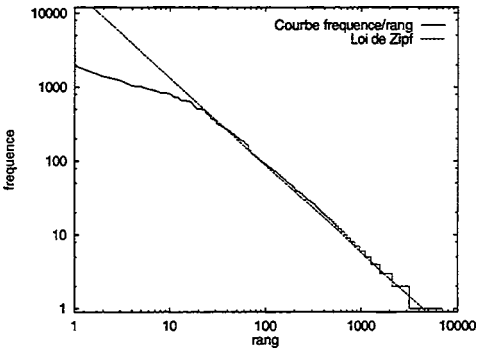


FIG. 11.2 – *Courbe fréquence/rang en échelle doublement logarithmique (diagramme de Paretto)*

annexe B. Le fait qu'elle s'observe dans de nombreuses distributions naturelles ou non (par exemple pour la distribution des tailles de villes, mais aussi pour la distribution en fréquence des formes de textes qui seraient tapés par des singes sur une machine à écrire) incite à une grande prudence dans l'utilisation qu'on peut en faire. Une manière d'exprimer cette loi est qu'il existe peu de formes très fréquentes, et beaucoup de formes très peu fréquentes. Sous forme cumulative (figure 11.3), cette représentation permet de voir que 1 mot sur 2 dans notre texte exemple appartient au 139 formes les plus fréquentes. On voit aussi que 1 forme sur 3 appartient aux 25 formes les plus fréquentes.

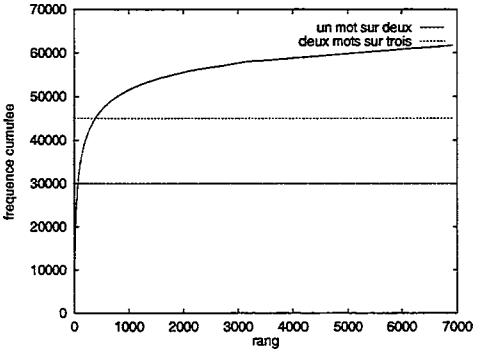


FIG. 11.3 – *Courbe fréquence/rang cumulée sur la fréquence*

Enfin, on peut étudier la répartition du vocabulaire sur le texte qui permet d'obtenir des informations assez fines. Par exemple, en partitionnant le texte en zones (paragraphes, articles), il est possible de faire apparaître le vocabulaire propre à chaque partie et le vocabulaire commun. La comparaison du vocabulaire entre les différentes zones, permet d'associer à chaque mot dans chaque zone une valeur appelée *spécificité* qui met en valeur une présence anormalement importante ou faible du mot dans la zone considérée. La réprésentation de la répartition du vocabulaire dans une partition du texte est appelée

[9] *tableau lexical entier.* Dans un texte homogène non divisible, l'étude de la loi de distribution des mots pleins permet de faire ressortir des contextes différents [17]. A cette répartition il est possible d'associer deux notions de «durée de vie» du vocabulaire et de couverture, permettant de faire ressortir des formes très localisées (ces deux notions sont définies dans la dernière version de INTEX©).

Relation entre différents items, cooccurrences

Les diverses mesures précédentes sont faites sur chaque mot et des méthodes d'analyse ultérieures permettent de comparer les comportements de chacun des mots. Les mesures que nous présentons maintenant se basent sur la présence simultanée de plusieurs formes. Ces mesures sont de deux types différents: les segments (ou quasi-segments) répétés et les relations de cooccurrence.
Un *segment répété* est une séquence composée de plusieurs formes (*multi-words unit*) dont on étudie la répétition. Nous avons représenté dans le tableau 11.2 des extraits des listes de formes simples, des séquences de deux formes et des séquences de trois formes. Le nombre de séquences augmente avec le nombre de formes, mais par contre le nombre de formes suffisamment fréquentes (5 par exemple) pour valider la répétition diminue.

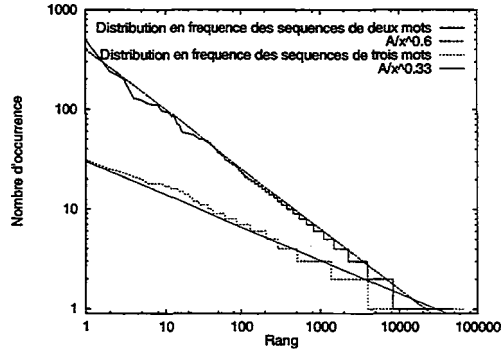


FIG. 11.4 – *Courbe fréquence/rang pour les segments répétés. On remarque que la courbe est proche d'une loi de la forme* $A.x^{-\alpha}$, *mais ici* $\alpha \not\approx 1$.

Des extensions de la notion de *segment* ont aussi été proposées avec les quasi-segments qui représentent des séquences non connexes [1].
Le deuxième moyen de faire des mesures sur des couples de formes (ou de segments répétés) est d'étudier leurs corrélations statistiques. On évalue l'influence de la présence d'une ou plusieurs formes sur la distribution d'une autre forme.
Plusieurs méthodes/formules ont été proposées, la définition d'information mutuelle proposés par CHURCH et HANKS [3]: $I(x,y) = \log \frac{P(x,y)}{P(x)P(y)}$.
Nous avons proposé dans [17] une méthode permettant de distinguer des effets à courte distance et des effets à plus longue distance. Cela permet en particulier de distinguer les effets de contexte, et les effets grammaticaux.
Sur notre corpus de référence, nous avons calculé les informations mutuelles entre des noms comme *foot*, *Chirac*, et *Moscou* et tous les autres «mots pleins» suffisamment

12076 formes simples		*47420* séquences de deux formes		*66274* séquences de trois formes	
2002 de fréquence ≥ 5		*1495* de fréquence ≥ 5		*289* de fréquence ≥ 5	
4023	de	*598*	de la	*42*	n est pas
1948	la	*500*	de l	*31*	il y a
1492	l	*242*	à la	*25*	dans le cadre
1488	à	*201*	à l	*23*	et de la
1395	le	*129*	et de	*21*	n a pas
1339	et	*124*	c est	*20*	de l État
1151	d	*114*	d un	*18*	infanterie de marine
1143	les	*111*	dans le	*18*	et de l
1078	des	*111*	d une	*18*	de Jacques Chirac
873	en	*97*	qu il	*18*	d infanterie de
869	du	*95*	C est	*17*	millions de francs
801	un	*94*	a été	*17*	de l école
755	est	*85*	dans les	*16*	qu il a
696	a	*84*	dans la	*16*	de l association
680	une	*71*	n est	*16*	Il y a
558	pour	*69*	sur le	*15*	que l on
505	qui	*60*	y a	*15*	le cadre de
481	dans	*58*	sur la	*15*	au cours de
458	au	*58*	et le	*14*	à 18 h
146	Jean	*50*	Jacques Chirac	*9*	régiment du génie
105	Saint	*43*	Lionel Jospin	*9*	la première fois
86	France	*35*	aujourd hui	*9*	au premier tour
76	maire	*21*	premier tour	*9*	appelle à voter
75	jeunes	*20*	mercredi soir	*8*	tout le monde
74	Jacques	*20*	après midi	*8*	maison de quartier
71	enfants	*19*	en place	*8*	chiffre d affaires
70	Chirac	*19*	au cours	*8*	Hervé de Charette
69	Jospin	*17*	parce que	*7*	sur le thème
67	travail	*16*	à Nantes	*7*	parents d élèves
67	premier	*16*	deux ans	*7*	mois de prison
58	Nantes	*16*	conseil municipal	*7*	communauté de communes
55	président	*16*	Loire Atlantique	*7*	comité de soutien
55	liste	*15*	à partir	*7*	ans de prison
53	élèves	*15*	rendez vous	*7*	Maine et Loire
53	Pierre	*14*	Jean Claude	*7*	Il s agit
52	notre	*13*	ce soir	*6*	sur le terrain
51	association	*13*	Jean Pierre	*6*	régiment d artillerie
50	ville	*13*	20 h	*6*	prison avec sursis
49	école	*12*	second tour	*6*	mise en place

TAB. 11.2 – *Listes des formes par fréquence décroissante. Dans la moitié inférieure du tableau, nous avons conservé uniquement les "mots pleins".*

fréquents du lexique. Nous avons ensuite trié ces informations mutuelles, et nous représentons dans le tableau 11.3 les formes associées aux informations mutuelles les plus fortes.

11.2.2 L'utilisation des outils : des valeurs numériques au sens

Les différentes mesures que nous venons de présenter conduisent à l'accumulation de quantités énormes de données qu'il faut utiliser.
Nous citons ici, sans entrer dans aucun détail technique, différentes méthodes [16, 9] qui permettent de «synthétiser» ces données. Supposons que l'on veuille comparer n formes différentes du texte. Nous venons de voir que nous disposons de la comparaison de la répartition de ces différentes formes sur le texte, ou de relations de corrélations.
En partant d'un *tableau lexical entier*, ou de relations de corrélations entre ces formes et des formes de référence, il est possible de construire une distance permettant de comparer les formes entre elles. Nous obtenons alors une matrice de taille $n \times n$ représentant les $n(n-1)$ distances possibles. L'espace géométrique dans lequel il existe une solution à ce problème de distance dans le cas général, est un espace de dimension $n-1$ dont la visualisation n'est pas possible! Plusieurs méthodes permettent de dégager une information visuelle :

- Classification automatique des formes (*clusters*). Des classes de formes sont établies en vue de dégager des comportements homogènes [15], [14].

- Analyse multi-dimensionnelle permettant d'obtenir une projection de cet espace sur un plan principal. Par exemple, l'analyse en composantes principales ou l'analyse factorielle des correspondances [2], [6].

- Construction de graphes lexicaux [20] à partir des relations de corrélations. Deux formes sont reliées si leur coefficient de corrélation est supérieur à une certaine valeur. Cette méthode permet d'obtenir des graphes dans lesquels deux formes sont reliées s'il y existe «un lien quantitatif[5] entre elles».

