UNIVERSITE PARIS 7

MODELES POUR LA REPRESENTATION ET L'INTERROGATION DE DONNEES TEXTUELLES ET DE CONNAISSANCES

Thèse de Doctorat d'Etat ès Sciences

spécialité : Informatique

présentée par Marcel CORI

Thèse soutenue le 29 juin 1987 devant le jury composé de:

Président: D. PERRIN
Examinateurs: H. BESTOUGEFF
N. COT
A. CULIOLI
M. GROSS
M. KAY
D. KAYSER

UNIVERSITE PARIS 7

MODELES POUR LA REPRESENTATION ET L'INTERROGATION DE DONNEES TEXTUELLES ET DE CONNAISSANCES.

Thèse de Doctorat d'Etat ès Sciences

spécialité : Informatique

présentée par Marcel CORI

Thèse soutenue le 29 juin 1987 devant le jury composé de:

Président: D. PERRIN
Examinateurs: H. BESTOUGEFF
N. COT
A. CULIOLI
M. GROSS
M. KAY
D. KAYSER

Je remercie

Dominique Perrin, responsable de l'UFR d'Informatique de Paris 7, qui préside le jury de cette thèse,

Antoine Culioli, directeur du Laboratoire de linguistique formelle auquel j'appartiens depuis sa création, qui a toujours manifesté de l'intérêt pour mes recherches,

Maurice Gross, qui a la grande gentillesse de faire partie du jury de cette thèse,

Martin Kay, qui a accepté d'apporter un point de vue d'Outre-Atlantique à ce jury,

Norbert Cot, qui a bien voulu porter un jugement bienveillant sur mon travail,

Daniel Kayser, dont les remarques critiques m'ont été d'un grand enrichissement et m'ont permis d'améliorer la présentation de ma thèse,

Hélène Bestougeff, enfin, dont les conseils toujours judicieux, l'aide efficace et le dynamisme communicatif m'ont permis de venir à bout de cette thèse.

1. Introduction

L'objectif de ce travail a été la construction de modèles formels et algorithmiques pour la **représentation** de données textuelles et de connaissances et pour leur **interrogation**. Plus précisément, il s'est agi de définir un formalisme qui permette de rendre compte des **connaissances sous-jacentes** à des textes (phrases ou suites de phrases).

Les critères qui ont présidé aux choix effectués ont été les suivants:

- représenter différents types de connaissances,
- définir très précisément les objets servant à représenter les connaissances,
- obtenir des procédures d'interrogation, c'est-à-dire d'extraction de données en fonction de questions, qui soient efficaces. Représentation et interrogation sont par conséquent deux processus indissociables.

Cela nous a conduit à proposer des modèles qui combinent à des réseaux sémantiques des bases de connaissances fondées sur des règles. Ces modèles sont décrits très précisément à l'aide de concepts de la théorie des graphes ou des grammaires de graphes.

1.1 Représentation de données textuelles ou représentation de connaissances

Il est très difficile de distinguer ce que l'on entend par représentation de données textuelles de ce que l'on entend par représentation de connaissances.

En premier lieu parce que les divers modèles de représentation de données textuelles se placent à un certain niveau de "profondeur", c'est-à-dire qu'ils s'éloignent plus ou moins de l'apparence superficielle du texte. Or, on peut dire que plus un modèle donne une représentation profonde, plus il se rapproche d'un modèle de représentation des connaissances.

L'utilisation des modèles les plus purement syntaxiques, et donc les plus superficiels, a été en vogue dans les années 60 et au début des années 70. Citons les grammaires indépendantes du contexte de [Chomsky, 57] utilisées par [Veillon, 70], les grammaires de dépendances de [Tesnière, 59] adaptées par [Courtin, 77], ou encore les grammaires catégorielles de [Bar Hillel, 70]. Ces modèles ont montré leurs limites, y compris pour des tâches telles que la traduction assistée par ordinateur.

Dès que l'on passe aux grammaires transformationnelles et que l'on admet une certaine équivalence entre

le chat mange la souris

et *la souris est mangée par le chat,*

on fait un pas vers la représentation des connaissances sous-jacentes au texte. On s'approche encore plus de la représentation des connaissances avec les grammaires casuelles de [Fillmore, 68] utilisées par exemple par [Sabah, 81], le maximum de profondeur étant atteint, pourrait-on dire, avec les travaux de Schank [Schank, 75] et sa décomposition des verbes selon des primitives sémantiques.

On s'aperçoit alors que le passage vers une représentation des connaissances s'accompagne d'une double **perte**. Perte d'abord parce que certaines nuances reflétées par la manière dont le texte est construit sont négligées. Il n'est par exemple pas indifférent qu'une phrase en français soit à l'actif ou au passif. Et il est clair qu'il y a un appauvrissement quand on traduit:

seriez vous assez aimable pour me donner le nom de l'écrivain qui a obtenu le prix Nobel de littérature en 1986 ?

par: *est-prix-nobel(x,littérature,1986)*

Perte également parce qu'on choisit une interprétation du texte, et donc qu'on en élimine un certain nombre. Cette interprétation étant liée dans certains cas à un contexte socio-culturel, comme chez Schank où il est déduit par exemple du fait qu'un individu coupe de l'herbe qu'il utilise une tondeuse à gazon.

En deuxième lieu, on ne peut jamais obtenir une représentation de connaissances qui soit "pure", indépendante de toute représentation de données textuelles. Représenter les connaissances sous-jacentes à un texte, c'est traduire ce texte en un certain langage. Langage qui, même s'il est formel, ne peut se dégager entièrement de la langue sur laquelle il repose. Dans tout système de représentation des connaissances on garde des unités lexicales qui appartiennent à une langue.

Nous parlerons de représentation de connaissances (par opposition à représentation de données textuelles) quand le formalisme de représentation se prêtera à des **déductions.** Autrement dit si à partir de plusieurs connaissances on pourra construire une nouvelle connaissance, c'est-à-dire une connaissance non explicitement formulée à l'origine.

A contrario, on peut dire qu'il n'y a pas représentation des connaissances lorsque l'interrogation ne permet d'obtenir en réponse à des questions que la restitution, sous une forme éventuellement modifiée en surface, de données qui ont été fournies antérieurement.

1.2 Les systèmes de représentation des connaissances

Très grossièrement, on peut séparer les systèmes de représentation des connaissances en deux catégories: d'une part les systèmes fondés sur des logiques, dont le prototype est PROLOG [Colmerauer, 77], d'autre part les réseaux sémantiques.

Dans la première catégorie se placent notamment les systèmes fondés sur les logiques non monotones [McDermott-Doyle, 80], les logiques modales, les logiques floues [Zadeh, 83], mais également la plupart des bases de connaissances des systèmes experts (voir [Laurière, 82]). Ces systèmes ont en commun d'être constitués d'un nombre fini de règles, en principe indépendantes, qui permettent de produire des connaissances en nombre illimité. Répondre à une question, cela revient à démontrer un théorème.

Un défaut de ces systèmes est qu'ils mettent toutes les règles au même niveau, ce qui fait qu'il est aussi difficile d'extraire une connaissance élémentaire qu'une connaissance plus complexe: difficile au sens du procédé mis en oeuvre aussi bien que de la combinatoire qui en résulte.

D'où l'intérêt des réseaux sémantiques dans lesquels les connaissances concernant un objet donné sont regroupées. Il en résulte un gain d'efficacité, certaines déductions étant effectuées très rapidement.

Les réseaux sémantiques ont été introduits en 1968 par M.R. Quillian [Quillian, 68]. De nombreux autres réseaux ont été définis par la suite (par exemple [Hendrix, 79], [Bobrow-Winograd, 77], [Shapiro, 79], [Brachman-Schmolze, 84]) sans que l'on puisse toujours très bien savoir ce qu'ils devaient à leurs prédécesseurs, les différents systèmes se développant de manière concurrente. Aux réseaux sémantiques on peut également rattacher les "frames" [Minsky, 75] qui structurent la connaissance d'une manière analogue aux réseaux, ou même les représentations à la Schank [Schank, 75].

Malgré leur diversité, on peut relever des traits communs à tous les réseaux, traits qui permettent de classer un modèle parmi les réseaux ou pas: c'est l'existence de noeuds (ou sommets) étiquetés qui représentent des objets ou des propositions et de liens (ou arcs) également étiquetés joignant entre eux ces sommets et servant à marquer des relations entre les objets représentés par les sommets. Les arcs, comme les sommets, peuvent être de natures diverses.

Dans de nombreux réseaux on retrouve une structuration classificatoire de connaissances du type:

tout éléphant est un mammifère

tout mammifère est un animal

sous la forme d'une structure hiérarchique, c'est-à-dire d'un arbre ou d'un graphe sans circuit, dans lequel les arcs marquent une relation appelée "IS-A" (*is an instance of*) ou "AKO" (*a kind of*).

En 1979 S.E. Fahlman a émis l'idée que des machines parallèles permettraient de répondre très rapidement à des questions par le déplacement de marqueurs booléens à travers un graphe qui devait être sans circuit [Fahlman, 79]. Son modèle est bâti autour d'une hiérarchie d'objets typiques et individuels et il permet d'envisager des exceptions aux règles générales, c'est-à-dire d'effectuer des déductions non monotones.

Le principal défaut des réseaux sémantiques, c'est leur absence de statut, ceci selon deux points de vue. D'une part, on ne sait pas toujours très bien ce que représentent les noeuds, les liens, quelle est leur signification. Il semble que dans certains systèmes on puisse rajouter à la demande de nouvelles sortes d'arcs ou de noeuds qui ont une nouvelle signification. Cela explique la parution d'un certain nombre d'articles ([Woods, 75], [Brachman, 79]) dont l'objectif était d'essayer justement de déterminer le statut de ces réseaux.

D'autre part, et cela découle en partie de ce qui précède, l'objet réseau sémantique n'est pas vraiment défini, on ne peut en cerner les limites. A quelques exceptions près ([McSkimin-Minker, 79], [Sowa, 84]), on ne sait pas précisément ce qu'est un réseau.

Ce n'est que dans un deuxième temps que l'on a ébauché des descriptions explicites des objets manipulés. La démarche a été exactement inverse de celle qui a présidé à l'utilisation de la logique. La logique avait un statut avant de servir à représenter des connaissances. Alors qu'avec les réseaux sémantiques on est parti de l'intuition, de la possibilité de réaliser des programmes relativement performants, et ensuite on a essayé de décrire précisément ce qu'on faisait.

Ainsi, ce n'est que plusieurs années après que Fahlman ait conçu son système que des formalisations, fondées sur les logiques, sont apparues ([Etherington-Reiter, 83], [Touretzky, 86], [Froidevaux, 85]). D.S. Touretzky, par exemple, justifie son travail de la manière suivante:

"Its primary aim is to present a formal mathematical theory of a popular reasoning strategy that to date has been defended mostly by appeals to intuition (...) Until recently, despite their widespread use, inheritance systems with exceptions remained unformalized; the lack of a formal theory hid some defects in the behavior of existing systems."

Toutefois, les formalisations qui ont été proposées des réseaux de [Fahlman, 79] ne portent que sur une toute petite partie de ces réseaux, uniquement sur la structure classificatoire. Ainsi, les propositions ne sont pas traitées. Il n'est pas non plus indiqué comment manipuler les objets typiques complexes, c'est-à-dire définis à partir de la combinaison de notions (noms ou adjectifs).

Les modèles que nous proposons ne permettent pas de traiter autant de connaissances que prétend en traiter Fahlman (ou d'autres). Mais ils sont définis pas à pas et très précisément. Et ils s'étendent à une plus large classe de connaissances que les modèles formels apparus récemment.

1.3 Trois modèles de représentation et d'interrogation

Nous proposons **trois** modèles de représentation et d'interrogation qui s'articulent mais que l'on peut décrire séparément.

1°/ Un modèle très proche de la surface, qui ne permet aucune déduction. Une caractéristique de ce modèle, c'est l'effacement de certains pronoms et conjonctions. Soit par exemple à représenter le texte suivant:

Jean croit qu'il aime Marie

Il faut d'abord savoir à quoi se rapporte le pronom *il*, à *Jean* ou à autre chose. Disons très clairement que dans le travail qui est présenté ici nous ne tentons absolument pas de développer des méthodes de résolution d'un tel problème. Nous supposons le problème résolu, en étant tout à fait conscient qu'il s'agit d'un problème d'une très grande difficulté.

Si l'on ne conserve que l'interprétation de ce texte selon laquelle le pronom se rapporte à *Jean*, il n'est pas nécessaire d'avoir deux entités distinctes dont l'une représenterait *Jean* et l'autre représenterait *il*. Cela entraîne que le texte n'est plus représenté par un simple arbre, mais par un graphe sans circuit

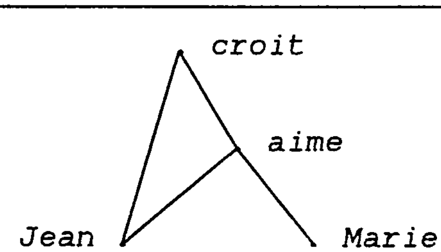


figure 1.1

plus général, comme on le voit en figure 1.1.

Plus exactement, la structure obtenue est un graphe ordonné, dans la mesure où l'ordre entre les successeurs d'un même sommet est significatif. Il n'est pas indifférent de savoir si c'est *Jean qui aime Marie* ou si c'est *Marie qui aime Jean*. D'autre part les sommets du graphe reçoivent plusieurs étiquettes, comme on le verra dans le chapitre 4. C'est pourquoi nous parlerons de graphe ordonné sans circuit multiétiqueté, ou GOSME.

Ce modèle, s'inspirant des représentations de A. Culioli [Culioli-Desclés, 75], permet d'insérer les structures prédicatives dans des conditions énonciatives. Ainsi on peut représenter

Jean a dit hier qu'il aimait Marie

par le graphe de la figure 1.2.

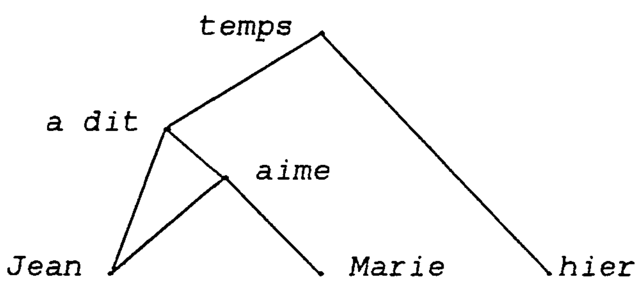


figure 1.2

Notons bien que cette représentation est très simpliste et qu'elle néglige un grand nombre de difficultés linguistiques. Mais elle nous permet de montrer quel type d'objet on utilise pour la représentation. Et d'indiquer que pour nous la réponse à une question n'est pas forcément positive ou négative. Par exemple, si l'on pose la question

est-ce que Jean aime Marie ?

le système détecte que la structure prédicative *Jean aime Marie* se trouve dans les données et il la restitue à l'interrogateur, accompagnée des conditions dans lesquelles elle est insérée.

2°/ Un modèle qui permet d'effectuer des déductions élémentaires. Les graphes sans circuit représentant des propositions sont intriqués avec des structures hiérarchiques - qui sont aussi des graphes sans circuit - exprimant l'héritage de propriétés entre objets typiques et objets individuels. Ainsi pourra-t-on déduire des connaissances:

le chat typique déteste le chien typique

Mitsou est un chat

Médor est un chien

représentées dans le graphe de la figure 1.3, que:

Mitsou déteste Médor

à moins qu'une connaissance particulière du genre

Pussy est un chat qui ne déteste pas Médor

vienne infirmer cette connaissance générale. Cela signifie que ce modèle admet une forme de raisonnement non monotone, forme qui est semblable à celle de [Touretzky, 86], mais plus générale.

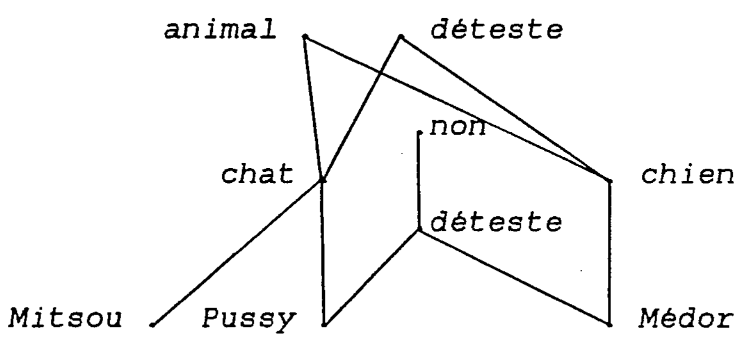


figure 1.3

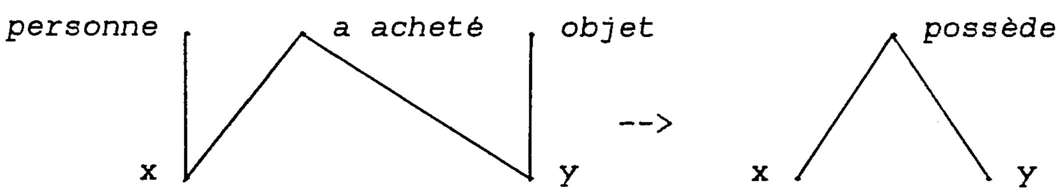
Dans une telle structure, tous les sommets n'ont pas le même statut. Ainsi, si l'ordre entre les successeurs d'un sommet représentant des propositions a de l'importance, l'ordre entre les successeurs d'un sommet représentant un objet typique n'a aucune importance. C'est pourquoi, les réseaux seront pour nous des graphes semi-ordonnés multiétiquetés.

Nous explicitons très précisément le processus de propagation de marqueurs qui permet de répondre à des questions, y compris dans le cas où des objets typiques complexes sont en cause, ou bien dans le cas où il y a des exceptions aux règles générales.

3°/ Un modèle qui permet d'effectuer des déductions plus complexes, en faisant appel à des règles lorsque c'est nécessaire. Par exemple, on ne peut représenter dans un réseau la connaissance:

si une personne a acheté un objet, alors elle possède cet objet

Mais, comme nous voulons ne faire appel à des règles que lorsque c'est indispensable, il faut que ces règles puissent se combiner avec les réseaux. C'est pourquoi les règles sont des règles de réécriture de graphes semi-ordonnés, ou règles de réécriture de réseaux. La connaissance ci-dessus est alors représentée par la règle schématisée en figure 1.4.



---------------------------figure 1.4---------------------------

L'interrogation consiste, pour le premier comme pour le deuxième modèle, à chercher dans le graphe représentatif des connaissances une ou plusieurs sous-structures qui soient à l'image de la question, elle-même représentée par un graphe. Ceci est formalisé en termes de morphismes de graphes. Un morphisme nous sera particulièrement utile: l'**agrandissement** de graphes (voir chapitre 2).

Dans le cas du troisième modèle, les questions sont encore représentées par des graphes, mais l'application des règles de réécriture permet de transformer une question en de nouvelles questions avant de chercher les réponses.

La réponse à une question peut, en définitive, soit être trouvée directement dans le réseau représentatif des connaissances, soit faire appel aux règles. L'interrogation est ainsi "à deux vitesses", et elle combine les avantages des deux méthodes de représentation des connaissances.

Nous proposons également des outils qui permettent, quel que soit le mode d'analyse syntaxique, de construire et de mettre à jour un réseau à partir de connaissances exprimées en un langage quasi-naturel. Ces outils sont encore à base de règles de réécriture de graphes.

1.4 Informatique et linguistique

Traitant des connaissances qui, au moins dans un premier temps, sont exprimées en langue naturelle, nous sommes obligés de définir notre place par rapport à la linguistique. Disons tout de suite nettement que si (1) nous ne prétendons en aucune façon faire oeuvre de linguiste, (2) nous ne prétendons pas pour autant ignorer la linguistique et les linguistes.

(1) Nous ne faisons pas oeuvre de linguiste: notre objectif ne consiste en effet pas à construire un modèle susceptible de décrire toute une langue ou à étudier un phénomène propre aux langues naturelles, tel que la détermination, la modalité ou le temps. Nous cherchons à définir des outils qui permettront de répondre à des objectifs finalisés, sur des ensembles de corpus d'un même type et dans des conditions de production d'énoncés bien délimitées.

(2) Cependant, les linguistes nous ayant précédé dans la représentation de données textuelles, et par conséquent dans la représentation des connaissances, les réflexions qu'ils ont déjà faites sont susceptibles de nous servir. Il serait stupide de croire s'avancer en terrain entièrement vierge. Toutefois, il y a des difficultés très importantes à l'adaptation des travaux linguistiques à des fins informatiques. Et ceci du fait même des différences de perspectives.

Les linguistes, en effet, s'attachent à l'étude de phénomènes très fins du langage, ne proposant pas de modèles achevés qui aient l'adhésion de l'ensemble de leur communauté scientifique. La linguistique progresse par la remise en question successive des théories existantes. Plusieurs écoles de pensées, qui sont parfois en désaccord sur des points essentiels, se développent simultanément.

L'informaticien ne sait donc pas où il doit chercher la "vérité linguistique". Il ne peut attendre que la linguistique résolve les problèmes qu'il rencontre avant de prendre ses décisions. Il entretient avec la linguistique un peu les mêmes rapports que le médecin avec la biologie. C'est dans une certaine ignorance qu'il est obligé d'agir.

Les enseignants en langues, langues maternelles ou langues étrangères, doivent également s'accommoder de l'état actuel de la linguistique dans leur pratique concrète. E. Genouvrier et J. Peytard donnent à une étude sur l'apport de la linguistique à l'enseignement du français [Genouvrier-Peytard, 1970] une conclusion que nous pouvons reprendre à notre compte en l'appliquant à l'informatique:

"Le praticien n'a pas à attendre du linguiste une vérité définitive, mais qui peut prétendre en disposer ? N'est-il pas précieux de déjà pouvoir demander aux vérités provisoires une efficacité sur le monde - en l'occurrence à celles de la linguistique sur la conquête par l'enfant de sa langue maternelle ?"

De même, les vérités provisoires de la linguistique contribuent à la réalisation de programmes de traitement informatique de la langue naturelle. Toutefois, et nous allons le voir, l'action de la linguistique sur l'activité de programmation n'est pas sans poser des problèmes.

Les systèmes de représentations des linguistes - sauf s'ils ont été réalisés dans cette perspective - ne sont pas adaptés à une utilisation immédiate par l'informatique.

Quand un linguiste dit qu'il a présenté ses réflexions de manière "formelle", il n'a en général fait qu'employer un symbolisme abstrait afin de schématiser un phénomène. Les termes mathématiques sont utilisés sans avoir reçu de définition mathématique, il en est fait plutôt un usage métaphorique. Ainsi la notion de "règle" vue par le linguiste est-elle très éloignée de celle de l'informaticien. Le critère de validité d'une théorie linguistique n'est ni son aptitude à être immédiatement informatisable, ni sa "propreté" mathématique.

Afin de parvenir, malgré ces difficultés, à des résultats acceptables, les informaticiens ont eu diverses attitudes. Certains, comme [Bonnet, 80], ont ignoré complètement la linguistique et sont parvenus, par des approches pragmatiques, à des réalisations qui les ont satisfaits. Le défaut de leurs travaux, c'est qu'ils ne sont nullement généralisables. D'autres ont recherché dans les théories linguistiques disponibles celle qui leur paraissait leur convenir le mieux. Le critère dans ce cas a plus été celui de l'adéquation des théories linguistiques aux contraintes du traitement informatique que l'adéquation de ces théories à la description des langues.

Quoi qu'il en soit, l'informaticien garde sa liberté par rapport à la théorie linguistique dont il s'inspire. Il construit ses propres outils qui peut-être n'ont plus que très peu à voir avec la pensée originelle du linguiste. Nous verrons dans le chapitre 4 comment nous reconstruisons, en nous inspirant notamment des représentations de A. Culioli, un système entièrement autonome et qui se justifie par des considérations uniquement informatiques.

Mais, une fois construits les outils, tout n'est pas terminé pour autant. On ne peut réaliser un système complet qui fonctionne sur une langue sans faire appel à des connaissances linguistiques précises. Il est nécessaire de faire agir ces outils sur des données fournies par l'expert en langues qu'est le linguiste.

Les structures mathématiques et informatiques que nous manipulons sont parfaitement définies. Mais cela ne veut pas dire que nous savons quelle structure doit être associée à n'importe quel énoncé de la langue naturelle. En l'absence de réponses précises, nous avons dû effectuer des choix provisoires dans le seul but de donner une illustration du fonctionnement de nos outils.

Nous admettons que les choix que nous avons effectués sont très souvent discutables d'un point de vue linguistique.

1.5 Pourquoi une approche systématique

Nous pensons qu'une approche systématique de problèmes tels que celui de la représentation des connaissances se doit de bien séparer les étapes du travail. Plus précisément, nous distinguons quatre étapes:

(1) Une étape où l'on définit les objets que l'on manipule. Une définition mathématique est le prix à payer d'une connaissance sans équivoque de ces objets, de la possibilité de les comparer à d'autres objets. Il est nécessaire que cette étape puisse être considérée comme entièrement indépendante des autres étapes. Ce qui ne veut bien entendu pas dire que le choix des objets à définir ne dépend pas de ce qu'on veut en faire.

C'est pourquoi, dans ce qui suit, nous donnons pas à pas les définitions mathématiques précises des objets que nous manipulons - en utilisant le formalisme de la théorie des graphes - et de leurs manipulations - en se servant de grammaires de graphes.

(2) Une étape où l'on indique comment se servir des définitions de (1) afin de représenter des connaissances ou des données textuelles, comment formaliser l'interrogation.

Il y a alors forcément un certain appel à l'intuition, intuition qui est étayée par les travaux linguistiques - ou, pour certains auteurs, par les travaux de la psychologie cognitive -, avec les difficultés que nous avons signalées.

A l'issue de cette étape on dispose de structures représentatives de données textuelles ou de connaissances qui sont des sous-classes des objets définis en (1) et dont on peut oublier, pour les traitements ultérieurs, les raisons intuitives pour lesquelles on les a construites.

Il ne faut pas que des changements éventuels dans la partie (2) fassent sortir de la classe des objets définis en (1).

(3) Une étape algorithmique: l'objectif de l'informaticien est d'avoir la possibilité d'effectuer des calculs sur des objets. La représentation ne peut être une fin en soi.

Si les objets ont été définis précisément, les algorithmes peuvent être décrits très rigoureusement.

(4) Une étape où l'on passe à la programmation, c'est-à-dire où l'on se place dans des conditions concrètes, en représentant par exemple certaines connaissances ayant trait à un domaine limité, où on choisit un langage de programmation dans lequel on traduit les algorithmes.

Il est clair que ce sont les perspectives du traitement en machine qui guident les choix effectués en (1), (2) et (3). Et d'une certaine manière, c'est le traitement en machine qui atteste de la validité de ces choix. D'où la nécessité d'un retour vers la pratique.

C'est dans le cadre de ce retour à la pratique que nous avons réalisé un programme expérimental qui ne prend en compte qu'une partie du modèle que nous définissons. Il est clair que nous ne cherchons pas à aller aussi loin dans la réalisation que des entreprises industrielles peuvent le faire. Mais nous avons la prétention de décrire précisément les chemins que nous empruntons.

La présentation de notre travail s'articule de la manière suivante:

Dans les chapitres 2 et 3 nous décrivons très explicitement les outils qui nous sont utiles: graphes ordonnés sans circuit multi-étiquetés (GOSME) pour la représentation des données textuelles, graphes semi-ordonnés sans circuit multi-étiquetés ou réseaux pour la représentation des connaissances élémentaires (chapitre 2), puis grammaires de graphes sans circuits (chapitre 3) qui serviront à la représentation de connaissances plus complexes, mais également à la construction des structures représentatives et à leur mise à jour.

Dans le chapitre 4 nous indiquons comment représenter les données textuelles et les connaissances, élémentaires ou plus complexes. Il ne s'agit pas de décrire les processus de traduction, mais de montrer, sur un certain nombre d'exemples,

les résultats auxquels on veut parvenir.

Le chapitre 5 traite de l'interrogation, donnant une formalisation mathématique de cette interrogation, ainsi que la description d'algorithmes, sous forme classique ou sous forme d'un déplacement en parallèle de marqueurs booléens.

Dans le chapitre 6 nous montrons comment construire et mettre à jour un réseau représentatif de connaissances élémentaires à partir de textes en langage quasi-naturel.

Dans le chapitre 7, enfin, nous donnons une description rapide du système expérimental que nous avons construit.

Principales notations utilisées

L'ensemble des éléments x appartenant à X qui vérifient la propriété P est noté

$$\{ x \in X \; ; \; P(x) \}$$

$X \backslash Y$ est l'ensemble des éléments de X qui ne sont pas dans Y:

$$X \backslash Y = \{x \in X \; ; \; x \notin Y\}$$

Si X est un ensemble, $\mathcal{P}(X)$ est l'ensemble des parties de X.

Soit $f : X \dashrightarrow Y$ une application. Si A est un sous-ensemble de X, f(A) désigne l'ensemble composé de tous les éléments f(x) pour $x \in A$.

$$f(A) = \{y \in Y \; ; \; \exists \; x \in A \quad f(x) = y\}$$

La composée de deux applications $f : X \dashrightarrow Y$ et $g : Y \dashrightarrow Z$ est notée gf.

$\mathbb{N}$ est l'ensemble des entiers naturels.
$\mathbb{N}^{+}$ est l'ensemble des entiers strictement positifs.

[n] est l'ensemble {1,2,...,n} si $n \neq 0$.
C'est l'ensemble vide sinon.

Etant donné un ensemble X, le monoïde libre engendré par X sera noté X^{*}.

$|\mu|$ est la longueur du mot μ de X^{*} .

λ est le mot vide du monoïde libre.

X^{+} est l'ensemble des mots de longueur strictement positive de X^{*}. On a par conséquent:

$$X^{+} = X^{*} \backslash \{\lambda\}$$

Si $\alpha = x_1...x_p$ appartient à un monoïde libre, x_i pourra encore être noté $\alpha^{(i)}$.

Si f : Y --> X^* est une application d'un ensemble quelconque dans un monoïde libre, et si l'on a $f(y) = x_1...x_p$,
x_i pourra encore être noté $f^{(i)}(y)$.

2 Les structures de base

Dans ce chapitre on se sert du formalisme de la théorie des graphes afin de définir des outils qui trouveront leur utilisation dans les chapitres suivants.

On procède par étapes afin de définir les deux structures les plus importantes pour la suite: les Σ-GOSME et les réseaux.

On rappelle les définitions de base sur les graphes (2.1), puis on introduit les graphes ordonnés (ou GO), c'est-à-dire tels que les successeurs d'un même sommet soient ordonnés (2.4), les graphes étiquetés, ou GE (2.5), et les graphes multi-étiquetés, ou GME (2.5.4), en s'intéressant spécialement aux graphes sans circuit. Ainsi parvient-on à la définition des GOSME, ou graphes ordonnés sans circuit multi-étiquetés (2.5.6).

A partir de de cette structure de base, on peut définir deux sortes de structures:
(1) on associe des opérateurs aux sommets du graphe avec un multi-étiquetage afin de définir les Σ-GOSME (2.7.4), structure que l'on peut mettre en liaison avec les Σ-algèbres.
(2) On définit les réseaux, d'une part en étiquetant les arcs des graphes (2.5.7), d'autre part en définissant une relation d'équivalence entre graphes ordonnés étiquetés de telle manière que seul l'ordre entre successeurs de certains sommets soit pertinent, introduisant ainsi les graphes semi-ordonnés (2.6).

Notre objectif est de représenter des données textuelles (ou des connaissances) à l'aide de graphes. Dans ce cadre, il est intéresant de pouvoir caractériser des analogies entre les textes représentés en observant des ressemblances entre les structures représentatives. Ces ressemblances s'exprimeront en termes de morphismes. Et, bien entendu, il y aura des analogies plus ou moins fortes entre données textuelles. C'est pourquoi nous introduisons une grande variété de morphismes sur les graphes.

Dans un ordre croissant de généralité nous considérons les morphismes de graphe, les P-morphismes, les Q-morphismes, les R-morphismes et les agrandissements. Ces derniers morphismes ont la particularité de transformer tout arc en un chemin.

Ces morphismes sont d'abord introduits sur les graphes les plus généraux (2.2). On précise les particularités de ces morphismes pour les structures introduites par la suite au fur et à mesure de l'introduction de ces structures.

Notons les QR-morphismes, définis uniquement pour les réseaux, qui sont intermédiaires entre les Q-morphismes et les R-morphismes (2.6.5).

Trouver la réponse à une question, cela consistera à extraire une sous-structure d'une structure représentative. C'est pourquoi il est accordé une grande importance aux sous-structures des graphes.

Outre les sous-graphes partiels et les sous-graphes qui sont des sous-structures bien connues, on introduit les sous-graphes presques totaux, les sous-graphes propres et les sous-graphes totaux.

De même que pous les morphismes, les sous-structures sont introduites d'abord pour les graphes généraux (2.3).

2.1 <u>Définitions préliminaires sur les graphes</u>

Les définitions classiques sur les graphes sont tirées de [Berge,70] ou de [Harary,69].

2.1.1 Un **graphe** G est défini comme étant un couple $<X,\gamma>$ où X est un ensemble (de **sommets**) et γ une application de X dans $\mathcal{P}(X)$.

Comme on le voit, on ne prend en compte que les 1-graphes (au sens de [Berge,70]). De plus, dans tout ce qui suit, on ne s'intéressera qu'aux graphes finis.

Pour tout sommet x de X, $\gamma(x)$ est **l'ensemble des successeurs** de x.

2.1.2 **L'ensemble des sommets terminaux** de G est constitué par tous les sommets x de X tels que $\gamma(x)$ soit vide.

On note $\quad Ter(G) = \{x \in X \ ; \ \gamma(x) = \emptyset\}$

2.1.3 **L'ensemble des sommets initiaux** de G est constitué par tous les sommets x de X qui ne sont successeurs d'aucun sommet.

On note $\mathrm{Ini}(G) = \{x \in X \; ; \; \forall y \in X \quad x \notin \gamma(y)\}$

2.1.4 **L'ensemble des descendants** d'un sommet donné x, noté $\hat{\gamma}(x)$ est défini comme suit:

(i) $x \in \hat{\gamma}(x)$

(ii) $\forall y,z \in X \quad y \in \hat{\gamma}(x)$ et $z \in \gamma(y) \Longrightarrow z \in \hat{\gamma}(x)$

<u>Propriété</u>: L'ensemble des successeurs d'un sommet est inclus dans l'ensemble des descendants de ce sommet.

L'ensemble des descendants **au sens strict** d'un sommet donné x, noté $\ddot{\gamma}(x)$ est défini comme suit

$\ddot{\gamma}(x) = \hat{\gamma}\gamma(x)$

L'ensemble **ascendants immédiats** d'un sommet donné x, noté $\gamma^{-1}(x)$, est défini comme suit:

$\gamma^{-1}(x) = \{y \in X \; ; \; x \in \gamma(y)\}$

2.1.5 Tout couple $<x,y>$ de sommets tel que $y \in \gamma(x)$ est un **arc** du graphe.

x en est l'**origine**, y l'**extrémité**.

Si x = y, l'arc est une **boucle**.

2.1.6 Toute suite $<x_0,x_1,\ldots,x_p>$ de sommets vérifiant

$\forall i \in [p] \quad x_i \in \gamma(x_{i-1})$

est un **chemin** du graphe.

Le chemin a pour **origine** x_0 et pour **extrémité** x_p.

p est la **longueur** du chemin.

<u>Propriété</u>: Il existe un chemin d'origine x et d'extrémité y si et seulement si y appartient à $\hat{\gamma}(x)$.

<u>Remarque</u>: Tout chemin $<x_0,x_1,\ldots,x_p>$ est tel que:

$\forall\ i,j \in \{0,1,\ldots p\} \quad i \leqslant j \implies x_j \in \hat{\gamma}(x_i)$

2.1.7 Tout chemin $\langle x_0,x_1,\ldots,x_p\rangle$ de longueur au moins égale à 1 et où $x_0 = x_p$ est un **circuit** du graphe.

Propriété: Tout couple $\langle x_i,x_j\rangle$ de sommets qui appartiennent à un même circuit est tel que $x_i \in \hat{\gamma}(x_j)$ et $x_j \in \hat{\gamma}(x_i)$.

Preuve: On a $x_p \in \hat{\gamma}(x_i)$ et $x_j \in \hat{\gamma}(x_0)$.
Comme $x_0 = x_p$,
$x_j \in \hat{\gamma}(x_i)$ et de même $x_i \in \hat{\gamma}(x_j)$.

Réciproque: Deux sommets distincts x et y qui vérifient $x \in \hat{\gamma}(y)$ et $y \in \hat{\gamma}(x)$ appartiennent à un même circuit.

Preuve: Il existe un chemin d'origine x et d'extrémité x qui est de longueur au moins égale à 1 puisqu'il passe par y, distinct de x.

Conséquence: Un graphe est donc **sans circuit** si et seulement si

(i) $\quad \forall\ x \in X \quad x \notin \gamma(x)$

(ii) $\quad \forall\ x,y \in X \quad x \in \hat{\gamma}(y)$ et $y \in \hat{\gamma}(x) \implies x = y$

2.1.8 Propriété: Si un graphe $G = \langle X,\gamma\rangle$ fini non vide est sans circuit, les ensembles Ini(G) et Ter(G) sont non vides.

Preuve: Si Ter(G) était vide, chaque sommet x_0 de X aurait au moins un successeur. Il en résulterait l'existence d'une suite infinie $x_0,x_1,\ldots,x_p,\ldots$ pour laquelle

$$\forall\ i \in \mathbb{N} \quad x_{i+1} \in \gamma(x_i)$$

Comme X est fini, nécessairement deux éléments de cette suite sont identiques, ce qui entraîne l'existence d'un circuit.

Si Ini(G) était vide, cela impliquerait l'existence d'une suite infinie $x_0,x_1,\ldots,x_p,\ldots$ pour laquelle

$$\forall\ i \in \mathbb{N} \quad x_i \in \gamma(x_{i+1})$$

et donc l'existence d'un circuit.

2.1.9 Si $G = \langle X,\gamma \rangle$ est un graphe sans circuit, le **rang** $r(x)$ de tout sommet x de G est la longueur maximale des chemins ayant pour extrémité x.

Le **rang** $r(G)$ **du graphe sans circuit** est le maximum des $r(x)$, $x \in X$.

2.2 Morphismes

Soient $G = \langle X,\gamma \rangle$ et $G' = \langle X',\gamma' \rangle$ deux graphes et $f : X \dashrightarrow X'$

une application.

2.2.1 Définition: f est un **morphisme de graphes** si et seulement si

$$\forall x \in X \qquad f\gamma(x) = \gamma' f(x)$$

Propriété: La composée de deux morphismes de graphes est un morphisme de graphes.

Preuve: Immédiate.