D'autres méthodes, plus simples, dégagent directement du vocabulaire des mots-clefs potentiels (formes ou segments répétés), sur des bases de fréquence élevée et de comportement non grammatical.
Toutes ces méthodes font sortir du texte des informations et des représentations visuelles qui permettent à un lecteur humain d'obtenir des informations fines sur le texte étudié. Dans la partie suivante nous décrivons cette notion d'information.

11.3 Nature de l'information

11.3.1 Information et entropie

Dans un processus de communication linéaire (typiquement un texte), la notion d'information, au sens de la théorie de SHANNON et WEAVER [19], est liée à la surprise de la réception de chaque nouvelle unité.

5. On peut espérer qu'il s'agit aussi d'un lien sémantique.

foot		*Chirac*		*Russe*	
Jean	0.042	politiques	0.015	production	0.033
premier	0.088	volonté	0.015	vivre	0.033
place	0.093	économique	0.015	Allemands	0.035
Nantes	0.106	loi	0.016	anniversaire	0.035
association	0.109	lundi	0.016	Dora	0.035
liste	0.111	maintenant	0.016	deuxième	0.035
tour	0.121	nature	0.016	moyens	0.035
mercredi	0.123	nouveaux	0.016	prisonniers	0.035
heures	0.140	ouvert	0.016	retraité	0.035
occasion	0.151	parler	0.016	Mai	0.038
petit	0.160	propositions	0.016	42	0.039
Henri	0.164	respect	0.016	Dominique	0.039
équipe	0.174	sociétés	0.016	Union	0.039
hommes	0.175	solidarité	0.016	février	0.039
ancien	0.186	solution	0.016	oeuvres	0.039
Paul	0.198	série	0.016	système	0.039
ailleurs	0.215	fonction	0.016	peuple	0.040
conseiller	0.228	nombreuses	0.016	technique	0.041
80	0.400	Brahim	0.016	art	0.043
Cela	0.430	Dimanche	0.016	1991	0.043
Quant	0.430	Gérard	0.016	avis	0.043
ouverture	0.430	René	0.016	corps	0.043
deuxième	0.466	avis	0.016	derniers	0.043
jeu	0.466	choisi	0.016	espoir	0.043
quinzaine	0.466	derniers	0.016	force	0.043
retraité	0.466	morts	0.016	grands	0.043
sortie	0.466	veulent	0.016	loi	0.043
22	0.509	vite	0.016	lundi	0.043
Dominique	0.509	Éducation	0.016	maintenant	0.043
Thierry	0.509	élections	0.016	série	0.043
laquelle	0.560	lancé	0.017	élections	0.043
matériel	0.560	sortant	0.017	sortant	0.045
élections	0.560	abstention	0.018	morts	0.047
sortant	0.594	Charette	0.018	Russie	0.049

TAB. 11.3 – *Informations mutuelles entre différentes formes*

Supposons qu'une personne A envoie des messages indépendants et d'une manière aléatoire à une personne B. Par définition, l'information contenu dans le message est mesurée par l'entropie du message :

$$H = -P(\mathcal{M}) \log_2 P(\mathcal{M})$$

où $P(\mathcal{M})$ est la probabilité pour A d'émettre ce message $\mathcal{M}$. SHANON prouve que cette entropie est exactement le nombre de «bits» minimum qu'il faut à A pour envoyer chacun de ses messages de manière à minimiser la taille totale de l'ensemble des messages émis[6]. La quantité d'information contenu dans un message est donc reflétée par l'entropie : plus un message sera surprenant, plus il prendra de place à coder, et plus il contiendra d'information. Cette notion de surprise peut directement être associée à la notion opposée qui est la notion de redondance[7]. Plus les liens entre les différentes parties du message sont forts et moins il y a d'information.
Le codage correspondant à ce nombre minimum de bits correspondent à une version épurée du message ne contenant plus que l'*information pure*. Nous aurons besoin, par la suite, du lemme élémentaire suivant :

Lemme 5 Dans ce message codé : la succession des différents bits de données est complètement aléatoire.

Preuve : *Considérons la succession de bits $b_0 ... b_n$ correspondant au code du message $\mathcal{M}$, et supposons que la distribution des différents b_i est complètement aléatoire. Pour tout i, $P(b_i = 0) = \frac{1}{2}$. L'entropie par bit est donc :*

$$H_i = -P(b_i = 0) \log P(b_i = 0) + (-P(b_i = 1) \log P(b_i = 1)) = 1$$

L'entropie du code complet est donc :

$$H' = \sum H_i = n = H$$

Il est facile de voir que si la distribution n'était pas aléatoire alors l'entropie serait inférieure, donc le nombre de bits nécessaire serait inférieur, ce qui serait contradictoire avec la définition de l'entropie d'un message. □

Extraire l'information contenue dans un texte et comprimer sa taille sont donc deux activités complémentaires indissociables. De premiers essais de compression de texte à l'aide d'outils statistiques [11] montrent déjà que l'utilisation d'outils linguistiques permet de dépasser la limite quasi-figée de la compression de texte avec des outils purement statistiques.

11.3.2 Entropie et langage naturel

Il est assez évident que cette version très informatique de la communication entre deux individus n'est pas intéressante sous sa forme binaire. En fait, on peut la reformuler en effectuant un «changement de base». Si on considère que l'unité minimale du «code» est le mot du langage naturel (issu d'un lexique $\mathcal{A}$ (et non plus le bit), tout ce que nous venons d'énoncer reste valable en base $||\mathcal{A}||$. *Grosso-modo*, cela correspond à coder le

6. Nous supposons donc que la loi P est connue à l'avance par $\mathcal{A}$ et $\mathcal{B}$.
7. La redondance n'apporte pas d'information, mais permet de faire ressortir statistiquement une information.

message initial en langage naturel.

On peut comprendre intuitivement ce processus en comparant deux textes différents de taille analogue : une fable de La Fontaine, et un article de quotidien. Pour le premier texte, le nom de la fable peut suffire à la coder complètement (ou éventuellement les premiers et les derniers mots *Maître Corbeau sur un arbre perché...qu'on ne l'y prendroit plus.*). Toute personne connaissant La Fontaine pourra reconstituer la fable complète, ou trouver comment la reconstituer. La surprise de trouver les mots de la fable entre *Corbeau* et *prendroit plus* est donc nulle. Dans le deuxième cas, il n'y a pas de possibilité de faire la même chose. Le titre de l'article ne sera pas suffisant, ni même les premiers mots. De nouveaux éléments surprenants peuvent apparaître tout du long de l'article. Il y a donc «plus d'information» dans ce texte que dans le précédent.

> La notion d'information contenue dans un texte est donc liée à la connaissance du lecteur.

Un processus d'extraction automatique d'information, est une boîte noire prenant un texte en entrée et produisant des mots ou des phrases en sortie. Ce résultat sera ou ne sera pas de l'information proprement dite pour l'utilisateur suivant ses connaissances propres. Dans notre exemple précédent, si un processus d'extraction automatique fait apparaître un lien statistique entre le corbeau et le renard dans le premier texte, il ne s'agira pas d'information. Par contre, une même relation observée dans le second texte aurait un statut d'information.
En nous plaçant dans le cadre d'une extraction automatique d'information de textes homogènes (style journalistique), nous pouvons supposer que la forme n'a pas d'importance pour l'utilisateur (que ce soit sa langue maternelle ou une langue étrangère, un tel utilisateur cherchera à appréhender le contenu informationnel du texte). Nous voulons montrer que pour séparer la forme du contenu, le processus devra lui-même avoir connaissance de la grammaire utilisée : c'est-à-dire qu'il ne pourra pas trouver lui-même dans le texte, l'information suffisante à extraire les deux composantes.

Pour prendre un autre exemple pratique, considérons la règle d'accord *déterminant/nom*. Pour quelqu'un ne connaissant pas le français, la présence des deux *s* dans le groupe nominal *les mots* est une double surprise indépendante. Pour quelqu'un connaissant la règle d'accord, seul le premier *s* est une surprise, le second n'apporte aucune information et ne doit pas être non plus comptée comme une redondance. On peut donc supprimer ce deuxième *s*. D'un point de vue entropique, la connaissance de la règle d'accord diminue donc l'entropie, et nous rapproche d'une version plus pure en information du texte.

Dire que nous voulons séparer la partie grammaticale de la partie sémantique[8] dans un texte, c'est exactement dire que, en connaissant la grammaire, nous voulons comprimer le texte de telle manière que les différents objets restant soient totalement libres de la contrainte linguistique (par définition du message comprimé, voir lemme 5).
L'idéal serait donc de supprimer complétement la forme linguistique (la représentation par mots) pour atteindre une représentation indépendante, sur laquelle nous pourrions appliquer les outils statistiques décrits. Cet objectif est un objectif idéal et utopique

8. Cette séparation est très sommaire, il faudrait aussi tenir compte d'une partie stylistique intermédiaire.

car l'interaction entre grammaire et sémantique n'est pas nulle. Cependant, le modèle de l'entropie montre que toute source d'information linguistique extérieure à un texte, réduit la part de surprise dans le texte, et donc nous permet de mieux s'approcher du sens. Dans la partie suivante, nous essayons de mettre en évidence l'information contenue dans différents phénomènes linguistiques, et nous montrons quelle pourrait être la réduction de ces phénomènes.

11.4 La composante linguistique

Commençons par mettre en évidence l'écart réél entre le contenu sémantique recherché et sa représentation par phrases.

11.4.1 Le décompte des unités d'information

Supposons que nous puissons comprimer tout message en langage naturel, en une ou plusieurs phrases, elles-mêmes en langage naturel. D'après le lemme 5 : la distribution des différents objets dans cette phrase doit être complètement aléatoire. Cela signifie qu'il ne doit y avoir aucun lien entre les différents objets.

Considérons donc des phrases et essayons de compter le nombre d'unités d'informations présentes dans cette phrase et leurs liens possibles.

(77) *Max voit Léa.*
(78) *Jean lit un roman.*
(79) *J'ai dit que Paul souhaite que tu viennes demain.*
(80) *Jean prend le taureau par les cornes.*
(81) *Jean prend ses responsabilités.*
(82) *Jean a pris une bonne décision.*

- (77) et (78) : dans la phrase (77), chacun des trois mots semble être complètement indépendant ; dans la phrase (78) (sans tenir compte pour le moment du déterminant), nous avons aussi trois objets. Mais le lien entre *lire* et *roman* est assez fort : la surprise de trouver *roman* comme complèment d'objet du verbe *lire* est assez faible. En fait, cela peut s'exprimer par le fait que le verbe *lire* opère une sélection sur la distribution de son complèment d'objet direct : il doit s'agir d'un nom concret «lisible». Cette sélection fait que le choix est relativement restreint, et que *roman* en est un des rares éléments. En fait, cette sélection existe aussi dans la première phrase, mais d'une manière beaucoup moins stricte. Le complément d'objet direct de *voir* est un concret quelconque. Il existe une sélection sur les sujets des deux verbes : dans la première phrase le sujet doit être animé, dans la seconde le sujet doit être humain.
  D'un point de vue stockage de l'information, cela se représente de la manière suivante :

$$\begin{aligned} P(\textit{Max voit Léa}) = \; & P(\textit{Phrase élémentaire}{=}N_0 \textit{ voir } N_1) \\ & \times P(\textit{sujet animé}{=}\textit{Max}) \\ & \times P(\textit{Objet concret}{=}\textit{Léa}) \end{aligned}$$

  Les quantités d'information contenue dans les deux groupes nominaux *Max* et *Léa* sont donc différentes.

Remarquons que parler de sélection des arguments par le verbe est traditionnellement admis. Cette sélection ne respecte pas l'ordre des mots de la phrase. On pourrait aussi dire qu'un sujet humain sélectionne un verbe humain, ou qu'un objet direct «lisible» sélectionne un verbe de lecture. Cela obligerait une redéfinition des phrases élémentaires basées autour des noms qui ne serait pas naturelle. Cependant, cela ne changerait pas l'entropie de la phrase.
Si nous voulons maintenant tenir compte du déterminant dans la deuxième phrase, l'information qu'il apporte n'est pas vraiment le nombre, puisque la phrase *Max lit deux livres* est étrange sans l'ajout du complément *à la fois*. Donc l'information portée par *un* est seulement l'indétermination.

- (79) : cette phrase est complexe car elle comporte deux complétives. On pourrait décomposer cette phrase en trois phrases élémentaires. Cependant on peut se poser la question, quel est le verbe principal de la phrase, ou d'une manière dérivée, où est l'information principale dans cette phrase?
  Le premier verbe de la phrase est un performatif (*je dis que*), son contenu informationnel est donc très limité. En fait, le supprimer ne modifie pas le sens de la phrase.
  Le second verbe : *souhaiter* pourrait être vu comme une simple modalité sur le verbe *venir*. Considérons en effet les trois phrases suivantes :
  (?)Paul souhaite qu'il vienne demain.
  Paul souhaite venir demain.
  Paul viendrais bien demain.
  Elles sont toutes les trois synonymes, la première est maladroite mais correspond au schéma de notre exemple précédent. La deuxième réduit la complétive à une infinitive et montre distinctement la présence de *souhaiter* comme auxiliaire du verbe *venir*. La dernière phrase incorpore le souhait dans l'utilisation du conditionnel. Un processus de lemmatisation purement grammatical perdrait donc cette information de modalité du verbe *venir*.
  Même si l'équivalent avec le conditionnel n'existe pas dans notre exemple, il est légitime de dire que *venir* est le verbe principal. En tout cas, au niveau d'approximation actuel de l'extraction d'information, la phrase élémentaire porteuse de sens est la troisième *je venir demain*. Cette extraction est contraire à la décomposition grammaticale en verbe principal et verbes secondaires.

- (80) : dans cette phrase, nous ne pouvons trouver que deux unités d'information, le sujet *je*, et tout le reste *prend le taureau par les cornes* (plus la modification du temps que nous négligeons pour le moment).

- (81) : dans l'expression *prendre ses responsabilités*, la partie variable *ses* ne comporte pas d'information puisque la forme de l'adjectif possessif se déduit automatiquement du sujet. Il n'y a donc encore que deux unités dans cette phrase.

- (82) : ici, la séquence canonique est *prendre une décision* `C1IPN`. Il y a cependant une modification de décision non prédictible. Il y a donc trois unités d'information : le choix de la phrase élémentaire N_0 *prendre décision*, le sujet humain N_0, et le modificateur de *décision*, par exemple les modifications par des adjectifs (*bonne*, *mauvaise*, (?)*joyeuse*, etc.) ou par des relatives : *qui s'impose*, etc.

Pour évaluer la quantité d'information contenue dans chaque élément d'une phrase, il faut que cette phrase soit syntaxiquement analysée et que cette analyse extrait la phrase élémentaire sous-jacente, ainsi que les propriétés distributionnelles de ses arguments. Cette remarque, plutôt théorique, est-elle utilisable directement?

Dans la phrase *Jean lit un livre*, la quantité d'information contenue dans *un livre* est extrêmement faible: il s'agit d'une redondance (syntaxique) du verbe *lire* et de l'indétermination de son complément. Nous avons vu qu'il était possible d'envisager le problème dans le sens inverse: *lire* peut-être sélectionné comme un verbe privilégié de *livre*, et dans ce cas c'est lui qui porterait un minimum de sens. Cette analyse se justifie pour d'autres noms:

(83) *Jean fait une course (e.g. un 100 mètres).*

(84) *(?)Jean court une course.*

Ici, en effet la phrase équivalente à *Jean lit un livre* est la phrase (84) qui est maladroite. La phrase (83) est préférable. On dit que le verbe *faire* est un verbe support du nom *course*. L'analyse «logico/fonctionnelle» de cette phrase devrait donc être *faire_course(Jean)*. Dans ce cas, la phrase (84) équivalente à (83) doit recevoir la même analyse, et donc le verbe *courir* dans cette phrase a le même statut de verbe support. On retrouve la même situation avec *Jean (fait+écrit) un livre*. Pour *lire* si le verbe support «canonique» n'existe pas, l'analyse reste la même. Pour éliminer cette redondance syntaxique, *qui n'est pas une redondance d'information*, il faudra faire un choix: soit on ne conserve que le nom, soit on ne conserve que le verbe. Aucune des deux solutions n'est totalement satisfaisante dans le cas d'une analyse très stricte. Si on supprime le verbe, on perd l'information de lecture, (qui pourrait aussi être celle de feuilleter). Si on supprime le nom, on perd l'information disant qu'il s'agit d'un livre et non pas d'une bande dessinée. On a donc bien effectivement deux objets sémantiques différents, mais ces deux objets ont une intersection forte. Pour comprimer linguistiquement cette phrase, on pourrait envisager d'introduire une modalité[9] modifiant l'action unique *lire un texte*.

> En conclusion, quel que soit le choix effectué, on ne pourra pas transmettre la même information sans redondance syntaxique, ou sans omission en gardant une forme textuelle à l'information. La redondance persistente pertubera l'extraction d'information ultérieure en faisant apparaître des contraintes linguistiques. L'omission appauvrira les résultats.

11.4.2 La formation des phrases

Si nous reprenons le processus de description des phrases données en 3, l'information proprement dite est située dans la décomposition en phrases élémentaires ou même dans la forme antérieure que nous avons représentée comme des fonctions pseudo/logiques. Considérons la phrase suivante:

Christian Ramussen, jeune Danois de 18 ans, n'a pas eu de chance.

Elle pourrait avoir la décomposition logico/fonctionnelle suivante:

9. De la même manière que dans *je dois venir*, il s'agit du verbe *venir* portant la modalité *devoir*.

R_1 $CR =$ homme(Christian Ramussen)
R_2 Nationalité$(CR, Danois)$
R_3 âge$(CR, 18)$
R_4 $\neg$(avoir_de_la_chance(CR))

Regardons maintenant différentes étapes possibles de la reconstitution des phrases associées à cette représentation :

P_1 – Christian Ramussen
– Mr Christian Ramussen
– Mr Ramussen
– Le prénom de Ramussen est Christian

P_2 – le Danois Christian Ramussen
– Christian Ramussen, Danois,
– Christian Ramussen de nationalité Danoise.
– Christian Ramussen porte la nationalité Danoise.
– Christian Ramussen est Danois.

P_3 – Christian Ramussen, jeune homme de 18 ans, Danois
– Christian Ramussen, jeune Danois de 18 ans
– le Danois Christian Ramussen, 18 ans
– Christian Ramussen est danois, et il a 18 ans.
– CR est un jeune. Ce jeune est Danois. Ce jeune a 18 ans.

P_4 – N_0 a de la chance
– N_0 être chanceux

P_5 – N_0 n'a pas de chance
– N_0 n'est pas chanceux=N_0 est (un) malchanceux
– le malchanceux N_0
– Christian Ramussen, jeune Danois de 18 ans, n'est pas chanceux.
– Le malchanceux jeune Danois, Christian Ramussen, de 18 ans
– Christian Ramussen, qui a 18 ans et qui porte la nationalité Danoise, n'a pas eu de chance.
– Mr. Ramussen est âgé de 18 ans. Il est Danois.
Il n'a pas eu de chance.

Entre l'information sous sa forme «brute» et les différentes formes textuelles, nous remarquons des différences fondamentales :

- La notion finale de phrase n'est pas significative, la même information peut-être portée par un groupe nominal (donc une partie de phrase), une phrase simple, une phrase complexe (c'est-à-dire plusieurs phrases coordonnées ou subordonnées).