2.2.2 Définition: f est un **Q-morphisme** de graphes si et seulement si

$$\forall x \in X \qquad \gamma(x) \neq \emptyset \Longrightarrow f\gamma(x) = \gamma' f(x)$$

Exemples

Tous nos exemples de graphes ou de graphes étiquetés seront présentés sous forme de tableaux et de schémas, éventuellement uniquement sous forme de schémas, toutes les étiquettes n'étant pas forcément portées sur chaque schéma.

Deux exemples de Q-morphismes sont donnés par les figures 2.1 et 2.2.

2.2.3 Propriété: Si f est un Q-morphisme de graphes, l'image de tout sommet non terminal de G est un sommet non terminal de G'.

Preuve: Immédiate.

Propriété: La composée de deux Q-morphismes est un Q-morphisme.

x	a
$\gamma(x)$	$\emptyset$

x	a
$f(x)$	a'

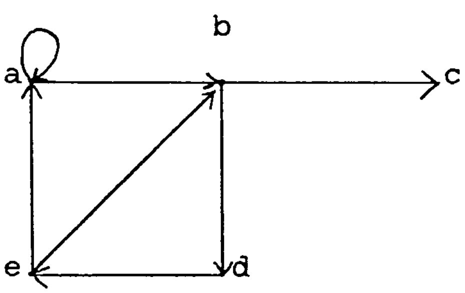
x	a'
$\gamma'(x)$	{a'}

a. a'.

figure 2.1

x	a	b	c	d	e
$\gamma(x)$	{a,b}	{c,d}	$\emptyset$	{e}	{a,b}
$f(x)$	a'	b'	c'	d'	a'

x	a'	b'	c'	d'	f'
$\gamma'(x)$	{a',b'}	{c',d'}	{d',f'}	{a'}	{d'}



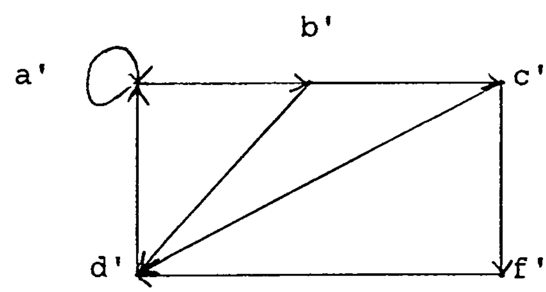


figure 2.2

Preuve: Découle immédiatement de la propriété précédente.

2.2.4 Propriété: Tout morphisme de graphes est un Q-morphisme.

Preuve: Immédiate.

2.2.5 Définition: f est un **P-morphisme** de graphes si et seulement si c'est un Q-morphisme vérifiant:

$$\forall\, x \in X \quad \gamma(x) = \emptyset \;\Rightarrow\; \ddot{\gamma}'f(x) \cap f(X) = \emptyset$$

Autrement dit, l'image d'un sommet terminal de G ne peut avoir pour descendants au sens strict les images de sommets de X.

Notons que les deux Q-morphismes donnés précédemment en exemples ne sont pas des P-morphismes. Un exemple de P-morphisme est donné en figure 2.3.

x	a	b	c
$\gamma(x)$	{b,c}	{c}	$\emptyset$
f(x)	a'	b'	c'

x	a'	b'	c'	d'	e'
$\gamma'(x)$	{b',c'}	{c'}	{d',e'}	{e'}	{d'}

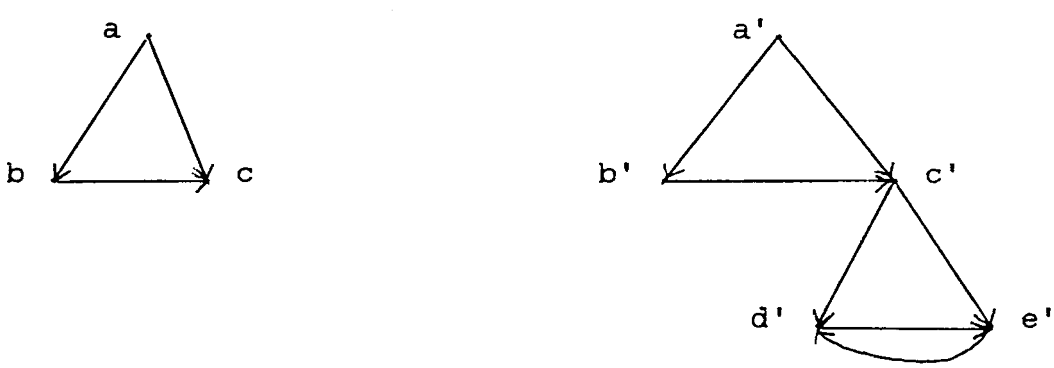


figure 2.3

Propriété: Si f est un P-morphisme, x et y deux sommets de G, alors tout chemin d'origine f(x) et d'extrémité f(y) est l'image d'un chemin d'origine x et d'extrémité y de même longueur.

Preuve: Soit $\langle x'_1,\dots,x'_p\rangle$ un chemin de G' avec $x'_1 = f(x)$ et $x'_p = f(y)$ et soit $\langle x'_1,\dots,x'_l\rangle$ le plus grand chemin possible extrait de ce chemin qui soit image d'un chemin $\langle x_1,\dots,x_l\rangle$ de G.

Si l est différent de p, x_l appartient à Ter(G). Dans le cas contraire, en effet, on aurait

$$f\gamma(x_1) = \gamma'(x_1')$$

et il existerait un x_{1+1} dont x_{1+1}' serait l'image et qui appartiendrait à $\gamma(x_1)$. $<x_1',\ldots,x_1'>$ ne serait pas le plus grand chemin possible.

Par conséquent:

$$\ddot{\gamma}'(x_1') \cap f(X) = \emptyset$$

et il ne peut y avoir de chemin entre x_1' et x_p'.

<u>Réciproque</u>: Un Q-morphisme tel que, pour tous x et y appartenant à X, tout chemin d'origine f(x) et d'extrémité f(y) est image d'un chemin d'origine x et d'extrémité y est un P-morphisme.

<u>Preuve</u>: Soit $x \in X$ tel que

$$\ddot{\gamma}'f(x) \cap f(X) \neq \emptyset \ .$$

Il existe nécessairement y' qui soit image par f d'un y de X et il existe un chemin d'origine f(x) et d'extrémité f(y) de longueur au moins égale à 1.

Il existe donc un chemin d'origine x et d'extrémité y, ce qui montre que x ne peut appartenir à Ter(G) et donc que f est un P-morphisme.

<u>Propriété</u>: La composée de deux P-morphismes est un P-morphisme.

<u>Preuve</u>: Découle immédiatement de la réciproque précédente.

2.2.6 <u>Définition</u>: f est un **R-morphisme** de graphes si et seulement si

$$\forall\ x \in X \qquad f\gamma(x) \subset \gamma'f(x)$$

<u>Propriété</u>: La condition nécessaire et suffisante pour que l'image de tout arc de G soit un arc de G' est que f soit un R-morphisme.

<u>Preuve</u>: La condition caractérisant les R-morphismes peut s'écrire:

$$\forall\ x \in X \quad \forall\ y \in \gamma(x) \qquad f(y) \in \gamma'f(x)$$

ou encore:

$$\forall\ x,y \in X \quad y \in \gamma(x) \Rightarrow f(y) \in \gamma' f(x)$$

ce qui est la condition qui exprime que l'image de tout arc de G est un arc de G'.

Propriété: La condition nécessaire et suffisante pour que l'image de tout chemin de G soit un chemin de G' est que f soit un R-morphisme.

Preuve: Immédiate.

Propriété: La composée de deux R-morphismes est un R-morphisme.

Preuve: Immédiate.

Propriété: Tout Q-morphisme de graphes est aussi un R-morphisme.

Preuve: Immédiate.

2.2.7 Définition f est un **agrandissement** de graphes si et seulement si

$$\forall\ x,y \in X \quad y \in \gamma(x) \Longrightarrow f(y) \in \hat{\gamma}' f(x)$$

Tout arc du graphe G a pour image un couple de sommets de G' reliés par un chemin. Comme ce chemin a éventuellement une longueur nulle, le mot "agrandissement" n'aura pas toujours la signification concrète qu'on peut lui attribuer intuitivement.

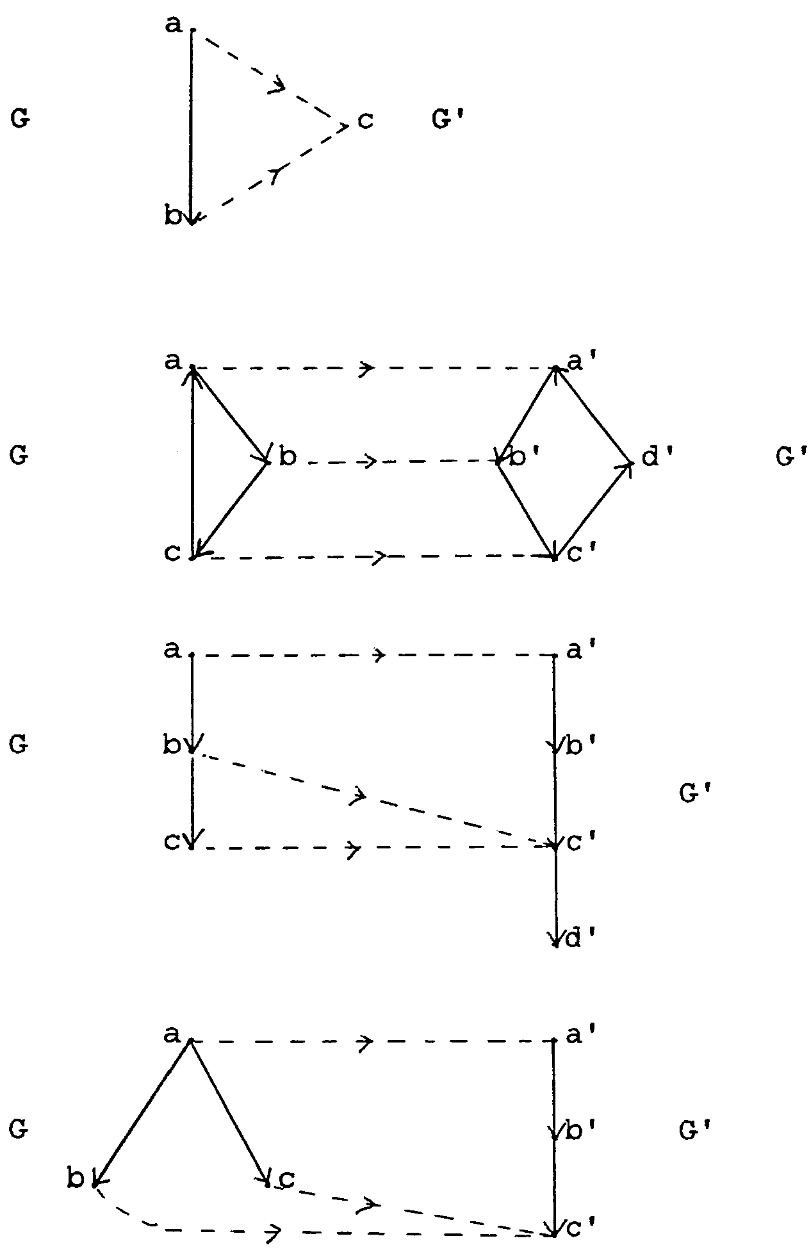
Plusieurs exemples d'agrandissements sont donnés en figure 2.4.

2.2.8 Propriété: Tout R-morphisme de graphes est aussi un agrandissement.

Preuve: Pour tout sommet x on a :

$$\forall\ y \in \gamma(x) \qquad f(y) \in \gamma' f(x)$$

et donc $f(y) \in \hat{\gamma}' f(x)$.



-------------------------<u>figure 2.4</u>--------------------------

2.2.9 <u>Propriété</u>: La condition nécessaire et suffisante pour que f soit un agrandissement est que pour tout x et tout y de X tels que $y \in \hat{\gamma}(x)$ on ait $f(y) \in \hat{\gamma}' f(x)$.

<u>Preuve</u>: Soit $\langle x_0, x_1, \dots, x_p, x_{p+1} \rangle$ une suite telle que $x_0 = x$, $x_{p+1} = y$ et

$$\forall\ i \in [p+1] \quad x_i \in \gamma(x_{i-1}) \quad .$$

On a nécessairement:

$\forall\ i \in [p+1] \quad f(x_i) \in \hat{\gamma}'f(x_{i-1})$
et donc, par transivité, $f(y) \in \hat{\gamma}'f(x)$.

Réciproquement,
$y \in \gamma(x) \implies y \in \hat{\gamma}(x) \implies f(y) \in \hat{\gamma}'f(x)$.

Remarque: Si l'application f est injective, à tout chemin (qui ne comporte pas de boucle) entre deux sommets x et y donnés, on pourra faire correspondre un chemin de longueur au moins égale entre f(x) et f(y) dans G'. Dans ce cas, on aura à faire à un agrandissement au sens strict.

2.2.10 Propriété: Si f est un agrandissement de G dans G', tout circuit dans G a pour image au moins un circuit dans G'.

Preuve: $x \in \hat{\gamma}(x) \implies f(x) \in \hat{\gamma}'f(x)$

Conséquence: Si G' est un graphe sans circuit, il en est de même pour G.

En revanche, il est tout à fait possible que G soit sans circuit et que G' admette un circuit comme on le voit sur l'exemple de la figure 2.5.

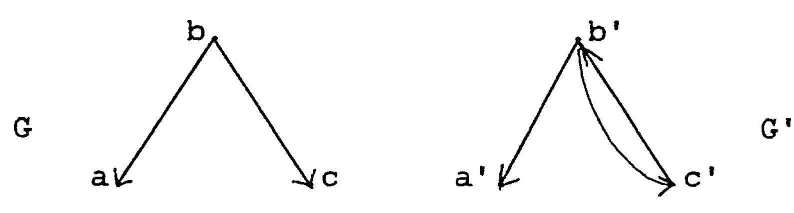


figure 2.5

2.2.11 Propriété: La composée de deux agrandissements est un agrandissement.

Preuve: Soient $G = \langle X,\gamma \rangle$, $G' = \langle X',\gamma' \rangle$ et $G'' = \langle X'',\gamma'' \rangle$ trois graphes; $f : X \to X'$ et $f' : X' \to X''$ deux agrandissements.

Pour tous x et y appartenant à X on a:
$y \in \hat{\gamma}(x) \implies f(y) \in \hat{\gamma}'f(x) \implies f'f(y) \in \hat{\gamma}''f'f(x)$.

2.3 Sous-structures

Soient $G = \langle X,\gamma\rangle$ et $G_1 = \langle X_1,\gamma_1\rangle$ deux graphes tels que X_1 soit inclus dans X.

2.3.1 Définition: G_1 est le **sous-graphe** de G engendré par X_1 si et seulement si

$$\forall\ x \in X_1 \qquad \gamma_1(x) = \gamma(x) \cap X_1$$

2.3.2 Propriété: Si G_1 est un sous-graphe de G, l'injection canonique de X_1 dans X détermine un R-morphisme de G_1 dans G.

Preuve: On a bien

$$\forall\ x \in X_1 \qquad \gamma_1(x) \subset \gamma(x)$$

2.3.3 Définition: G_1 est un **sous-graphe partiel** de G si et seulement si l'injection canonique de X_1 dans X détermine un R-morphisme de G_1 dans G.

On retrouve bien la notion traditionnelle de sous-graphe partiel. En effet, deux sommets de X_1 peuvent ne pas être joints par un arc dans G_1 alors qu'ils sont joints par un arc dans G.

Propriété: Si G_1 est un sous-graphe de G, c'en est aussi un sous-graphe partiel.

Preuve: Résulte immédiatement de 2.3.2.

2.3.4 Définition G_1 est un **sous-graphe total** de G si et seulement si

$$\forall\ x \in X_1 \qquad \gamma_1(x) = \gamma(x)$$

Propriété: G_1 est alors un sous-graphe de G.

Preuve: Immédiate.

2.3.5 Propriété: G_1 est un sous-graphe total de G si et seulement si l'injection canonique de X_1 dans X est un morphisme de graphes.

Preuve: Immédiate.

Propriété: Pour que G_1 soit un sous-graphe total de G, il faut et il suffit que $X_1 = \hat{\gamma}(X_1)$ et que G_1 soit le sous-graphe de G engendré par X_1.

Preuve: Si G_1 est un sous-graphe total de G, c'est évidemment le sous-graphe de G engendré par X_1.

Par ailleurs, soient $x_1 \in X_1$ et $x \in \hat{\gamma}(x_1)$.

Il existe une suite $x_1, x_2, \ldots, x_p$ avec $p>0$ et $x_p = x$ telle que:

$$\forall i \in [p-1] \quad x_{i+1} \in \gamma(x_i)$$

On démontre par récurrence que tous les x_i appartiennent à X_1:

- x_1 appartient à X_1
- si $x_i \in X_1$, $\gamma(x_i) = \gamma_1(x_i) \subset X_1$;

donc x_{i+1} appartient à X_1.

Par conséquent $\hat{\gamma}(X_1) \subset X_1$ et donc $\hat{\gamma}(X_1) = X_1$.

Réciproquement, soit X_1 tel que $\hat{\gamma}(X_1) = X_1$ et soit $G_1 = \langle X_1, \gamma_1 \rangle$ le sous-graphe de G engendré par X_1. On a:

$\forall\, x \in X_1 \quad \gamma(x) \subset X_1$

Par conséquent $\gamma_1(x) = \gamma(x)$, et G_1 est un sous-graphe total de G.

Propriété: Soit $G = \langle X, \gamma \rangle$ un graphe et Y un sous-ensemble de X. Il existe un plus petit sous-graphe total $G_1 = \langle X_1, \gamma_1 \rangle$ de G tel que Y soit inclus dans X_1.

Preuve: D'après la propriété précédente, tout sous-graphe total $G_1 = \langle X_1, \gamma_1 \rangle$ de G tel que $Y \subset X_1$ est tel que $\hat{\gamma}(Y) \subset X_1$.

Or $\hat{\gamma}(Y)$, toujours d'après la propriété précédente, engendre bien un sous-graphe total de G puisque $\hat{\gamma}\hat{\gamma}(Y) = \hat{\gamma}(Y)$.

Ce graphe est bien le plus petit sous-graphe total vérifiant la condition.

<u>Définition</u>: G_1 sera alors **le sous-graphe total engendré par** Y.

<u>Propriété</u>: $Ini(G_1) \subset Y$

<u>Preuve</u>: Pour tout x appartenant à X_1,

$\exists\, y \in Y \quad x \in \hat{\gamma}(y)$

Si x n'appartient pas à Y, x est différent de y et par conséquent ne peut appartenir à $Ini(G_1)$.

<u>Propriété</u>: Si G_1 est un sous-graphe total de G, il existe un plus petit sous-graphe total $G_1' = \langle X_1', \gamma_1' \rangle$ de G tel que X_1 soit inclus dans X_1' et que:

$\forall\, x \in X \quad \forall\, y \in Ini(G_1) \quad y \in \hat{\gamma}(x) \implies x \in X_1'$

<u>Preuve</u>: Soit Y_1 l'ensemble:

$Y_1 = \{x \in Ini(G);\ \hat{\gamma}(x) \cap Ini(G_1) \neq \emptyset\}$

Si G_1' existe, il doit contenir tous les éléments de Y_1.

Or, d'après la démonstration de la propriété précédente, le sous-graphe de G engendré par $\hat{\gamma}(Y_1)$ est le plus petit sous-graphe total de G admettant pour sommets les éléments de Y_1.

Comme X_1 est inclus dans $\hat{\gamma}(Y_1)$, G_1 est bien le plus petit sous-graphe total de G vérifiant les conditions demandées.

<u>Corollaire</u>: Si G_1 est un sous-graphe partiel de G, il existe un plus petit sous-graphe total $G_1' = \langle X_1', \gamma_1' \rangle$ de G tel que X_1 soit inclus dans X_1' et que:

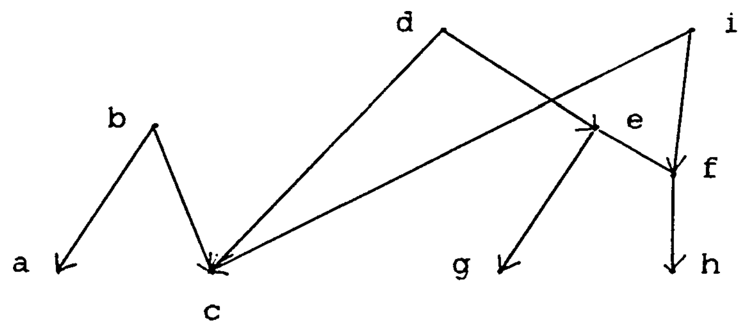
$\forall\, x \in X \quad \forall\, y \in Ini(G_1) \quad y \in \hat{\gamma}(x) \implies x \in X_1'$

<u>Preuve</u>: Résulte immédiatement des deux propriétés précédentes.

<u>Définition</u>: G_1' sera dit le sous-graphe total **englobant** G_1.

<u>Exemple</u>: Soit G le graphe défini en figure 2.6 et soit G_1 le sous-graphe de G engendré par {e,f,g}. Le sous-graphe G_1' de G engendré par {c,d,e,f,g,h} est le sous-graphe total de G englobant G_1.

x	a	b	c	d	e	f	g	h	i
$\gamma(x)$	$\emptyset$	$\{a,c\}$	$\emptyset$	$\{c,e\}$	$\{f,g\}$	$\{h\}$	$\emptyset$	$\emptyset$	$\{c,f\}$



---------------------------figure 2.6---------------------------

2.3.6 Définition: G_1 est un **sous-graphe presque total** de G si et seulement si l'injection canonique de X_1 dans X est un Q-morphisme.

Il s'agit d'un sous-graphe partiel particulier.

Propriété: Pour que G_1, sous-graphe presque total de G, en soit un sous-graphe total, il faut et il suffit que

$\forall\ x \in X_1 \quad x \in Ter(G_1) \implies x \in Ter(G)$

Preuve: Si G_1 est un sous-graphe total de G, il est clair que:

$\gamma_1(x) = \emptyset \implies \gamma(x) = \emptyset$.

Réciproquement, si les sommets terminaux de G_1 sont aussi terminaux dans G, cela entraîne:

$\forall\ x \in X_1 \quad \gamma_1(x) = \gamma(x)$.

2.3.7 Définition: G_1 est un sous graphe **propre** de G si et seulement si l'injection canonique de X_1 dans X est un P-morphisme.

Propriété: G_1 est alors un sous-graphe de G.

Preuve: Soient x et y appartenant à X_1 tels que $y \in \gamma(x)$.

-si $\gamma_1(x) \neq \emptyset$, alors $y \in \gamma_1(x)$

-si $\gamma_1(x) = \emptyset$, alors $\ddot{\gamma}(x) \cap X_1 = \emptyset$,
ce qui est contradictoire avec $y \in \ddot{\gamma}(x) \cap X_1$.

<u>Propriété</u>: Tout sous-graphe total G_1 de G est un sous-graphe propre de G.

<u>Preuve</u>: Immédiate.

2.3.8 La figure 2.7 présente un graphe G dont G_1 est un sous-graphe, G_2 un sous-graphe partiel, G_3 un sous-graphe total, G_4 un

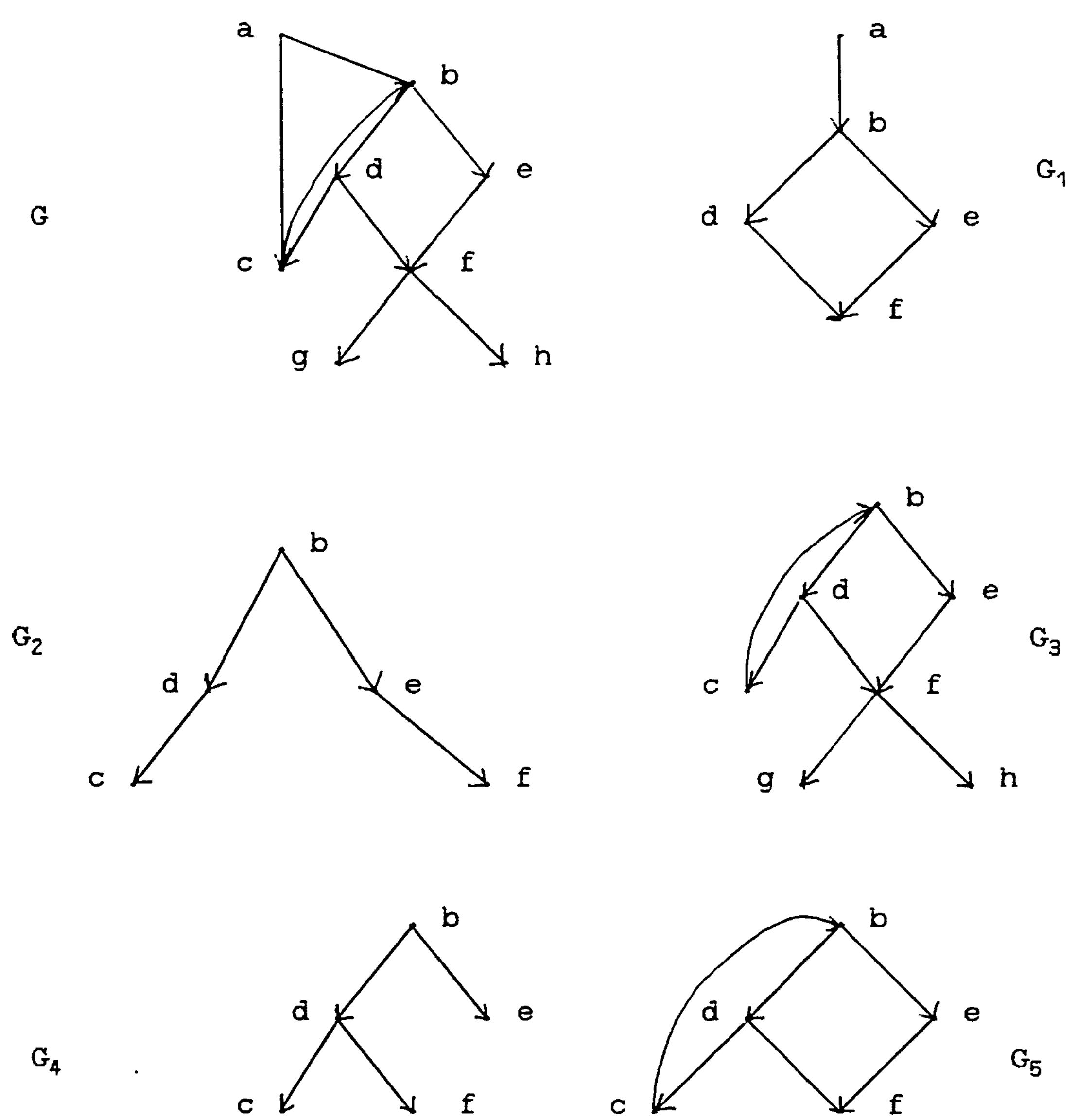


figure 2.7

sous-graphe presque total et G_5 un sous-graphe propre.

On obtient finalement une hiérarchie entre sous-structures d'un graphe. Cette hiérarchie est schématiquement représentée en figure 2.8.

---

sous-graphe

sous-graphe total ==> sous-graphe propre sous-graphe partiel

sous-graphe presque total

-----------------------------figure 2.8---------------------------

2.4 Graphes ordonnés

2.4.1 Définition: Un **graphe ordonné** (ou GO) est défini comme étant un couple $<X,\delta>$ où X est un ensemble et δ une application de X dans X^*.

On retrouve la notion de "DOG" (Directed Ordered Graph) introduite notamment par M.A. Arbib et Y Give'on [Arbib-Give'on, 68].

L'"ordre" concerne la **suite des successeurs** de tout sommet x de X.

On peut maintenant considérer que plusieurs arcs distincts peuvent joindre deux sommets.

2.4.2 Etant donné un GO $G = <X,\delta>$, on définit le **graphe sous-jacent** à G, noté $\overline{G}$, comme étant le graphe $<X,\overline{\delta}>$ où $\overline{\delta}$ est l'application définie à partir de δ en faisant correspondre à chaque suite de sommets l'ensemble des éléments de la suite.

Si on dit "le graphe G", c'est que l'on pense au graphe sous-jacent.

On peut donc définir, d'après les définitions correspondantes sur les graphes sous-jacents, l'ensemble des successeurs d'un sommet x, l'ensemble des sommets terminaux, l'ensemble des sommets initiaux et l'ensemble des descendants d'un sommet donné x. Ce dernier ensemble pourra être noté: $\hat{\delta}(x)$ au lieu de $\bar{\delta}(x)$.

Dans le cas où le graphe sous-jacent est sans circuit, M.A. Arbib et Y Give'on parlent de "DOAG" (Directed Ordered Acyclic Graph). Nous préfèrerons cette appellation à celle de DAG généralement employée en compilation (voir notamment [Aho-Ullman,77]) parce que DAG ne rend pas compte du fait que les successeurs de tout sommet sont ordonnés. Pour donner une traduction francaise à DOAG nous dirons "GOS" (Graphe Ordonné Sans circuit).

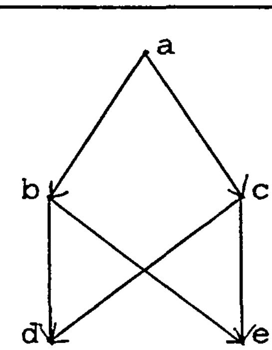
2.4.3 <u>Relations θ_x</u>

Pour tout sommet x d'un graphe ordonné $G = \langle X,\delta\rangle$, on définit une relation notée θ_x. Si y_1 et y_2 sont deux sommets de G, on a $y_1 \ \theta_x \ y_2$ si et seulement si il existe x_1 et x_2 appartenant à X et vérifiant:

(i) $\exists\ \mu, \nu, \rho \in X^*$ $\quad \delta(x) = \mu x_1 \nu x_2 \rho$
(ii) $y_1 \in \hat{\delta}(x_1)$
(iii) $y_2 \in \hat{\delta}(x_2)$

<u>Remarque</u>: Cette relation n'est ni réflexive, ni transitive, ni symétrique, ni antisymétrique.

<u>Exemples</u>: Dans les exemples de graphes ordonnés qui seront présentés, on indiquera l'ordre entre successeurs d'un même



<u>figure 2.9</u>

sommet par un ordre entre les arcs qui ont pour origine ce sommet: l'ordre entre ces arcs sera lu de gauche à droite au voisinage de leur origine. Pour le graphe de la figure 2.9, les couples qui sont dans la relation θ_a sont les suivants:
<b,c>,<d,e>,<e,d>,<d,d>,<e,e>,<d,c>,<e,c>,<b,e>,<b,d>.

Pour le graphe de la figure 2.10, les couples qui sont dans la relation θ_a sont:
<b,c>,<d,e>,<b,e>,<d,c>,<a,c>,<c,c>,<e,c>,<a,e>,<c,e>,<e,e>.

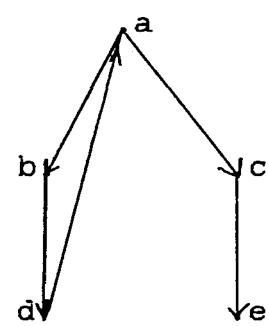


figure 2.10

2.4.4 Morphismes de graphes ordonnés

Soient $G = \langle X,\delta\rangle$ et $G' = \langle X',\delta'\rangle$ deux GO et $f : X \dashrightarrow X'$ une application.

f est un **morphisme de GO** si et seulement si

$$\forall x \in X \quad f\delta(x) = \delta' f(x)$$

Propriété: f détermine un morphisme de graphes sur les graphes sous-jacents.

Preuve: Immédiate.

Remarque: La réciproque n'est pas vraie.

f est un **Q-morphisme de GO** si et seulement si

$$\forall x \in X \quad \delta(x) \neq \lambda \implies f\delta(x) = \delta' f(x)$$

Propriété: f détermine un Q-morphisme de graphes sur les graphes sous-jacents.

Preuve: Immédiate.

Remarque: La réciproque n'est pas vraie.

f est un **R-morphisme de GO** si et seulement si

$\forall x \in X \qquad f\delta(x) \subset \delta' f(x)$

Il s'agit ici d'une inclusion de suites, c'est-à-dire que l'ordre des successeurs d'un sommet donné x doit être conservé.

Pour tout $x \in X$, il faut que si

$\delta(x) = x_1 \ldots x_p$

et $\delta' f(x) = x'_1 \ldots x'_q$

on ait une suite $<j_1, \ldots, j_p>$ croissante extraite de $<1, \ldots, q>$ telle que:

$\forall k \in [p] \qquad x'_{j_k} = f(x_k)$.

f est un **agrandissement de GO** si et seulement si

(i) $\forall x,y \in X \quad y \in \delta(x) \implies f(y) \in \delta' f(x)$

(ii) $\forall x,y,z \in X \ \forall \mu \cdot \nu \cdot \rho \in X^* \qquad \delta(x) = \mu y \nu z \rho \implies f(y) \theta_{f(x)} f(z)$

Propriété: f détermine alors un agrandissement sur les graphes sous-jacents.

Preuve: Immédiate.

2.4.5 Sous-structures

De même que pour les graphes non ordonnés, on définit:

- des sous-GO
- des sous-GO totaux
- des sous-GO presque totaux
- des sous-GO propres
- des sous-GO partiels.

2.5 Graphes étiquetés

2.5.1 Un **graphe étiqueté** (ou GE) sur L sera donné à l'aide d'une application $\xi : X \to L$ où L est un ensemble (d'étiquettes).

On aura donc un triplet $H = <X, \gamma, \xi>$.

Il se peut que tous les sommets du graphe ne soient pas étiquetés. Dans ce cas l'application ξ est telle que:

$$\xi \; : \; X^0 \dashrightarrow L$$

X^0 étant un sous-ensemble de X.

Si $X = X^0$, le graphe étiqueté est dit **complet.**

2.5.2 Morphismes de graphes étiquetés

Soient $H = \langle X,\gamma,\xi\rangle$ et $H' = \langle X',\gamma',\xi'\rangle$ deux graphes étiquetés sur le même ensemble L, $\xi : X^0 \dashrightarrow L$ et $\xi' : X'^0 \dashrightarrow L$ étant les deux applications étiquetantes. Soit f une application de X dans X'.

f est un **morphisme de GE** si et seulement si c'est un morphisme de graphes qui vérifie de plus

$$\forall\, x \in X^0 \qquad \xi(x) = \xi' f(x)$$

On définit de même les Q-morphismes de GE, les P-morphismes de GE, les R-morphismes de GE ainsi que les agrandissements de GE.

2.5.3 Sous-structures

De même que pour les graphes non étiquetés, on définit:
- des sous-GE
- des sous-GE totaux
- des sous-GE presque totaux
- des sous-GE propres
- des sous-GE partiels.

Mais, dans le cas présent, nous donnons une définition nouvelle qui nous sera utile par la suite.

Définition: Etant donnés un graphe étiqueté $H = \langle X,\gamma,\xi\rangle$ sur un ensemble d'étiquettes L et K un sous-ensemble de L, $H_1 = \langle X_1,\gamma_1,\xi_1\rangle$ est un **sous-GE K-total** de H si et seulement si c'est un sous-graphe partiel de H vérifiant

$$\forall\, x \in X_1 \qquad \xi(x) \in K \Longrightarrow \gamma_1(x) = \gamma(x)$$

<u>Propriété</u>: Pour que H_1 soit un sous-graphe total de H, il faut et il suffit qu'il en soit un sous-graphe L-total.

<u>Preuve</u>: Immédiate.

2.5.4 <u>Graphes multiétiquetés</u> (GME)

L'étiquetage des sommets d'un graphe peut être multiple. Il suffit que:

$$L = L^1 \times L^2 \times \ldots \times L^p$$

On dira dans ce cas que le graphe est **multiétiqueté** sur la suite d'ensembles $<L^1, \ldots, L^p>$.

A ce moment là, l'application ξ se décompose en p composantes:

$$\xi^i : X \dashrightarrow L^i$$

On pourra écrire: $\xi = <\xi^1, \ldots, \xi^p>$.

Il n'est pas nécessaire que chaque sommet de G possède les p étiquettes. On définit X^i, sous-ensemble de X composé de tous les sommets qui possèdent la ième étiquette. L'application ξ^i devient dans ces conditions:

$$\xi^i : X^i \dashrightarrow L^i$$

Afin que l'application ξ reste définie, on se donne un symbole distingué π qui n'appartient à aucun des L^i. ξ devient une application

$$\xi : X \dashrightarrow L^1 \cup \{\pi\} \times L^2 \cup \{\pi\} \times \ldots \times L^p \cup \{\pi\}$$

$$x \mapsto <s_1, s_2, \ldots, s_p>$$

Pour tout $i \in [p]$, si $x \in X^i$ $\quad s_i = \xi^i(x)$

si $x \notin X^i$ $\quad s_i = \pi$

On admet encore l'écriture: $\xi = <\xi^1, \ldots, \xi^p>$.

Soit $<i_1, \ldots, i_q>$ une sous-suite de $<1, \ldots, p>$. Un graphe multiétiqueté sur $<L^1, \ldots, L^p>$ sera dit $<i_1, \ldots, i_q>$-**complet** si et seulement si

$\forall\, j \in [q] \quad X^{ij} = X$.

2.5.5 <u>Morphismes de graphes multiétiquetés</u>

Soient $H = \langle X,\gamma,\xi\rangle$ et $H' = \langle X',\gamma',\xi'\rangle$ deux graphes multiétiquetés tels que:

$$\xi = \langle \xi^1,\xi^2,\ldots,\xi^p\rangle$$

$$\xi' = \langle \xi'^1,\xi'^2,\ldots,\xi'^q\rangle$$

$$\forall\, i \in [p] \qquad \xi^i : X^i \dashrightarrow L^{l_i}$$

$$\forall\, j \in [q] \qquad \xi'^j : X'^j \dashrightarrow L^{m_j}$$

et soit f un morphisme de graphes de $\langle X,\gamma\rangle$ dans $\langle X',\gamma'\rangle$.

Soient $\langle i_1,\ldots,i_r\rangle$ une suite extraite de $\langle 1,\ldots,p\rangle$ et soit $\langle j_1,\ldots,j_r\rangle$ une suite composée de r éléments distincts pris dans [q].

f est un $(\langle i_1,\ldots,i_r\rangle,\langle j_1,\ldots,j_r\rangle)$-**morphisme de GME** de H dans H' si et seulement si pour tout k appartenant à [r] on a:

(i) $L^{l_{i_k}} = L^{m_{j_k}}$

(ii) $\forall x \in X^{i_k}\ f(x) \in X'^{j_k}$ et $\xi^{i_k}(x) = \xi'^{j_k}f(x)$

Supposons maintenant que les applications ξ et ξ' soient telles que

$$\xi = \langle \xi^1,\xi^2,\ldots,\xi^p\rangle$$

$$\xi' = \langle \xi'^1,\xi'^2,\ldots,\xi'^p\rangle$$

$$\forall\, i \in [p] \qquad \xi^i : X^i \to L^i$$

$$\xi'^i : X'^i \to L^i$$

f est un **morphisme de GME** si et seulement si c'est un $(\langle 1,\ldots,p\rangle,\langle 1,\ldots,p\rangle)$-morphisme.