- La forme de chaque «parcelle» d'information dans la forme textuelle peut varier, et ce sans introduire la relation sémantique de synonymie (ne pas avoir de chance (structure verbale), ne pas être chanceux, (être) malchanceux (adjectif-nom)).

- La localisation des différentes parcelles d'information n'est pas forcément limitée à des mots simples (*nationalité Danoise*), ni même à des séquences connexes provenant des mots simples (*jeune...de 18 ans*, *malchanceux* qui porte deux informations).

- Certains des mots rajoutés ne sont pas des mots vides au sens statistique (*porte, nationalité, âgé*).

- Certains des mots dits grammaticaux peuvent modifier totalement le sens du message (*Danois de 18 ans→Danois depuis 18 ans*).

11.4.3 L'approximation de la grammaire

Nous reprenons ici les caractères mis en évidence précédemment et nous indiquons les approximations qui les prennent en compte:

- Certaines unités provenant de phrases élémentaires assurent la liaison de leurs différents arguments à l'aide de mots bien définis ne représentant sémantiquement que cette «liaison». Ce problème est traité par la suppression des mots dits grammaticaux. Ces mots peuvent soit définis *a priori* (stop list), soit être extraits automatiquement du texte sur une base de fréquence élevée, ou, d'une manière plus raffinée, sur la base d'une répartion quasi-aléatoire.

- Certaines unités provenant de phrases élémentaires sont exprimées par plusieurs mots pleins (par opposition aux mots grammaticaux). Ce problème peut être traité par la localisation de segments répétés (multi-words units) qui sont des suites connexes de mots, ou même par la localisation de quasi-segments répétés (les différentes unités reviennent à distance variable et dans un ordre non fixé).

- Certaines unités identiques provenant de phrases élémentaires peuvent exister sous des formes fléchies différentes. Une lemmatisation automatique de mots (utilisation de la racine ou de règles très générales) rend ces formes équivalentes.

- Certaines phrases élémentaires peuvent être représentées indifféremment par une ou plusieurs phrases. La notion de phrase n'est pas une notion pertinente pour la localisation de l'information. On peut utiliser des fenêtres de collocation ne tenant pas compte de la segmentation naturelle en phrases. Les fenêtres sont uniquement définies par une taille à gauche, et une taille à droite.

- Certaines unités de phrases élémentaires s'observent sous des étiquettes syntaxiques diverses. Nous en avons vu un exemple de nom/verbe, et de verbe/adjectif. On peut regrouper automatiquement les mots par racine morphologique (voir [13]).

- Certaines unités de phrase ne sont pas indépendantes (voir page 262) par exemple *lire* et *livre* dans *lire un livre*. On peut contourner ce problème en travaillant uniquement sur une partie du discours donnée comme les noms. Cela nécessite néanmoins un étiquetage.

Toutes ces approximations tiennent compte des différents phénomènes linguistiques. Considérons en détail le cas de la flexion.

Le problème de la flexion des noms est résolu au LADL par codage dans le DELAS de la catégorie flexionnelle à laquelle chaque nom appartient [10]. Chacune de ces catégories est associée à une règle d'accord. Par exemple, la catégorie $N1$ regroupe tous les noms maculins prenant un pluriel en $-s$, la catégorie $N5$ regroupe tous les noms maculins en

−ail prenant un pluriel en *−aux* (*travail*), la catégorie $N16$ regroupe les noms maculins en *−e* prenant un pluriel en *−i* (*nuraghe*[10]), etc. La couverture complète des différents cas est ainsi assuré par quelque 80 règles.
Comparons cette description détaillée à l'approximation : «*tous les noms prennent un −s au pluriel*».
Dans notre dictionnaire, cette «règle» est valable à 97,6 %. Si on ajoute une deuxième règle indiquant que les mots terminant par $-s$ ou $-x$ sont invariables, la validité de l'analyse passe à 98,6 %. Dans la pratique cette règle est intéressante dans le sens inverse, c'est-à-dire pour trouver les noms singuliers associés aux noms pluriels du texte (lemmatisation). Nous avons donc systématiquement supprimé les *−s* des noms pluriel observés[11] dans notre texte et laissé les autres identiques. L'efficacité d'une tel algorithme est de 97,5 %, le bruit (mauvaise suppression d'un *−s*) est de 0,1 %, et le silence (mot laissé sous sa forme, alors que la forme singulier est différente) est de 2,4 %.
Dans ce cas précis, on évalue exactement l'information contenue dans la catégorie flexionnelle des mots : pour chaque nom $h = \sum_i -P(N_i).\log_2 P(N_i)$. On obtient $h = 0,38$ bit. Si on ne rajoutait donc aucune information de corrélation entre la forme d'un nom et son pluriel, il faudrait en moyenne 0,38 bits par mot pour coder la classe flexionnelle de chaque nom.
Si maintenant, on distingue quelques terminaisons spéciales des noms singuliers *-au*, *-ail* et *-al*, *-x* et *-s*, la probabilité que ces pluriels soient réguliers hors de ces catégories est 99,6%. A l'intérieur de chacune de ces catégories on calcule aussi la probabilité pour que les pluriel soient aussi ceux attendus (i.e. *-aux*, *-aux* et *-aux*, *-x* et *-s*), et on recalcule l'entropie cumulée de tous les mots : $H = 1710$. L'entropie par mot passe donc à $h = 0,03$ bits par mots ! Cela signifie qu'il faut en tout 214 octets pour coder la flexion de tous les noms (près de 60 000 entrées). Avec cette approximation encore élémentaire (distinction de 5 types de pluriel), la précision atteinte est de l'ordre de 99,8%, le codage de toute l'information sur la flexion des noms peut tenir dans 214 octets. Ce qui veut dire que la quantité d'information contenu dans cette ressource linguistique est très faible, et qu'il peut-être tout à fait légitime de l'approximer car les résultats obtenus par une méthode linguistique complète (concernant l'extraction d'information) ne seront pas sensiblement meilleurs. En d'autres termes, *la connaissance des pluriels associés à chacun des noms permettra de réduire au maximum l'entropie de tout message en français de* 214 *octets.*

Si nous prouvions que l'information contenue dans chacune des ressources linguistiques utilisables était du même ordre de grandeur, il serait tout à fait légitime de concevoir des systèmes d'extraction d'information complet n'incluant aucune ressource linguistique. Nous montrons maintenant ce qu'il en est pour le phénomène «figé».

11.4.4 Les figés

La question que nous nous posons est la suivante : «*Quel est l'apport entropique de la connaissance d'une liste de mots composés, et plus généralement d'une description des unités figées telles que nous les avons décrites dans la partie II ?*». Si nous montrons que cet apport est important, la question principale serait alors : «*Comment utiliser ces ressources pour obtenir des résultats plus significatifs ?*».

10. Construction cyclopéenne de l'âge du bronze, en Sardaigne.
11. En supposant que l'on puisse distinguer les noms et que l'on connaisse leur nombres.

Pour répondre à cette question, nous considérerons exclusivement les noms composés. En effet, il s'agit d'une large classe bien représentative . Nous considérons d'abord l'utilisation triviale des unités figées qui consiste à les considérer comme des unités minimales pour tout calcul statistique. Puis nous approfondissons leur utilisation en explicitant leur figement.

L'utilisation triviale des unités figées

La première utilisation simple d'une liste de mots composées pour l'extraction d'information est de marquer dans un texte toutes les unités composées présentes et de les considérer comme des unités minimales pour un algorithme classique. Nous avons expérimenté cette opération sur le système ACRONYM [14] en l'associant avec des opérations de lemmatisation. Ce système manipule la notion de nym, la définition d'un «nym» étant: deux mots sont des nyms, s'ils apparaissent tous les deux d'une manière privilégiée dans le contexte d'un troisième mot. Les nyms sont ensuite classé par «cluster», et une analyse factorielle discriminante permet de faire ressortir certains groupes sémantiques. Plusieurs effets sont observés:

A- Apparition de noms composés dans la liste des nyms les plus statistiquement pertinents;

B- La largeur de la fenêtre de collocation se trouve naturellement élargie dans les phrases contenant des composés sur plusieurs mots (c'est-à-dire que la distance entre deux objets de part et d'autre d'un nom simple ou d'un nom composé est la même, ce qui semble plus pertinent);

C- Disparition de certains nyms non composés importants;

Les points positifs de cette démarche sont les points **A** et **B**. Le deuxième point est cependant très difficile à mesurer (le nombre de nyms statistiquement pertinents augmente légèrement). **C** par contre est nettement négatif puisqu'il appauvrit les résultats. Il est en concurrence avec **A**, mais il faut remarquer que souvent, quand un mot simple comme *ministre* est masqué par des mots composés, les mots composés associés n'apparaissent pas forcément car ils sont nombreux et donc nettement plus rares. On dit que **C** est du à un **effet de masquage** des mots simples par les mots composés.
ACRONYM gère également la notion de segment répétés (*multi words units*). D'une manière prévisible, tous les noms composés que nous avions vu apparaître dans les listes de nyms apparaissent, en tant que segments répétés, sans avoir de dictionnaire de mots composés. En effet, les mots composés que nous avons localisés sont ceux qui apparaissent suffisamment fréquemment, et donc ils sont tous repérables par une fréquence élevée dans la liste des *multi words units*. Un argument permettant de justifier l'utilisation d'un dictionnaire de noms composés est qu'en français les flexions de ces noms composés ne sont pas triviales, et que les segments répétés ne peuvent pas identifier deux mots composés de lemmes identiques mais de formes fléchies différentes. Cet argument de flexion non triviale des mots composés ne tient guère si on applique une «lemmatisation automatique» comme celle montrée plus haut. En fait, nous n'avons même pas besoin d'une fonction donnant le lemme d'un mot composé, mais d'une fonction comme celle de la figure 11.5

qui indique si deux composés font référence à la même forme. Cette fonction prend en argument deux chaînes de la forme $d\ p\ e$ avec $p = de + à + sur + sous + par + ... + \epsilon$ et qui renvoit une valeur booléenne disant si ces composés potentiels font référence à la même forme. Sur les 113,000 segments de longueur 2 et 3 que nous avons décrits page 254, aucune erreur de regroupement n'a été observée avec ce simple algorithme.

```
1.  Fonction Ident(chaine c = d p e,chaine c' = d' p' e'):boolean
2.  Retourne MemeMot(d,d') Et p = p' Et MemeMot(e,e') ;
3.  Fin
4.  Fonction MemeMot(chaine c, chaine c'):boolean
5.  Retourne
6.          c = c' - s Ou c = c' Ou (c = m - a(i + ε)l Et c' = m - aux)
7.      Ou c' = c - s Ou c' = c Ou (c' = m - a(i + ε)l Et c = m - aux) ;
8.  Fin
```