La différence avec les morphismes de GE est que l'on peut avoir pour un $x \in X$ tel que

$$\xi(x) = \langle s_1, \ldots, s_p \rangle$$

$$\xi'(x) = \langle s'_1, \ldots, s'_p \rangle$$

avec un $i \in [p]$ tel que $s_i = \pi$ et $s'_i \neq \pi$

On définit de même des morphismes correspondant aux Q-morphismes, aux P-morphismes, aux R-morphismes et aux agrandissements: la définition sera la même, à ceci près que f devra être un Q-morphisme, un P-morphisme, un R-morphisme ou un agrandissement de $\langle X,\gamma \rangle$ dans $\langle X',\gamma' \rangle$.

2.5.6 Etiquetage des graphes ordonnés

Définition: Un **graphe ordonné étiqueté** (GOE) est un triplet $H = \langle X,\delta,\xi \rangle$ où ξ est une application de X (ou d'un sous-ensemble de X) dans un ensemble L d'étiquettes.

On définit, de la même manière que les GME, les **graphes ordonnés multiétiquetés** (ou GOME), les **graphes ordonnés sans circuit multiétiquetés** (ou GOSME) ainsi que les différents morphismes sur ces structures.

Définition: Soient $H = \langle X,\gamma,\xi \rangle$ et $H' = \langle X',\gamma',\xi' \rangle$ deux graphes étiquetés sur le même ensemble L, $\xi : X^0 \dashrightarrow L$ et $\xi' : X'^0 \dashrightarrow L$ étant les deux applications étiquetantes. Soit f une application de X dans X'et soit M un sous-ensemble de L.

f est un M-agrandissement de H dans H' si et seulement si $\forall\, x,y,z \in X\ \forall\, \mu,\ \nu,\ \rho \in X^*$

$$\xi(x) \in M \text{ et } \delta(x) = \mu x_1 \nu x_2 \rho \implies f(y)\ \theta_{f(x)}\ f(z)$$

On définit de même, si H est un graphe multiétiqueté sur $\langle L^1,\ldots,L^p \rangle$, un $\langle i_1,\ldots,i_q \rangle$-agrandissement, $\langle i_1,\ldots,i_q \rangle$ étant une sous-suite de $\langle 1,\ldots,p \rangle$.

2.5.7 Etiquetage des arcs

On peut étiqueter les arcs des GOE ou des GOME à l'aide des applications ξ^1. L^1 doit simplement être tel que:

$$L^1 = T^* ,$$

T étant un ensemble d'étiquettes d'arcs. L'application ξ^1 doit être alors définie sur tout l'ensemble X et vérifier:

$$\forall x \in X \qquad |\xi^1(x)| = |\delta(x)| \quad .$$

Par définition, l'application ξ^1 est dite **arc-étiquetante.**

Si $\delta(x) = x_1 \ldots x_p$ et $\xi^1(x) = t_1 \ldots t_p$, chaque arc $\langle x, x_j \rangle$ $(j \in [p])$ est étiqueté par t_j.

Dans ces conditions plusieurs arcs de même origine et même extrémité pourront être différenciés puisqu'ils pourront être étiquetés de manière différente.

2.5.8 Préservation de l'étiquetage des arcs dans les morphismes de GOE ou de GOME

Plaçons-nous dans le cas des GOE, le cas des GOME s'en déduisant aisément.

Définition: Soient $H = \langle X, \delta, \xi \rangle$ et $H' = \langle X', \delta', \xi' \rangle$ deux GOE sur T^* où ξ et ξ' sont des applications arc-étiquetantes et soit f une application de X dans X'. f **conserve l'étiquetage des arcs** si et seulement si, pour tout $x \in X$, on a:

$$\forall\, i \in [p] \quad \exists\, j \in [q] \quad f(x_i) = x'_j \quad \text{et} \quad t_i = t'_j$$

avec:

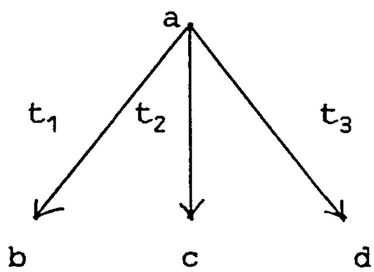
$$\delta(x) = x_1 \ldots x_p \qquad \xi(x) = t_1 \ldots t_p$$
$$\delta' f(x) = x'_1 \ldots x'_q \qquad \xi' f(x) = t'_1 \ldots t'_q$$

Propriété: f est un R-morphisme de graphes de $\langle X, \overline{\delta} \rangle$ dans $\langle X', \overline{\delta}' \rangle$.

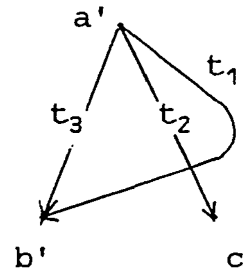
Preuve: Il est clair que:

$$\forall\, x \in X \ \forall\, x_i \in \overline{\delta}(x) \ \exists\, x'_j \in \overline{\delta}' f(x) \ f(x_i) = x'_j$$

Exemples: L'application f de la figure 2.11 ne conserve pas



H

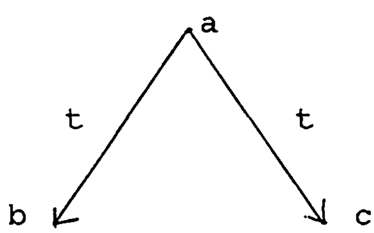


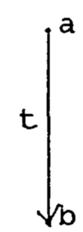
H'

x	a	b	c	d
$\delta(x)$	bcd	λ	λ	λ
$\xi(x)$	$t_1t_2t_3$	λ	λ	λ
f(x)	a'	b'	c'	b'

x	a'	b'	c'
$\delta'(x)$	b'c'b'	λ	λ
$\xi'(x)$	$t_3t_2t_1$	λ	λ

figure 2.11





x	a	b	c
$\delta(x)$	bc	λ	λ
$\xi(x)$	tt	λ	λ
f(x)	a'	b'	b'

x	a'	b'
$\delta'(x)$	b'	λ
$\xi'(x)$	t	λ

figure 2.12

l'étiquetage de chaque sommet. Dans ce cas on a à faire à un morphisme de GO, mais pas à un morphisme de GOE. Tandis que l'application f de la figure 2.12 est un morphisme de graphes de

$<X,\overline{\delta}>$ dans $<X',\overline{\delta}'>$, mais n'est pas un morphisme de GO.

Propriété: Soient $H = <X,\delta,\xi>$ et $H' = <X',\delta',\xi'>$ deux GOE sur T^* tels que ξ' soit une application arc-étiquetante. S'il existe un morphisme f de GOE de H dans H', alors ξ est une application arc-étiquetante et f conserve l'étiquetage des arcs.

Preuve: f étant un morphisme de GOE, on a:

$$\forall x \in X \qquad f\delta(x) = \delta'f(x) \quad \text{et} \quad \xi(x) = \xi'f(x)$$

et, comme ξ' est arc-étiquetante:

$$\forall y \in X' \qquad |\xi'(y)| = |\delta'(y)|$$

On en déduit:

$$\forall x \in X \qquad |\xi'f(x)| = |\delta'f(x)| = |f\delta(x)| = |\xi(x)|$$

et donc $|\xi(x)| = |\delta(x)|$, ce qui montre que l'application ξ est arc-étiquetante.

D'autre part, en notant:

$$\delta(x) = x_1...x_p \qquad \xi(x) = t_1...t_p$$
$$\delta'f(x) = x_1'...x_p' \qquad \xi'f(x) = t_1'...t_p'$$

on a:

$$\forall i \in [p] \qquad f(x_i) = x_i' \quad \text{et} \quad t_i = t_i'$$

ce qui montre que f conserve l'étiquetage des arcs.

2.6 Graphes semi-ordonnés et réseaux

Il se peut très bien que l'ordre entre les successeurs de sommets soit significatif pour certains sommets et pas pour d'autres. En particulier, qu'il ne soit pas significatif pour les sommets qui recoivent certaines étiquettes.

2.6.1 K-équivalence de GOE

Soit L un ensemble d'étiquettes et soit K un sous-ensemble de L.

Définition: Deux GOE $H = <X,\delta,\xi>$ et $H'= <X,\delta',\xi>$ qui ne diffèrent que par l'application δ seront dits **K-équivalents** si et seulement si:

(i) $\forall x \in X \quad \delta(x) = x_1...x_p \implies \delta'(x) = x_{j_1}...x_{j_p}$

$<j_1,...,j_p>$ étant une permutation des p premiers nombres

(ii) $\forall x \in X \quad \xi(x) \in K \implies \delta(x) = \delta'(x)$

Il y a bien une relation d'équivalence entre graphes ordonnés.

2.6.2 Graphe K-ordonné

Les classes d'équivalence moyennant cette relation seront, par définition, des graphes **K-ordonnés**.

On désignera une classe d'équivalence par l'un de ses représentants.

Remarque: Si K est vide, la structure est de même type que les graphes étiquetés non ordonnés, à ceci près que plusieurs arcs distincts peuvent joindre deux mêmes sommets.

Définition: Soient $H = \langle X,\delta,\xi\rangle$ et $H' = \langle X',\delta',\xi'\rangle$ deux graphes K-ordonnés, et soit f une application de X dans X'. f est un **agrandissement de graphes K-ordonnés** si et seulement si

(i) $\forall x,y \in X \quad y \in \delta(x) \implies f(y) \in \hat{\delta}'f(x)$

(ii) $\forall x,y,z \in X \; \forall \mu\cdot\nu\cdot\rho \in X^*$

$\delta(x) = \mu y \nu z \rho$ et $\xi(x) \in K \implies f(y)\theta_{f(x)}f(z)$

2.6.3 Cas des graphes multiétiquetés

L'application ξ est telle que:

$\xi = \langle \xi^1,\dots,\xi^p\rangle$

Il se peut que seulement certaines des applications ξ^i interviennent pour déterminer si l'ordre des successeurs des sommets est significatif ou pas.

Dans ce cas, il est commode de grouper en tête ces applications. Supposons qu'il y en ait k; on a alors

$\xi = \langle \xi^1,\dots,\xi^k,\xi^{k+1},\dots,\xi^p\rangle$.

Soient $K^1,\dots,K^k$ des sous-ensembles de $L^1,\dots,L^k$. On dira que les graphes sont **$\langle K^1,K^2,\dots,K^k\rangle$-ordonnés**.

Si c'est seulement l'application ξ^1 qui est significative, on pourra dire que les graphes sont des graphes K^1-ordonnés.

2.6.4 Etiquetage des arcs des graphes semi-ordonnés

Soient H et H' deux graphes multiétiquetés K^1-équivalents. Ils sont, indifféremment l'un ou l'autre, le représentant d'un graphe K^1-ordonné.

Supposons que pour un sommet donné x de X on ait:

$$\delta(x) = x_1x_2 \quad \text{et} \quad \delta'(x) = x_2x_1$$

Si l'application ξ^2 est arc-étiquetante, l'étiquetage des arcs peut ne plus être le même dans le graphe H et dans le graphe H'. En effet si on a

$$\xi^2(x) = t_1t_2 \ ,$$

l'arc $\langle xx_1\rangle$ aura pour étiquette t_1 dans H_1 et t_2 dans H_2.

Par conséquent, si l'on veut que l'étiquetage des arcs soit conservé, il faut définir une nouvelle forme d'équivalence entre GOME.

A cette fin, il est nécessaire de distinguer au préalable les applications arc-étiquetantes (ou plutôt celles qui servent à étiqueter les arcs) des autres applications ξ^i.

On notera donc de tels GOME $H = \langle X,\delta,\xi,\rho\rangle$ avec

$$\xi : \quad X \dashrightarrow L^1 \times \ldots \times L^h$$

$$\rho : \quad X \dashrightarrow L^{h+1} \times \ldots \times L^p$$

où chacune des applications $\rho^i : X \dashrightarrow L^{h+i}$ est arc-étiquetante.

La structure définie sera appelée GOAME.

2.6.5 Réseaux

Deux GOAME $H = \langle X,\delta,\xi,\rho\rangle$ et $H' = \langle X,\delta',\xi,\rho'\rangle$ seront $\langle K^1,\ldots,K^k\rangle$-équivalents ($k\leqslant h$) si, en plus des conditions (2.6.1) (i) et (ii), pour tout i tel que $1\leqslant i\leqslant p-h$, pour tout x appartenant à X, on ait:

$$\left.\begin{array}{l} \delta(x) = x_1\ldots x_q \\ \delta'(x) = x_{j_1}\ldots x_{j_q} \\ \rho^i(x) = t_1\ldots t_q \end{array}\right\} \Longrightarrow \rho'^i(x) = t_{j_1}\ldots t_{j_q}$$

On définit un GOAME $<K^1,\ldots,K^k>$-ordonné comme étant une classe d'équivalence moyennant cette relation et on le désigne par un de ses représentants.

Dans le cas où le graphe est sans circuit et k = 1, cette structure sera encore appelée un **K^1-réseau.**

Définition: Etant donné un K_1 réseau H, H_1 sera un **sous-réseau total** de H si et seulement si c'est un réseau engendré par un sous-GE K^1-total de H.

Définition: Etant donnés deux K^1-réseaux $H = <X,\delta,\xi,\rho>$ et $H' = <X',\delta',\xi',\rho'>$, f est un **QR-morphisme de réseaux** si et seulement si c'est un (<1>,<1>)-morphisme de GME de H dans H' qui conserve l'étiquetage des arcs et qui de plus vérifie:

$\forall\ x \in X \quad f\delta(x) \subset \delta' f(x)$

$\forall\ x \in X\backslash Ter(H)\ \xi' f(x) \in K^1 \Longrightarrow f\delta(x) = \delta' f(x)$

La définition des agrandissements de réseaux se déduit immédiatement de celle des agrandissements des graphes K-ordonnés.

2.7 Opérateurs associés aux graphes: Σ-GOSME et Σ-grilles

2.7.1 Un **schéma d'opérateurs** Σ sera donné par un ensemble U (d'opérateurs) ainsi qu'une application $\nu : U \dashrightarrow \mathbb{N}$ (c'est-à-dire par un ensemble "gradué" au sens de [Arbib-Give'on, 68]).

Un opérateur u appartenant à U sera dit **n-aire** lorsqu'il vérifiera $\nu(u) = n$.

2.7.2 Soit S un ensemble de **sortes**. Σ sera dit un **S-schéma d'opérateurs** si et seulement si il existe une application

$$\begin{array}{rcl} \tau : U & \longrightarrow & S \times S^* \\ u & \longmapsto & \tau(u) = <s,s_1\ldots s_n> \end{array}$$

Par définition, l'arité $\nu(u)$ de l'opérateur est égale à n.

$\tau(u)$ sera le **type** de u qui sera un opérateur **typé**.

La notion de S-schéma est équivalente à la notion de Σ-algèbre (voir [Desclés,80]).

2.7.3 On munit un schéma d'opérateurs d'une **sémantique** de la manière suivante: on se donne une famille d'ensembles indexée par S $(E^s)_{s \in S}$ et, à certains triplets tels que

$$\langle u, s, s_1 \ldots s_{\nu(u)} \rangle$$

on associe une application

$$u^{\cdot} \;:\; E^{s_1} \times E^{s_2} \times \ldots \times E^{s_{\nu(u)}} \dashrightarrow E^s \quad .$$

Si Σ est un S-schéma, c'est à chacun des triplets $\langle u, s, s_1 \ldots s_{\nu(u)} \rangle$ où $\tau(u) = \langle s, s_1 \ldots s_{\nu(u)} \rangle$ que doit être associée une application.

2.7.4 On se donne un graphe ordonné sans circuit $H = \langle X, \delta, \xi \rangle$ multiétiqueté de la manière suivante:

$$\xi^1 : X \dashrightarrow S$$
$$\xi^2 : X^2 \dashrightarrow U$$

où S est un ensemble de sortes, U un ensemble d'opérateurs et X^2 un sous-ensemble de X vérifiant:

$$X \setminus Ter(H) \subset X^2$$

On notera σ l'application ξ^1 et ω l'application ξ^2.

Dans le cas où $\Sigma = \langle U, \nu \rangle$ est un schéma d'opérateurs, H sera un **Σ-GOSME** si et seulement si:

$$\text{(i)} \quad \forall\, x \in X^2 \quad |\delta(x)| = \nu\omega(x)$$

On retrouve, si l'on ne tient pas compte de l'application σ, la notion de Σ-DOAG introduite dans [Arbib-Give'on,68].

Dans le cas où $\Sigma = \langle U, \tau \rangle$ est un S-schéma, H sera un Σ-GOSME si et seulement si

$$\text{(ii)} \quad \forall\, x \in X^2 \quad \tau\omega(x) = \langle \sigma(x), \sigma\delta(x) \rangle$$

Dans ce cas la condition (i) est nécessairement vérifiée.

Que Σ soit un schéma ou un S-schéma, on pourra noter le Σ-GOSME $H = \langle X, \delta, \sigma, \omega \rangle$.

Le Σ-GOSME sera dit **complet** si et seulement si $X^2 = X$.

2.7.5 Soit $G = \langle X,\delta\rangle$ un graphe ordonné sans circuit tel que Ini(G) soit composé de deux sommets distincts v et w, et soit $G^0 = \langle X^0,\delta^0\rangle$ le sous-GO de G engendré par $X^0 = X\setminus\{v,w\}$.

G est une **grille** si et seulement si

(i) $\overline{\delta}(v) = \mathrm{Ini}(G^0)$

(ii) $\overline{\delta}(w) = \mathrm{Ter}(G)$

Dans ce cas $\delta(v)$ est la **suite initiale** de G et $\delta(w)$ en est la **suite terminale.**

2.7.6 Soit $G = \langle X,\delta\rangle$ une grille et soient σ et ω deux apllications:

$\sigma : X^0 \dashrightarrow S$

$\omega : X^2 \dashrightarrow U$

avec $X^2 = X^0 \setminus \mathrm{Ter}(G)$.

$H = \langle X,\delta,\sigma,\omega\rangle$ est une **Σ-grille** si et seulement si $H^0 = \langle X^0,\delta^0,\sigma^0,\omega^0\rangle$ est un Σ-GOSME.

2.7.7 Si Σ est muni d'une sémantique, on peut associer à toute Σ-grille $H = \langle X,\delta,\sigma,\omega\rangle$ une **application** f_H à condition qu'à tout triplet $\langle\omega(x),\sigma(x),\sigma\delta(x)\rangle$ ($x \in X^2$) soit associée une application

$(\omega(x))^{\cdot} : E^{\sigma(x_1)}x\ldots x\ E^{\sigma(x_p)} \dashrightarrow E^{\sigma(x)}$

où $\delta(x) = x_1\ldots x_p$.

Cette application pourra être notée: $\omega^{\cdot}(x)$.

A ce moment-là, on définit, par induction descendante sur k ($r(H^0)\geqslant k\geqslant 0$), pour tout sommet x de H^0 de rang k l'application f_x

$f_x : E^{\sigma(z_1)}x\ldots x\ E^{\sigma(z_n)} \dashrightarrow E^{\sigma(x)}$

(où $\delta(w) = z_1\ldots z_n$) de la manière suivante:

$\langle e_1,\ldots,e_n\rangle \mapsto e' = f(e_1,\ldots,e_n)$.

Si x est un sommet terminal de H, il existe j appartenant à [n] tel que $x = z_j$; on pose:

$e' = e_j$

Sinon la suite $\delta(x) = x_1 \ldots x_p$ est non vide. Chaque élément de cette suite est de rang supérieur à k, on a donc pu faire correspondre à $<e_1, \ldots, e_n>$ un p-uple

$<e_1', \ldots, e_p'> \in X\, E^{\sigma(x_i)}$

par les applications associées aux sommets de rang supérieur à k.

On peut donc poser:

$$e' = \omega'(x)\ (e_1', \ldots, e_p').$$

L'application f_H associée à la Σ-grille sera définie de la manière suivante:

$f_H : E \dashrightarrow F$ avec

$E = E^{\sigma(z_1)} x \ldots x\ E^{\sigma(z_n)}$

$F = E^{\sigma(a_1)} x \ldots x\ E^{\sigma(a_m)}$

et $\delta(v) = a_1 \ldots a_m$

$f_H(e_1, \ldots, e_n) = <f_{a_1}(e_1, \ldots, e_n), \ldots, f_{a_m}(e_1, \ldots, e_n)>$

3 Grammaires de graphes sans circuit

Les grammaires de graphes nous serviront à formaliser la représentation des connaissances dans un cas complexe (4.3), mais également à décrire la construction de réseaux représentatifs de connaissances plus élémentaires (chapitre 6). Dans les deux cas on fera appel à des règles de réécriture de réseaux (3.4.4).

Toutefois, on introduit ces règles pas à pas en commençant (3.1) par les grammaires portant sur les graphes les plus généraux (non ordonnés). On se pose ensuite le problème de la conservation du caractère sans circuit des graphes à travers les transformations (3.2), puis on étudie le cas des graphes ordonnés (3.3), celui des graphes étiquetés et multi-étiquetés (3.4), arrivant ainsi aux réseaux.

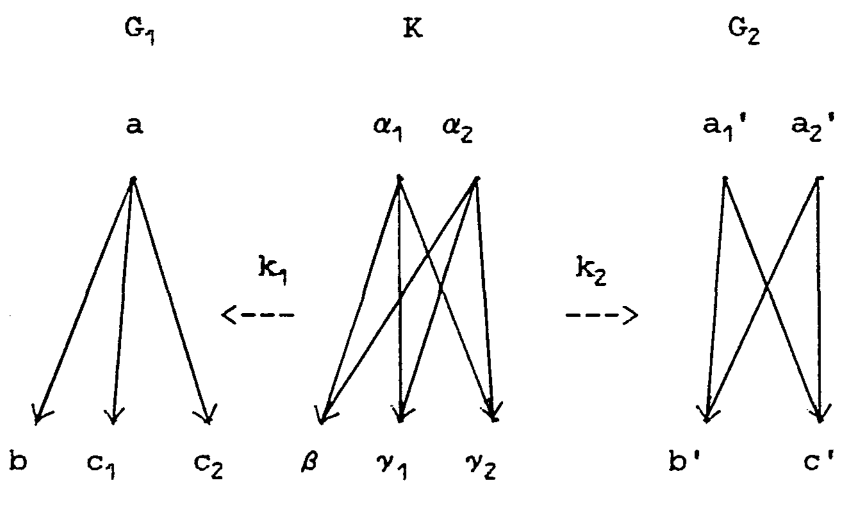
3.1 Description d'un modèle général

3.1.1 Les grammaires de graphes selon H. Ehrig

Les premières grammaires de graphes - ou "Web Grammars" - ont été introduites en 1969 par J.L. Pfaltz et A. Rosenfeld [Pfaltz-Rosenfeld, 69]. Une "règle de réécriture de graphes" est alors un triplet (G_1,G_2,E) où G_1 et G_2 sont deux graphes. L'application de cette règle à un graphe G se fait en remplacant dans G un sous-graphe équivalent à G_1 par G_2, E servant à préciser de quelle manière G_2 s'emboîte dans le graphe G d'où a été retiré G_1. Mais E est écrit, pour chaque règle, en langue naturelle; ce qui signifie que les règles ne sont pas encore réellement formalisées.

Par la suite, des grammaires dont le formalisme était de plus en plus élaboré ont été présentées. Mais la description la plus générale des grammaires de graphes est due, depuis 1973 [Ehrig-Pfender-Schneider, 73], à H. Ehrig et ses collaborateurs (voir [Ehrig, 79] pour une présentation synthétique).

Selon eux, une règle est composée non seulement d'un couple de graphes (G_1,G_2), mais aussi d'un troisième graphe K et de deux morphismes de graphes k_1 : K --> G_1 et k_2 : K --> G_2 qui servent à indiquer les arcs et sommets "communs" à G_1 et G_2.



---------------------------figure 3.1----------------------------

Pour transformer un graphe G dont G_1 est un sous-graphe, il y a nécessité de se donner un graphe D et un morphisme injectif d : K --> D de telle manière que G soit obtenu à partir de G_1 et D par la mise en commun des éléments qui sont image des éléments de K. G', transformé de G, est construit pareillement à partir de G_2 et D.

Un exemple de règle est donné en figure 3.1.
$k_1(\alpha_1) = k_1(\alpha_2) = a$ indique qu'il y a "éclatement" du sommet a en les sommets a_1' et a_2'.
$k_2(\gamma_1) = k_2(\gamma_2) = c'$ indique qu'il y a "fusion" des sommets c_1 et c_2 en c'.

3.1.2 Une autre définition des règles de réécriture

Nous donnons une définition spécifique des règles de réécriture de graphes, définition qui est plus simple que celle de H. Ehrig et qui est plus adaptée aux transformations de structures que l'on veut effectuer.

En premier lieu, pour nous, il suffira de donner l'ensemble sous-jacent au graphe K, la transformation des arcs étant déduite des graphes G_1 et G_2.

Dès lors, K, k_1 et k_2 déterminent uniquement une correspondance entre les sommets de G_1 et ceux de G_2. On peut donc les remplacer par une relation entre éléments de X_1 et éléments de X_2.

Définition: Une règle de réécriture de graphes est constituée par un triplet $<G_1,G_2,\Phi>$ où $G_1 = <X_1,\gamma_1>$ et $G_2 = <X_2,\gamma_2>$ sont deux graphes et Φ un sous-ensemble de $X_1 \times X_2$.

Exemple: La règle de l'exemple précédent est décrite par les deux graphes G_1 et G_2 ainsi que par la relation composée des couples: $<a,a_1'>$, $<a,a_2'>$, $<b,b'>$, $<c_1,c'>$, $<c_2,c'>$.

3.1.3 La dérivation directe

Nous nous restreindrons au cas où il n'y a pas d'éclatement dans les transformations. Une règle devient alors un triplet $<G_1,G_2,\zeta>$ où ζ est une fonction (non partout définie dans le cas général) de X_1 dans X_2. On notera $\zeta^{-1}(X_2)$ le domaine de définition de ζ .

Supposons que l'on veuille appliquer la règle $<G_1,G_2,\zeta>$ à un graphe G.

Il s'agit en premier lieu de trouver dans G une sous-structure G_1' qui soit à l'image de G_1. Cela signifie qu'il doit exister un morphisme de G_1 dans G_1', et par conséquent un morphisme ψ de G_1 dans G. Or, dans le chapitre précédent, nous avons défini toute une variété de morphismes, ainsi que de sous-structures. Cela entraîne qu'il y aura une grande variété de recherches de G_1' possibles, et donc une grande variété de types de dérivations directes. Selon que ψ est un R-morphisme, un Q-morphisme ou un P-morphisme, nous parlerons de R-dérivation, de Q-dérivation ou de P-dérivation.

Nous nous plaçons d'abord dans le cas où G_1' est un sous-graphe partiel de G et où il existe un R-morphisme de G_1 dans G_1', cas qui est réalisé si et seulement si il existe un R-morphisme ψ de G_1 dans G. Ce cas englobe tous les autres cas, hormis celui des agrandissements.

On construit le graphe $G' = <X',\gamma'>$ en "remplaçant" G_1' par un graphe G_2' "à l'image" de G_2, selon un procédé que l'on va indiquer. On dira que G' est le **transformé** de G **par** la règle $R = <G_1,G_2,\zeta>$ **pour** le morphisme ψ.

On se donne l'ensemble Ω constitué des couples $\langle x,y\rangle$ de $X_1' \times X_2$ qui vérifient la propriété suivante:

$$\exists\, z \in X_1 \quad x = \psi(z) \quad \text{et} \quad y = \zeta(z)$$

On définit dans Ω la relation S par:

$$\langle u,v\rangle \; S \; \langle x,y\rangle \iff u = x \quad \text{ou} \quad v = y$$

Soit V l'ensemble quotient de Ω par $\overline{S}$, fermeture transitive de S (V est inclus dans l'ensemble des parties de Ω) et soit W l'ensemble constitué de tous les ensembles composés d'un seul couple tel que $\langle \epsilon,y\rangle$, ϵ étant un symbole distingué et y étant un élément quelconque de $X_2 \setminus \zeta(X_1)$.

X_2' est défini comme étant l'union de V et W.

Par conséquent: - à tout élément x de X_1' correspond au maximum un élément α de X_2' tel qu'il existe un couple de la forme $\langle x,y\rangle$ appartenant à α;

- à tout élément y de X_2 correspond un élément et un seul de X_2' tel qu'il existe un couple de la forme $\langle x,y\rangle$ appartenant à α.

On pose:

$$X' = (X \setminus X_1') \cup X_2'$$

et on définit γ' comme suit:

- si $x \in X \setminus X_1'$

$$\gamma'(x) = [\gamma(x) \setminus X_1'] \cup \{\alpha \in X_2';\ \exists \langle y,z\rangle \in \alpha \quad y \in \gamma(x)\}$$

Cela signifie que si $\langle x,y\rangle$ est un arc de G et si x n'est pas pris dans la transformation,

(a) si y n'est pas non plus pris dans la transformation, l'arc $\langle x,y\rangle$ est conservé;

(b) si y est pris dans la transformation, l'arc $\langle x,y\rangle$ est remplacé par l'arc obtenu en joignant à x le correspondant de y dans la transformation; il est clair que si y est détruit, il n'y aura pas de correspondant de l'arc $\langle x,y\rangle$.

- si $x \in X_2'$

$$\gamma'(x) = Y_1 \cup Y_2 \cup Y_3 \text{ avec:}$$

$$Y_1 = \{y \in X \setminus X_1';\ \exists \langle u,v\rangle \in x \quad y \in \gamma(u)\}$$

Cela signifie que si <x,y> est un arc de G, si y n'est pas pris dans la transformation, on obtient l'arc construit en joignant le correspondant de x à y.

$Y_2 = \{\alpha \in X_2'; \exists <u,v> \in x \; \exists <y,z> \in \alpha \quad z \in \gamma_2(v)\}$

Les arcs de G_2, ou ce qui leur correspond, doivent se retrouver dans G'.

$Y_3 = \{\alpha \in X_2'; \; \exists <u,v> \in x \; \exists <y,z> \in \alpha$

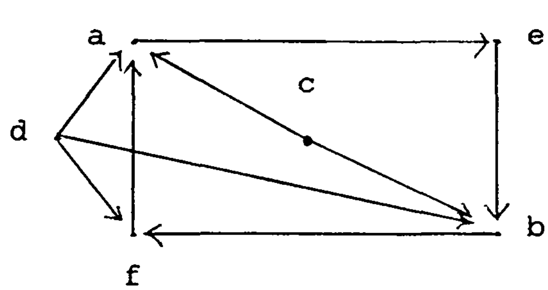
$\qquad y \in \gamma(u)$ et $\psi^{-1}(y) \cap \gamma_1\psi^{-1}(u) = \emptyset\}$

Si x et y sont deux sommets pris dans la transformation, mais l'arc <x,y> n'a pas de correspondant dans X_1, alors le lien entre x et y est maintenu entre le correspondant de x et le correspondant de y.

On s'aperçoit qu'un arc de G ne peut être détruit que si sa destruction est explicitement mentionnée dans la règle de transformation, ou si la destruction de son origine ou de son extrémité est mentionnée

Exemples: 1. On se donne le graphe G (figure 3.2) et la règle $<G_1,G_2,\zeta>$ (figure 3.3).

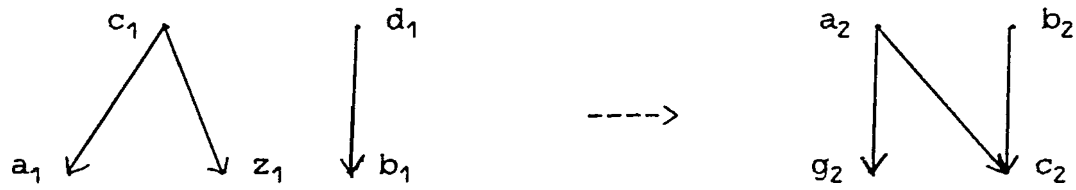
L'application ψ (figure 3.4) détermine un R-morphisme de G_1 dans G.



G

x	a	b	c	d	e	f
$\gamma(x)$	{e}	{f}	{a,b}	{a,b,f}	{b}	{a}

---------------------------figure 3.2---------------------------



G_1 G_2

x	a_1	b_1	c_1	d_1	z_1
$\gamma_1(x)$	$\emptyset$	$\emptyset$	$\{a_1,z_1\}$	$\{b_1\}$	$\emptyset$

x	a_2	b_2	c_2	g_2
$\gamma_2(x)$	$\{c_2,g_2\}$	$\{c_2\}$	$\emptyset$	$\emptyset$

x	a_1	b_1	c_1	d_1	z_1
$\zeta(x)$	a_2	b_2	c_2	-	a_2

--------------------------figure 3.3--------------------------

x	a_1	b_1	c_1	d_1	z_1
$\psi(x)$	a	b	c	d	b

--------------------------figure 3.4--------------------------

On a:

$X_1' = \{a,b,c,d\}$

$\Omega = \{<a,a_2>,<b,a_2>,<b,b_2>,<c,c_2>\}$

Les éléments de X_2' sont par conséquent les suivants:

$a' = \{<a,a_2>,<b,a_2>,<b,b_2>\}$

$c' = \{<c,c_2>\}$

$g' = \{<\epsilon,g_2>\}$

Et, comme $X\backslash X_1' = \{e,f\}$, on a:

$X' = \{a',c',g',e,f\}$

Il reste à définir l'application γ':

$\gamma'(e) = \{a'\}$

$\gamma'(f) = \{a'\}$
$\gamma'(a') = \{e,f,c',g'\}$
$\gamma'(c') = \emptyset$
$\gamma'(g') = \emptyset$
On obtient le graphe G' de la figure 3.5.

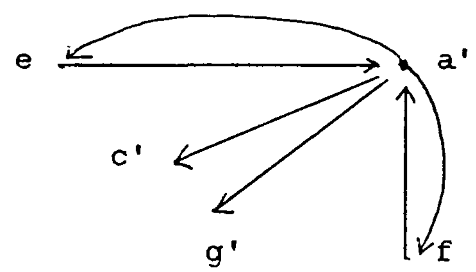
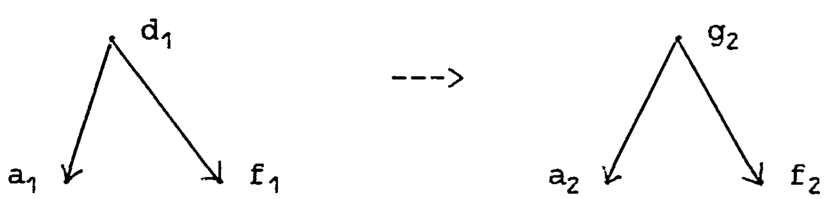


figure 3.5

2. On se donne toujours le graphe G de la figure 3.2, mais cette fois la règle $<G_1,G_2,\zeta>$ de la figure 3.6. On obtient par transformation le graphe G' de la figure 3.7.



$\zeta(a_1) = a_2$ et $\zeta(f_1) = f_2$

figure 3.6

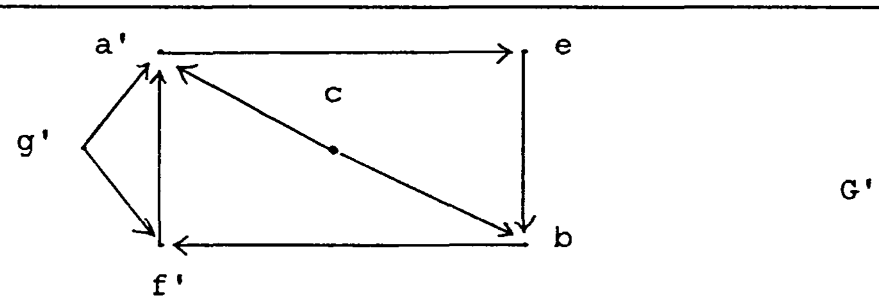


figure 3.7

Notons que l'arc <f,a> non impliqué dans la transformation garde un correspondant <f',a'> dans G'.

<u>Propriété</u>: Il existe un R-morphisme de graphes de G_2 dans G'.

<u>Preuve</u>: A tout élément y de X_2 correspond un et un seul élément α de X_2' tel qu'il existe un couple de la forme <x,y> qui appartienne à α. Ceci définit une application de X_2 dans X'; notons cette application ψ'.

Il est clair que, par construction de γ', $\gamma_2(y)$ a pour image par ψ' un sous-ensemble de $\gamma'\psi'(y)$.

<u>Remarque</u>: A tout élément x de $\psi\zeta^{-1}(X_2)$ on peut associer un élément de X_2': l'élément unique auquel appartient un couple de la forme <x,y>. Soit $\zeta'(x)$ cet élément. En étendant ζ' aux éléments de $X \setminus X_1'$ par l'application identique, on construit une fonction (non partout définie dans le cas général) de X dans X'. Ce qui fait que l'on peut représenter la dérivation directe selon le schéma de la figure 3.8, où ψ et ψ' sont des R-morphismes de graphes, et où ζ et ζ' sont des fonctions.

$$\begin{tikzcd} G_1 \arrow[r, "\psi"] \arrow[d, "\zeta"'] & G \arrow[d, "\zeta'"] \\ G_2 \arrow[r, "\psi'"'] & G' \end{tikzcd}$$

<u>figure 3.8</u>

D'autre part on peut définir une relation d'équivalence entre éléments de $\psi\zeta^{-1}(X_2)$: deux éléments sont équivalents si et seulement si leur image par ζ' est identique. On notera Cl(x) la classe d'équivalence à laquelle appartient un élément x.

De même on peut définir une relation d'équivalence entre éléments de X_2 selon leur image par ψ'. On notera $Cl_2(y)$ la classe d'équivalence à laquelle appartient un élément y.

Propriété: Si ψ est une application injective, ψ' est aussi une application injective.

Preuve: Si ψ' n'est pas une application injective, cela signifie qu'il existe deux couples $<u,v>$ et $<x,y>$ appartenant à un même α de X_2' et tels que v soit différent de y. Il existe donc une suite:

$<u_0,v_0>,<u_1,v_1>,\ldots,<u_p,v_p>$ avec

$u_0 = u,\ v_0 = v,\ u_p = x,\ v_p = y$ et

$\forall\ i \in [p]\quad u_i = u_{i-1}$ ou $v_i = v_{i-1}$

Dans cette suite, il y a forcément un i tel que:

$v_i \neq v_{i-1}$ et $u_i = u_{i-1}$

Donc u_i est image de deux éléments distincts de X_1, ce qui est contradictoire avec le fait que ψ est une application injective.

3.1.4 Cas où il n'y a pas de fusions de sommets:

Si ζ est une fonction injective, cela signifie que la transformation ne produit pas de fusion de sommets. On peut alors identifier les éléments de X_2 qui sont images d'éléments de X_1 aux éléments de X_1 dont ils sont images. Auquel cas, il suffira d'écrire la règle sous la forme $R = <G_1,G_2>$.

Du coup, un élément x de X_2' a l'une des deux formes suivantes:

$x = \{\epsilon,v\}$ avec $v \in X_2\backslash X_1$

ou $x = \{<y,v_1>,<y,v_2>,\ldots,<y,v_q>\}$

On peut dans ce cas assimiler x, soit à l'élément v de $X_2\backslash X_1$ dans le premier cas, soit à l'élément y de X dans le deuxième cas. Ceci à condition que X et $X_2\backslash X_1$ soient disjoints, cas auquel on peut toujours se ramener. On pose alors:

$X' = [X\backslash(X_1\backslash X_2)] \cup (X_2\backslash X_1)$

et l'application γ' peut être définie d'une nouvelle manière:

- si $x \in X\backslash\psi(X_1)$

  $\gamma'(x) = \gamma(x) \cap X'$

- si $x \in X_2\backslash X_1$

  $\gamma'(x) = [\gamma_2(x)\backslash X_1)] \cup \psi(\gamma_2(x) \cap X_1)$

- si $x \in \psi(X_1 \cap X_2)$

$\gamma'(x) = Y_1 \cup Y_2 \cup Y_3$ avec:

$Y_1 = \gamma(x) \cap [X \backslash \psi(X_1)]$

$Y_2 = [\gamma_2\psi^{-1}(x) \backslash X_1] \cup \psi[\gamma_2\psi^{-1}(x) \cap X_1]$

$Y_3 = [\gamma(x) \backslash \psi\gamma_1\psi^{-1}(x)] \cap \psi(X_1 \cap X_2)$

On vérifie aisément que l'on est ramené aux définitions du paragraphe précédent.

3.1.5 Règles conservatrices

Définition: Une règle $R = \langle G_1, G_2, \zeta\rangle$ est une règle **conservatrice** si et seulement si ζ est une application injective qui soit un R-morphisme de graphes.

On peut alors noter la règle $R = \langle G_1, G_2\rangle$.

Propriété: Si $R = \langle G_1, G_2\rangle$ est une règle conservatrice, alors G est un sous-graphe partiel de son transformé G' par R.

Preuve: Dans ce cas, $X_1 \backslash X_2$ est vide, et donc:

$X' = X \cup (X_2 \backslash X_1)$

Par conséquent, si $x \in X \backslash \psi(X_1)$, on a:

$\gamma'(x) = \gamma(x)$

Si $x \in \psi(X_1 \cap X_2)$, autrement dit si $x \in \psi(X_1)$, soit y appartenant à $\gamma(x)$. Deux cas se présentent:

(1) si $y \notin \psi(X_1)$ alors y appartient à Y_1, et par conséquent à $\gamma'(x)$;

(2) si $y \in \psi(X_1)$, alors soit $z \in \psi^{-1}(y)$

- si $z \in \gamma_1\psi^{-1}(x)$

alors $z \in \gamma_2\psi^{-1}(x)$ puisque l'application ζ est un R-morphisme;

d'où: $\psi(z) \in Y_2$

et $y \in \gamma'(x)$;

- si $z \notin \gamma_1\psi^{-1}(x)$

alors $y \notin \psi\gamma_1\psi^{-1}(x)$

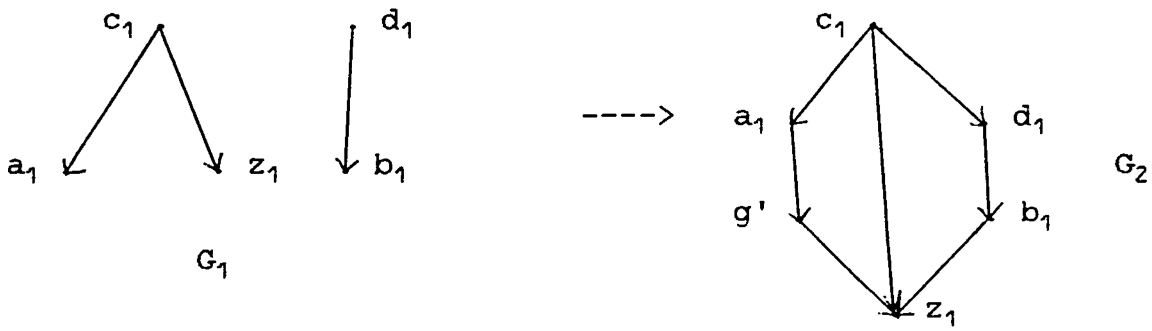
donc $y \in Y_3$ et $y \in \gamma'(x)$.

Dans tous les cas donc on a:

$y \in \gamma(x) \implies y \in \gamma'(x)$

ce qui démontre la propriété.

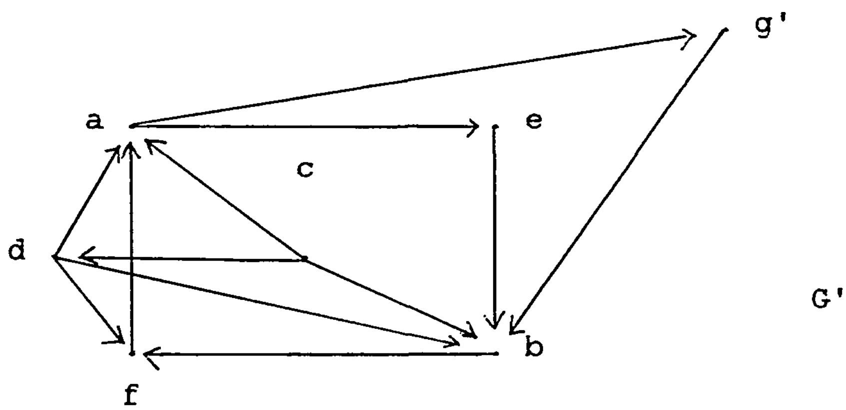
Exemple: On reprend le graphe G de la figure 3.2 et on lui applique la règle R = <G_1,G_2> représentée en figure 3.9. On obtient alors le graphe transformé G' de la figure 3.10.



x	a_1	b_1	c_1	d_1	z_1
$\gamma_1(x)$	$\emptyset$	$\emptyset$	$\{a_1,z_1\}$	$\{b_1\}$	$\emptyset$

x	a_1	b_1	c_1	d_1	z_1	g'
$\gamma_2(x)$	$\{g'\}$	$\{z_1\}$	$\{a_1,d_1,z_1\}$	$\{b_1\}$	$\emptyset$	$\{z_1\}$

--------------------------figure 3.9--------------------------



--------------------------figure 3.10--------------------------

Remarques: 1. On peut, afin d'abréger l'écriture, ne pas indiquer dans G_2 les arcs qui sont déjà dans G_1, ni les sommets de G_1 qui ne sont ni origine ni extrémité d'un arc de G_2 qui n'est pas dans G_1.

Il suffira de préciser alors que l'on applique la règle **de manière conservatrice**. Cela signifie que:

$$X' = X \cup X_2 \setminus X_1$$

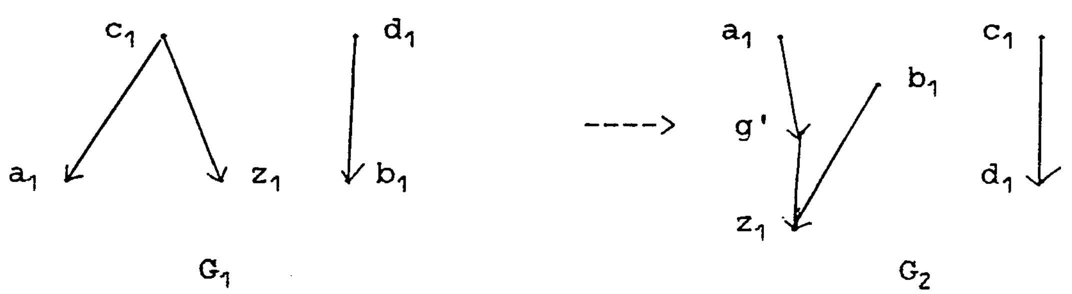
l'application γ' étant définie de la manière suivante:

- si $x \notin \psi(X_1)$ alors $\gamma'(x) = \gamma(x)$
- si $x \in \psi(X_1)$ alors soit y l'élément de X_1 dont x est image par ψ;

$$\gamma'(x) = \gamma(x) \cup \psi[\gamma_2(y) \cap X_1] \cup \gamma_2(y) \setminus X_1$$

Il est clair que l'on obtient le même transformé que si l'on applique de manière non conservatrice la règle dont l'écriture n'est pas abrégée.

Exemple: La règle de l'exemple précédent est représentée de manière abrégée par la figure 3.11.



x	a_1	b_1	c_1	d_1	z_1
$\gamma_1(x)$	$\emptyset$	$\emptyset$	$\{a_1, z_1\}$	$\{b_1\}$	$\emptyset$

x	a_1	b_1	c_1	d_1	z_1	g'
$\gamma_2(x)$	$\{g'\}$	$\{z_1\}$	$\{d_1\}$	$\emptyset$	$\emptyset$	$\{z_1\}$

--------------------------figure 3.11--------------------------

2. Pour le moment l'application ψ est un R-morphisme de graphes, ce qui englobe tous les cas, sauf celui de l'agrandissement. Si on veut que ψ puisse être un agrandissement, il est clair que l'on ne peut modifier simplement les chemins de G en fonction de transformation d'arcs de G_1. C'est pourquoi l'application des

règles n'aura de sens dans ce cas que si les règles sont conservatrices. On parlera dans ce cas de A-dérivation directe.

3.2 Conservation du caractère sans circuit

Dans cette partie nous poursuivons les études entreprises dans [Cori, 80] et dans [Cori, 82a].

3.2.1 Préliminaires

Définition: Une règle de réécriture de graphes sans circuit sera constituée d'un triplet $R = \langle G_1, G_2, \zeta\rangle$ dans laquelle $G_1 = \langle X_1, \gamma_1\rangle$ et $G_2 = \langle X_2, \gamma_2\rangle$ sont deux graphes sans circuit et ζ une fonction de X_1 dans X_2.

En effet, soit $G = \langle X, \gamma\rangle$ un graphe sans circuit à transformer par la règle R. Il est clair que si G_1 admet un circuit, on ne peut trouver de R-morphisme de X_1 dans X. Par ailleurs, si G_2 admet un circuit, il y aura également un circuit dans le graphe transformé G', puisqu'il existe un R-morphisme de G_2 dans G'.

Si l'on a une règle de réécriture de graphes sans circuit $R = \langle G_1, G_2, \zeta\rangle$ et un R-morphisme ψ de X_1 dans X, on va séparer l'ensemble X en trois parties en posant:

$$X_1' = \psi(X_1)$$

$$Z = (X \setminus X_1') \cap \hat{\gamma}(X_1')$$

$$W = (X \setminus X_1') \setminus Z$$

Il est clair que ces trois ensembles sont disjoints et que X est formé de leur réunion.

On aura évidemmment:

$$X' = W \cup Z \cup X_2'$$

les trois ensembles W, Z et X_2' étant encore disjoints.

Lemme: On a les relations suivantes:

$$\gamma'(Z) \subset Z \cup X_2'$$

et $\gamma'(X_2') \subset Z \cup X_2'$

Preuve: Soit x appartenant à Z; on a:

$$\gamma'(x) \cap (X \setminus X_1') = \gamma(x) \setminus X_1'$$

et, si $y \in \gamma(x)$ et $x \in \hat{\gamma}(X_1')$

y appartient à $\hat{\gamma}(X_1')$ et par conséquent à Z.

Soit x appartenant à X_2'; on a:

$\gamma'(x) = Y_1 \cup Y_2 \cup Y_3$

Y_2 et Y_3 étant inclus dans X_2' et:

$Y_1 = \{y \in X \setminus X_1';\ \exists <u,v> \in x \quad y \in \gamma(u)\}$

Pour tout y de Y_1 il existe donc un u de X_1' tel que y appartient à $\gamma(u)$. y appartient donc à $\hat{\gamma}(X_1')$ et par là même à Z.

3.2.2 <u>P-dérivation</u>

<u>Propriété</u>: Soient $R = <G_1,G_2,\zeta>$ une règle dans laquelle $G_1 = <X_1,\gamma_1>$ et $G_2 = <X_2,\gamma_2>$ sont deux graphes sans circuit, $G = <X,\gamma>$ un graphe sans circuit et ψ une application de X_1 dans X_2. Si ψ est un P-morphisme de graphes injectif, alors le transformé G' de G par R pour ψ est également un graphe sans circuit.

<u>Preuve</u>: On va d'abord montrer que:

$\gamma'(Z) \subset Z$

On a en effet, si x appartient à Z:

$\gamma'(x) \cap X_2' = \{\alpha \in X_2';\ \exists <y,z> \in \alpha \quad y \in \gamma(x)\}$

y devant appartenir à X_1'; et comme x appartient à $\hat{\gamma}(X_1')$

$\exists\ u \in X_1' \quad x \in \hat{\gamma}(u)$

$\exists\ y \in X_1' \quad y \in \gamma(x)$

Il existe donc un chemin entre u et y. Or u et y sont images chacun d'au moins un sommet de X_1. Soient v et z deux sommets de X_1 tels que:

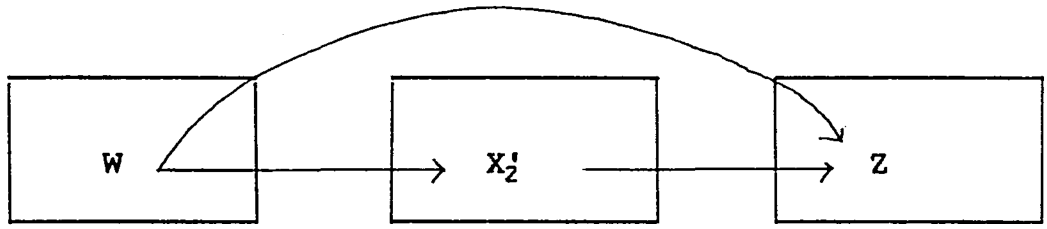
$\psi(v) = u$ et $\psi(z) = y$

Comme ψ est un P-morphisme tout chemin entre u et y de $\psi(X_1)$ doit être l'image par ψ d'un chemin de G_1. Cela implique que x appartient à X_1', ce qui est contraire à l'hypothèse.

On en déduit que les arcs possibles dans G' entre les sommets de W, Z ou X_2' se présentent selon le schéma de la figure 3.12.

Un circuit dans G' ne peut par conséquent exister qu'entre des sommets appartenant à un même ensemble W, Z ou X_2'.

Or, il ne peut y avoir de circuit entre éléments de W ou de Z car cela entraînerait l'existence d'un circuit dans G.



-------------------------figure 3.12-------------------------

Les seuls circuits possibles sont donc entre éléments de X'_2. Soit x appartenant à X'_2. On a:

$\gamma'(x) \cap X'_2 = Y_2 \cup Y_3$

Soit $\alpha \in Y_3$; il existe forcément $\langle u,v\rangle \in x$ et $\langle y,z\rangle \in \alpha$ tels que $y \in \gamma(u)$ et $\psi^{-1}(y) \cap \gamma_1\psi^{-1}(u) = \emptyset$.

Il existe par conséquent t et w appartenant à X_1 tels que:

$u = \psi(t) \qquad y = \psi(w)$

$v = \zeta(t) \qquad z = \zeta(w)$

w ne peut appartenir à $\gamma_1(t)$; or $\psi(w)$ appartient à $\gamma\psi(t)$; ceci est contradictoire avec le fait que ψ est un P-morphisme.

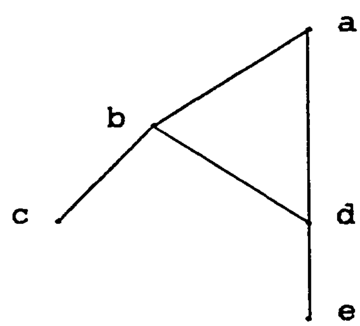
Donc $\gamma'(x) \cap X'_2 = Y_2$

et tout arc joignant deux sommets de X'_2 est image, par le R-morphisme ψ', d'un arc joignant deux sommets de X_2. Comme ψ' est injective, s'il y avait un circuit entre sommets de X'_2, il y aurait également un circuit dans G_2, ce qui est contradictoire avec le fait que R soit une règle de réécriture de graphes sans circuit.

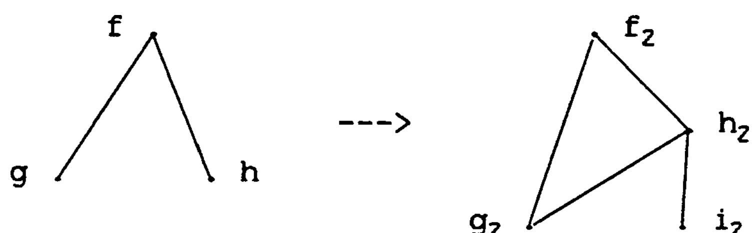
Corollaire: Il existe un P-morphisme de G_2 dans G'.

Preuve: Dans la démonstration de la propriété précédente, on vient de montrer que tout chemin entre sommets de X'_2 est image d'un chemin de G_2. ψ' est donc bien un P-morphisme.

Exemple: On considère le graphe sans circuit G de la figure 3.13, ainsi que la règle $R = \langle G_1,G_2,\zeta\rangle$ de la figure 3.14. On considère d'autre part les 3 applications ψ_1, ψ_2 et ψ_3 de X_1 dans X de la figure 3.15. ψ_1 n'est pas un P-morphisme, ψ_2 est un P-morphisme non injectif et ψ_3 est un P-morphisme injectif: c'est uniquement



---------------------------figure 3.13---------------------------



x	f	g	h
$\zeta(x)$	f_2	g_2	h_2

---------------------------figure 3.14---------------------------

x	f	g	h
$\psi_1(x)$	a	b	d

x	f	g	h
$\psi_2(x)$	d	e	e

x	f	g	h
$\psi_3(x)$	b	c	d

---------------------------figure 3.15---------------------------

dans l'application de la règle R pour ψ_2 que le graphe transformé reste sans circuit.

3.2.3 Règles libres

Propriété: Soit R = $\langle G_1,G_2,\zeta\rangle$ une règle dans laquelle G_1 = $\langle X_1,\gamma_1\rangle$ et G_2 = $\langle X_2,\gamma_2\rangle$ sont deux graphes sans circuit. Pour que quel que soit le graphe sans circuit G = $\langle X,\gamma\rangle$ et quel que soit le R-morphisme ψ de X_1 dans X, le transformé G' de G par R pour ψ reste sans circuit, il faut et il suffit que la règle R vérifie la condition suivante:

$\forall\ y,z \in X_1 \cap \zeta^{-1}(X_2) \quad z \notin \hat{\gamma}_1(y) \Longrightarrow \zeta(z) \notin \hat{\gamma}_2\zeta(y)$

Preuve: Soit $R = \langle G_1,G_2\rangle$ une règle ne vérifiant pas la condition ci-dessus. Cela veut dire qu'il existe y et z appartenant à $\zeta^{-1}(X_2)$ tels que:

$z \notin \hat{\gamma}_1(y)$ et $\zeta(z) \in \hat{\gamma}_2\zeta(y)$

Soit $G = \langle X_1,\gamma\rangle$ le graphe tel que:

$\forall\ x \in X_1\backslash\{z\} \quad \gamma(x) = \gamma_1(x)$

$\gamma(z) = \gamma_1(z) \cup \{y\}$.

Il est clair que G ne comporte pas de circuit.

L'application identique détermine bien un R-morphisme de G_1 dans G.

Dans G', obtenu par application de la règle R à G pour l'application identique, on aura:

$\psi'\zeta(z) \in \hat{\gamma}'\psi'\zeta(y)$

et $\psi'\zeta(y) \in \gamma'\psi'\zeta(z)$,

c'est-à-dire qu'il y aura un circuit.

Réciproquement, le lemme 3.2.1 indique que s'il y a des circuits dans G' ils peuvent être:

- soit entre sommets de X_2' et sommets de Z;
- soit entre sommets de X_2'.

Dans le premier cas, il faut qu'il existe deux couples $\langle u,v\rangle$ et $\langle x,y\rangle$ de Ω tels que

$y \in \hat{\gamma}_2(v)$ et $x \notin \hat{\gamma}(u)$

sinon le circuit serait déjà dans G.

Il existe donc w et z appartenant à X_1 tels que:

$x = \psi(w) \qquad u = \psi(z)$

$y = \zeta(w) \qquad v = \zeta(z)$

w ne peut appartenir à $\hat{\gamma}_1(z)$, sinon, comme ψ est un R-morphisme, x appartiendrait à $\hat{\gamma}(u)$.

Donc y n'appartient pas à $\hat{\gamma}_2(v)$, d'après l'hypothèse.

Supposons maintenant (deuxième cas) qu'il existe un circuit dans G' comprenant uniquement des sommets de X_2'. Dans ce cas il existe une suite de couples de Ω:

$\langle u_0,v_0\rangle,\langle u_1,v_1\rangle,\ldots,\langle u_{2p},v_{2p}\rangle$

telle que $u_0 = u_{2p}$, $v_0 = v_{2p}$ et, pour tout i appartenant à [p]:

$v_{2i} \in \hat{\gamma}_2(v_{2i-1})$

$u_{2i+1} \in \hat{\gamma}(u_{2i})$

$\forall w_{2i} \in \psi^{-1}(u_{2i}) \quad \forall w_{2i+1} \in \psi^{-1}(u_{2i+1}) \quad w_{2i+1} \notin \hat{\gamma}_1(w_{2i})$

Cela entraîne, d'après l'hypothèse de départ, que w_{2i} appartient à $\hat{\gamma}_1\psi^{-1}(u_{2i-1})$ et donc, comme ψ est un R-morphisme, que u_{2i} appartient à $\hat{\gamma}(u_{2i-1})$.

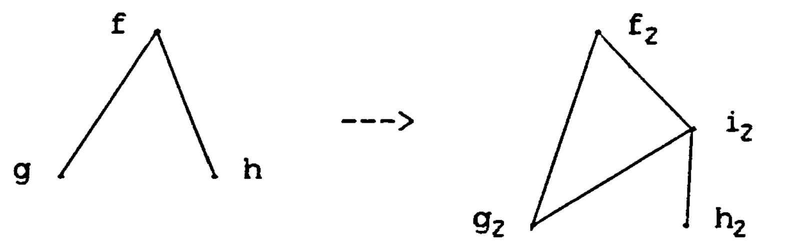
Finalement, pour tout i,

$u_{2i+1} \in \hat{\gamma}(u_{2i-1})$

ce qui entraîne qu'il y a un circuit dans G.

Définition: On dira dans ce cas que la règle est **libre**.

Exemple: On reprend le graphe G de la figure 3.13. La règle de la figure 3.14 n'était pas une règle libre. Considérons la règle R de la figure 3.16: c'est une règle libre et le transformé de G par cette règle, aussi bien pour l'application ψ_1 que pour l'application ψ_2 de la figure 3.15 reste sans circuit.



x	f	g	h
$\zeta(x)$	f_2	g_2	h_2

figure 3.16

3.3 Grammaires de graphes ordonnés

3.3.1 Les conditions d'application des règles

Définition: Une règle de réécriture de graphes ordonnés sera constituée d'un triplet $R = \langle G_1, G_2, \zeta \rangle$ dans laquelle $G_1 = \langle X_1, \delta_1 \rangle$ et $G_2 = \langle X_2, \delta_2 \rangle$ sont deux graphes ordonnés et ζ une fonction de X_1 dans X_2.

Soit $G = \langle X, \delta \rangle$ un GO, et R une règle de réécriture. On cherche à construire le GO $G' = \langle X', \delta' \rangle$ transformé de G par la règle R.

Dans le cas des graphes ordonnés, on ne peut se contenter d'appliquer les règles de la manière décrite précédemment. En effet, un problème spécifique se pose: comment déterminer l'ordre entre successeurs d'un même sommet du graphe transformé quand l'ensemble des successeurs de ce sommet est construit comme l'union de deux ensembles. On ne sait ce que devient cette union sur des suites. Il n'y a pas de raison d'imposer un ordre arbitraire.

C'est pourquoi on voudra que l'ordre des successeurs d'un sommet de G' soit déterminé:

- soit par l'ordre de ses successeurs dans G s'il s'agit d'un sommet non touché par la transformation;
- soit par l'ordre de ses successeurs dans G_2 s'il s'agit d'un sommet qui provient de la transformation.

On se fixe une contrainte globale: qu'aucun $\delta'(x)$ ne soit construit par union ou par extraction. Il faudra par conséquent que tous les arcs originaires d'un sommet donné touchés par la transformation soient touchés par la transformation, ou aucun. C'est-à-dire que ψ devra être un Q-morphisme de GO de G_1 dans G. La dérivation directe est donc une Q-dérivation.

Par ailleurs, si l'on veut que l'ordre des successeurs d'un sommet non impliqué dans la tansformation soit inchangé, il faut éviter que le successeur d'un sommet non impliqué dans la transformation puisse être détruit. Cela s'écrit:

(i) $\forall\ x \in X_1 \quad \delta^{-1}\psi(x)\backslash X_1' \neq \emptyset \implies x \in \zeta^{-1}(X_2)$

Pour éviter d'avoir une concurrence entre δ et δ_2 pour un sommet donné de G', il faut également que la condition suivante soit réalisée:

(ii) $\forall\ x \in Ter(G_1) \cap \zeta^{-1}(X_2\backslash Ter(G_2)) \quad \delta\psi(x) = \lambda$

De plus, il ne faut pas que le successeur d'un sommet impliqué dans la transformation mais tel que les arcs dont il est origine ne sont pas impliqués dans la transformation puisse être détruit. Cela s'écrit:

(iii) $\forall\ x \in X_1 \backslash \zeta^{-1}(X_2)\ \forall\ y \in \zeta^{-1}(X_2)$

$$\delta_1(y) \neq \lambda \text{ ou } \delta_2\zeta(y) \neq \lambda \text{ ou } \psi(x) \notin \psi\delta(y)$$

Il reste à voir ce qui se passe dans le cas où l'application ψ n'est pas injective, ou dans le cas où il y a des fusions de sommets induites par la règle. Il ne faut pas que $\delta(x)$ pour un sommet de G' soit construit par l'union de deux $\delta_2(y)$:
(iv) $\forall\ x \in X_2\ \forall\ y \in Cl_2(x)\quad Cl_2\delta_2(x) = Cl_2\delta_2(y)$
(l'égalité étant une égalité entre suites d'ensembles)

Enfin, il ne faut pas, si δ' est construit à partir de δ, qu'il y ait concurrence entre deux $\delta(x)$. D'où la condition:
(v) $\forall\ x \in Ter(G_1) \cap \zeta^{-1} Ter(G_2)\quad \forall\ y \in Cl\psi(x)\ \ \delta\psi(x) = \delta(y)$

Ces cinq conditions sont des conditions suffisantes à l'application d'une règle à un graphe ordonné, de telle manière que la construction du graphe transformé soit compatible avec ce qui a été défini pour les graphes non ordonnés.

3.3.2 <u>La dérivation directe</u>

On pose encore $X' = (X \backslash X_1') \cup X_2'$
- si $x \in X \backslash X_1'$
soit $\delta(x) = x_1 \ldots x_p$;
on pose: $\delta'(x) = x_1' \ldots x_p'$
chaque x_i' étant défini de la manière suivante:
si $x_i \notin X_1'$ alors $x_i' = x_i$;
sinon, il existe un et un seul α appartenant à X_2' tel qu'il existe un couple $<x_i,u>$ appartenant à α (en raison de la condition (i)); on pose $x_i' = \alpha$
- sinon $x \in X_2'$;
soit $<u,v> \in x$; la longueur de $\delta_2(v)$ est indépendante du choix du couple $<u,v>$ de x (en raison de la condition (iv)). Si $\delta_2(v)$ est non vide, on pose:

$$\delta_2(v) = v_1 \ldots v_q$$

Chacun des v_i appartient à un et un seul $x_i = \psi'(v_i)$ de X_2'.
On pose:

$$\delta'(x) = x_1 \ldots x_q$$

Ce résultat est indépendant du choix du couple <u,v>, en raison de la condition (iv).
Si $\delta_2(v) = \lambda$, deux cas se présentent. Ou bien $\zeta^{-1}Cl_2(v)$ est inclus dans $Ter(G_1)$, auquel cas $\delta(u)$ est indépendant du couple <u,v> de x (d'après la condition (v)).

Soit $\delta(u) = u_1 \ldots u_p$

On pose: $\delta'(u) = u_1' \ldots u_p'$ avec:
si u_i n'appartient pas à X_1', $u_i' = u_i$
si u_i appartient à X_1', il existe un et un seul α appartenant à X_2' tel qu'il existe un couple $<u_i,u>$ appartenant à α (en raison de la condition (iii)); on pose $u_i' = \alpha$
Dans le cas enfin où $\zeta^{-1}Cl_2(v)$ n'est pas inclus dans $Ter(G_1)$, on pose: $\delta'(x) = \lambda$

3.3.3 <u>Règles propres</u>

Les conditions d'application des règles aux graphes ordonnés dépendent du graphe G auxquelles on veut les appliquer et de l'application ψ. Il existe des règles qui vérifient ces conditions quels que soient le graphe G et le Q-morphisme ψ. Ce sont les règles qui vérifient les propriétés suivantes:

(i) $\zeta^{-1}(X_2) = X_1$
(ii) $\forall x \in X_1 \quad \delta_1(x) = \lambda \implies \delta_2\zeta(x) = \lambda$
(iii) $\forall x \in X_1 \quad \delta_2\zeta(x) = \zeta\delta_1(x)$
(iv) $\forall x,y \in Ter(G_1) \quad x \neq y \implies \zeta(x) \neq \zeta(y)$

La condition (i) implique les conditions (i) et (iii) de 3.3.2. La condition (ii) implique la condition (ii). La condition (iii) implique la condition (iv). Et la condition (iv) implique la condition (v).

<u>Définition</u>: On dira dans ce cas que la règle est **propre**.

3.3.4 <u>Grammaires de graphes ordonnés sans circuit</u>
<u>Propriété</u>: Une règle propre est une règle libre.

<u>Preuve</u>: Si $R = <G_1,G_2,\zeta>$ est une règle propre, ζ est un morphisme de graphes. C'est par conséquent un P-morphisme et la condition

relative aux règles libres est vérifiée.

3.4 Grammaires de graphes étiquetés

3.4.1 Grammaires de GE

Définition: Une règle de réécriture de graphes étiquetés sera constituée d'un triplet $R = \langle H_1, H_2, \zeta \rangle$ dans laquelle $H_1 = \langle X_1, \gamma_1, \xi_1 \rangle$ et $H_2 = \langle X_2, \gamma_2, \xi_2 \rangle$ sont deux graphes étiquetés et ζ une fonction de X_1 dans X_2.

Soit $H = \langle X, \gamma, \xi \rangle$ un GE, et R une règle de réécriture. Soit ψ un R-morphisme de GE (2.5.2) de H_1 dans H. On cherche à construire le GE $H' = \langle X', \gamma', \xi' \rangle$ transformé de H par la règle R. Le problème nouveau qui se pose dans la construction de H' est la détermination de ξ'.

Si x appartient à $X \setminus X_1'$, on aura évidemment:

$$\xi'(x) = \xi(x)$$

Si x appartient à X_2', on décide que si x est l'image de sommets de X_2 pour lesquels ξ_2 n'est pas définie, ces sommets correspondant à des sommets de X_1 pour lesquels ξ_1 n'est pas non plus définie, c'est l'application ξ qui détermine $\xi'(x)$. Sinon, c'est l'application ξ_2.

Il peut alors y avoir des problèmes de concurrence entre deux valeurs prises par ξ_2 quand ψ n'est pas injective, ou des concurrences entre deux valeurs prises par ξ quand il y a des fusions de sommets. C'est pourquoi on donne les deux conditions suivantes:

(i) $\forall\, x \in X_2\ \forall\, y \in Cl_2(x)\quad \xi_2(x) = \xi_2(y)$

(ii) $\forall\, x \in (X_1 \setminus X_1^0) \cap \zeta^{-1}(X_2 \setminus X_2^0) \cap \psi^{-1}(X^0)$

$$\forall\, y \in Cl\psi(x) \cap X^0 \quad \xi\psi(x) = \xi(y)$$

Si ces conditions sont vérifiées, on peut définir l'application ξ' de la manière suivante:

$\forall\, x \in X \setminus X_1'\quad \xi'(x) = \xi(x)$

$\forall\, x \in X_2^0\quad \xi'\psi'(x) = \xi_2(x)$

$\forall\, x \in (X_1 \setminus X_1^0) \cap \zeta^{-1}(X_2 \setminus X_2^0) \cap \psi^{-1}(X^0)\quad \xi'\psi'\zeta(x) = \xi\psi(x)$

Les sommets de H' pour lesquels ξ' n'est pas défini de cette manière n'appartiennent pas au domaine de définition de H'.

3.4.2 Grammaires de GME

Supposons que H, H_1 et H_2 soient des graphes multiétiquetés. On a alors:

$\xi = <\xi^1,\dots,\xi^p>$

avec, pour tout i appartenant à [p], une application:

$\xi^i: X^i \dashrightarrow L^i$

Il n'est pas nécessaire que H_1 reçoive les mêmes étiquettes que H. Il est en revanche nécessaire que H_1 reçoive un sous-ensemble des étiquettes reçues par H. On aurait alors:

$\xi_1 = <\xi_1^1,\xi_1^2,\dots\xi_1^q>$

et pour tout j appartenant à [q] il existe i_j appartenant à [q] tel que

$\xi_1^j : X_1^j \dashrightarrow L^{i_j}$

ψ devra être un $(<1,\dots,q>,<i_1,\dots,i_q>)$-R-morphisme de GME. On parlera alors de $<i_1,\dots,i_q>$-R-dérivation directe.

H_2 devra en revanche recevoir toutes les étiquettes reçues par H. La construction de la suite d'applications ξ' s'effectuera comme indiqué au paragraphe précédent.

3.4.3 Etiquetage des arcs

Soit $H = <X,\delta,\xi>$ un graphe ordonné étiqueté dans lequel ξ est une application arc-étiquetante sur L, et soit $R = <H_1,H_2,\zeta>$ une règle de réécriture dans laquelle ξ_1 et ξ_2 sont également des applications arc-étiquetantes sur L.

L'application de la règle à H nécessite que soient vérifiées les conditions (i) à (v) de 3.3.1, ainsi que les conditions (i) et (ii) de 3.4.1, puisqu'on a affaire à un graphe qui est à la fois ordonné et étiqueté.

Toutefois, on ne peut appliquer la dérivation directe telle qu'elle a été définie pour les graphes ordonnés, en étiquetant les sommets comme indiqué en 3.4.1. L'application ξ' construite ne serait en effet plus arc-étiquetante. Cela provient du fait que si un sommet terminal de H_1 est transformé en un sommet terminal de H_2, son correspondant dans H conserve dans la transformation la suite de successeurs qu'il a dans H. Alors que d'après la définition de ξ' de 3.4.1, ce correspondant devra être étiqueté par λ, qui est l'étiquette reçue par un sommet terminal de H_2.

C'est pourquoi il faut modifier pour les applications arc-étiquetantes la condition (ii) de 3.4.1, qui devient:

(ii) $\forall\ x \in \mathrm{Ter}(H_1) \cap \zeta^{-1}\mathrm{Ter}(H_2) \quad \forall\ y \in \mathrm{Cl}\psi(x) \quad \xi\psi(x) = \xi(y)$

On peut alors définir l'application ξ' de la manière suivante:

- si $x \in X \backslash X_1'$ $\quad \xi'(x) = \xi(x)$
- sinon, $x \in X_2'$; soit $<u,v>$ appartenant à x.

Si $\delta_2(v)$ est différent de λ, on sait que $\mathrm{Cl}_2(v)$ est indépendant du choix du couple $<u,v>$ et $\xi_2(v)$ est indépendant du choix de v dans sa classe. On peut donc poser:

$$\xi'(x) = \xi_2(v)$$

Si $\delta_2(v) = \lambda$ et $\zeta^{-1}\mathrm{Cl}_2(v)$ est inclus dans $\mathrm{Ter}(H_1)$, $\delta(u) = u_1 \ldots u_p$ est indépendant du choix de $<u,v>$ appartenant à x, de même que $\xi(u)$ d'après la condition que l'on vient de poser. Bien que $\xi_2(v)$ soit défini dans ce cas et égal à λ, on pose

$$\xi'(x) = \xi(u)$$

afin que ξ' soit arc-étiquetante.

Dans le dernier cas, $\delta'(x)$ est égal à λ. On pose donc

$$\xi'(x) = \lambda$$

3.4.4 <u>Grammaires de réseaux</u>

Soit $H = <X,\delta,\xi,\rho>$ un K^1-réseau à transformer par une règle de réécriture de K^1-réseaux $R = <H_1,H_2,\zeta>$.

Les réseaux étant des graphes semi-ordonnés et multiétiquetés, il est clair que les conditions relatives aux graphes ordonnés devront être vérifiées pour certains sommets, et pas pour d'autres, c'est-à-dire par les sommets x tels que $\xi^1\psi(x)$ appartiénne à K^1. De plus, il ne sera pas nécessaire que l'application ψ soit un Q-morphisme: il suffira que ce soit un QR-morphisme de réseaux (2.6.5). On parlera alors de QR-dérivation de réseaux.

Ceci étant posé, en appliquant ce qui a été dit pour la dérivation directe des graphes ordonnés, des graphes multiétiquetés, y compris dans le cas où les arcs sont étiquetés, sur la conservation du caractère sans circuit des graphes dans les transformations, on en déduit la QR-dérivation directe pour les réseaux.

Dans le cas où une règle est appliquée de manière conservatrice (3.1.5), on définit les A-dérivations directes de réseaux.

4 La représentation des données textuelles et des connaissances

Notre objectif est, rappelons-le, la **représentation** de données textuelles en machine sous une forme telle que l'on puisse obtenir la restitution de connaissances sous-jacentes à ces données en réponse à des **questions**.

Le passage par une représentation ne va pas de soi. Certains auteurs [Coulon-Kayser, 81] en effet, préconisent de stocker les textes tels quels et de ne se livrer à des opérations dessus qu'au moment de l'interrogation. A notre sens, l'utilisation d'une représentation permet un gain d'efficacité dans le processus d'interrogation car on peut obtenir une représentation unique de plusieurs textes dont la forme est différente, mais dont le sens peut être considéré comme identique. Et surtout, on peut représenter les questions selon un formalisme analogue à celui des données textuelles de telle manière que l'on puisse extraire des données enregistrées des réponses qui soient **à l'image** des questions.