FIG. 11.5 – *Fonctions indiquant si deux segments répétés correspondent au même «lemme».*

Conclusion : Nous venons de montrer que l'utilisation simple des mots composés dans un système comme *ACRONYM* permet de raffiner légérement les résultats mais le raffinement provient de la seule information : «des segments de plus d'un mot peuvent être intéressants» (plus éventuellement une fonction de regroupement de ces segments comme celle de la figure 11.5), mais pas vraiment de la liste des mots composées elle-même. De plus, l'effet de masquage apporté par la reconnaissance des mots composés détruit certaines informations extraites sans les mots composés. Nous montrons maintenant que pour exploiter l'information provenant du dictionnaire de mots composés, il faut se livrer à des mesures beaucoup plus fines sur les objets reconnus.

Analyse de mots composés

Nous rappelons la définition de nom composé que nous utilisons : «*toute séquence nominale dont le calcul sémantique ou syntaxique de composition n'est pas prévisible*».
Avec cette définition, *carte bleue*, *carte blanche* et *carte magnétique* sont trois mots composés de même forme *NA, nom-adjectif.* Ces trois composés sont néanmoins de trois types différents :

- *carte bleue.* Son sens ne se déduit pas d'une formule générale à partir des deux mots *carte* et *bleue. carte* n'est pas le nom-tête, puisqu'on n'a pas *une carte bleue est une carte.* Par contre, on a *une carte bleue est une carte bancaire.* L'adjectif n'a pas non plus sa fonction de modifieur classique, puisqu'il existe des cartes bleues qui sont roses.

- *carte blanche* : n'existe que dans l'expression *donner carte blanche à*, on peut néanmoins le trouver seul : *Avec ma carte blanche, je passerai partout* qui s'analyse comme *Avec la carte blanche que j'ai reçue, je passerai partout.* La définition ne fait pas intervenir ni le nom *carte*, ni l'adjectif *blanche.*

- *carte magnétique*. Ici, on peut avoir la construction *une carte magnétique est une carte (portant des pistes magnétiques)*. Donc, le nom semble avoir un comportement général de nom-tête ; cependant, l'adjectif *magnétique* n'est pas utilisé dans son sens général mais dans le sens non productif : *qui possède une piste magnétique*.

Ces séquences sont donc toutes les trois des séquences composées. Pour les manipuler (humainement), il faut en posséder les définitions et les structures syntaxiques dans lesquelles elles interviennent. Il y a donc une pertinence linguistique à les regrouper dans une liste de noms composés, et ce regroupement comporte une part d'information que nous voulons utiliser dans un processus d'extraction d'information. Le caractère naturel de de regroupement ces noms est confirmé par le test de décision de l'appartenance d'un nom composé au DELAC qui est totalement reproductible[12].
Cette définition très large des mots composés est celle adoptée au LADL. Elle a conduit à la construction du DELAC qui comporte de l'ordre de 150 000 entrées. Cette ressource est donc considérable sur le plan de l'information. Du point de vue entropique, la décision qu'une combinaison *nom-adjectif* est figée ou libre est très coûteuse puisqu'il y a environ 60 000 noms, environ 20 000 adjectifs, et seulement[13] 20 000 noms composés de type NA, ce qui fait une entropie de 317 000 à comparer à l'entropie de la flexion 1 500.
Avant d'utiliser une telle ressource, deux questions naturelles se posent : «*Est-il possible de faire une approximation de la même manière que celle faite pour la flexion de manière à construire automatiquement la plupart de ces mots composés ?*» et «*Peut-on les retrouver automatiquement dans un texte sans ressource extérieure ?*»
Nous avons partiellement répondu à ces questions : comme nous localisons les segments répétés, nous appréhendons les mots composés se répétant suffisamment fréquemment.[14] Cependant, il ne sera pas possible de décider de leur caractère figé. Ainsi *deux ans* apparaît de la même façon dans la table des segments répétés de longueur 2 que *conseil municipal* et *Loire-Atlantique*. L'information apportée n'est cependant pas la même dans ces trois exemples : dans *deux ans* l'utilisateur ne pouvait pas prévoir le lien entre les deux mots *deux* et *ans* : il y a donc de l'information dans le repérage de ce segment répété en plus de l'information de fréquence élevée. Dans *conseil municipal* et *Loire Atlantique*, il y a de la surprise dans la fréquence relativement élevée de ces deux mots, mais il n'y a aucune surprise dans leurs formations. D'où une première mesure non triviale[15] :

Tout segment répété fréquent qui n'est pas un composé comporte une information non syntaxique de co-occurrence forte entre ces composantes.

Montrons maintenant que le «masquage» provoqué par les mots composés peut aussi servir de source d'information potentielle. Sur notre corpus d'un million de mots, nous avons trouvé 23 674 noms composés de la forme N_1A, AN_1 ou $N_1(D+A+P)N_2$. Pour chacun de ces noms composés et dans un contexte droit de taille fixée, nous avons compté le nombre d'occurrences de N_1, seul ainsi que le nombre d'occurrences du mot composé (éventuellement avec une flexion différente). Par exemple, pour un contexte droit de taille

12. La définition précédente accompagnée de quelques exemples est suffisante, indépendamment du fait que nous appelons ces séquences des mots composés.

13. Nous comptons les formes lemmatisées.

14. De plus, nous avons vu que ceux qui n'apparaissaient pas assez fréquemment n'étaient pas statistiquements intéressants.

15. C'est-à-dire non réalisable sans le dictionnaire des noms composés.

50 mots, nous avons trouvé que 2 063 noms simples N_1 qui «reprenaient» le nom principal du nom-composé, et seulement 936 mots composés qui «reprenaient» le nom-composé initial. Voici quelques exemples de ces reprises :

(85) CERTES, un gouvernement démocratique peut revendiquer une marge de manoeuvre et de secret afin de défendre l'"intérêt national$_{NA}$". Mais la raison d'Etat a des limites, surtout quand cet intérêt est ...

(86) En novembre, le chômage a augmenté de 0,1 %, en données corrigées$_{NA}$. La France comptait 3 285 700 demandeurs d'emploi, soit 3 200 de plus qu'en octobre. En données brutes$_{NA}$, ...

(87) ... la protection de leur vie privée$_{NA}$ et qu'ils ont seuls le droit de fixer les limites de ce qui peut être publié sur leur vie intime$_{NA}$, ...

(88) ... l'article L. 211-1-1 du code rural$_{NA}$ et sanctionnée des peines prévues aux articles L. 215-1 et suivants du même code, soit un emprisonnement de six mois et/ou une amende de 60 000 F.

(89) ... incriminé par le droit communautaire$_{NA}$ et non incriminée par le droit français$_{NA}$.

(90)(*) Les violences subies par les femmes ont pris depuis longtemps aux Etats-Unis$_{NA}$ la dimension d'un problème national, mais plus rarement fait-on état du nombre d'hommes victimes d'affrontements conjugaux.

Dans (90), le nom *état* ne correspond pas à la même notion que *Etats-Unis*. Nous avons évalué le pourcentage de telles «reprises» erronées[16] pour ce contexte de 50 mots. Il est d'environ 1 %. Dans les phrases (86), (87) et (89) la «reprise» se fait par un autre nom composé. Dans (85), et (88) la reprise est faite par le nom seul. Notons dans ces deux phrases la présence du démonstratif montrant bien le lien entre les deux groupes nominaux.

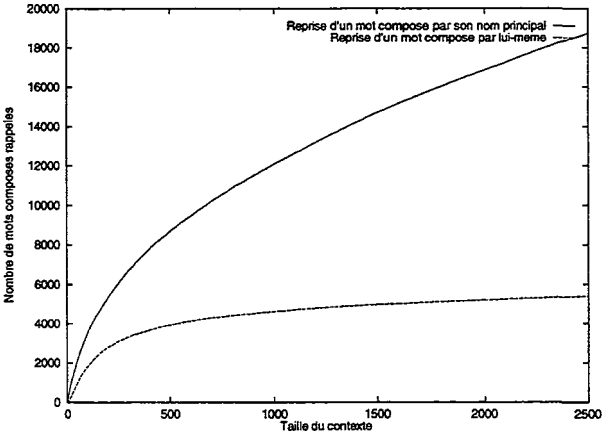


FIG. 11.6 – *«Reprise» des mots composés par leurs noms principaux et par eux-mêmes.*

Nous représentons sur le graphe de la figure 11.6, ces deux quantités variant en fonction de la taille du contexte. Il est clair qu'après 2 000 mots, le nom membre du nom composé ne peut plus faire référence au mot composé (la taille moyenne des longs articles sur ce

16. Nous ne parlons pas de référence directe entre deux groupes nominaux, mais d'une identité de concepts.

corpus est de 600 mots). Nous distinguons sur la courbe deux phases : contexte inférieur à 500 mots, et contexte supérieur à 500 mots. Dans la première phase, la croissance est forte mais la pente est décroissante. Dans la deuxième phase, la croissance est quasi-linéaire, ce qui correspond à une distribution aléatoire du nom recherché, ou du nom composé recherché (ce qui prouve qu'il n'y a plus de lien entre les deux groupes nominaux).