Comprendre un texte, cela consistera alors à en effectuer la traduction dans ce formalisme intermédiaire, la vérification que le texte a été bien compris ne pouvant se faire que par l'analyse des réponses qui seront fournies aux questions. Mais la compréhension n'est pas un processus univoque. Sans évoquer des textes complexes (discours politique, par exemple) desquels chacun retient ce qu'il a envie de retenir et que chacun interprête comme il a envie de l'interpréter, un énoncé très simple peut être compris plus ou moins **profondément.** Nous poserons par définition que la compréhension d'un énoncé est d'autant plus profonde qu'on peut répondre à son sujet à un plus grand nombre de questions. Cette définition ne se situe nullement sur le terrain de la psychologie cognitive, elle a uniquement une visée pratique.

Supposons que l'on ait pour donnée la phrase suivante:

le tigre a avalé un poulet.

Il n'y aurait pas du tout compréhension si l'on n'était capable de répondre aux questions suivantes:

Qu'est-ce que le tigre a avalé ?

Qui a avalé le poulet ?

Par qui le poulet a-t-il été avalé ?.

Un deuxième pas dans la compréhension est franchi quand on est capable de répondre à:

Qui a avalé un animal ?

Y a-t-il un félin qui a avalé quelque chose ?.

On est alors capable de combiner aux connaissances apportées par l'énoncé des connaissances préalables. La compréhension sera encore plus profonde quand on pourra répondre à la question:

Qu'est devenu le poulet ?

en disant par exemple que *le poulet est mort.* On a une certaine compréhension de la signification du verbe.

Il faut prendre garde au fait qu'une compréhension n'est pas forcément **meilleure** parce qu'elle est plus profonde. Plus la compréhension est profonde et plus le monde auquel elle s'applique se rétrécit: on effectue des choix et, par conséquent, on élimine des interprétations possibles. Si l'on a conclu de l'énoncé pris en exemple que le poulet a été digéré, et donc qu'il n'existe plus, on fait des contre-sens dans la compréhension de contes tels que *Le petit chaperon rouge* puisque selon certaines versions du conte le petit chaperon rouge et sa grand-mère ressortent indemnes du ventre du loup qui les a avalés! Quant au *poulet*, ce n'est pas forcément un *animal*...

C'est pourquoi nous allons donner trois modèles pour la représentation, correspondant à trois niveaux de compréhension, sans privilégier l'un des trois:

(1) un premier modèle qui permet uniquement de représenter des relations entre objets directement tirées d'un texte. Aucune déduction n'est possible: on ne peut ressortir que ce qu'on a entré, éventuellement sous une forme modifiée. Ce modèle, volontairement très restreint, servira de base aux modèles suivants;

(2) un modèle qui sera capable de prendre en compte des "connaissances", essentiellement celles qui pourront être mises sous la forme de relations hiérarchiques entre classe d'objets.

Il sera par conséquent possible de produire des inférences simples à l'aide de ce modèle;

(3) un modèle dans lequel il sera possible de décomposer les verbes et qui permettra par conséquent d'effectuer des déductions plus complexes.

La distinction entre ces trois modèles n'est pas due à la volonté de retrouver la distinction classique entre syntaxe, sémantique et pragmatique, ou toute autre distinction du même type: elle correspond simplement à la complexité des outils informatiques utilisés. Le premier modèle utilise des Σ-GOSME, le deuxième modèle utilise des réseaux, quant au troisième modèle il est à base de règles de réécriture de réseaux.

4.1 Description d'un premier modèle de représentation des données textuelles

4.1.1 Introduction

On représente les données textuelles à un certain degré de profondeur.

Cela signifie que l'on s'éloigne des représentations syntaxiques superficielles. Selon notre perspective, il ne sert à rien de représenter différemment deux énoncés qui ne diffèrent que par la surface, c'est-à-dire qui apportent les mêmes connaissances, ou des connaissances suffisamment proches pour qu'on puisse négliger les différences.

Ainsi les énoncés (1) et (1'), comme les énoncés (2) et (2'), auront une même représentation:

(1) *Jean aime Marie*

(1') *Marie est aimée par Jean*

(2) *Jean aime Marie et elle le sait*

(2') *Jean aime Marie et Marie sait que Jean aime Marie*

Il en résulte que l'ordre des unités lexicales dans un énoncé n'est pas une information qui doit être retenue dans la représentation. De même, les pronoms sont utiles dans la langue naturelle, mais rien n'oblige à ce qu'ils soient conservés dans la représentation, du moins quand ils servent à marquer un lien anaphorique: ce lien pourra être indiqué autrement dans la représentation.

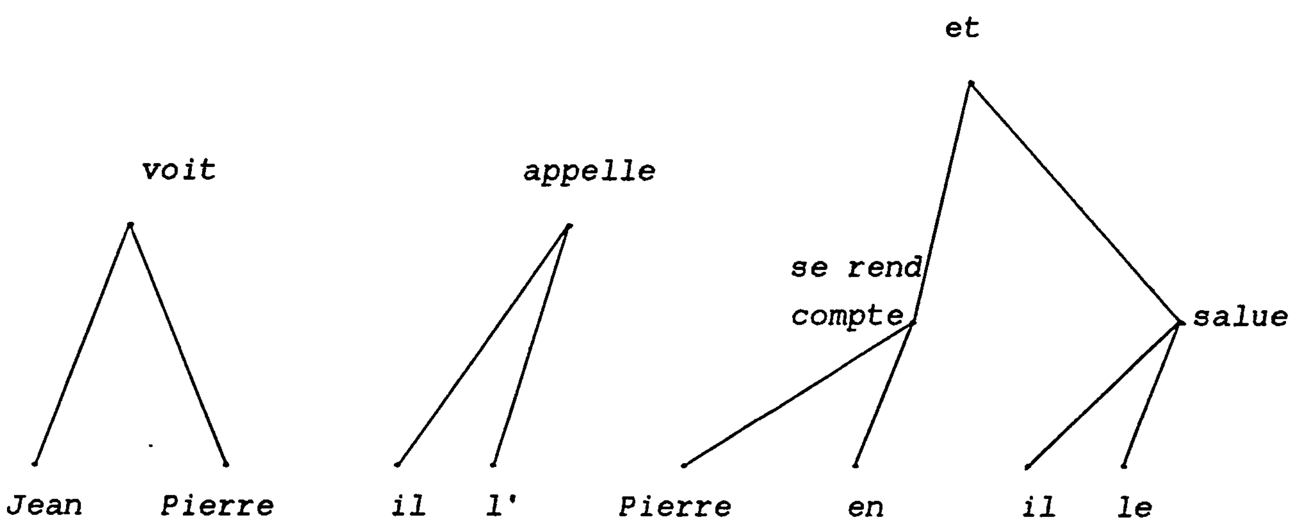
Il ne sera donc pas possible, à partir de la représentation, de restituer des données textuelles telles qu'elles auront été entrées, mais seulement de restituer la connaissance sous-jacente à ces données. C'est d'ailleurs pourquoi il n'y aura aucune difficulté à ce que les connaissances élémentaires que l'on voudra stocker soient représentées dans un formalisme analogue à celui des données textuelles, et que la combinaison des deux conduise à une structure unique.

L'effacement des pronoms conduit à passer des structures arborescentes, classiquement utilisées pour représenter des données textuelles, aux graphes sans circuit plus généraux. Nous allons montrer pourquoi, dans la perspective de l'efficacité des traitements informatiques, l'utilisation de telles structures est préférable.

On prend comme exemple le texte suivant:

Jean voit Pierre. Il l'appelle. Pierre s'en rend compte et il le salue.

Ce texte peut être représenté par un ensemble d'arbres qui, quel que soit leur type, engendreraient la même difficulté. La figure



-------------------------figure 4.1-------------------------

4.1 montre ce que cela donne avec des arbres de dépendance.

Il est clair qu'avec une telle représentation, on ne peut pas répondre à des questions aussi simples que:

Qui appelle Pierre?

Qui Jean appelle-t-il?

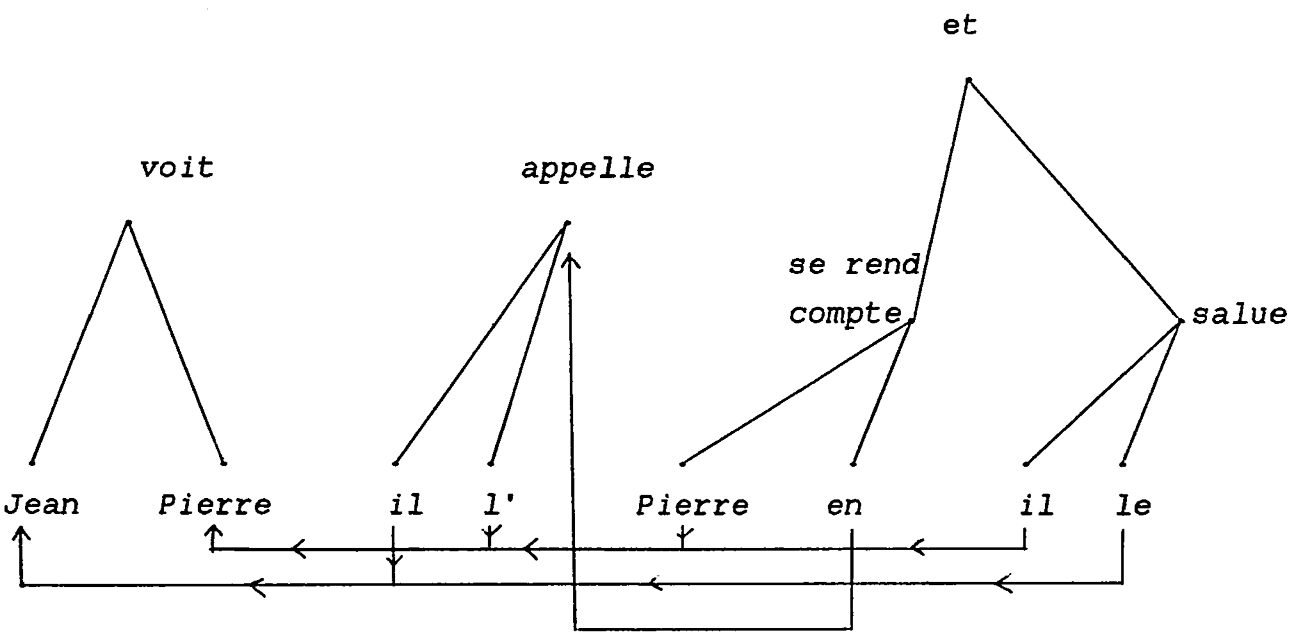
De quoi Pierre se rend-il compte?

Pierre se rend-il compte que quelqu'un l'appelle?

parce qu'on n'a pas indiqué à quoi se référaient les pronoms.

Evidemment, rien ne prouve que l'on dispose de procédures permettant de savoir à coup sûr quel objet est remplacé par un pronom, et ce n'est pas l'objet de notre travail de rechercher de telles procédures. Mais si on ne peut résoudre ce problème pour un texte, il est vain de vouloir poser des questions à propos de ce texte.

Il est donc indispensable de marquer la liaison entre chaque pronom et l'objet qu'il remplace. En utilisant à cette fin des pointeurs, on obtient la structure de la figure 4.2.



<u>figure 4.2</u>

Dès lors, il est possible de répondre aux différentes questions; mais les algorithmes à mettre en oeuvre sont relativement compliqués.

Ainsi, pour trouver la réponse à la dernière question, il faut:

(1) rechercher toutes les occurrences de *Pierre*

(2) rechercher les prédécesseurs droits de ces sommets. S'ils sont étiquetés par *se rend compte*, on en cherche les autres successeurs.

Dans le cas présent, on trouve un successeur étiqueté par *en*. On recherche sur quoi pointe ce sommet: sur le sommet étiqueté par *appelle*.

(3) rechercher les descendants de ce dernier sommet qui sont étiquetés par *il* et *l'*.

Par les pointeurs, on est renvoyé à *Jean* et *Pierre*. Il reste à vérifier que ce *Pierre*-ci est le même que celui rencontré précédemment.

S'il y a, comme on le voit, une certaine complexité à répondre à une question simple, c'est parce qu'on représente un objet unique par plusieurs sommets distincts. Or, la reprise par des pronoms, qui est nécessaire dans la langue naturelle, ne s'impose pas dans la représentation informatique.

C'est pourquoi on fusionne les sommets liés par des pointeurs et on obtient un graphe sans circuit (figure 4.3).

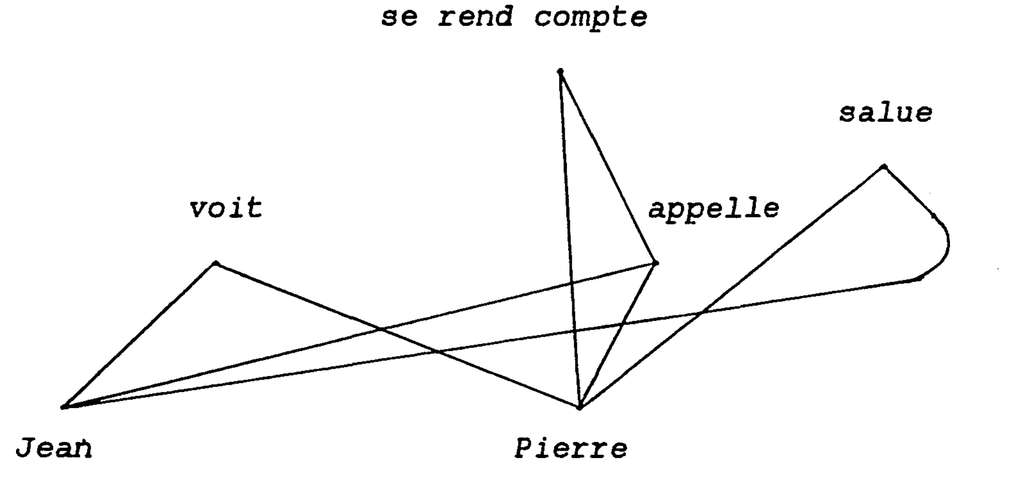


figure 4.3

L'algorithme d'extraction de la réponse est alors plus simple:

(1) Il y a une seule occurrence de *Pierre*

(2) Parmi les prédécesseurs de ce sommet, il faut trouver celui qui est étiqueté par se *rend compte*

(3) Ensuite il n'y a plus qu'à descendre et à vérifier qu'il y a bien correspondance avec les éléments de la question.

Dans ce processus, la recherche de la réponse se ramène à l'extraction d'une sous-structure du graphe qui a servi à enregistrer les données qui soit "à l'image" de la question, c'est-à-dire qui soit obtenue à partir de la question par un certain morphisme. La question sera en effet également représentée par un graphe sans circuit, comme nous le verrons dans le chapitre 5.

Avec des arbres, on ne pourrait obtenir de manière naturelle que la représentation de la question soit contenue dans la représentation des données.

La structure représentative de données textuelles sera un Σ-GOSME comme nous allons le voir en 4.1.4. Cependant, nous n'introduisons pas directement cette structure, commençant par associer des arborescences élémentaires aux unités lexicales.

4.1.2 Les sortes primitives

On part d'une perspective analogue à celle des grammaires catégorielles [Bar-Hillel, 70], c'est-à-dire qu'on se donne deux sortes primitives, $\underline{i}$ pour "objet individuel" (au lieu de "nom" dans les grammaires catégorielles) et $\underline{p}$ pour "proposition". "Objet individuel" est employé par opposition à "objet typique" qui sera introduit dans la deuxième partie de ce chapitre.

Si l'on préfère parler d'objet individuel plutôt que de nom c'est parce que, voulant représenter la connaissance sous-jacente aux énoncés, ce n'est pas la classe grammaticale du mot qui nous intéresse, mais sa "valeur référentielle", l'objet du monde dont l'énoncé rend compte qui lui correspond.

On suppose donc que le monde est composé d'un certain nombre d'objets individualisables. Supposition très réductrice, comme nous allons le voir plus loin (4.2.1). On suppose même, dans le cadre de cette première approche, qu'à chaque nom correspond un objet et un seul du monde. C'est dire que nous voulons ignorer pour l'instant ce qui concerne la détermination. L'énoncé

Jean mange une pomme

sera interprété exactement de la même manière que

Jean mange la pomme.

Cela revient à ne prendre en compte, parmi les noms, que les noms propres ou les noms communs précédés de l'article défini *le* ou *la* et à considérer qu'à un nom correspond un objet et un seul du monde auquel on s'intéresse, un individu.

S = {$\underline{i}$,$\underline{p}$} sera, provisoirement, l'ensemble des sortes.

4.1.3 Le dictionnaire: unités lexicales, types et arborescences élémentaires associés

Partant des objets individuels, on construit des **syntagmes** à l'aide de verbes, prépositions, conjonctions et adverbes. Un syntagme sera soit un syntagme nominal - ou plutôt un syntagme "objectal" -, soit un syntagme propositionnel.

Par exemple, avec les objets individuels *Jean* et *Marie*, et le verbe *aime*, on construit le syntagme propositionnel (ou tout simplement la proposition)

Jean aime Marie.

Le verbe *aime* est considéré comme un opérateur permettant de construire un syntagme propositionnel à partir de deux syntagmes nominaux. Chaque verbe, préposition, etc, sera considéré comme un opérateur. A chaque opérateur va être associé un type, par exemple <$\underline{p}$,$\underline{nn}$> pour *aime*. Pour que les notations soient homogènes, on considèrera que les objets individuels sont des opérateurs 0-aires qui construisent des syntagmes objectaux.

Ces considérations expliquent la structuration que nous donnons au dictionnaire. Un **dictionnaire** est constitué d'un ensemble de triplets $<u,s,s_1s_2...s_p>$ où

- u est une **unité lexicale**, c'est-à-dire une suite de caractères. Si A est l'alphabet, u prend ses valeurs dans l'ensemble A^+. L'ensemble U des unités lexicales est un sous-ensemble de A^+;

- le couple $<s,s_1s_2...s_p>$ de S x S* est un **type**. Cela signifie qu'à une suite d'objets de sortes s_1, s_2,...,s_p on peut faire correspondre un objet de sorte s par l'élément considéré du dictionnaire.

Un dictionnaire est par conséquent un S-schéma, selon la définition donnée en 2.7.2. On notera ce S-schéma Σ. Notons qu'à chaque entrée du dictionnaire est associée, en plus du type, une unité lexicale. Ainsi, à la même unité lexicale il peut correspondre deux types différents. C'est en raison de l'existence de mots homographes ou de la possibilité d'emploi de certains mots de plusieurs manières différentes.

Les **objets individuels** sont les éléments primitifs du système de représentation. On leur associe le type $<\underline{i},\lambda>$ qui permet, à partir d'une suite vide, de construire un objet de sorte $\underline{i}$. Par exemple:

Jean

la France

le Massif central

les Alpes

la lune

l'informatique

la pomme

ce dernier exemple donné avec les restrictions explicitées en 4.1.2.

Les **verbes** construisent des propositions, en général à partir de noms (2, 1 ou 0 noms). Par exemple:

appelle :	$<\underline{p},\underline{ii}>$	(*Jean appelle Pierre*)
marche :	$<\underline{p},\underline{i}>$	(*Jean marche*)
pleut :	$<\underline{p},\lambda>$	(*il pleut*)

dépend le sommet z de sorte j, on trouve un ensemble vide. On peut dans ces conditions remplacer la sorte j par i.

Ce remplacement ne sera pas possible si on applique la même règle de réécriture à la connaissance

chaque femme qui habite Trifouilly est la soeur de chaque homme qui habite Trifouilly

On obtient le réseau de la figure 4.93.

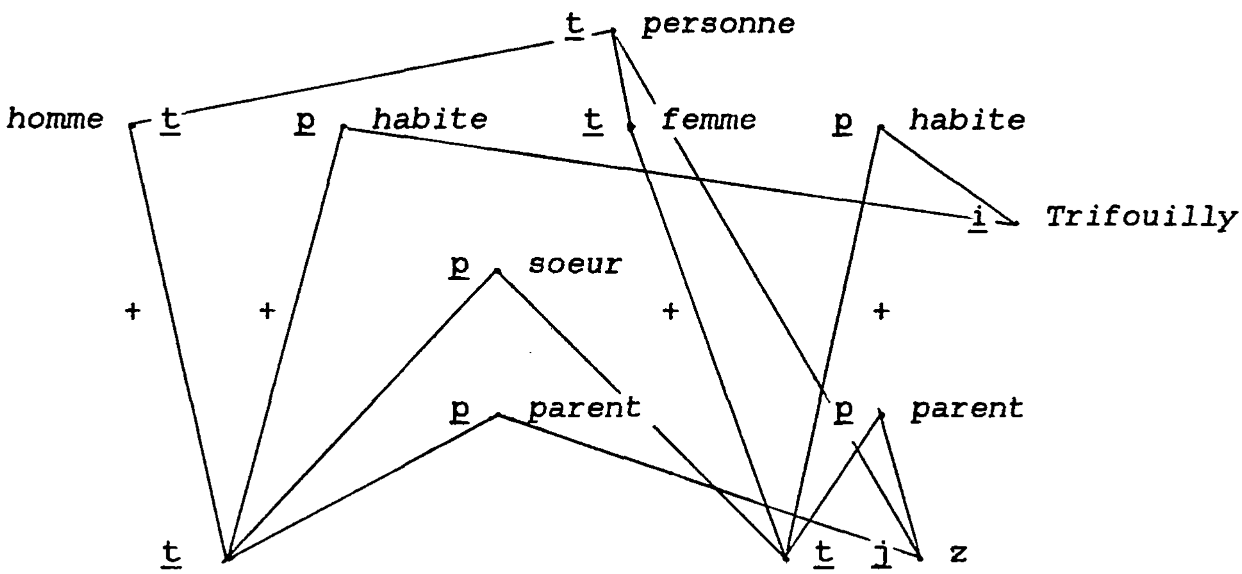


figure 4.93

Dans ce cas, le sommet z est un sommet qui dépend de 2 sommets typiques. Il doit donc garder la sorte j.

Quand on dispose d'un réseau H ainsi que d'un ensemble de règles de réécriture de réseaux, on a la possibilité (théorique) d'obtenir par dérivation toutes les connaissances possibles. Mais, ces connaissances ne sont pas en nombre fini. Il n'est donc pas possible d'effectuer toutes les dérivations. Et, de toutes façons, ce n'était pas le but recherché.

En réalité, on ne se servira des règles que lorsque l'on en aura besoin pour répondre à une question donnée, comme nous allons le voir dans le chapitre suivant.

Il se peut cependant qu'un verbe construise une proposition à partir de noms et de propositions (elles-mêmes construites à partir de noms). Ainsi:

se *rend compte* : <p,np> (*Jean se rend compte que Pierre l'appelle*)

<u>Remarque</u>: On ne peut pas représenter simplement de tels verbes par des prédicats du premier ordre.

Les **adverbes**, lorsqu'ils sont attachés à un verbe, modifient les verbes de telle manière que soit construite une proposition différente de la proposition construite sans l'adverbe. On fera donc comme si l'adverbe construisait à partir d'une proposition donnée, une proposition modifiée.

Par exemple:

vite : <p,p> (*Jean marche vite*)

Les **prépositions** enfin, dans certains de leurs emplois, permettent d'attacher un - ou plusieurs - arguments supplémentaires à un verbe. A partir de la proposition qui aurait été construite sans cet argument supplémentaire et de cet argument, on construit une nouvelle proposition. Le type associé aux prépositions dans ce cas est donc <p,pn>. Par exemple:

à : <p,pn> (*Jean présente Pierre à Paul*)

Avec un tel emploi des prépositions, on attache au maximum deux arguments primitifs à un verbe.

Il se pose néanmoins deux problèmes:

1. Dans certains cas, on ne peut concevoir le verbe indépendamment de la préposition. Par exemple, de *Jean va à Paris*, on ne peut déduire que *Jean va*. Il serait donc peu cohérent de construire l'énoncé à partir du sous-énoncé. De plus la question *est-ce que Jean va?* produirait la réponse oui.

C'est pourquoi nous préfèrerons considérer va à comme une unité lexicale unique dont le type sera naturellement <p,nn>. Ce qui ne voudra pas dire nécessairement que le verbe ne peut pas s'employer sans la préposition. Prenons l'exemple:

Jean pense à Pierre

On peut très bien ne pas vouloir déduire de cet énoncé que *Jean pense*, bien que cette dernière formulation soit possible. On aura alors deux unités lexicales distinctes: *pense* et *pense à*.

2. Il peut sembler que certains verbes admettent deux compléments d'objet direct, ce qui ne permettrait pas de leur appliquer notre représentation. Par exemple:

François Mitterrand nomme Jacques Chirac premier ministre

Dans de tels cas, nous supposerons qu'il y a des prépositions implicites. Ici: *comme*

<u>Remarque</u>: Si plusieurs types sont associés à une même unité, c'est qu'il y a une ambiguïté, au moins sur l'utilisation syntaxique de l'unité. Tel ne sera cependant pas le cas pour les verbes transitifs employés sans complément d'objet. Si l'on dit *Jean mange*, nous supposerons que *Jean mange quelque chose* mais que, simplement, ce *quelque chose* n'est pas mentionné.

En revanche, entre *Jean vit* et *la France vit des événements dramatiques*, il y a plus qu'une nuance dans l'emploi de l'unité *vit*. Notamment, il serait abusif de décréter que *Jean vit quelque chose*. Il faudra donc pouvoir associer les deux types $<\underline{p},\underline{i}>$ et $<\underline{p},\underline{ii}>$ à l'unité lexicale *vit*.

Pour cela, on aura deux entrées dans le dictionnaire correspondant à la chaîne de caractères *vit*.

A chaque triplet $<u,s,s_1...s_p>$ du dictionnaire va être associé un graphe ordonné sans circuit multiétiqueté $G = <X,\delta,\xi>$ qui sera en fait réduit à une arborescence élémentaire.

$X = \{x_0,x_1,...,x_p\}$ est n'importe quel ensemble de cardinal p+1

$$\delta(x_0) = x_1...x_p$$

$$\forall\ i \in [p] \qquad \delta(x_i) = \lambda$$

$\xi = <\xi^1,\xi^2>$,avec

$$\xi^1 : X \dashrightarrow S$$

$$\xi^1 (x_0) = s$$

$$\forall\, i \in [p] \qquad \xi^1(x_i) = s_i$$

$$\xi^2 : \{x_0\} \dashrightarrow U \qquad \xi^2(x_0) = u$$

On remarque que les conditions relatives aux Σ-GOSME sont vérifiées en raison de la manière dont ces arborescences sont définies. On pourra donc noter la structure $G = \langle X,\delta,\sigma,\omega \rangle$ conformément à la notation introduite en 2.7.4.

Des exemples d'arborescences sont présentés en figure 4.4.

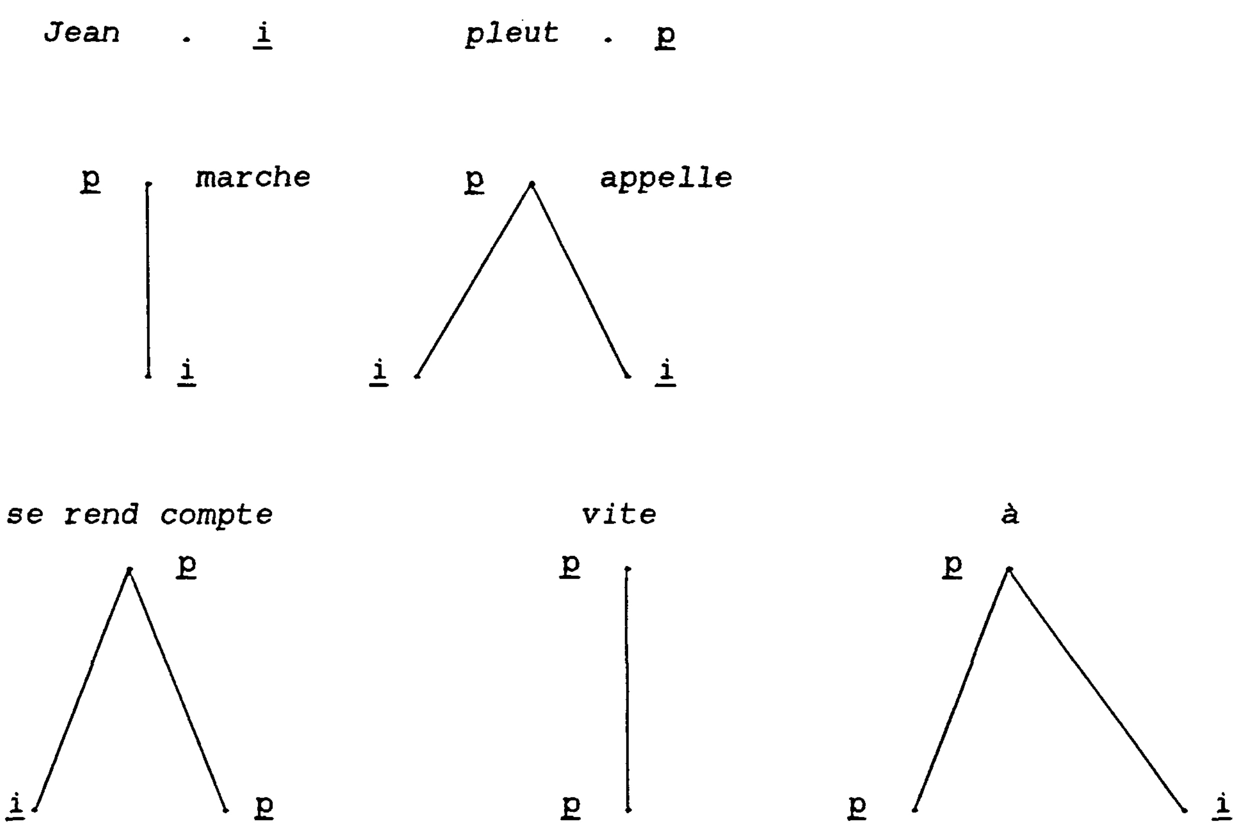


figure 4.4

4.1.4 <u>La représentation des propositions</u>

On part de l'hypothèse que tout texte est constitué d'un ensemble de syntagmes imbriqués les uns dans les autres.

Nous avons déjà une représentation des syntagmes objectaux qui sont réduits à un objet individuel: il s'agit, soit de l'opérateur de type <$\underline{i}$,λ>, soit du Σ-GOSME réduit à un unique sommet.

Pour représenter la proposition

Jean voit Pierre

on pourrait procéder par "greffe" et "intrication" des opérateurs <*Jean*,$\underline{i}$,λ>, <*Pierre*,$\underline{i}$,λ> et <*voit*,$\underline{p}$,$\underline{ii}$> (voir [Desclés, 80]). Mais on obtiendrait ainsi des structures très vite peu manipulables, en particulier très difficiles à interroger (voir [Blosseville-Bouilloux, 80]).

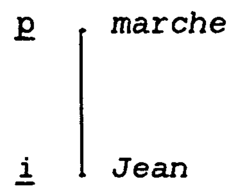
C'est pourquoi on préfèrera donner une représentation sous forme de graphe et, plus exactement, sous forme de Σ-GOSME.

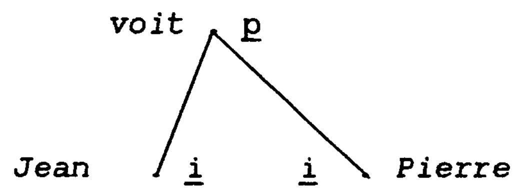
Dans le chapitre 6 nous indiquons, sur un sous-ensemble très restreint du français, quelles sont les procédures qui permettent de passer des données textuelles à la représentation. Cependant, l'objet du présent travail n'est pas de traiter ce problème en toute généralité. Pour l'instant nous ne décrivons pas comment on construit, à partir des arbres élémentaires associés aux unités lexicales qui composent un texte, la structure plus complexe censée représenter le texte. Nous nous contentons, comme dans la figure 4.5, d'exhiber des exemples de structures en indiquant quel genre d'énoncé elles représentent.

Notons que le choix des représentations est très lié au choix des types associés aux unités lexicales. C'est même dans la perspective de construire les structures qui suivent que nous avons associé certains types aux unités lexicales.

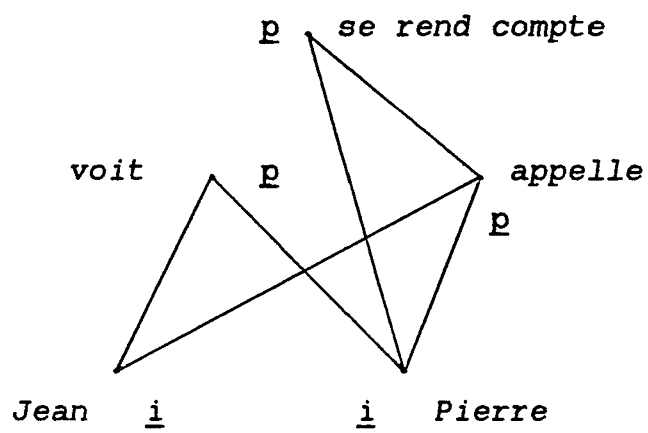
<u>Remarques</u>: 1. Les objets de sorte $\underline{i}$ ne sont pas répétés dans la représentation.

2. Sur les deux derniers exemples, on constate que notre représentation est relativement proche de la syntaxe. En effet,

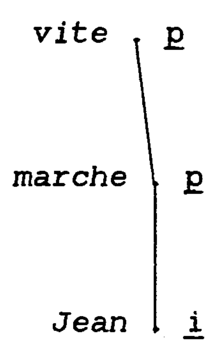


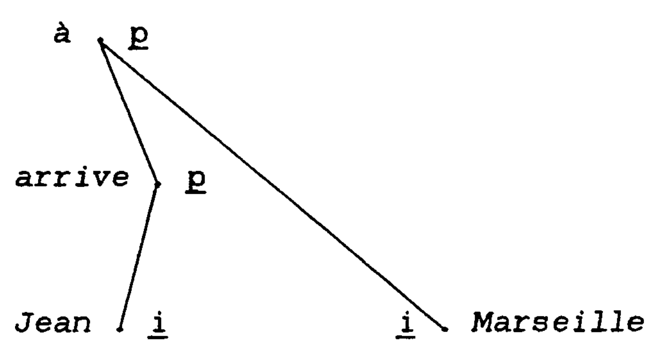


Jean marche　　　　*Jean voit Pierre*



Jean voit Pierre. Il l'appelle. Pierre s'en rend compte.



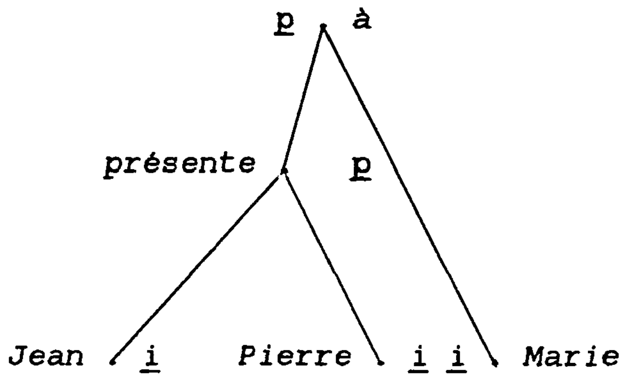


Jean marche vite　　　　*Jean arrive à Marseille*

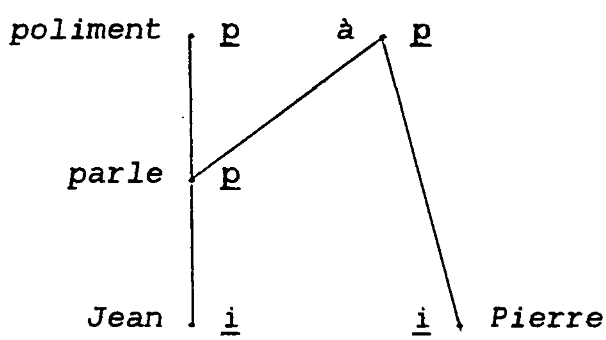
--------------------------figure 4.5--------------------------

si au lieu de *marche vite*, on avait eu *court* et si au lieu de *arrive à*, on avait eu *atteint*, un arc de chacune des représentations aurait été remplacé par un sommet unique. La représentation ne dépend donc pas uniquement du type de relations entre les objets, mais également de la manière dont ces relations sont décrites dans une langue donnée.

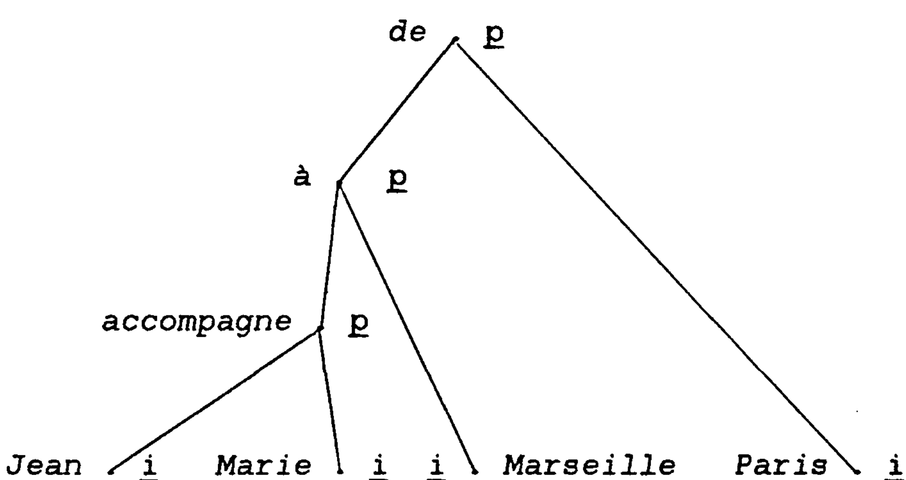
Nous le voyons également sur les exemples de la figure 4.6.



Jean présente Pierre à Marie



Jean parle poliment à Pierre



--------------------------<u>figure 4.6</u>--------------------------

Le dernier graphe est une des représentations de l'énoncé ambigu:

Jean accompagne Marie de Paris à Marseille

<u>Remarque</u>: La position respective des prépositions *de* et *à* n'est pas fortuite. Elle permet d'extraire de la représentation les structures représentatives de l'énoncé

Jean accompagne Marie à Marseille

comme celle de l'énoncé

Jean accompagne Marie

mais elle ne permet pas d'isoler une structure qui représenterait

? Jean accompagne Marie de Paris

? Jean accompagne de Paris à Marseille

? Jean accompagne à Marseille

De même, de

Jean parle poliment à Pierre,

on pourra déduire

Jean parle à Pierre

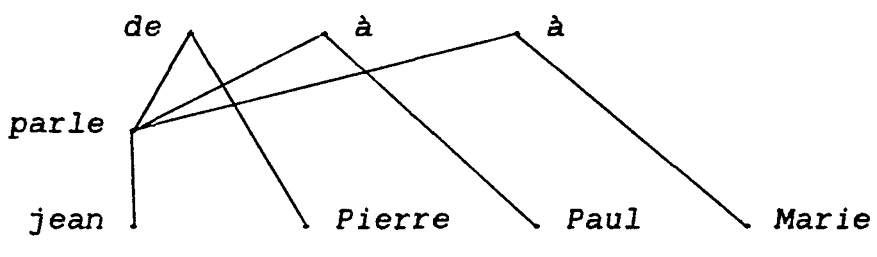
Jean parle poliment

Jean parle

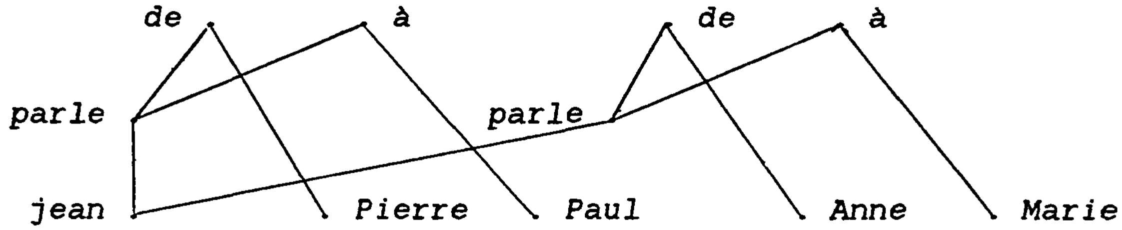
Ce mécanisme d'extraction de sous-structures sera d'une importance capitale dans l'extraction des réponses en fonction de questions. Il détermine les choix que nous effectuons pour la représentation.