En fait pour analyser correctement cette courbe, il faut distinguer trois phénomènes différents :

- Apparition aléatoire des séquences recherchées. Si les mots sont distribués aléatoirement sur le texte, leur appartion se manifeste par une croissance linéaire. Nous l'observons nettement pour des contextes de taille supérieure à 1 000 mots.
- Reprise par référence d'un mot composé par son nom principal. Cet effet a forcément une très courte portée (quelques phrases).
- Effet contexte. Le premier mot composé que nous trouvons modifie la distribution locale des mots sur l'article (voir plus haut), donc la présence des mots du même domaine apparaissent d'une manière favorisée dans tout l'article.

Seuls ces deux derniers phénomènes nous intéressent, nous les appelons tous deux **effet de reprise**. Le premier phénomène est clairement négligeable par rapport aux deux suivants pour les contextes inférieurs à 1 000 mots. La comparaison des deux courbes nous montre que l'effet de reprise est beaucoup plus faible, si nous recherchons le même mot composé (environ 2 à 3 fois).
Ainsi le masquage provoqué par la reconnaissance des noms composés est facilement contournable lors de l'extraction d'information, il suffit de relier les noms simples aux noms composés dont ils sont issus, on augmente alors la fréquence de ces noms composés. Nous avons calculé que cette heuristique valable dans le contexte d'un article (ou en approximant, dans un contexte de 500 mots) introduit le taux d'erreur négligeable de 2 % (les cas d'erreur sont ceux où les mots simples ne font pas référence au nom composé).

Réciproquement, quand les mots composés ne peuvent pas être repris abrégés, il est naturel de masquer les composantes (c'est à dire de ne plus effectuer de calcul dessus). Pour représenter les relations de reprise de manière rigoureuse, il faut savoir quels sont les mots composés qui peuvent être «repris» et ceux qui ne le peuvent pas. Des travaux systématiques sont en cours [5] sur ce sujet, ils concernent l'ensemble des noms composés du DELAC et ont pour but de montrer les modifications possibles des noms composés. Ces travaux montrent par ailleurs, qu'il est possible de créer des classes d'équivalence qui opèrent des regroupements syntaxiques (qui ne sont pas loin d'être sémantiques) dans les listes de noms composés. Ainsi, une *amende salée* et une *amende lourde* sont deux mots composés pour lesquels le mot *amende* fait référence au même objet syntaxique (ce qui autorise entre autre la conjonction *une amende lourde, voire salée*). De plus la propriété *une amende lourde est une amende* permet de valider la reprise du nom composé par le nom principal seul.
Cette étude fait de plus connaître les modifications de forme possible des mots composées, parmi ces modifications, il y a les permutations (*amende lourde* = *lourde amende* mais *grande surface* ≠ *surface grande*), les insertions (*ligne directe* → *ligne de téléphone directe*), etc.

11.5 Conclusion

L'extraction d'information dans un texte est donc un phénomène purement statistique. Cependant, pour fournir à l'utilisateur de l'information sur le contenu et non sur la forme du texte ou sur la langue, il faudrait «comprimer» linguistiquement le texte, c'est-à-dire en retirer toute l'information linguistique. Nous avons montré que ce n'était pas possible si l'on conserve une forme de langue naturelle pour le message «comprimé». Pour cette raison, utiliser les frontières des phrases simples comme limites pour l'analyse statistique n'est pas plus pertinent qu'utiliser une segmentation artificielle. Il est cependant possible d'utiliser des ressources linguistiques pour réduire l'entropie. Nous avons montré que des dictionnaires permettant de lemmatiser exactement les noms avaient une importance très faible dans cette minimisation, comparée à un algorithme de «lemmatisation» approximatif. De la même façon, l'utilisation de séquences figées connexes d'une manière «brute» (regroupement des différentes mots de la séquence) avait un intérêt quasiment nul et même potentiellement négatif, sur l'extraction d'information. Pour pouvoir utiliser l'information contenue dans une liste de mot composés, par exemple, il faut faire des mesures beaucoup plus fines et d'une certaine manière «défiger» un peu ces mots composés. Cette conclusion était prévisible, la notion de figement faisant l'objet d'un continuum, il serait peu plausible que l'on observe un comportement sémantique et syntaxique complètement différent entre unités figées et unités libres. Par contre, la complexité des séquences non connexes fait que leur reconnaissance simple, sans ajout d'information, de ces séquences ne peut pas être approximée par des méthodes statistiques locales. On a vu dans quelle mesure la reconnaissance des expressions figées apportait une quantité considérable d'information sur la forme du texte, et devrait donc améliorer nettement les analyses statistiques sur son contenu.

Bibliographie

[1] M. Bécue and R. Peiro. Les quasi-segments pour une classification automatique des réponses ouvertes. In *Actes des 2ndes Journées Internationales d'analyse des données textuelles*, pages 310–325. ENST, 1993.

[2] F. Benzecri and J.P Benzecri. *Pratique de l'analyse de données.* DUNOD, Paris, 1980.

[3] K.W. Church and P. Hanks. Word association norms, mutual information and lexicography. *Computational Linguistics*, 16:22–29, 1990.

[4] J. C. Henderson E. Brill, R. Florian and L. Mangu. Beyond n-grams : Can linguistic sophistication improve language modeling ? In *COLING-ACL'98*, volume 1, pages 186–190. Université de Montréal, 1998.

[5] A. Fotopoulou and J. Senellart. Transformation sur les mots composés. Communication au 17e colloque du lexique-grammaire des langues romanes, 1998.

[6] T. Foucart. *Analyse Factorielle de Tableaux Multiples.* MASSON, Paris, 1984.

[7] M. Gross and J. Senellart. Nouvelles bases statistiques pour les mots du français. In *JADT 98, 4èmes Journées internationales d'Analyse statistique des Données textuelles*, pages 335–350. Université de Nice - Sophia Antipolis, Février 1998.

[8] V. Hatzivassiloglou. *The Balancing Act*, chapitre Do we need Linguistics When We Have Statistics? A Comparative Analysis of Contributions of Linguistics Cues to a Statistical Word Grouping System. MIT Press, 1994.

[9] L. Lebart and A. Salem. *Statistique textuelle.* DUNOD, 1994.

[10] C. Leclere. Dictionnaire électroniques du français. In *Langue française*, volume 87, chapitre Un système de dictionnaires électroniques pour les mots simples su français, pages 11–22. Larousse, 1990.

[11] C. Martineau. *Compression sur les mots.* Thèse de doctorat, Université de Marne la Vallée, 1999.

[12] F. Miller et al. Wordnet : an on-line lexical database. *Interntational Journal of Lexicography*, 3(4):235–245, 1990.

[13] M. Reinert. Alceste, une méthodologie d'analyse des données textuelles et application : Aurélia de gérard de nerval. *Bull. de Méthod. Sociol.*, 26:24–54, 1990.

[14] A. Renouf et al. The acronym project: Discovering the textual thesaurus. In *Papers from English Language Research on Computerized Corpora (ICAME'96)*, pages 171–187, Amsterdam, 1996.

[15] M. Roux. *Algorithmes de classification.* MASSON, Paris, 1985.

[16] A. Salem. *Méthodes de la statistique textuelles.* Thèse d'État, Université Paris 3, 1993.

[17] J. Senellart. Indexation de grands corpus en tenant compte de données linguistiques. Rapport de stage d'Option, Ecole Polytechnique, 7 1995.

[18] J. Senellart. Statistique prudence. In *Acte des Premières Journées INTEX.* LADL, 1996.

[19] C.E. Shannon. *A Mathematical theory of communication.* University of Illinois Press, 1949.

[20] Tournier. Textes propagandiste et coocurrences. hypothèses et méthodes pour l'étude de la sloganisation. In *Mots*, 11, pages 155–187. 85.

Cinquième partie

Conclusion, Perspectives

Les résultats numériques obtenus par notre programme de reconnaissance d'expressions figées dans des textes sont de trois types :

- Une proportion d'environ un mot sur deux du français écrit fait partie d'une unité minimale complexe pour laquelle une grammaire doit nécessairement être appliquée. Cela prouve qu'aucune analyse de la langue ne peut ignorer de telles ressources.
- Le nombre de ces grammaires construites automatiquement à partir des tables du lexique-grammaire est de l'ordre de la centaine de milliers.
- Chacune de ces grammaires reconnaît en moyenne deux occurrences dans notre texte. Or la moitié des séquences reconnues sont de ce type «binaire». Donc une grammaire dont les règles sont à base de catégories reconnaîtra moins de la moitié des séquences que nous avons reconnues.

Nous avons montré que ces grammaires nécessitent une application par passes successives et qu'il n'y a pas lieu de les ordonner. Nous avons développé un formalisme adapté à leur application et étudié des algorithmes adaptés pour que l'étape de reconnaissance puisse être rapide, même avec des structures de texte comportant plusieurs millions de mots. Plusieurs problèmes ont été rencontrés ouvrant des perspectives de travail intéressantes :

- À chaque transformation appliquée au texte, toutes les grammaires sont à nouveau *a priori* applicables, puisqu'une transformation peut introduire de nouvelles formes, ou effectuer des opérations complexes comme des permutations. Il devrait être possible de n'appliquer localement que les grammaires concernés par la partie transformée. Pour cela, il faudrait pouvoir indexer les grammaires de la même façon que nous avons indexé l'automate du texte. Les transformations se feront alors par «intersection» de deux index. Cette approche permettrait de combiner une analyse du texte descendante avec une analyse ascendante.
- Les expressions libres présentent une description analogue aux formes figés, et nécessite les mêmes outils. Des tentatives de reconnaissance de schéma de phrases libres devraient pouvoir être effectuées à partir des ressources du LADL.
- Des ressources d'expressions figées, comme les mots composés, nécessitent un «défigement» pour accroître la couverture de leur reconnaissance.
- L'imbrication des différents phénomènes linguistiques fait que la conception de grammaire de verbes est extrêmement compliquée à visualiser et à maîtriser. La génération automatique de graphes patrons devrait permettre de tenir compte de plus de phénomènes parallèles et de simplifier encore les grammaires de base. Cette étape de génération de graphes-patrons n'est pas très éloignée de la génération de textes à partir des grammaires.
- On a montré que la factorisation de sous-motifs dans des automates était une opération améliorant la vitesse du pattern-matching. Cette factorisation, essentiellement manuelle, est réalisable automatiquement sur des graphes quelconques.