En fonction de ces perspectives nous donnons la représentation (en figure 4.7) de l'énoncé

Jean parle de Pierre à Paul et à Marie



<u>figure 4.7</u>



---------------------------figure 4.8---------------------------

et celle (en figure 4.8) de l'énoncé

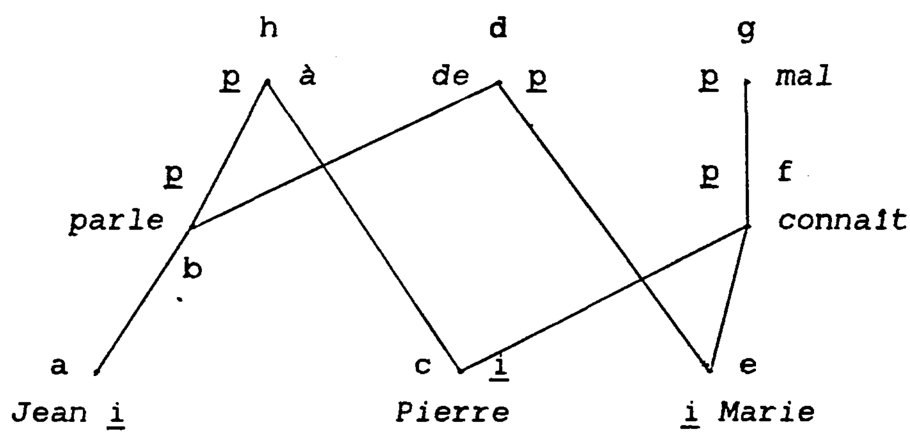
Jean parle de Pierre à Paul et d'Anne à Marie.

Remarquons que, dans ce dernier cas le syntagme *Jean parle* est représenté deux fois.

Donnons enfin, de manière tout à fait précise, la représentation d'un texte:

Jean parle de Marie à Pierre. Pierre connaît mal Marie

x	a	b	c	d	e	f	g	h
$\delta(x)$	λ	a	λ	be	λ	ce	f	bc
$\sigma(x)$	$\underline{i}$	$\underline{p}$	$\underline{i}$	$\underline{p}$	$\underline{i}$	$\underline{p}$	$\underline{p}$	$\underline{p}$
$\omega(x)$	*Jean*	*parle*	*Pierre*	de	*Marie*	*connaît*	*mal*	à



---------------------------figure 4.9---------------------------

C'est le Σ-GOSME H = $<X,\delta,\sigma,\omega>$ présenté en figure 4.9.

Il s'agit d'un Σ-GOSME complet (voir 2.7.4).

Pour tout sommet x du graphe, le sous-GOSME total engendré par {x} (2.3.5) représente un syntagme extrait du texte. La sorte du syntagme (objectal ou propositionnel) dépend de la sorte du sommet **x**. Le sous-graphe total engendré par {b} représente par exemple le syntagme *Jean présente Pierre*.

Plus généralement, tout "sous-texte" extrait d'un texte donné ne peut être représenté que par un sous-GOSME total du GOSME représentatif du texte global: toute autre sous-structure représenterait des verbes sans l'ensemble de leurs arguments, ou des adverbes non reliés à un verbe, etc...

4.1.5 <u>Prise en compte des conditions énonciatives et de la modalité</u>

Suivant en cela certaines théories linguistiques, nous considérerons qu'un énoncé est composé d'une structure prédicative répérée par rapport à une **situation énonciative** (voir [Culioli, 82]).

Ce point de vue s'accorde très bien avec notre perspective pratique qui est l'interrogation de données textuelles. Le problème se pose en effet très concrètement. Si l'on sait que

Jean dit que Paul aime Marie

et qu'on pose la question

Est-ce que Paul aime Marie ?

on ne peut pas répondre simplement par *oui* ou par *non*. Si l'on trouve la structure représentative de la question dans les données textuelles enregistrées, il faut encore déterminer quelles sont les conditions énonciatives dans lesquelles elle s'insère. C'est-à-dire, dans le cas présent, d'indiquer que le **sujet énonciateur** est *Jean*.

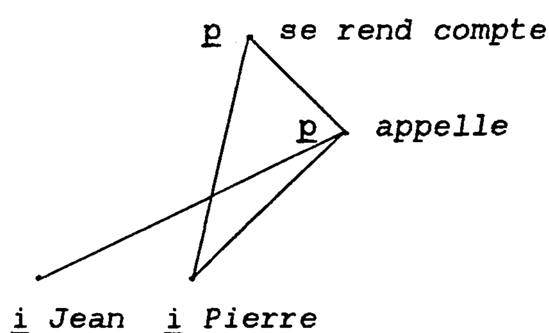
On peut en déduire, par conséquent, que quand on représente des données textuelles, on ne peut se contenter de deux (ou même de trois comme le propose A. Colmerauer [Colmerauer, 79]) valeurs de vérité. La notion même de valeurs de vérité ne semble pas adaptée.

Mais les conditions énonciatives ne sont pas composées que d'un sujet énonciateur. A la place de *dit*, on aurait pu avoir *croit*, *pense*, *affirme*, etc... Cela n'est pas indifférent dans la réponse à la question. Plus généralement, les propositions (complétives) qui sont objets d'un verbe de type <p,np> admettent le verbe et son sujet comme conditions d'énonciation. Cette particularité de la langue naturelle fait qu'elle ne peut pas être correctement représentée (même approximativement) par un système fondé sur le calcul des prédicats du premier ordre.

Dans l'exemple suivant

Pierre se rend compte que Jean l'appelle

la proposition *Jean appelle Pierre* a ses conditions énonciatives déterminées par la proposition principale *Pierre se rend compte*. La représentation s'effectue tout-à-fait naturellement à l'aide d'un Σ-GOSME comme on peut le voir sur la figure 4.10.



--------------------------figure 4.10--------------------------

On peut dire que, dans un tel cas, le verbe *se rend compte* marque une certaine **modalité** de la complétive.

Plus généralement, les conditions énonciatives peuvent être marquées par des expressions modales telles que *peut-être*, *il faut*, etc., expressions auxquelles nous attribuerons le type <p,p>. Cela nous permettra de donner par exemple à l'énoncé

il faut que Paul voie Marie

la représentation de la figure 4.11.

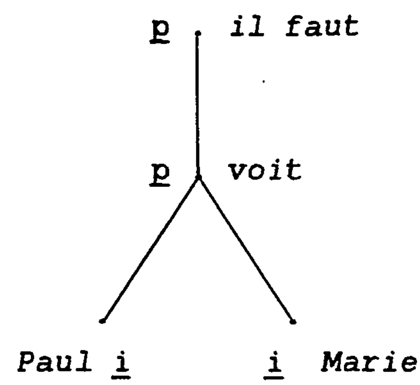


figure 4.11

Cette représentation permettra d'obtenir très facilement la réponse à la question

est-ce que Paul voit Marie ?

qui sera

il faut que Paul voie Marie.

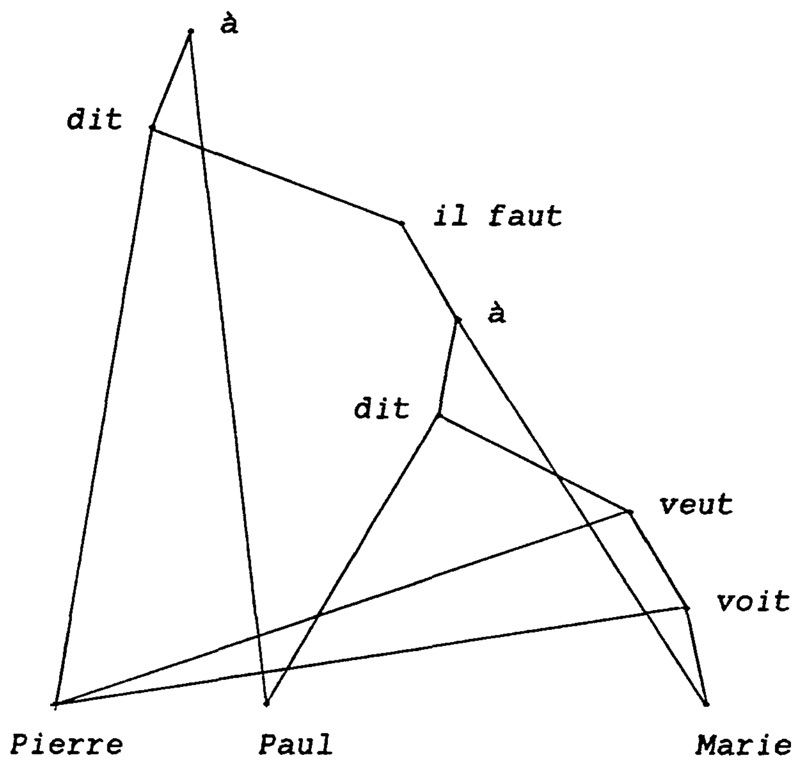
La recherche des réponses à une telle question reviendra en effet à chercher dans les données la structure prédicative *Jean voit Marie* et à déterminer dans quelles conditions elle est insérée.

Il peut y avoir plusieurs conditions énonciatives distinctes dans un énoncé donné, les unes imbriquées dans les autres: des conditions énonciatives peuvent porter sur un énoncé qui comporte déjà des conditions énonciatives, et ainsi de suite. Par exemple:

Pierre dit à Paul: il faut que tu dises à Marie que je veux la voir.

La représentation de tels énoncés ne pose aucun problème, comme on le voit avec la figure 4.12.

Dans un texte donné, il y a forcément des propositions (principales) dont rien ne marque les conditions énonciatives. On dira que ces propositions sont repérées par rapport à une **situation énonciative origine**. En d'autres termes, cela signifie que, dans le monde où on se place, la proposition est considérée comme vraie, sans aucune restriction.



---------------------------figure 4.12------------------------

Considérons la donnée

Jean aime Marie

L'absence de tout contexte restrictif se traduira dans la représentation par le fait que le sommet étiqueté par *aime* n'aura aucun ascendant.

Mais si l'on veut représenter

Jean aime Marie et Marie croit que Jean l'aime

le sommet étiqueté par *aime* admettra comme prédécesseur le sommet étiqueté par *croit*, ce qui limitera la vérité de *Jean aime Marie*. C'est pourquoi on donnera la représentation de la figure 4.13. où *vrai* est une **pseudo-unité lexicale.** C'est-à-dire qu'elle intervient dans la représentation comme une unité lexicale, bien qu'elle ne figure pas comme telle dans les données textuelles. Le type associé à *vrai* sera <p,p>.

De même, on utilise une pseudo-unité lexicale de même type, *faux*. Elle permettra de représenter des énoncés tels que:

Marie croit que Jean l'aime, mais ce n'est pas vrai

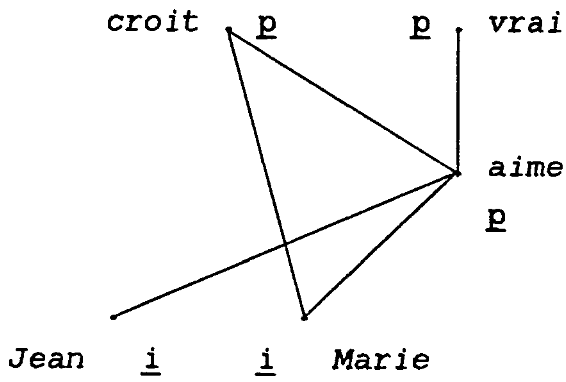


figure 4.13

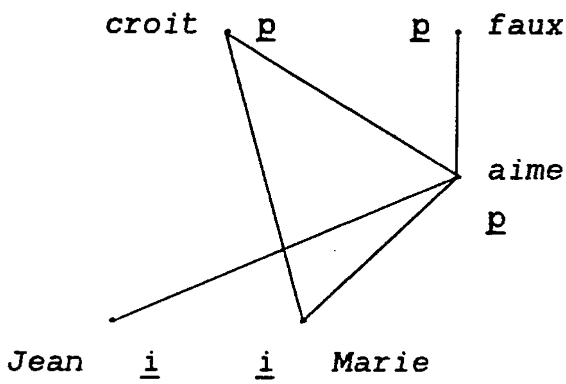


figure 4.14

La représentation de cet énoncé est donnée en figure 4.14.

Ainsi a-t-on une représentation de la négation qui porte sur des propositions: la négation est assimilée à une modalité comme une autre.

Remarque: L'usage de pseudo-unités lexicales peut paraître briser l'homogénéité de la représentation puisque les sommets sont étiquetés par des étiquettes d'un genre différent. En réalité, il n'y a pas de rupture véritable entre unités lexicales et pseudo-unités lexicales qui sont toutes deux des suites de caractères. La présence de la suite de caractères dans les données textuelles n'est pas un critère déterminant. En effet, déjà, nous avons transformé les suites de caractères présentes dans les données pour n'introduire dans les représentations que des formes "standard" à l'exclusion de toute forme fléchie, nous avons exclu de la représentation certaines suites de carctères présentes dans les données (pronoms). Et dans la structure représentative des données, les sommets ont tous le même statut.

Le **temps** pourra également être considéré comme une modalité particulière. En effet, si l'on sait que

Jean aimait Marie

et que l'on demande

est-ce que Jean aime Marie ?

il faudra pouvoir retrouver la structure représentant la question dans la représentation des données et restreindre la vérité de la structure prédicative au passé.

On aurait alors la représentation de la figure 4.15

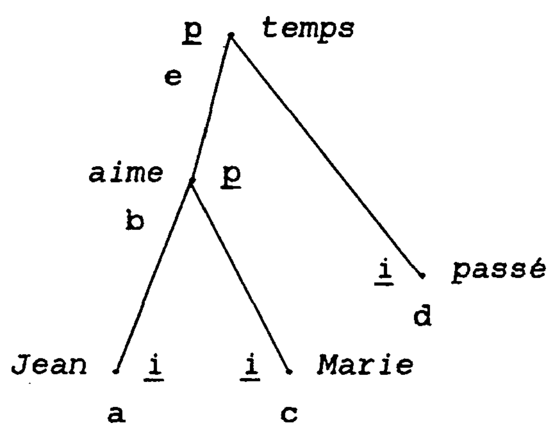


figure 4.15

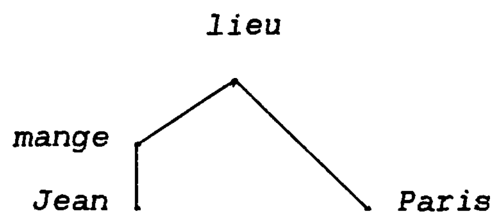
où *temps* et *passé* sont des pseudo-unités lexicales dont les types respectifs associés sont <p,pi> et <i,λ>.

Il est clair que la représentation du temps est, dans cet exemple, très grossière. Toutefois, le sommet d peut, s'il est inséré dans certaines relations, décrire beaucoup plus précisément les conditions temporelles dans lesquelles s'insère la structure prédicative. Nous allons voir plus loin (4.2.6) comment avoir une description plus précise. Mais nous ne prétendons pas résoudre tous les problèmes liés au temps. Cela demande une étude informatico-linguistique spécifique (voir notamment [Bestougeff-Ligozat, 85]).

De même pourra-t-on avoit une première représentation de la localisation. Par exemple, pour

Jean mange à Paris

on aura la représentation de la figure 4.16.



--------------------------figure 4.16--------------------------

Nous aurons une meilleure représentation de cet énoncé en 4.2.6.

Remarque: La représentation obtenue est totalement différente des représentations de:

Jean parle à Pierre

et de:

Jean va à Paris.

Ainsi, trois énoncés qui ont apparemment la même syntaxe reçoivent une représentation distincte. C'est pour cela qu'on peut dire que les énoncés sont représentés à un certain niveau de profondeur.

Toutefois, la représentation que nous donnons des "compléments circonstanciels" n'est pas sans poser des problèmes:

(1) en premier lieu, il n'est pas évident de séparer les compléments de lieu des compléments de temps, comme par exemple dans la phrase:

Jean chante quand il est sous la douche.

C'est encore plus difficile si on considère d'autres types de compléments circonstanciels, de "manière", de "cause", etc... Par exemple:

S'il est sous la douche, Jean chante.

Or, notre représentation exige qu'un choix soit effectué. Ceci pour pouvoir répondre à des questions telles que:

Où Jean chante-t-il ?

Quand Jean chante-t-il ?

Pourquoi Jean chante-t-il ?.

(2) On ne peut même pas distinguer toujours les "compléments circonstanciels" des "compléments d'objet indirects" ou les "compléments de nom". Par exemple:

Jean a marché sur les pieds de Pierre

Il mangeait sa soupe dans un bol

Il étudia la vie de Lénine entre 1917 et 1920,

ce dernier énoncé étant ambigu en dehors de toute information supplémentaire.

Les choses se compliquent encore plus si l'on observe que l'énoncé

Jean va à Paris

donne lieu à la question

Où Jean va-t-il ?

bien qu'il soit évident que le complément soit un complément d'objet indirect, et même un complément "indispensable".

L'existence de ces problèmes provient de la faiblesse des notions que nous empruntons aux grammaires scolaires. En fait, il y a un continuum entre tous les compléments, et la classification des compléments circonstanciels est tout à fait arbitraire. Mais, malheureusement, il n'existe pas - à notre connaissance - de travaux linguistiques exhaustifs et incontestables qui apportent de nouvelles notions claires, sur lesquels on puisse s'appuyer.

4.2 La représentation des connaissances

Ce que nous appelons "représentation des connaissances" revient simplement à traduire les connaissances exprimées dans une langue donnée en une représentation abstraite qui soit relativement éloignée de la surface. Il n'y a donc pas rupture entre représentation de données textuelles et représentation de connaissances: il s'agit de la même opération à laquelle on donne un nom en général différent selon le niveau de profondeur de la représentation.

L'objectif est, rappelons-le, pour nous, de traiter avec le maximum d'efficacité le problème de l'interrogation, et nous verrons que concrètement on peut arriver à des résultats appréciables.

Dans ce qui suit nous présentons en détail un modèle dont la construction a été ébauchée dans [Cori, 1982c]. Nous examinons les uns à la suite des autres différents problèmes de représentation qui nous conduisent à introduire deux nouvelles sortes ainsi que quatre étiquettes d'arcs. La structure que l'on obtient finalement afin de représenter les connaissances est un {p}-réseau. Elle est décrite en 4.2.9.

4.2.1 Objets typiques et héritage des propriétés

Prenons un exemple très simple; si l'on a enregistré comme donnée:

Marie possède un chien

et que l'on veut répondre à la question

Marie possède-t-elle un animal ?

il faut savoir que *tout chien est un animal*. Il faut donc avoir au préalable enregistré cette information - évidente pour un humain, mais bien entendu pas pour une machine - et être capable de la mettre en liaison avec l'information précédente.

Nous proposons, afin de représenter de telles connaissances, de construire une hiérarchie dans laquelle tout sommet hérite des propriétés de tous ses ascendants. Ceci n'a rien d'original et se retrouve dans presque tous les "réseaux sémantiques".

Ainsi, nous pourrons intriquer ce type de connaissances (classificatoires) avec les connnaissances apportées par les énoncés. De là résultera une simplification extrêmement importante du processus d'interrogation qui s'exprimera sous la forme d'une équation unique (voir (5.3.2)).

L'approche la plus systématique de cette représentation hiérarchique des connaissances est celle de S.E. Fahlman

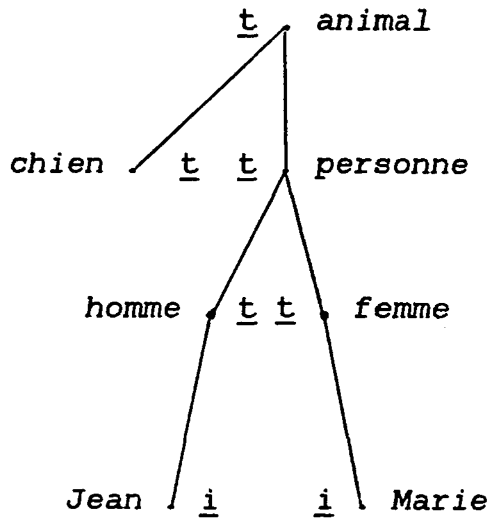
[Fahlman, 77]. Mais sa présentation manque de rigueur (voir critiques dans [Shapiro, 80]) et les formalisations de son système qui sont apparues par la suite ([Etherington-Reiter, 83], [Touretzky, 86], [Froidevaux, 85]) ne concernent qu'une toute petite partie du système. C'est pourquoi, nous avons été obligés de redéfinir entièrement un modèle, par pas mal de côtés différent de celui de Fahlman, qui s'appuie sur le formalisme mathématique précis introduit dans le chapitre 2.

Comme Fahlman, nous distinguons les **objets typiques** des objets individuels. A cette fin, nous nous donnons une nouvelle sorte, $\underline{t}$, qui sera attribuée aux sommets représentant des objets typiques. A tout sommet typique x correspond un ensemble E_x d'objets du monde que l'on veut représenter. Un certain nombre de propriétés sont attachées à l'objet typique de l'ensemble E_x: ce sont les propriétés que vérifient <u>tous</u> les éléments de l'ensemble (à certaines exceptions près, comme nous le verrons en 4.2.7).

Les sommets typiques ainsi que les sommets individuels sont reliés ensemble par des arcs. Si y est un descendant de x, l'ensemble E_y qui lui correspond (ensemble composé d'un seul élément si y est un sommet individuel) est inclus dans E_x. L'objet typique de l'ensemble E_y (ou l'objet individuel) vérifie par conséquent toutes les propriétés que vérifie l'objet typique de E_x. C'est ainsi que s'exprime **l'héritage des propriétés** pour les objets typiques (et individuels). Il est clair que les graphes qui seront obtenus ne comporteront pas de circuit, car sinon cela signifierait que la relation d'inclusion entre ensembles n'est pas une relation d'ordre. Un premier exemple de graphe représentatif d'une hiérarchie d'objets est donné en figure 4.17.

Nous avons représenté par ce graphe les connaissances suivantes:

tout chien est un animal
toute personne est un animal
tout homme est une personne
toute femme est une personne
Jean est un homme



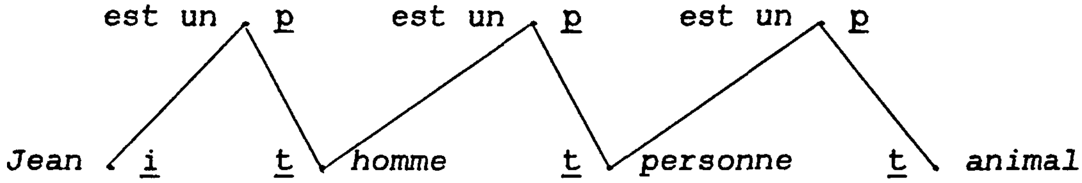
---------------------------figure 4.17---------------------------

Marie est une femme

On retrouve la relation introduite dans pas mal de réseaux sémantiques sous le nom de ISA ("is an instance of") (voir par exemple [Norman-Rumelhart, 75]) ou de AKO ("a kind of") (voir par exemple [Roberts-Goldstein, 77]).

Les arcs qui ont pour origine un sommet typique (et pour extrémité un sommet typique ou un sommet individuel) représentent donc une certaine utilisation du verbe *être*. Il s'agit d'une utilisation du verbe *être* en tant que copule.

Il aurait pu, dès lors, sembler naturel d'utiliser une représentation du type de celle utilisée dans la partie précédente pour représenter les verbes. On aurait eu à ce moment-là une représentation telle que celle de la figure 4.18



---------------------------figure 4.18---------------------------

Mais, avec une telle représentation, on perdrait en efficacité algorithmique dans la recherche des réponses aux questions, le chemin ente *Jean* et *animal* étant plus difficile à retrouver.

Du point de vue linguistique, il n'est pas gênant de traiter la copule différemment des autres verbes: en effet, dans certaines langues telles que le russe, le verbe est omis (voir par exemple [Lyons, 70]). Toutefois, le verbe doit réapparaître quand on veut marquer des distinctions quant au temps, au mode, quand on veut insérer la relation marquée par le verbe être dans des relations plus larges.

Et, effectivement, ce que nous gagnons en efficacité algorithmique, nous le payons en problèmes de représentation quand nous voudrons par exemple modaliser des propriétés de ce type (voir 4.2.7).

Le graphe obtenu est, dans l'exemple de la figure 4.17, une simple arborescence. On aura des graphes sans circuit plus généraux lorsqu'un objet (individuel ou typique) sera inséré dans plusieurs hiérarchies qui alors seront intriquées. Par exemple, si l'on veut dire que

Evariste Galois est un mathématicien français,

l'objet individuel *Evariste Galois* descendra à la fois de la hiérarchie des *scientifiques* par l'intermédiaire des *mathématiciens* et de la hiérarchie des *Européens* par l'intermédiaire des *Français* (figure 4.19).

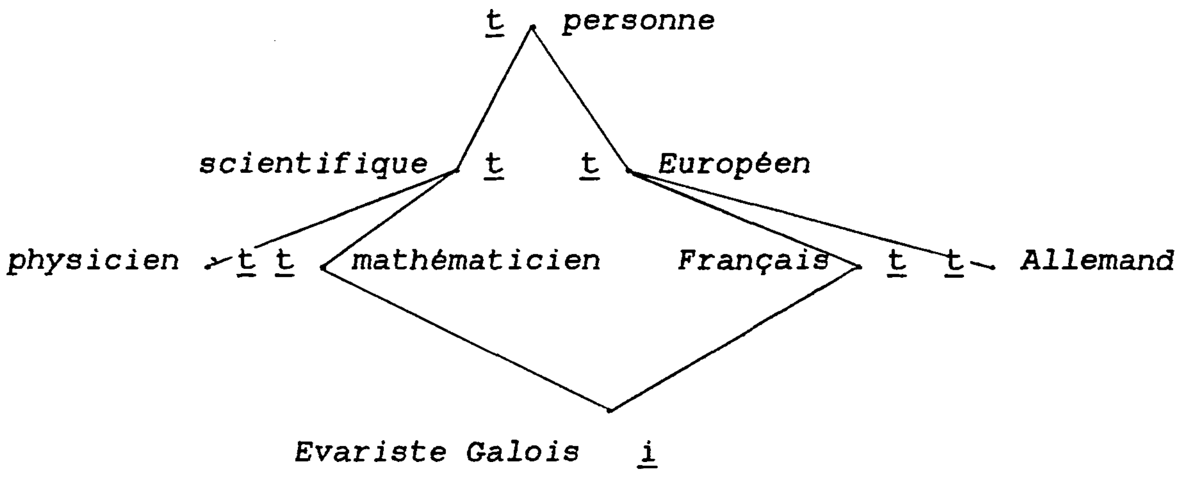


figure 4.19

De même pourrait-on insérer le *professeur d'informatique d'Université* typique dans la hiérarchie des *scientifiques* et dans celle des *fonctionnaires*.

L'ordre entre successeurs d'un même sommet n'est pas significatif. La structure représentative des connaissances hiérarchiques est donc un graphe sans circuit (non ordonné) multiétiqueté $G = \langle X, \gamma, \varepsilon^1, \varepsilon^2 \rangle$ avec:

$\varepsilon^1 : X \dashrightarrow \{\underline{t}, \underline{i}\}$

$\varepsilon^2 : X \dashrightarrow U$

où U est un ensemble d'unités lexicales.

<u>Remarques</u>: 1. Pour le moment, un sommet individuel ne peut être origine d'aucun arc: il est nécessairement terminal. Il peut être successeur d'un sommet typique ou d'un sommet de sorte $\underline{p}$ (ou des deux). Un sommet typique n'a pas forcément de successeur.

2. Il ne faut pas confondre l'objet typique d'un ensemble et l'ensemble lui-même. Si l'on sait par exemple que

un cheval bon marché est rare

il ne faut pas affecter la propriété d'être *rare* au *cheval bon marché* (typique). Il s'agit d'une propriété de l'ensemble correspondant à l'objet typique.

C'est pourquoi, le sommet typique représente l'élément caractéristique de l'ensemble et pas l'ensemble. L'ensemble sera représenté, si nécessaire, par un sommet individuel. Nous verrons plus loin (4.2.5) comment marquer l'appartenance de l'objet typique à l'ensemble qu'il caractérise.

3. L'objet typique d'un ensemble n'existe pas. Il ne s'agit pas, comme le soldat inconnu, d'un élément pris au hasard dans un ensemble et destiné à le représenter. L'objet typique ne vérifie que toutes les propriétés vérifiées par tous les éléments de l'ensemble, il ne vérifie aucune propriété particulière.

Ainsi, la *personne* typique, bien qu'elle ait une taille, ne mesure ni 170 cm, ni 180 cm.

4. L'unité lexicale associée à un objet typique est soit un nom, soit un adjectif qualificatif. Dans une première approximation, il n'est pas nécessaire de représenter différemment les adjectifs des noms: on peut dire que les uns comme les autres rendent compte de **notions**. Les différences dans l'usage syntaxique des noms et adjectifs ne nous importent pas. Presque tous les adjectifs, d'ailleurs, peuvent être employés sous une forme substantivée.

Nous considèrerons par exemple que les deux énoncés suivants:

Evariste Galois est un mathématicien français

Evariste Galois est un Français mathématicien

sont suffisamment proches pour recevoir la même représentation.

Nous verrons toutefois plus loin (4.2.2) que les adjectifs posent malgré cela un problème spécifique.

5. Il n'est pas toujours évident de distinguer les objets typiques des objets individuels.

Comment par exemple interpréter l'énoncé:

Jean aime le vin ?

Faut-il considérer que *le vin* est un objet individuel? Mais alors, on ne pourrait déduire de cet énoncé que

Jean aime le Bordeaux.

Si c'est le Bordeaux qui acquiert le statut d'objet individuel, on ne peut en déduire que

Jean aime le Saint-Emilion.

Et ainsi de suite...

Certes, *le vin* est un objet individuel de type particulier puisqu'on ne peut pas en cerner les contours matériels. Mais pour des objets plus concrets et plus facilement individuables, le problème se pose encore. Par exemple pour:

Jean aime Marie quand elle a sa robe verte ,

ou pour

Jean aime Paris au mois d'août .

4.2.2 Déduction et étiquetage des arcs

Par les arcs qui joignent deux sommets typiques, on a déjà représenté une certaine forme d'implication logique. En effet, que le sommet étiqueté par *homme* soit un successeur du sommet étiqueté par *personne* (figure 4.17) aurait pu être noté dans un système de représentation fondé sur le calcul des prédicats par:

homme(x) ==> *personne*(x)

Il est alors équivalent de dire qu'il existe une liaison entre un sommet typique et un sommet individuel ou que la constante (i.e. l'objet individuel) vérifie le prédicat unaire associé à l'objet typique. Sur notre exemple:

homme(*Jean*)

Dans ce cas, on constate que l'antécédent de l'implication est nécessairement constitué d'une formule atomique unique. Il est clair qu'une telle restriction est trop importante si l'on veut avoir un système déductif un minimum efficace.

C'est pourquoi nous voulons, en utilisant les mêmes méthodes, avoir des déductions admettant plusieurs antécédents. Si l'on veut représenter par exemple

toutes les pommes jaunes sont fades

il faut créer un sommet représentatif de la *pomme jaune* typique (qui descendra nécessairement de la *pomme* typique et de l'*objet jaune* typique) et le relier par un arc à l'*objet fade* typique. On pourrait alors avoir la représentation de la figure 4.20.

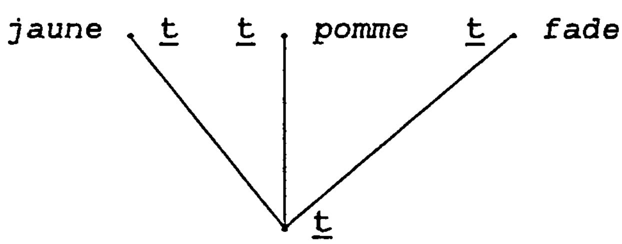


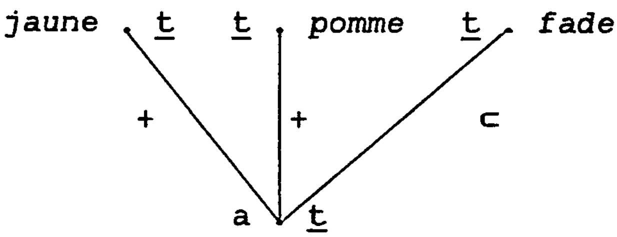
figure 4.20

Cette représentation est ambiguë. Elle peut en effet être aussi bien celle de la connaissance proposée que celle de

toute pomme fade est jaune

ou de *tout objet jaune et fade est une pomme* .

Afin de lever cette ambiguïté, on va **étiqueter les arcs** du graphe, en distinguant ceux qui servent à construire un nouvel objet à partir de plusieurs objets existants (comme *pomme jaune*) de ceux qui serviront à marquer une inférence comme celle proposée. Les premiers arcs seront étiquetés par + et les seconds par ⊂. On aura alors, pour la déduction proposée, la représentation de la figure 4.21.



--------------------------figure 4.21--------------------------

Il est clair dans ce cas que le sommet a représente la notion *pomme jaune* et pas la notion *pomme fade* ou la notion *pomme jaune et fade*.

Dès lors, on pourrait vouloir construire tous les sommets représentant une notion combinée, c'est-à-dire décrite par une combinaison de notions primitives (noms ou adjectifs). Mais, si l'on avait n notions primitives, on obtiendrait un total de

$$C_n^1 + C_n^2 + \dots + C_n^n = 2^n - 1$$

sommets représentant des notions. Ce qui dépasserait très vite les capacités de n'importe quelle machine, tout en n'ayant aucun intérêt puisque l'immense majorité des combinaisons ne serait jamais considérées.

C'est pourquoi nous ne construirons que les sommets représentant des combinaisons de notions que nous appellerons **distinguées**. Une combinaison de notions sera dite distinguée si il lui est affecté une propriété ou si elle est composée d'une seule notion. Sur notre dernier exemple, la combinaison {*pomme*, *jaune*} est distinguée, mais pas la combinaison {*fade*, *jaune*}.

Notons que si l'on avait la connaissance

toutes les pommes sont jaunes et fades
la combinaison {*jaune, fade*} ne serait toujours pas distinguée, puisqu'on aurait alors la représentation de la figure 4.22.

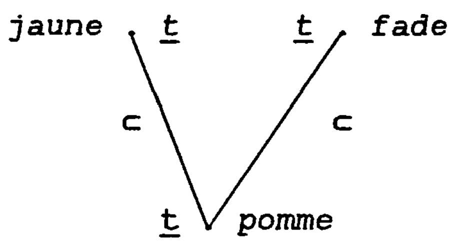


figure 4.22

L'inclusion détermine une relation d'ordre sur les combinaisons distinguées. A chaque combinaison distinguée C correspondra un sommet du graphe et un seul. Ce sommet sera construit comme suit. Si C ne se réduit pas à une seule notion, on lui associe toutes les combinaisons distinguées $C_1,\ldots,C_p$ qui sont telles que

$\forall\ i \in [p] \quad C_i \subset C$

et il n'existe pas une combinaison C' différente de C_i et C vérifiant $C_i \subset C' \subset C$.

Notons que l'on a bien

$C = C_1 \cup C_2 \cup \ldots \cup C_p$

Le sommet représentatif de C sera construit comme descendant par un arc étiqueté par "+" de tous les sommets représentant C_1, $C_2,\ldots,C_p$.

<u>Exemple</u>: Sur la figure 4.23 sont représentées les combinaisons

{*homme, riche*}
{*homme, riche, beau*}
{*homme, beau, intelligent*}
{*homme, riche, beau, intelligent*}.

<u>Règle</u>: Soit $C = \{N_1,\ldots,N_n\}$ une combinaison distinguée représentée par un sommet a et telle que les notions primitives N_1, $N_2,\ldots,$ N_n constituant C sont représentées respectivement par les sommets $a_1,\ldots,$ a_n. Si b est un sommet qui descend des n

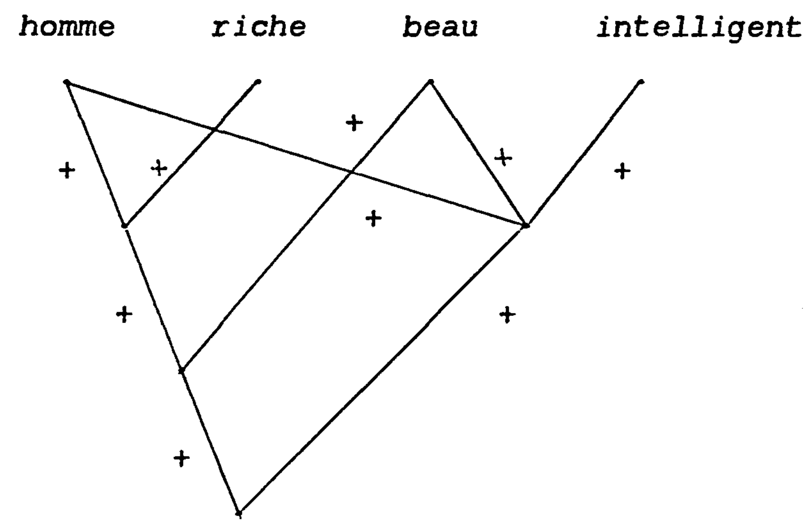


figure 4.23

sommets $a_1, \ldots, a_n$, alors il doit descendre de a.

Cela revient à dire en quelque sorte que si un objet a n propriétés, il a forcément l'ensemble des n propriétés.

Exemples: 1. Soient les connaissances suivantes:

tous les meubles solides sont chers

toutes les chaises métalliques sont solides

toutes les chaises sont des meubles

représentées par le graphe de la figure 4.24.

Il est nécessaire que le sommet b descende du sommet a, sans cela on ne trouverait pas que les chaises métalliques sont chères.

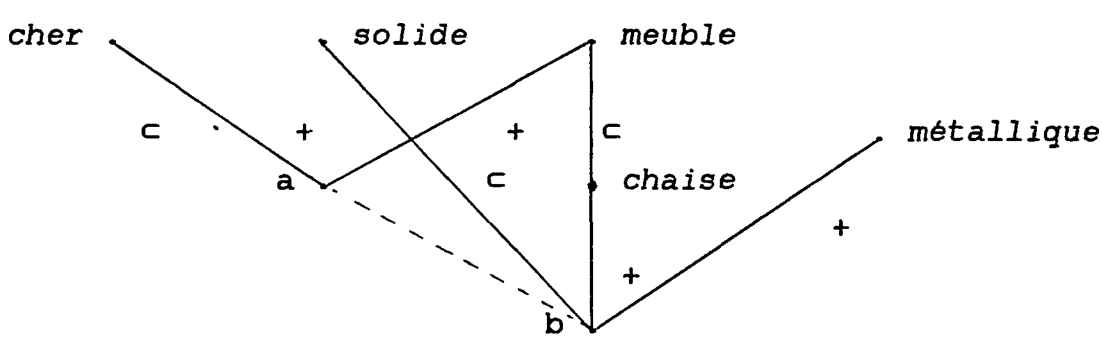


figure 4.24

2. Soient les connaissances suivantes:

tous les scientifiques intelligents sont pédants

tout mathématicien est un scientifique

Jean est un mathématicien intelligent

représentées par le graphe de la figure 4.25.

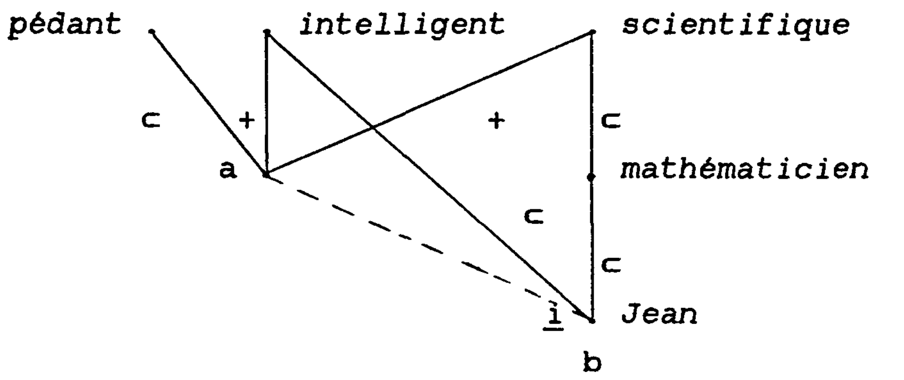


figure 4.25

Il est encore nécessaire ici que le sommet b descende du sommet a.

Remarques: 1. Dans ce paragraphe, nous nous sommes contentés de reprendre un problème abordé dans [Fahlman, 79], problème dont nous poursuivons l'étude, en y incluant les propositions qui servent à définir des objets typiques, en 4.2.6. Nous donnons une expression rigoureuse et systématiques aux solutions suggérées par Fahlman.

Fahlman parle de "classes définies" et considère des sommets d'un type particulier, ou *EVERY. Il distingue bien les propriétés définitoires d'un objet, qu'il appelle "spécifications", des autres propriétés, qu'il qualifie d'"incidentes". Toutefois, il affirme que le problème ne se pose qu'au moment de l'introduction de nouveaux objets dans le réseau alors que, comme nous allons le voir en 5.4.2, le problème se pose également au cours de l'interrogation.

Notons que dans KL-ONE ([Brachman-Schmolze, 84]) il est également fait une distinction entre "concepts primitifs" et "concepts définis", mais que les concepts définis sont définis par tous leurs ascendants immédiats. Il ne semble pas que l'on puisse faire la distinction pour ces concepts entre propriétés définitoires et propriétés non définitoires.

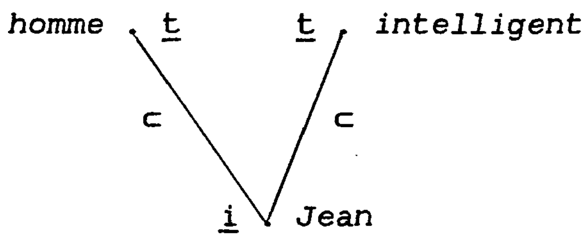
2. Pour obtenir une représentation homogène, nous étiquetterons par "⊂" les arcs reliant un sommet individuel aux sommets typiques qui en décrivent les propriétés. Cela nous donne par exemple la représentation de

Jean est un homme intelligent (figure 4.26)

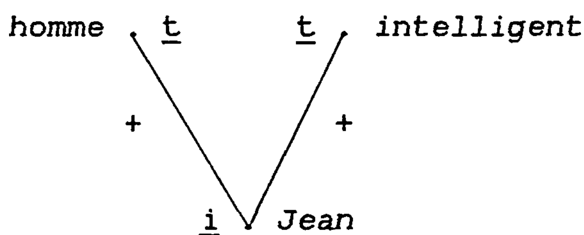
et de distinguer cette représentation de celle de

Jean est l'homme intelligent (figure 4.27)

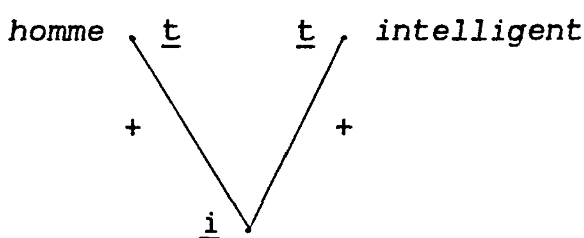
(en supposant que dans l'univers représenté il y a un seul *homme*



---------------------------figure 4.26---------------------------



---------------------------figure 4.27---------------------------



---------------------------figure 4.28---------------------------

intelligent). On pourra d'ailleurs représenter l'*homme intelligent* lorsqu'il n'y en a qu'un même si l'on n'en connaît pas le nom (figure 4.28).

3. De la représentation obtenue en figure 4.21 on pourra déduire que

toute pomme jaune est une pomme

ou que *toute pomme jaune est jaune.*

Mais il ne faut pas que de

Evariste Galois est un grand mathématicien

on puisse déduire que

Evariste Galois est grand.

L'adjectif, dans cet exemple, agit comme un modificateur du nom *mathématicien*, et pas comme une notion intrinsèque. Il n'exprime pas à lui tout seul une qualité de l'objet *Evariste Galois*. De même un *petit éléphant* ne sera pas nécessairement un objet *petit*, en tout cas moins *petit* qu'une *grande fourmi*!

La représentation que l'on donnera (figure 4.29) utilisera une nouvelle étiquette d'arcs: "I".

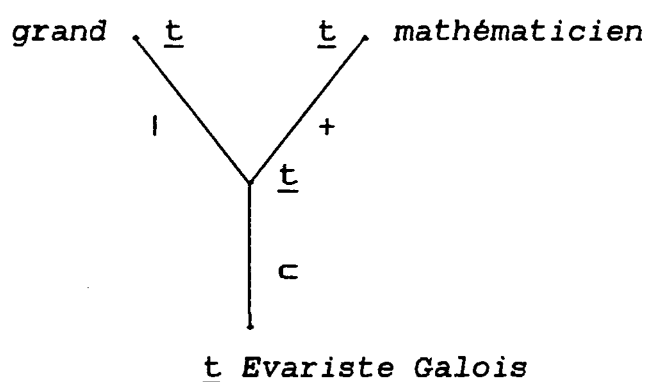


figure 4.29

Notons que de la représentation (figure 4.30) choisie alors pour le texte

Evariste Galois est un grand mathématicien français

on ne pourra pas déduire que

Evariste Galois est un grand Français.

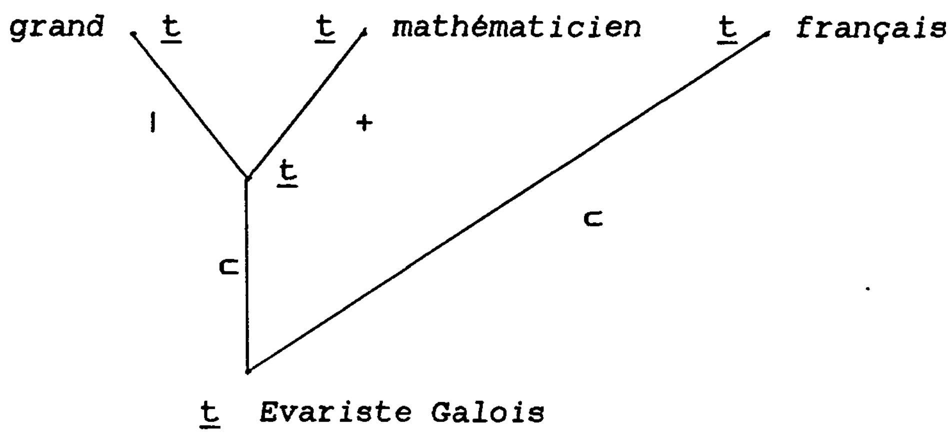


figure 4.30

Finalement, la distinction entre adjectifs et noms sera moins significative dans notre perspective qu'une distinction entre les noms et adjectifs qui expriment une propriété absolue d'un objet d'une part, les adjectifs qui expriment une propriété de l'objet relativement à l'appartenance de cet objet à une classe d'autre part.

Remarquons qu'il est difficile de déterminer relativement à quelle classe la propriété est définie. Par exemple *un homme grand* est *grand* par rapport à l'ensemble des *personnes*, ce qui ne sera pas le cas d'un *enfant grand*. On posera donc que si un sommet typique descend d'un autre sommet typique par un arc étiqueté par |, l'adjectif ne se combine qu'aux ascendants immédiats de ce sommet pour former une nouvelle notion.

Tous les arcs devant être étiquetés, mais les successeurs d'un même sommet n'ayant pas besoin d'être ordonnés, la structure représentative des connaissances est, pour l'instant, un $\emptyset$-réseau $H = \langle X, \delta, \xi, \rho \rangle$ où ρ est une application de X dans $\{+, \subset, |\}^*$.

4.2.3 Unités lexicales associées aux sommets typiques et individuels

Un objet typique ou individuel n'est pas défini par l'étiquette lexicale qu'il reçoit, et ceci pour deux raisons:

(1) Comme nous l'avons vu sur les différents exemples, un

sommet typique n'est pas obligatoirement étiqueté par une unité lexicale. Notamment quand l'ensemble correspondant à l'objet typique est construit par intersection d'ensembles et qu'il n'existe pas dans la langue de nom ou d'adjectif représentant l'objet typique. Par exemple la *fleur rouge* (typique) représentée en figure 4.31.

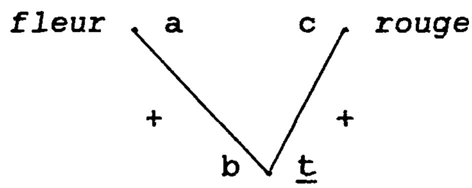


figure 4.31

Un sommet individuel est non étiqueté quand il représente un objet qui ne reçoit pas de nom (nom propre ou nom commun avec l'article défini marquant qu'il existe un seul objet de ce type dans l'univers). Par exemple pour une *fleur rouge* (figure 4.32).

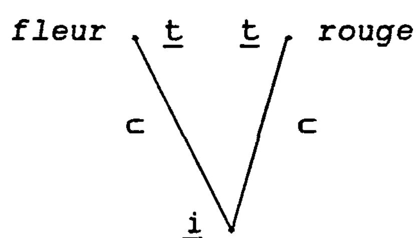


figure 4.32

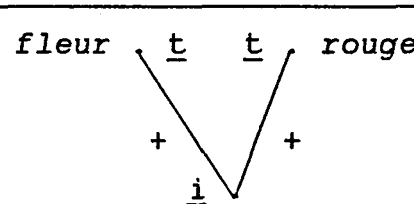
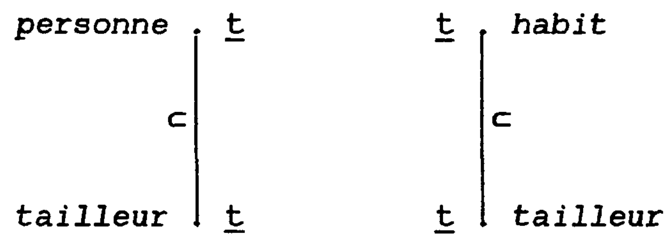


figure 4.33

Nous pouvons également représenter *la fleur rouge* quand cela signifie qu'il y a une seule *fleur rouge* dans l'univers, comme on le voit sur la figure 4.33.

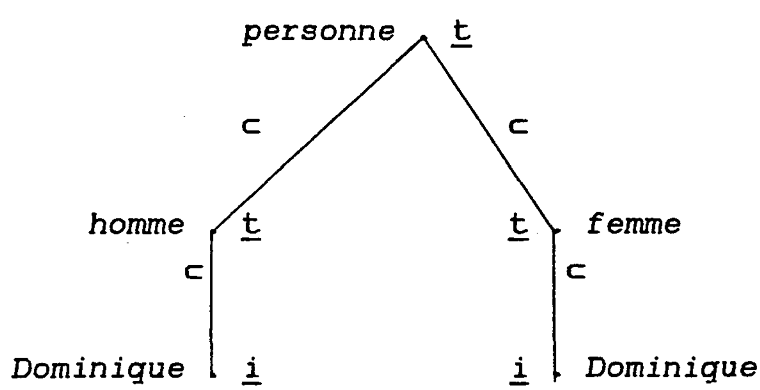
(2) Plusieurs sommets typiques ou plusieurs sommets individuels peuvent recevoir la même étiquette lexicale.

Par exemple un *tailleur* peut être une *personne* ou un *habit*, comme on le voit en figure 4.34. L'ambiguïté est levée par la connaissance des ascendants du sommet étiqueté par *tailleur*.



---figure 4.34---

De même peut-on distinguer entre des objets individuels qui reçoivent la même unité lexicale, comme par exemple on le voit en figure 4.35.



---figure 4.35---

4.2.4 Les propositions

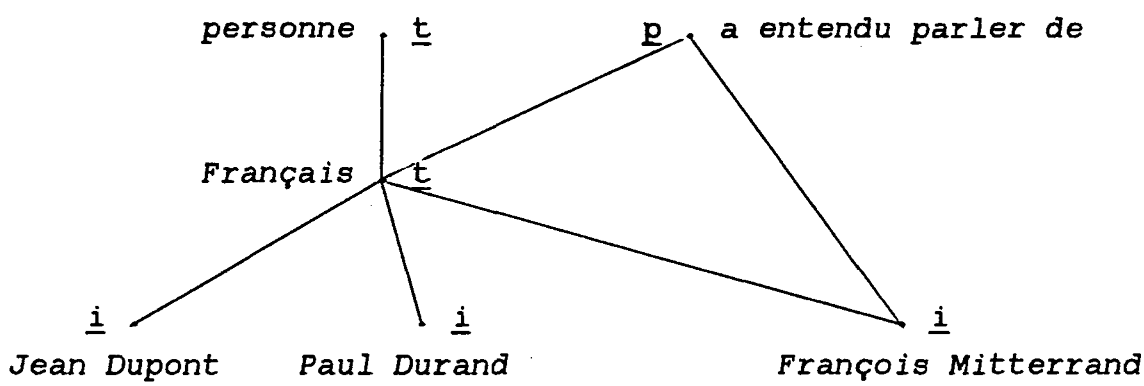
Dans la partie 4.1 nous avons indiqué comment représenter des propositions entre objets individuels. Nous savons à présent que ces représentations peuvent être intriquées avec des représentations de connaissances classificatoires.

Il est intéressant d'examiner maintenant comment on peut faire porter les propositions soit à la fois sur des objets individuels et des objets typiques, comme dans la représentation (figure 4.36) de

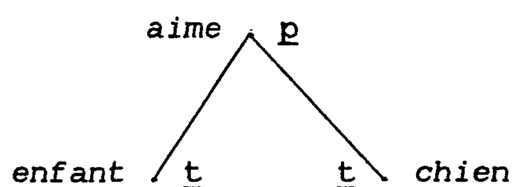
Tout Français a entendu parler de François Mitterrand

soit uniquement sur des objets typiques comme dans la représentation (figure 4.37) de

Les enfants aiment les chiens



--------------------------figure 4.36--------------------------



--------------------------figure 4.37--------------------------

De la première représentation on peut déduire que

Jean Dupont a entendu parler de François Mitterrand

ou *François Mitterrand a entendu parler de François Mitterrand*

tandis que la deuxième représentation peut être traduite avec plus d'exactitude par

tout enfant aime tout chien.

Remarquons que les verbes a *entendu parler de* ou *aime* reçoivent ici respectivement les types <p,ti> et <p,tt>, alors qu'ils recevaient précédemment tous les deux le type unique

<p,ii>. Ceci se vérifiera plus généralement pour tous les verbes de type <p,ii>. C'est pourquoi on définira une sorte globale n (comme nom) qui contiendra à la fois la sorte i et la sorte t (et plus tard la sorte j). Les verbes de type <p,nn> pourront alors avoir comme arguments soit des objets individuels, soit des objets typiques, soit les deux.

On peut maintenant représenter certaines relatives, celles qui, accollées à un nom, définissent une nouvelle classe d'objets. En effet, dans certains cas, les relatives ne font que marquer la juxtaposition de deux énoncés. Il en est presque toujours ainsi quand l'antécédent de la relative est un objet individuel, comme par exemple dans:

Jean qui aime Marie habite Marseille

Cette phrase peut être paraphrasée en:

Jean aime Marie et Jean habite Marseille

à moins qu'il ne s'agisse de distinguer entre plusieurs *Jean*.

Une telle paraphrase est en revanche impossible pour la phrase

Les hommes qui aiment Marie habitent Marseille.

dont une représentation possible est en figure 4.38.

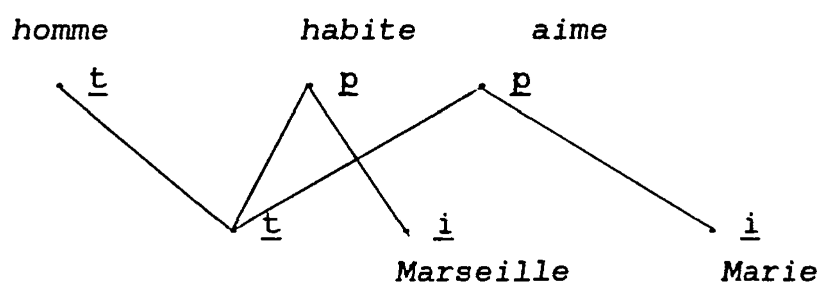
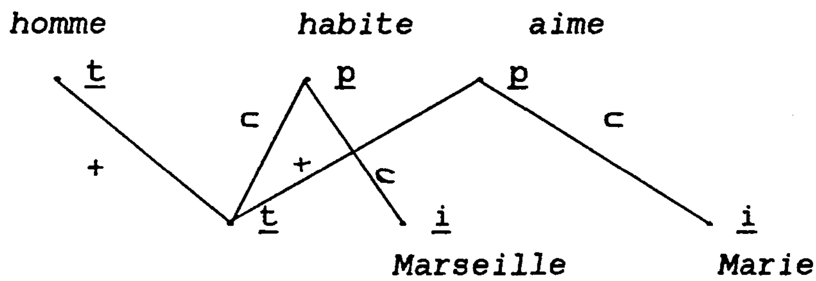


figure 4.38

Mais il est clair que de cette manière on a aussi bien représenté l'énoncé

Les hommes qui habitent Marseille aiment Marie.

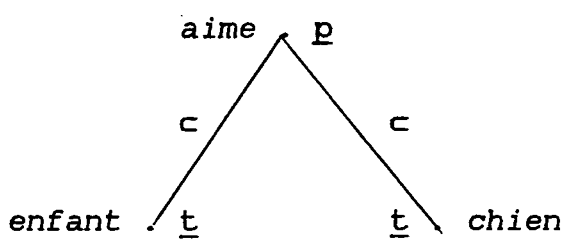
C'est pourquoi nous allons utiliser les étiquettes d'arcs "+" et "⊂" introduites au paragraphe 4.2.2. Cela donne pour l'énoncé initial la représentation de la figure 4.39.



---------------------------figure 4.39---------------------------

A partir de là, nous pourrons considérer que tout sommet typique est défini par l'ensemble de ses ascendants immédiats qui lui sont reliés par un arc étiqueté par "+". Les arcs étiquetés par "⊂" le reliant à des propriétés non définitoires. On aura alors une nouvelle représentation (figure 4.40) de l'énoncé

Les enfants aiment les chiens

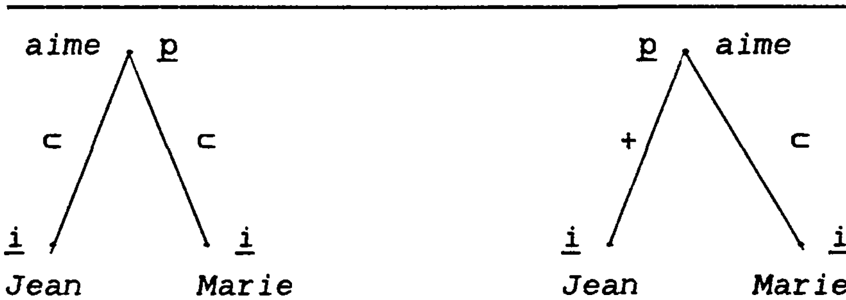


---------------------------figure 4.40---------------------------

Cet étiquetage d'arcs sera étendu aux objets individuels et permettra de distinguer entre

Jean aime Marie

et *Jean est celui qui aime Marie* (figure 4.41).



---------------------------figure 4.41---------------------------

Autres exemples:

Les hommes qui aiment la violence sont dangereux (figure 4.42)

les hommes que Jean admire sont intelligents (figure 4.43)

Notons que l'on a une autre représentation pour

Jean admire les hommes intelligents (figure 4.44).

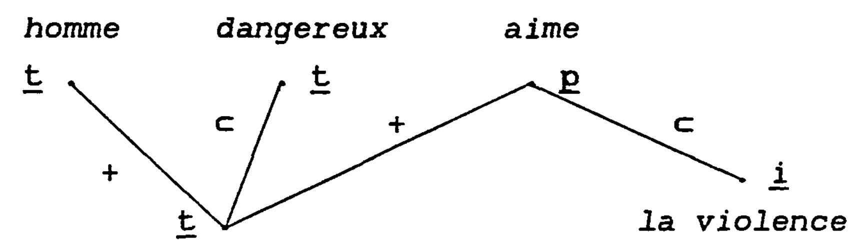


figure 4.42

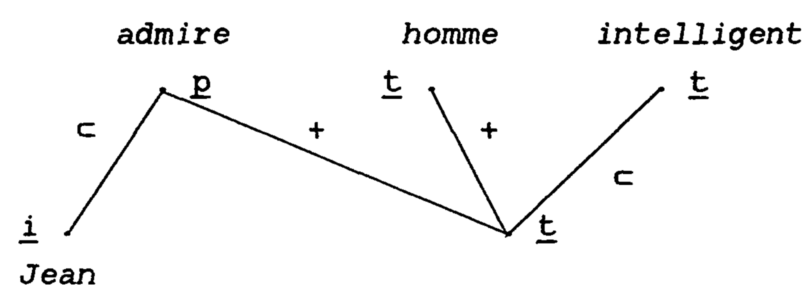


figure 4.43

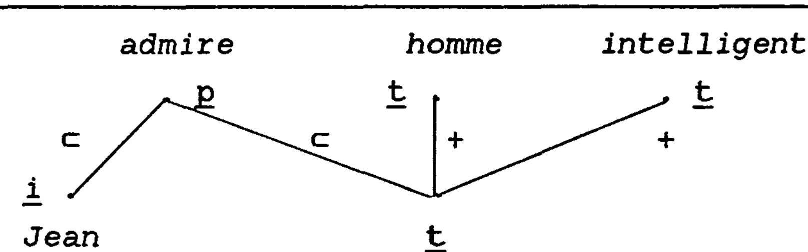


figure 4.44

Remarque: La représentation obtenue pour

Les éléphants qui volent sont roses (figure 4.45)

est très peu différente de celle qui aurait été obtenue en remplaçant *qui volent* par *volants*, la seule différence étant la sorte d'un sommet. Si l'on veut, on peut faire apparaître dans la représentation le sommet typique *objet volant* qui sera distinct de *objet qui vole* (figure 4.46).

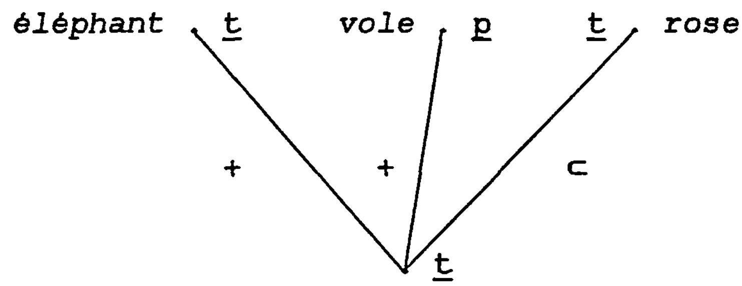


figure 4.45

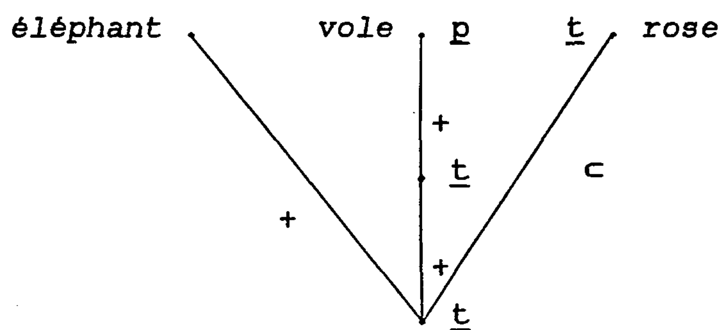


figure 4.46

On a créé, à partir du prédicat *vole* le sommet *objet volant* (*typique*). Une telle création est nécessaire si l'on veut affecter des propriétés à l'*objet volant*.

Plus généralement, à tout prédicat unaire, on peut faire correspondre le sommet qui représente l'objet typique qui possède la propriété caractérisée par le prédicat.

On peut même avoir cette distinction pour des prédicats binaires. Par exemple, si l'on veut représenter la connaissance

les enfants ont peur des serpents

il peut être nécessaire de représenter l'objet typique *qui a peur des serpents* ainsi que l'objet typique *dont les enfants ont peur*. Par exemple si l'on veut ajouter les connaissances suivantes:

ceux qui ont peur des serpents sont des peureux

ceux qui font peur aux enfants sont effrayants.
On obtient ainsi la représentation de la figure 4.47.

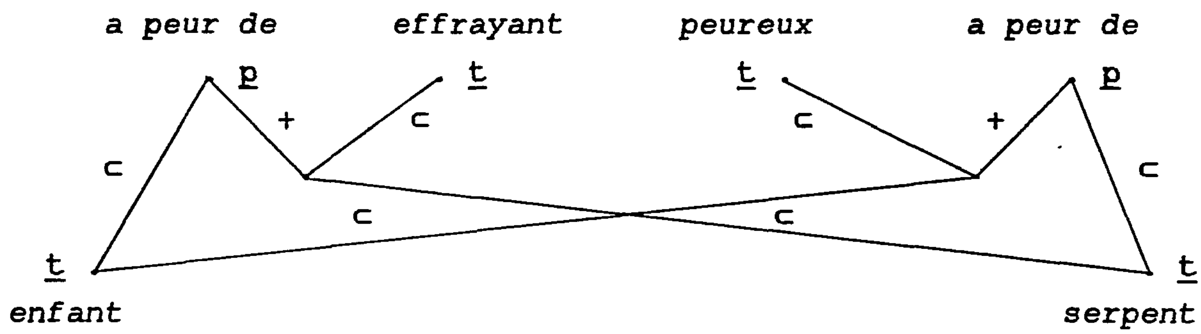


figure 4.47

Notons enfin que l'on peut représenter de manière analogue
l'objet typique que Jean presente à Marie (figure 4.48)
l'objet typique auquel Jean présente Pierre (figure 4.49)
l'objet typique qui présente Pierre à Marie (figure 4.50)

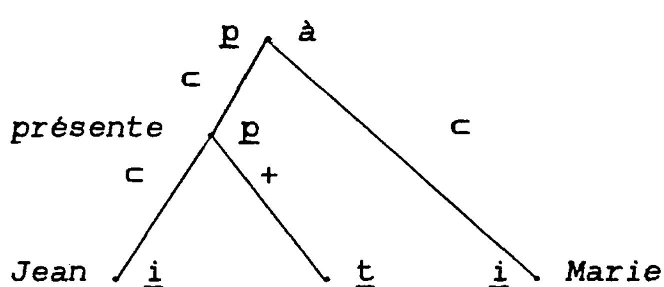


figure 4.48

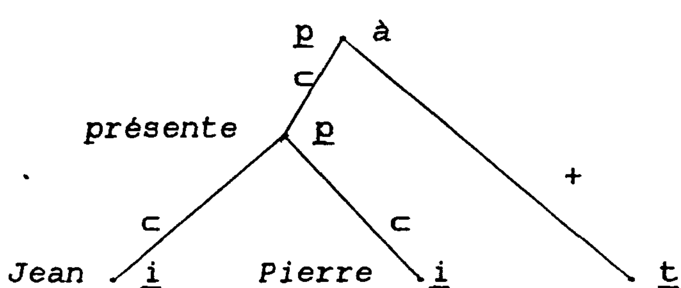
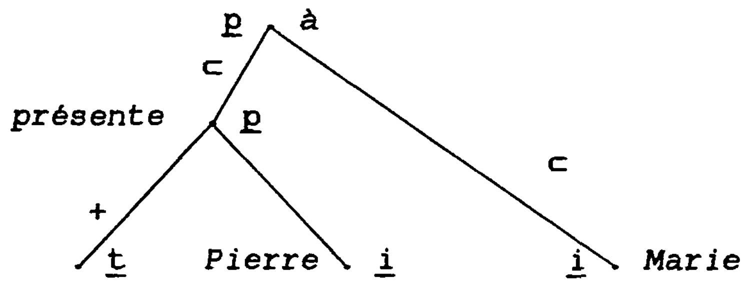


figure 4.49

C'est afin que finalement tous les arcs du graphe reçoivent



---figure 4.50---

une étiquette que nous affectons l'étiquette ⊂ aux arcs joignant deux sommets de sorte p. Dans la suite d'ailleurs, nous ommettrons fréquemment sur les figures les étiquettes d'arcs ⊂.

La structure représentative des connaissances est finalement un {p}-réseau H = <X,δ,ξ,ρ> : il est effectivement nécessaire que les successeurs des sommets de sorte p soient ordonnés, tandis que l'ordre entre successeurs des autres sommets est indifférent.

4.2.5 Quantification

On peut considérer que les énoncés qui ont été représentés dans le paragraphe précédent comportent une quantification universelle. En effet, avec une représentation fondée sur le calcul des prédicats on aurait des formules telles que:

∀ x *Français*(x) ==> *a_entendu_parler_de*(x,*François_Mitterrand*)
∀ x ∀ y *enfant*(x) et *chien*(y) ==> *aime*(x,y)
∀ x *homme*(x) et *aime*(x,*Marie*) ==> *habite*(x,*Marseille*)
∀ x *homme*(x) et *aime*(x,*la violence*) ==> *dangereux*(x)
∀ x *homme*(x) et *admire*(*Jean*,x) ==> *intelligent*(x)
∀ x *homme*(x) et *intelligent*(x) ==> *admire*(*Jean*,x)
∀ x *éléphant*(x) et *vole*(x) ==> *rose*(x)

Toutefois, dans toutes ces formules la quantification universelle porte sur l'ensemble de la formule. Nous pouvons également représenter certains énoncés pour lesquels il y aura une quantification universelle "interne" à la formule. Par exemple:

les hommes qui connaissent tous les ministres sont influents (figure 4.51)

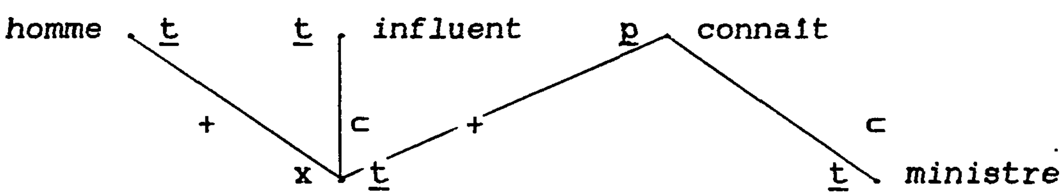


figure 4.51

Il est remarquable dans ce cas que la quantification universelle, qui est en réalité une quantification existentielle puisqu'elle est dans l'antécédent d'une implication, reste représentée par un sommets de sorte t. On peut donc dire que les sommets de sorte t donnent une représentation de la notion intuitive de quantification universelle.

Remarquons que l'on peut utiliser le même graphe, en changeant uniquement la sorte du sommet x de t en i si l'on veut dire qu'il y a un seul *homme qui connaît tous les ministres*.

Dès lors, il devient possible de rendre compte de l'appartenance d'un objet typique à l'ensemble qu'il caractérise. Plus précisément, cette appartenance sert à définir l'ensemble caractérisé par l'objet typique.

A cette fin on se donne une nouvelle pseudo-unité lexicale ϵ, grâce à laquelle on peut représenter par exemple l'*ensemble des éléphants* selon le graphe de la figure 4.52.

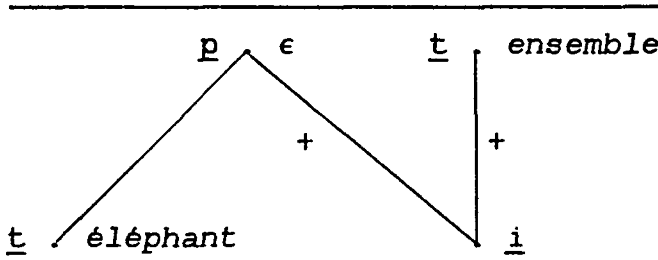


figure 4.52

L'*ensemble des éléphants* est un objet individuel: c'est le seul *ensemble* auquel, par définition, chaque *éléphant* appartient.

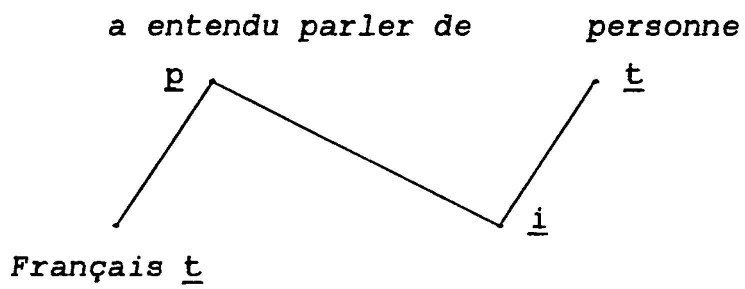
Cet ensemble pourrait éventuellement descendre de l'*ensemble typique situé en Afrique*, ou de l'*ensemble typique en voie de disparition*: ce n'est en effet pas *chaque éléphant qui est en voie de disparition*!

Il reste à voir comment, dans notre système de représentation, seront traités les énoncés qui requièrent une quantification existentielle.

Rappelons d'abord qu'un objet n'est vraiment quantifié existentiellement que s'il dépend d'un (ou plusieurs) objets quantifiés universellement. Il est dans ce cas fonction de Skolem de ces objets.

Dans le cas contraire, on est ramené à un type de représentation déjà étudié. Par exemple (figure 4.53):

Il y a une personne dont tous les Français ont entendu parler



---------------------------figure 4.53------------------------

Considérons maintenant les objets quantifiés existentiellement qui dépendent d'un unique objet quantifié universellement (c'est-à-dire, pour nous, d'un unique objet typique).

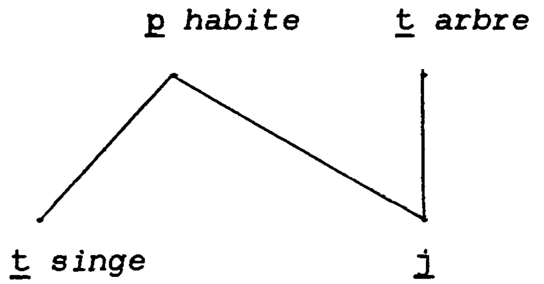
On attribue à ces objets la sorte j (objet individuel fonction d'un objet typique). On représente alors

tout singe habite un arbre

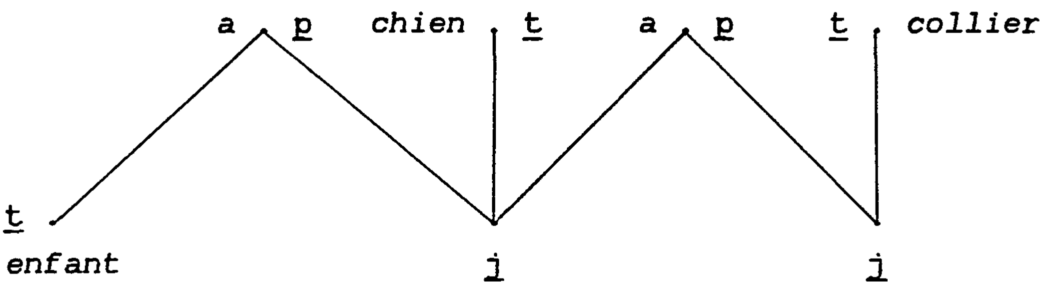
par le graphe de la figure 4.54.

Autre exemple (figure 4.55):

tout enfant a un chien qui a un collier



-------------------------figure 4.54-------------------------

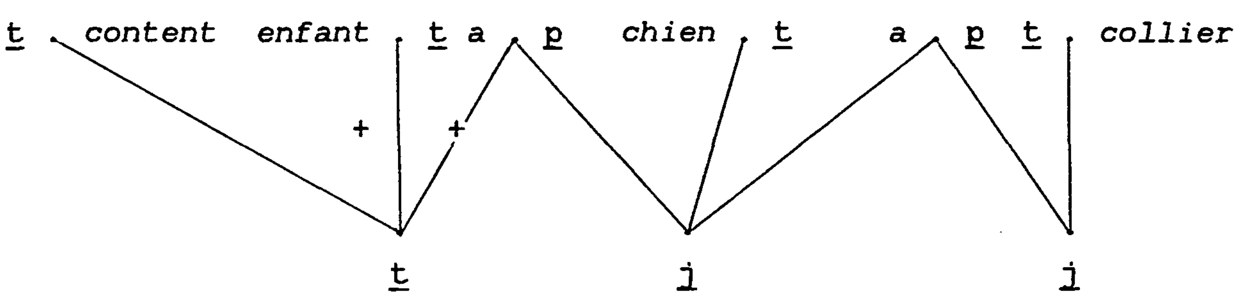


-------------------------figure 4.55-------------------------

Remarquons que *collier* dépend de l'objet quantifié universellement *enfant*.