Dans le cas général, le problème est vraisemblablement $\mathcal{NP}$-complet, mais des sous classes intéressantes pourraient être dégagées.

- Le nombre de grammaires sous formes d'automates finis ne va pas cesser de croître à mesure que la description progresse. Des outils de manipulation de bibliothèques importances de graphes deviendront alors nécessaires.

Annexe A

Grammaire utilisée par les transducteurs étendus

Nous précisons ici la syntaxe et la sémantique des transducteurs étendus présentés page 65.
Les différents objets manipulés peuvent être des 5 types suivants :
F : forme
$M = \{f, l.C : I\}$: mot
$N = \{f, v.C : I\}$: nombre
V : valeur numérique
S : liste de variable O_1,..., O_k. Une liste de taille 1 est équivalente à la seule donnée de la liste. Une liste de taille 0, correspond à une variable vide.

Les différentes variables #i représentant des sous-séquences de l'expression reconnues peuvent avoir les 4 types F, M, N et S. Le type V est propre aux champs numérique des nombres.

Sur ces structures sont définies plusieurs opérations que nous décrivons ensuite. Quand le type d'une variable n'est pas celui attendu par l'opération, il y a une conversion automatique. Cette conversion utilise les fonctions internes forme, lemme, catégorie et info. Ces fonctions ne sont pas les opérateurs définis page 74. Elles n'apparaissent pas explicitement. Ces conversions se font à l'aide du schéma suivant :

	forme	lemme	catégorie	info
F	F	F	F	F
$M = \{f, l.C : I\}$	f	l	C	I
$N = \{f, v.C : I\}$	f	-	C	I
S	forme(O_1)_... forme(O_k)	forme(S)	forme(S)	forme(S)

La conversion de type quelconque en type S est instantanée puisque nous avons fixé qu'une liste de longueur 1 est équivalente à son seul élément.

A.1 Valeurs numériques

Les valeurs numériques proviennent nécessairement de l'interprétation d'une séquence s par l'opération $v = [s]$. Si forme(s) est une suite de chiffres (sans autres caractères), v

est la valeur numérique associée. Sinon $v = 0$. Les opérateurs $+$, $-$, $/$ et $*$ sont définis sur les valeurs numériques avec leurs fonctions classiques.

A.2 Formes

Les formes se concaténent par l'opération $X_1_X_2$. Les formes se combinent aussi à l'aide des opérations $+$ et $-$ d'une manière compatible avec la gestion du champ information. Par exemple $Conc + Conc \rightarrow Conc$ et $(m + Hum) - m \rightarrow +Hum$.
L'opération $F_1 * F_2$ est définie ainsi : sa valeur est nulle si F_2 est nul, et sa valeur vaut F_1 sinon. Cette opération permet de modifier des informations portant sur un mot par exemple en fonction du chemin suivi. Sur le graphe de la figure A.1, les informations associées au déterminant sont `+Ddéf` si #1 est nul, ou bien `+Ddéf+Tous` sinon (c'est-à-dire si la séquence reconnue est de la forme *tous les N*.

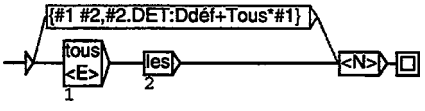


FIG. A.1 – *Utilisation de l'opérateur $*$ entre deux formes.*

A.3 Mot

Les mots sont construit avec la syntaxe suivante : $M \longrightarrow \{S_1, S_2.S_3(: S_4 + \epsilon)\}$
Le mot créé est alors :

$$\{\underline{forme}(S_1), \underline{lemme}(S_2).\underline{\text{catégorie}}(S_3) : \underline{\texttt{info}}(S_4)\}$$

L'opération $M/ < F >$ est définie de la manière suivante : $\{f, l.C : I\}/ < F > \rightarrow \{f, l.F : I\}$. Cette opération permet de changer la catégorie grammaticale d'un mot.
L'opération $M + F$ permet d'ajouter des informations : $\{f, l.C : I\} + F \rightarrow \{f, l.C : I + F\}$ (et pareillement pour $M - F$.
Par exemple, les deux graphes de la figure A.2 ont deux comportement différents. Dans A.2(a), la forme *être* est reconnue, et donc #1 vaut *être*. La séquence créé est donc $\{$*être*, *être.être*:`être`$\}$. Par contre, les séquences reconnues par le graphe A.2(b) sont des *mots* de lemme *être*. Et donc #1 est un mot, et donc la séquence créée par la transformation est le mot lui-même (en fait, un mot qui a la même forme, le même lemme, la même caégorie, et les mêmes informations sémantiques).

A.4 Nombre

Pour créer un nombre, le processus est analogue à celui de création de mot : $M \longrightarrow \{S_1, V.S_3(: S_4 + \epsilon)\}$
et le nombre créé est alors :

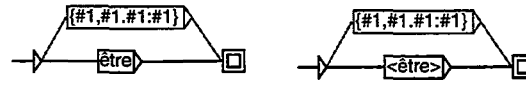
$$\{\underline{forme}(S_1), V.\underline{\text{catégorie}}(S_3) : \underline{\texttt{info}}(S_4)\}$$

Les mêmes opérations $/ < . >$, $+$ et $-$ sont définies sur les nombres.

A.5 Séquence

Le chemin créé par le transducteur étendu sera un chemin dont chaque transition correspond à un élément de la séquence créée.
Pour créer une séquence ou pour concaténer une séquence la syntaxe est :

$$S \longrightarrow (S + \epsilon + F + M + N + V)\ \ (F + M + N + V + S)$$



(a) Reconnaissance de la forme *être*

(b) Reconnaissance des mot de lemme *être*

FIG. A.2 – *Transformation analogue pour deux motifs reconnus différents.*

Annexe B

La loi de Zipf

Cette annexe décrit quelques propriétés de la loi de Zipf et donne une liste de références à cette loi. Une grande partie de ces références ont été trouvées sur le site Web `http://linkage.rockfeller.edu/wli/Zipf`.

La loi de Zipf, désignée ainsi après la mort du professeur de l'université de Harvard, GEORGE KINGSLEY ZIPF (1902-1950), est l'observation que la fréquence d'occurrence f_i d'un événement (P_i) en fonction de son rang r_i (où le rang est fixé par la fréquence décroissante) est une loi en $\frac{K}{i^a}$ avec $a \approx 1$.

Il est intéressant de noter que cette représentation fréquence/rang est introduite par ZIPF dans [16] par la phrase de la figure B.1 qui montre la prudence de ZIPF quant à l'origine de cette loi.

Cette loi est observable dans la distribution d'un grand nombre «d'évenements». Ainsi, la distribution des mots dans un texte, la distribution de la taille des villes, des salaires ou du nombre du nombre de «clic» dans une page Web [1] ! L'observation dans les domaines économiques est plus souvent appelé loi de Pareto qui l'a observée à la fin du vingtième siècle [11].

Pour le langage naturel, cette loi a aussi enormément été discutée, et formalisée. Quelques références importantes dans les années 60 (années autour desquelles les premiers gros «corpus» peuvent être analysés automatiquement) : B. MANDELBROT [3], [4], [7], G.A. MILLER [9] et un débat a rebondissement sur l'origine de cette loi entre H. A. SIMON et B. MANDELBROT dans la revue *Information and Control*: [13], [5], [14], ..., [6] et [15]. Cette loi un peu «mystérieuse» (la question sous-jacente à son observation étant : «*porte-t-elle de l'information linguistique ou même une loi universelle ?*») a même récemment donné lieu à un article localisant de l'information dans des génomes humains et faisant la supposition que cette localisation correspond au codage du language [8]. Des réponses à cet article (qui reprennent d'ailleurs des idées proposée par B. MANDELBROT) montrent que cette loi peut être celle de singe créant un livre par frappe aléatoire sur les touches d'une machine à écrire [2], [10]. Citons aussi l'article de R. PERLINE qui montre une autre manière de générer une loi de Zipf par division successive de l'intervalle unité.

Actuellement, alors que la taille des textes disponibles dépasse le gigaoctet, cette loi de Zipf devient plus difficile à être vérifiée *stricto sensu*. Nous montrons la distribution des mots sur la figure B.2 pour différentes tailles de textes. Nous voyons sur ce graphe que la loi est valable pour des textes pas trop gros et pour des rangs intermédiaires. Pour les petits rangs, la fréquence des mots appelés souvent mots grammaticaux est directement

proportionnelle à la taille du texte. Pour les trop faibles fréquences, au contraire, on observe pour des grands textes, une tendance qui se dessine et qui montre que ces mots sont sous-représentés par rapport à la loi attendue de Zipf. Il faut cependant se méfier en lisant le graphe B.2 : celui-ci est en échelle doublement logarithmique et donc la partie sur laquelle la pente de la courbe est proche de 1 ne concerne que peu de mots (quelques milliers), alors que la partie sur laquelle la courbe ne semble plus linéaire concerne tous les mots rares (plusieurs dizaines de milliers pour le plus gros texte). Pour ce texte, il est donc très difficile de dire que la loi de Zipf est vérifiée.

Remarquons que le coefficient K de la loi est la fréquence du mot le plus fréquent. La loi de Zipf peut nous donner certains ordres de grandeur[1] pour le texte :

1. **Taille du vocabulaire en fonction de la taille du texte.** La taille du vocabulaire étant N, la taille du texte T est exactement

$$T = \sum_{k=1...N} f_k = \sum_{k=1...N} \frac{K}{k^a}$$

Il faut distinguer deux cas : Si $a = 1$, la taille du texte est alors équivalente à $T \approx \log N$. Sinon $T \approx N^{-(a+1)}$. La croissance du vocabulaire est donc sous-linéaire par rapport à la taille du texte. Cette loi signifierait en particulier que la taille du vocabulaire n'est pas bornée. Même en prenant en compte les noms propres de toutes origines, cette conclusion est clairement fausse, puisque le vocabulaire est pratiquement borné (disons tous les mots de moins de 20 lettres). Cependant, cette loi se vérifie relativement bien pour N d'un ordre de grandeur de quelques millions de mots. Pour des textes de taille beaucoup plus grande, la croissance devient plus lente que les deux croissances $\log n$ et $N^{-(a+1)}$. Cette loi permet de minorer la taille de texte qu'il faut pour parcourir un dictionnaire de taille fixe. Cette analyse a été faite dans [12] sur les noms composés.

2. **Nombre d'hapax dans un texte.** Le nombre d'hapax est le nombre de mots de fréquence 1. Ce nombre n_1 est donc approximativement tel que $f_{r_1} = 1$, soit :

$$n_1 = K^{\frac{1}{a}}$$

où K est la fréquence du premier mot du texte. On observe de plus que K est proportionnel à la taille du texte. Le nombre d'hapax évolue donc de manière proportionnelle à la taille du texte (si a vaut 1).

3. **Nombre de mots de fréquence** f. De la même manière que précédemment

$$n_f \approx (\frac{K}{f})^{\frac{1}{a}} - (\frac{K}{f+1})^{\frac{1}{a}}$$

Dans le cas $a = 1$ et pour f grand, cela donne $n_f \approx \frac{K}{f} \cdot \frac{1}{f}$, soit $n_f . f^2 \approx K$. Cette loi est exactement la première des lois observées par ZIPF dans [16] (page 41 de la réédition de 1965).

As suggested by a friend, one can consider the words of a vocabulary as ranked in the order of their frequency, e.g. the first most frequent word, the second most frequent, the third most frequent, the five-hundredth most frequent, the thousandth most frequent, etc. We can indicate on the abscissa of a double logarithmic chart the number of the word in the series and on the ordinate its frequency.

FIG. B.1 – *Extrait de «the psycho-biology of language: an introduction to dynamic philology.» (George K. Zipf 1935).*

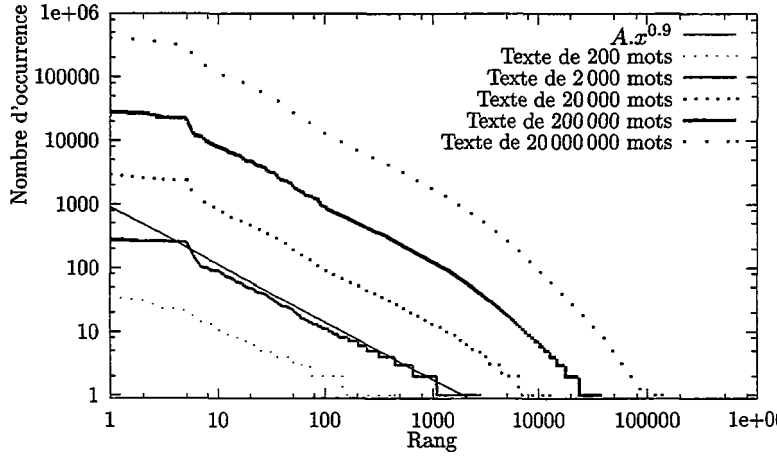


FIG. B.2 – *Courbe fréquence/rang pour plusieurs tailles de texte. On remarque que pour des très grands textes, la croissance des mots rares est plus lente que prévue.*

Conclusion : Nous avons donné quelques références relatives à cette loi de Zipf. Cette loi observée dans les années 30 sur des «petits» textes avait une bonne fiabilité. Actuellement, la taille des textes manipulés automatiquement montre que cette loi n'est pas si bien respectée. Il est possible que la tendance prise dans les textes actuels par la courbe de distribution soit due à la limitation du vocabulaire non prévue par la loi de Zipf[2]. Nous ne devons pas chercher à en extraire trop de renseignements. Pratiquement, le seul renseignement que nous avons utilisé pour nos algorithmes peut se résumer en : «*Il y a très peu de mots différents très fréquents et énormément de mots très rares dans un texte.* ». Ce retrait par rapport à l'observation de cette loi n'enlève rien de son aspect mystérieux. Des essais récents montre d'ailleurs qu'elle continue à être observée sur des sous-classes syntaxiques du vocabulaire en tenant compte d'unités minimales plus complexes que les mots simples.

1. Nous ne chercherons pas autre chose que des ordres de grandeur, vu l'approximation de la loi observée par rapport à la loi théorique de Zipf.

2. En repensant au modèle du singe sur la machine à écrire, nous devrions rencontrer de temps en temps des mots très longs (une centaine de caractères).

Bibliographie

[1] B.A. Huberman, P.L. Pirolli, et al. Strong regularities in world wide web surfing. *Science*, 280:95–97, 1998.

[2] W. Li. Random texts exhibit zipf's-law-like word frequency distribution. *IEEE Transactions on Information Theory*, 38(6):1842–1845, 1992.

[3] B. Mandelbrot. *Communication Theory*, chapter An information theory of the statistical structure of languages. Betterworth, 1943.

[4] B. Mandelbrot. Simple games of strategy occurring in communication through naturel languages. In *Transaction of IRE*, volume 3, pages 124–137, 1954.

[5] B. Mandelbrot. A note on a class of skew distribution function, analysis and critique of a paper by h.a. simon. *Information and Control*, 2:90–99, 1959.

[6] B. Mandelbrot. Post scriptum to 'final note'. *Information and Control*, 4:300–304, 1961.

[7] B. Mandelbrot. *Scientific Psychology : Principles and Approaches*, chapter Information theory and psycholinguistics. Basic Books, 1965.

[8] C. Mantegna et al. Linguistic features of noncoding dna sequences. *Physical Review Letters*, 73:3169–3172, 1994.

[9] G.A. Miller et E.B. Newman. Tests of a statistical explanation of the rank-frequency relation for words in written english. *American Journal of Psychology*, 71:209–218, 1958.

[10] P. Niyogi et R.C. Berwick. A note on zipf's law, natural languages, and noncoding dna regions. Technical Report 1530, MIT, Artificial Intelligence Laboratory, 1995.

[11] V. Pareto. *Cours d'économie politique*. Rouge, 1897.

[12] J. Senellart. Statistique prudence. In *Acte des Premières Journées INTEX*. LADL, 1996.

[13] H.A. Simon. On a class of a skew distribution functions. *Biometrika*, 42:425–440, 1955.

[14] H.A. Simon. Some further notes on a class of skew distribution functions. *Information and Control*, 3:80–88, 1960.

[15] H.A. Simon. Reply to dr. mandelbrot's post scriptum. *Information and Control*, 4:305–308, 1961.

[16] G.K. Zipf. *Psycho-Biology of Languages.* Houghton-Mifflin, 1935. réédité par MIT Press en 1965.

Index

&, 62
<*MOT*>, 62
<*X.ms*:+Hum>, 62

alphabet, **179**
ambiguïté, **13**, 58
automate
  plat, **182**
automate du texte, **58**, 223
  projection, 82
automate fini, voir graphe

capacité, **228**
catégorie grammaticale, **58**
chemin améliorant, **228**, 237
chevauchement, 84
composition, voir index
contraintes
  gestion, 71
coupe, 227, **228**
coupe minimale, **228**, 231

descripteur, **180**
description
  grammaire de, 50
dictionnaire, 36
  DELAF, 38
  utilisation, 38, 62
dualité, voir graphe, 233

ensemble pivot, **231**
entropie, **256**, 265, 268

factorisation, voir graphe, 198
figement, **91**, 266
flot, 227, **228**
flot admissible, **228**
flot maximal, **228**
flot propre, **233**
flux, **227**
forme, 58, **58**

grammaire locale, 18
graphe
  automate fini, repr. duale, 61
  factorisation, 64
  patron, **71**, 103
  récursif, 62

index, 181
  complet, **181**
  composition, **184**
  d'automate, **223**
  enrichissement de, **180**
  entrée de, **180**
  factorisation, 217
  intersection, **183**
  simple, **180**
Index-Index, **195**, 201
  approximatif, 195
Index-Texte, **195**, 201, 220
  d'ordre k, **195**
information, **58**
  flexionnelles, 59
  sémantique, 59
intersection, voir index
INTEX, 20, 23, 57, 61, 65, 75, 96, 254

lemme, **38**, **58**
lexique-grammaire, 31, 32, **48**, 101, 159
  table, **38**
lexique-grammire
  outils, voir graphe patron
loi de Zipf, 111, 176, 192, 200, 206, 252, **283**
longest match, 18, **23**

matrice chemin-transition, **231**
mot, **58**
  unité élémentaire, 36

nombres
    traitement, 75

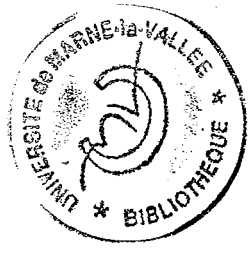
pattern matching, **192**
    points de départ, 227
permutation, 196
phrase
    complexe, **32**
    segmentation, 15
    élémentaire, 32, **32**, 41
points de départ, voir pattern matching
polysémie, 15
projection, voir automate du texte

reconnaissance, **31**, 50
réseau, **228**

sous-graphe, voir graphe
suffix-array, **193**, 201
suffixe
    arbre, **193**
    table, voir suffix-array

token, **179**
trait, voir information
transducteur, voir transformation
transformation
    linguistique, **41**
    transducteur étendu, 64
transition à étiquette unique, **233**

unité élémentaire, voir mot

Zipf, voir loi de Zipf

é
quilibrage, **239**