On peut, comme pour les objets quantifiés universellement, avoir une quantification existentielle plus "intérieure". Par exmple (figure 4.56) pour l'énoncé:

les enfants qui ont un chien qui a un collier sont contents



-------------------------figure 4.56-------------------------

Notons que la quantification existentielle qui est dans ce cas dans l'antécédent d'une implication est en réalité une quantification universelle, mais qu'elle reste représentée par un sommet de sorte j.

Le cas d'un objet quantifié existentiellement qui dépend de plusieurs objets quantifiés universellement ne se rencontre pas très couramment dans la représentation de connaissances liées à des domaines de la vie courante, alors qu'il se rencontre très fréquemment en mathématiques ou dans des disciplines scientifiques proches. Par exemple:

la combinaison d'un acide et une base forme un sel.

Afin de représenter cette connaissance, on se donne un prédicat de type <p,nn>, *combinaison*, et un prédicat de type <p,pn>, *forme*, qui sont deux prédicats liés au domaine de la chimie dans lequel on se place. On a alors la représentation de la figure 4.57.

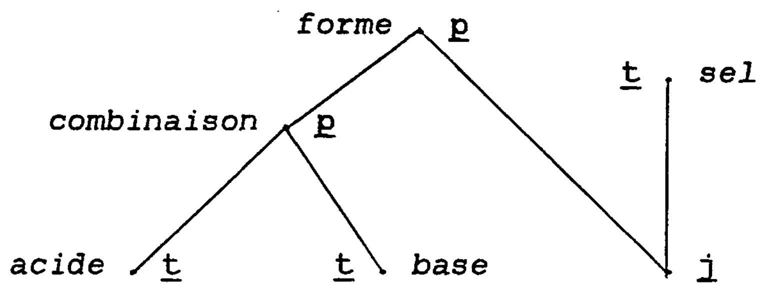


figure 4.57

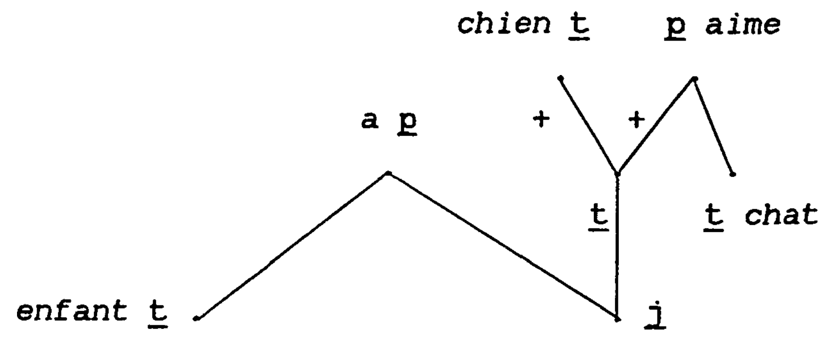
Lorsqu'on voudra exprimer à l'aide de prédicats des propriétés d'objets quantifiés existentiellement, on sera obligé, afin de ne pas introduire de confusions, de passer par un sommet typique intermédiaire.

Par exemple pour exprimer que

tout enfant a un chien qui aime tous les chats

selon la représentation de la figure 4.58. Sans sommet typique intermédiaire on représenterait la connaissance:

pour tout enfant et pour tout chat, il existe un chien que l'enfant a et qui aime le chat



--------------------------figure 4.58--------------------------

Si l'on veut retrouver l'objet (ou les objets) quantifié universellement duquel dépend un objet quantifié existentiellement, ou plutôt les sommets représentant ces objets, il suffit d'employer l'algorithme suivant, décrit de manière récursive.

Soit x un sommet de sorte j. On notera T(x) l'ensemble des sommets de sorte t desquels dépend x.

Soit X_x l'ensemble défini de la manière suivante:

(i) tous les ascendants immédiats de x de sorte p sont dans X_x

(ii) tous les ascendants immédiats de sorte p d'un sommet de X_x sont dans X_x

(iii) tous les descendants immédiats de sorte p d'un sommet de X_x sont dans X_x.

Soit Y_x et Z_x les ensembles définis comme suit:

$Y_x = \{y \in \delta(X_x);\ \varepsilon^1(x) = \underline{t}\}$

$Z_x = \{y \in \delta(X_x);\ \varepsilon^1(x) = \underline{j}\}$

On pose: $T(x) = Y_x \text{ T } T(Z_x)$

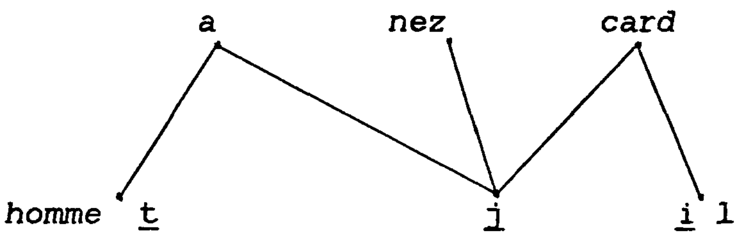
Remarques: 1. On peut préciser les quantités d'objets dépendant d'un objet typique en utilisant la relation *card* (pseudo-unité lexicale de sorte p). Nous le voyons sur les exemples qui suivent:

tout homme a un nez (et un seul) (figure 4.59)

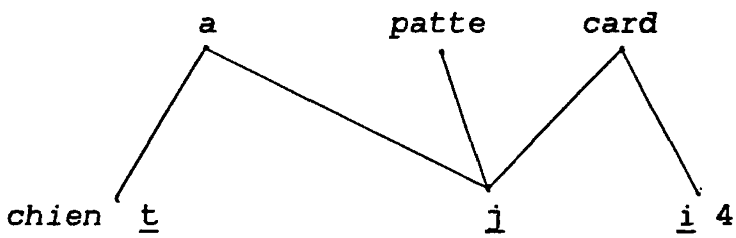
tout chien a 4 pattes (figure 4.60)

tout mammifère a des poils (figure 4.61)

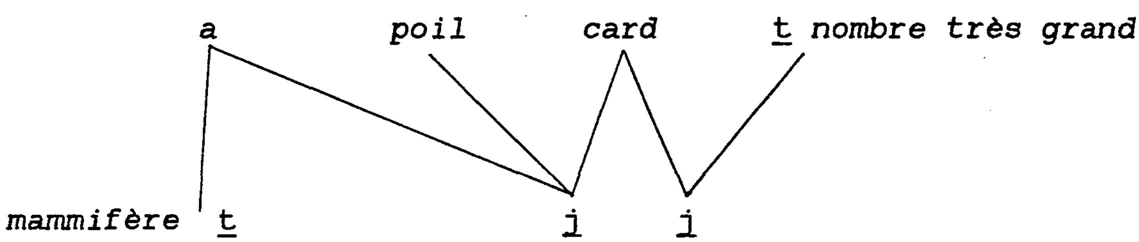
(Le nombre de poils dépend de chaque mammifère).



---------------------------figure 4.59---------------------------

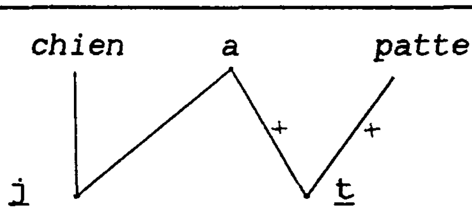


---------------------------figure 4.60---------------------------



---------------------------figure 4.61---------------------------

2. *La patte du chien typique* n'est pas *la patte typique de chien*. Dans le premier cas il s'agit d'un objet de sorte j alors que dans le deuxième cas il s'agit d'un objet de sorte t, représenté dans le graphe de la figure 4.62.



---------------------------figure 4.62---------------------------

On peut intriquer cette structure avec celle qui représente *la patte du chien typique* en donnant par exemple la représentation de (figure 4.63):

tout chien a 4 pattes de chien et toute patte de chien est possédée par un chien (et un seul).

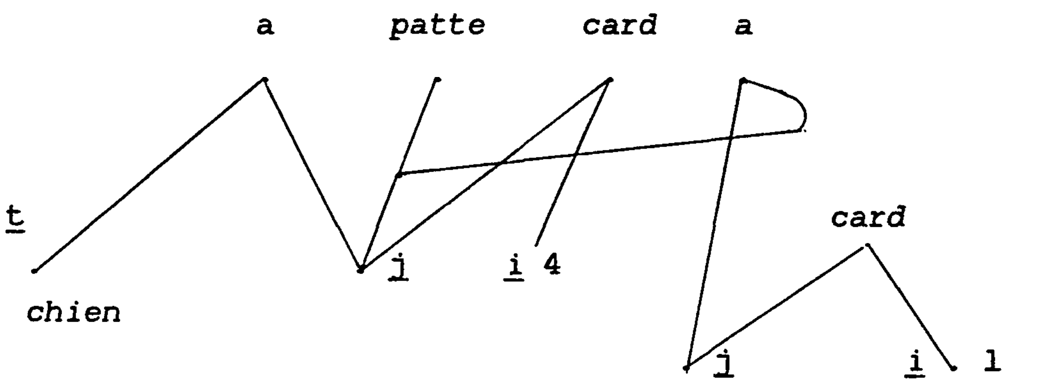


figure 4.63

3. Ce que l'on gagne en efficacité algorithmique en utilisant des réseaux sémantiques plutôt que des systèmes fondés sur le calcul des prédicats est perdu en difficultés dans le traitement de la quantification.

Les solutions les plus diverses ont été adoptées par les concepteurs de réseaux sémantiques. Très souvent les quantifications sont implicites. Ainsi, dans le système KL-ONE [Brachman-Schmolze, 84], les "rôles" induisent une quantification implicite, toujours du type suivant:

$\forall x \; \exists y \; P(x,y)$

permettant de représenter des connaissances du genre:

toute personne a une tête

On ne voit pas comment représenter

chaque éléphant a peur de chaque souris

et encore moins de faire la différence entre

tout enfant a un chien qui a un collier

et *tout enfant a un chien et tout chien a un collier* .

Les "réseaux partitionnés" de [Hendrix, 79] constituent une exception, dans la mesure où ils sont susceptibles de représenter toute formule du calcul des prédicats du premier ordre, et même plus que cela. Toutefois, de ce fait, l'objet "réseau partitionné" est un objet très complexe, à mi-chemin entre les réseaux sémantiques et les ensembles de formules logiques. La représentation de certaines connaissances telles que

toute personne qui connaît tous les ministres est influente

n'est ni immédiate, ni proche de l'intuition.

Quant à l'"algorithme de déduction" employé, il peut se révéler lourd dans des cas qui sont traités très simplement dans [Fahlman, 79], faisant appel à un processus d'unification. Notamment, à chaque fois que l'on passe par une quantification universelle. Ainsi le processus d'interrogation pour

Jean connaît tous les ministres

Paul est un ministre

qui connaît Paul ?

est relativement complexe alors que chez nous, comme on le verra par la suite (5.3), il est immédiat, y compris dans le cas où les connaissances nécessaires pour répondre à la question sont obtenues par héritage des propriétés.

4.2.6 <u>Sous-graphe définitoire d'un objet typique</u>

Tout sommet typique représente un objet typique.

Un sommet typique admettant une étiquette lexicale représente un objet typique **simple**.

Un sommet typique qui n'admet pas d'étiquette lexicale représente un objet typique **complexe**. Afin de savoir quel objet est représenté par un tel sommet typique il sera nécessaire de considérer un sous-graphe du graphe auquel appartient le sommet typique. Ce sera le **sous-graphe définitoire de l'objet typique.**

Soit x un sommet typique. Afin de construire le sous-graphe définitoire de l'objet typique représenté par x (ou sous-graphe définitoire du sommet x) on construit d'abord récursivement l'ensemble $Y(x)$ de ses sommets.

Si x admet une étiquette lexicale, l'objet typique est défini par un sous-graphe réduit au sommet x et tel que $\delta(x) = \lambda$. On a $Y(x) = \{x\}$.

Sinon, soient $x_1, x_2, \ldots, x_p$ les ascendants immédiats de x de sorte <u>t</u> et tels que les arcs $\langle x_i, x\rangle$ soient étiquetés par "+", soient y_1, $y_2, \ldots,$ y_q les ascendants de x de sorte p de x tels que les arcs $\langle y_i, x\rangle$ soient étiquetés par "+".

A chacun des sommets y_i de sorte p on associe un ensemble de sommets noté $Z(y_i,x)$ et défini de la manière suivante:

(i) si $\delta(y_i) = x$ alors $Z(y_i,x) = \{y_i\}$

(ii) sinon, soit z_i le successeur de y_i distinct de x

- si z_i est de sorte i, alors $Z(y_i,x) = \{y_i, z_i\}$
- si z_i est de sorte t, alors $Z(y_i,x) = \{y_i\} \cup Y(z_i)$
- si z_i est de sorte j, alors soient $u_i^1,\ldots,u_i^r$ les ascendants immédiats de z_i de sorte t et soient $v_i^1,\ldots,$ v_i^s les ascendants immédiats de z_i de sorte p distincts de y_i; on pose:

$$Z(y_i,x) = \{y_i,z_i\} \cup Y(u_i^1) \cup\ldots\cup Y(u_i^r) \cup Z(v_i^1,z_i) \cup\ldots\cup Z(v_i^s,z_i)$$

Finalement:

$$Y(x) = \{x\} \cup Y(x_1) \cup\ldots\cup Y(x_p) \cup Z(y_1,x) \cup\ldots\cup Z(y_q,x)$$

Le sous-réseau $H_1 = \langle X_1,\delta_1,\xi_1,\rho_1\rangle$ définitoire de x est tel que:

- pour tout y de Y(x) de sorte p, $\delta_1(y) = \delta(y)$;
- si y et z appartiennent à Y(x), si $\langle y,z\rangle$ est un arc de H étiqueté par +, alors $\langle y,z\rangle$ est également un arc de H_1;
- si y et z appartiennent à Y(x), si $\langle y,z\rangle$ est un arc de H, si z est de sorte j, alors $\langle y,z\rangle$ est également un arc de H_1.

Exemple: Soient les connaissances suivantes

Marie connait toutes les villes de France qui possèdent un vieux chateau qui a un pont-levis

les vieux chateaux sont pittoresques

représentées par le graphe de la figure 4.64.

On a:

$Y(b) = \{b\} \cup Y(k) \cup Z(h,b) \cup Z(i,b)$

$Y(k) = \{k\}$

$Z(h,b) = \{h,c\}$

$Z(i,b) = \{i,d\} \cup Y(f) \cup Z(j,d)$

$Y(f) = \{f,l,m\}$

$Z(j,d) = \{j,e,o\}$

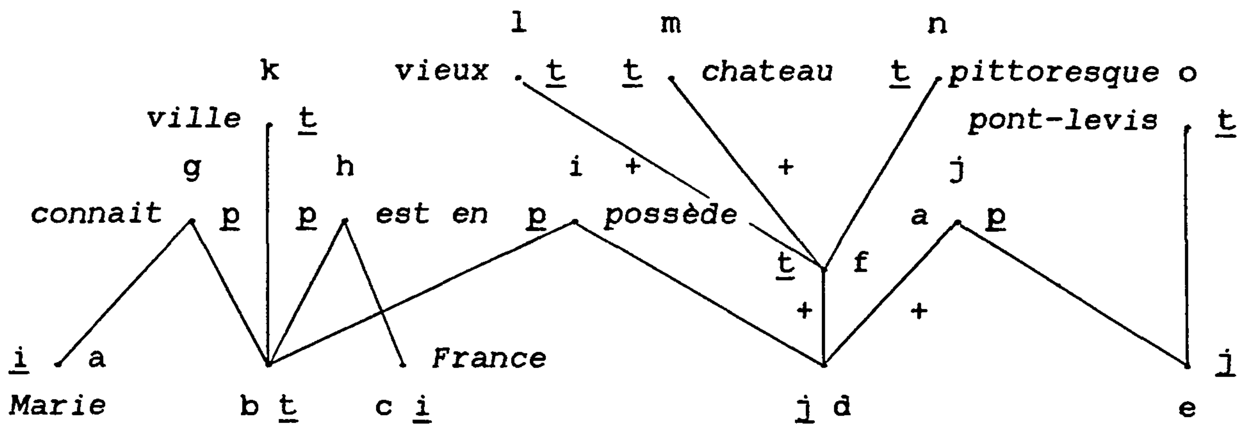


figure 4.64

et par conséquent:

$Y(b) = \{b,k,h,c,i,d,f,l,m,j,e,o\}$

La règle de cohérence exprimée au paragraphe 4.2.2 est maintenant modifée. Elle devient la règle suivante:

Règle: Soient $H = \langle X,\delta,\xi,\rho\rangle$ un graphe représentatif de connaissances et $H_1 = \langle X_1,\delta_1,\xi_1,\rho_1\rangle$ le sous-graphe définitoire d'un sommet typique x. S'il existe un sous-graphe $H_2 = \langle X_2,\delta_2,\xi_2,\rho_2\rangle$ de H distinct de H_1 et un agrandissement de réseaux ψ de H_1 dans H_2 qui conserve les unités lexicales $((\langle 2\rangle,\langle 2\rangle)$-agrandissement de graphes multi-étiquetés au sens de (2.5.5)) et qui vérifie:

(i) $\forall x \in X_1 \quad \xi_1^1(x) \in S\setminus\{\underline{j}\} \Longrightarrow \xi_2^1\psi(x) = \xi_1^1(x)$

(ii) $\forall x \in X_1 \quad \xi_1^1(x) = \underline{j} \Longrightarrow \xi_2^1\psi(x) \in \{\underline{i},\underline{j}\}$

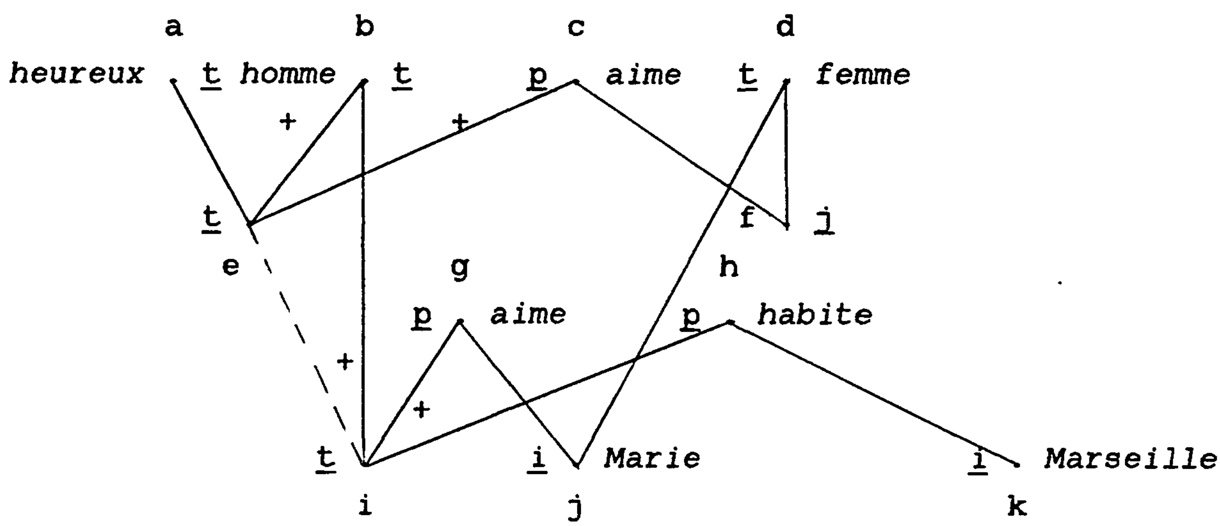
alors l'image $\psi(x)$ de x par cet agrandissement doit descendre de x.

La justification de cette règle est que si un objet possède toutes les propriétés définissant un objet typique, il doit hériter de toutes les autres propriétés de cet objet typique.

Exemples: 1. On se donne les connaissances suivantes:

les hommes qui aiment une femme sont heureux

Marie est une femme



---------------------------figure 4.65---------------------------

les hommes qui aiment Marie habitent Marseille
représentées par le graphe H de la figure 4.65.

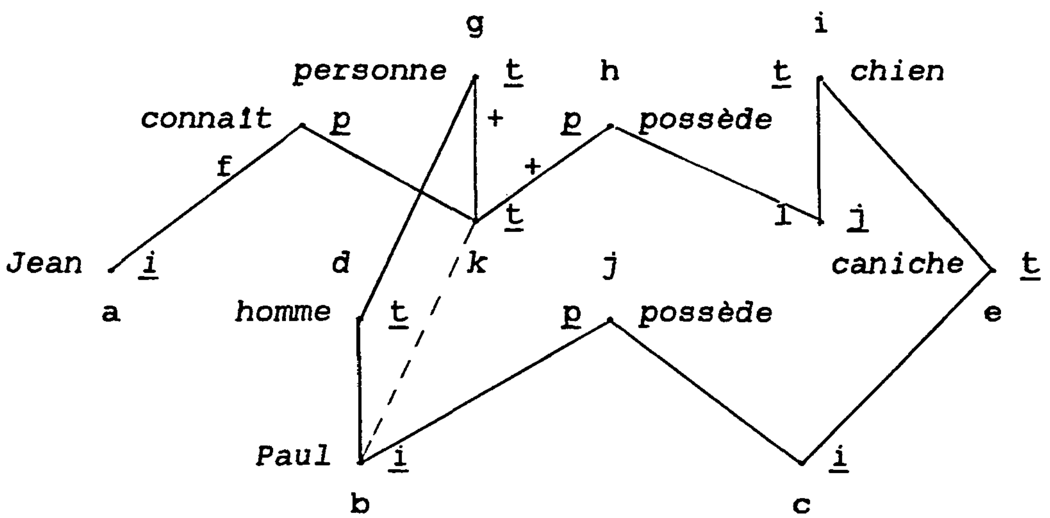
Soit H_1 le sous-graphe de H engendré par {b,c,d,e,f} et soit H_2 le sous-graphe de H engendré par {b,d,g,i,j}. H_1 est le sous-graphe définitoire du sommet typique e et il existe un agrandissement ψ de H_1 dans H_2 vérifiant les conditions requises tel que e ait pour image i. i doit donc descendre de e.

2. On se donne les connaissances suivantes:
Jean connaît toutes les personnes qui possèdent un chien
tout homme est une personne
tout caniche est un chien
Paul est un homme qui possède un caniche
représentées par le graphe H de la figure 4.66.

Le sous-graphe H_1 définitoire du sommet typique k est le sous-graphe de H engendré par {g,h,i,k,l}.

Il existe un agrandissement ψ H_1 sur le sous-graphe H_2 de H engendré par {b,c,d,e,g,i,j} (figure 4.67).

Il faut donc que b, image de k par ψ, descende de k. Sans cela, on ne pourrait savoir que Jean connaît Paul.



-------------------------figure 4.66-------------------------

x	g	h	i	k	l
$\psi(x)$	g	j	i	b	c

-------------------------figure 4.67-------------------------

4.2.7 Modalités, négation

Nous pouvons, de la même manière qu'en 4.1.5, insérer les prédicats qui portent sur des objets typiques, ou sur des objets typiques et des objets individuels, dans des conditions énonciatives, ou des modalités.

Cela nous permet de représenter des connaissances telles que:

Avant 1960 tous les instituteurs enseignaient la morale (figure 4.68) ou

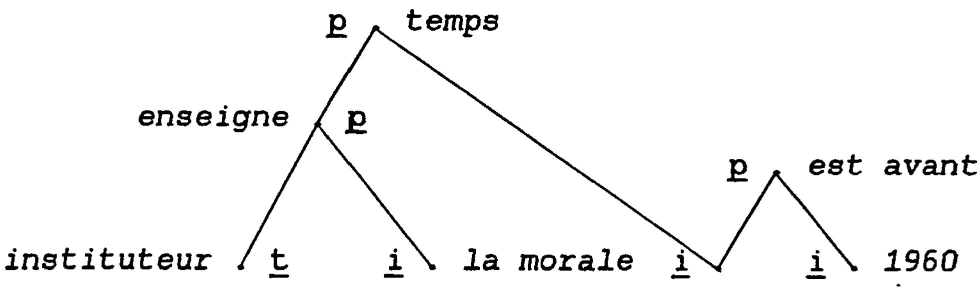
Paul croit que tous les Français admirent François Mitterand (figure 4.69).

Du premier énoncé, si l'on sait de plus que

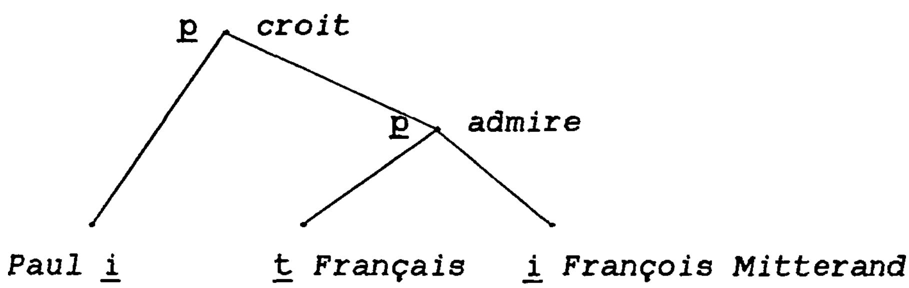
Jean Dupont est instituteur

on ne peut déduire que

Avant 1960 Jean Dupont enseignait la morale



---------------------------figure 4.68---------------------------



---------------------------figure 4.69---------------------------

parce qu'on ne sait pas si

Jean Dupont était instituteur avant 1960.

De même, du second énoncé on ne peut déduire avec la connaissance supplémentaire

Pierre est français

que *Paul croit que Pierre admire François Mitterand*

parce qu'on ne sait pas si

Paul sait que Pierre est français.

L'héritage des propriétés ne fonctionne donc pas pour des propriétés modalisées.

Notons que l'introduction des objets typiques et de l'héritage des propriétés nous permet d'avoir une meilleure représentation du temps et de la localisation que celle ébauchée précédemment (4.1.5).

A cette fin, on se donne deux objets typiques espace et *intervalle (de temps)*. Toute localisation d'un événement nécessitera l'usage d'un sommet descendant du sommet typique *espace*. Ainsi pourra-t-on représenter l'énoncé

Jean déjeune à Paris
par le graphe de la figure 4.70.

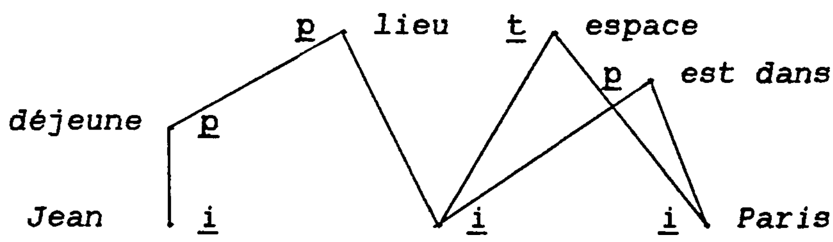


figure 4.70

De la sorte on indique que *Jean déjeune en un certain lieu*, et que ce lieu est situé à l'intérieur de *Paris*. Ce n'est pas n'importe où dans *Paris* que *Jean déjeune*. On aurait une représentation analogue de l'énoncé

Hier, Jean a déjeuné.

L'intérêt de cette nouvelle représentation de la localisation, c'est qu'elle permet d'effectuer des déductions; sachant que

Paris est en France et que

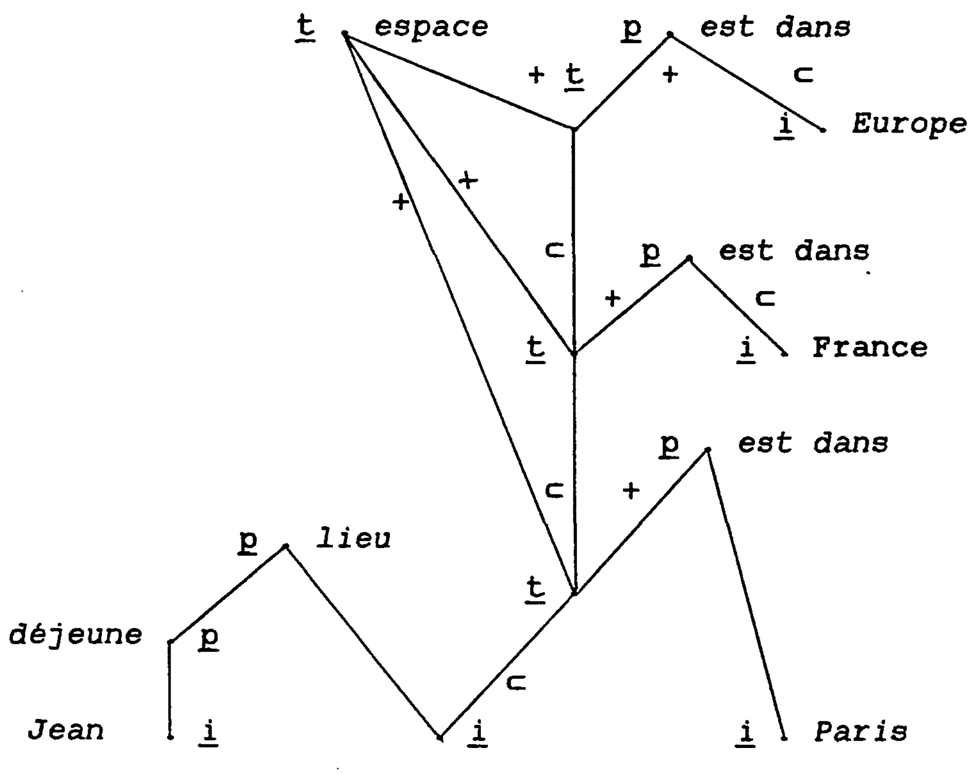
la France est en Europe,

on peut en déduire que Jean déjeune en Europe. Cette déduction passe par une hiérarchie d'espaces typiques: l'espace typique situé à Paris descend de l'espace typique situé en France, etc. On peut alors avoir la représentation de la figure 4.71.

Sur le graphe figurent:

- le sommet typique espace *qui est en Europe*
- descendant de ce sommet, le sommet typique *espace qui est en France*
- descendant de ce sommet, le sommet typique *espace qui est dans Paris*
- descendant de ce dernier sommet, le sommet individuel *espace dans lequel Jean déjeune*.

Nous pensons qu'avec une telle méthode, nous pouvons représenter un certain nombre de relations entre espaces telles

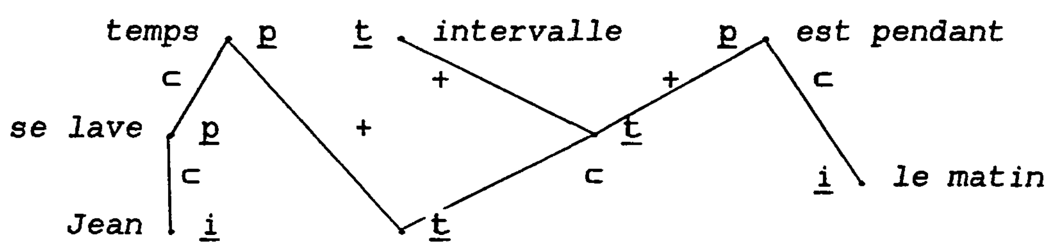


-------------------------<u>figure 4.71</u>-------------------------

la contiguïté, la proximité, etc... De même, nous pensons pouvoir représenter certaines relations entre intervalles de temps, telles que celles introduites par Allen [Allen, 83]. Et, également, d'avoir une certaine représentation de l'itération, par exemple pour l'énoncé

Jean se lave le matin

représenté par le graphe de la figure 4.72.



-------------------------<u>figure 4.72</u>-------------------------

L'intervalle de temps typique durant lequel *Jean se lave* descend de l'intervalle typique situé *le matin*.

permettra bien de trouver la connexion entre *Jean* et *intelligent*. Mais le passage par les arcs qui comportent cette nouvelle étiquette n'est pas aussi "libre" que par ceux qui comportent d'autres étiquettes, comme nous allons le voir dans le chapitre suivant.

Le cas extrême est celui de la négation portant sur le verbe être. Nous pouvons représenter de manière tout à fait semblable

Jean n'est pas intelligent (figure 4.74).

ou bien *aucun singe n'est une personne* (figure 4.75)

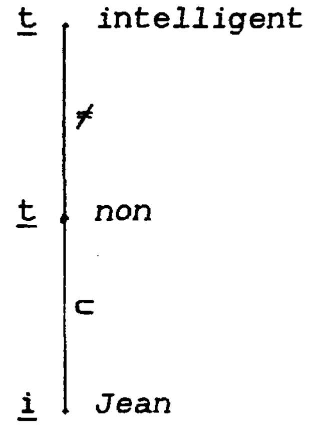


figure 4.74

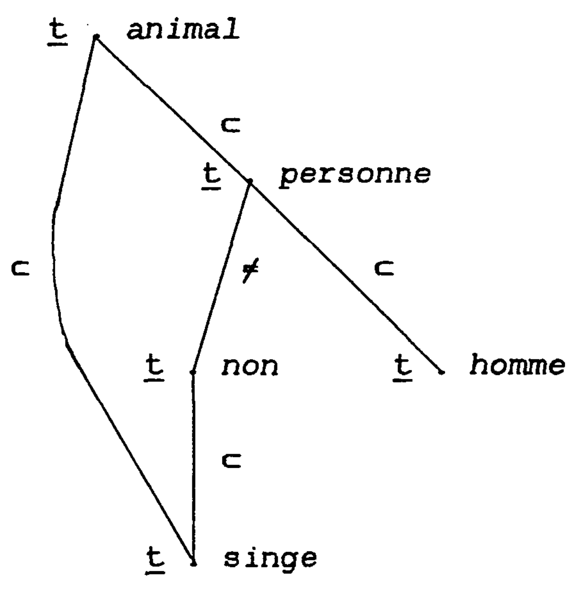


figure 4.75

On voit encore très bien sur ce dernier exemple qu'il n'y a pas héritage des propriétés pour les propriétés modalisées. En effet, il ne faut pas déduire de notre représentation que

aucun singe n'est un animal

On ne pourrait même rien dire sur les rapports entre notions *animal* et *singe* s'il n'existait un arc joignant directement les sommets représentatifs de ces notions.

On peut en revanche en déduire que

aucun singe n'est un homme

en raison des propriétés particulières de la modalité *non*. Il y a inversion du sens dans le parcours du graphe sans circuit: aucun descendant du sommet étiqueté par *personne* ne pourra être un *singe* - sauf si en redescendant du sommet on rencontre un nouvel arc étiqueté par ≠.

Nous aurons une représentation de même type pour les énoncés tels que:

Paul croit que tous les Français sont intelligents

(figure 4.76).

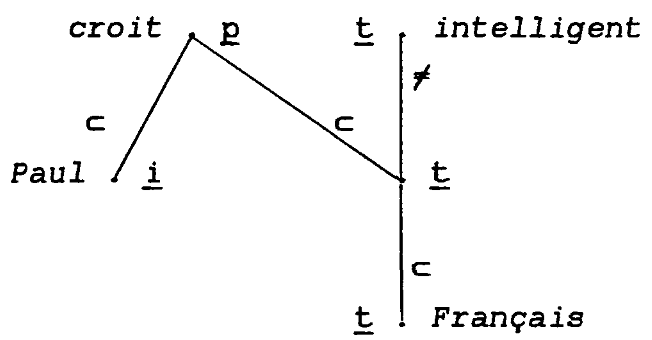
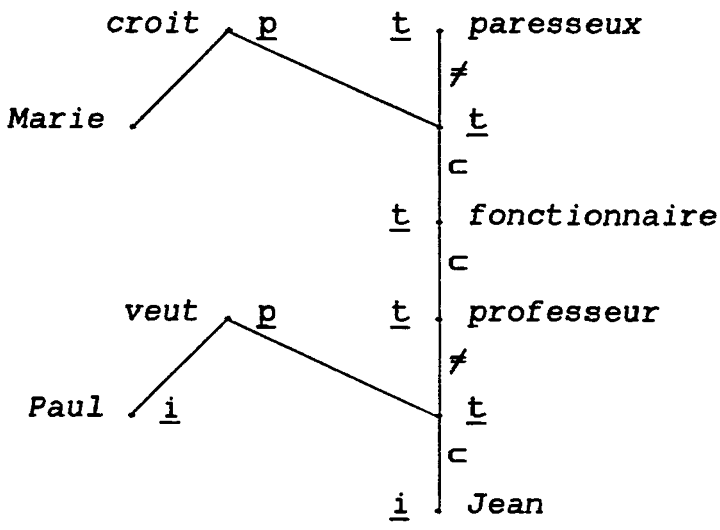


figure 4.76

Il faut toutefois être bien conscient qu'il y a un certain abus, car l'opérateur associé à *croit*, au lieu d'être de type <p,np> sera de type <p,nn>.

Donnons un autre exemple; on peut représenter sur un même graphe (figure 4.77) les connaissances suivantes:



--------------------------figure 4.77--------------------------

Paul veut que Jean soit professeur
Tous les professeurs sont fonctionnaires
Marie croit que tous les fonctionnaires sont paresseux

A partir d'une telle représentation, on peut répondre à la question

est-ce que Jean est paresseux ?

en restituant les connaissances enregistrées et en laissant l'interrogateur faire les déductions qu'il désire. L'extraction de ces connaissances d'après la question restent possible car il existe un chemin entre *paresseux* et *Jean* dans le graphe représentatif des connaissances.

4.2.8 Les exceptions

Rien, jusquà présent, n'interdit de représenter dans un même graphe des propositions qui soient contradictoires, comme par exemple (figure 4.78):

tous les mathématiciens sont intelligents
Jean est un mathématicien
Jean n'est pas intelligent

Ceci ne sera gênant que lorsque l'on posera la question:

est-ce que Jean est intelligent ?

Les deux réponses suivantes seront en effet extraites:

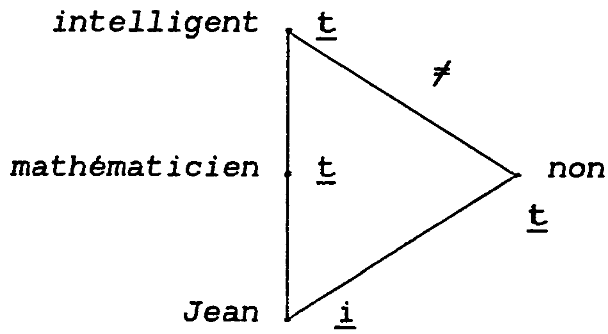


figure 4.78

Jean est intelligent

Jean n'est pas intelligent

le système d'interrogation ne pouvant que renvoyer à l'utilisateur les contradictions internes à la base de connaissances.

Mais il est possible de considérer que la propriété de *n'être pas intelligent*, directement affectée à *Jean*, l'emporte sur la propriété *d'être intelligent*, héritée par l'intermédiaire du *mathématicien* typique.

Dans cette hypothèse, un objet qui descend d'un objet typique n'hérite pas forcément de **toutes** les propriétés de cet objet typique. Un objet hérite des propriétés des objets typiques dont il descend à moins qu'une information incompatible avec une propriété de l'objet typique lui ait été affectée.

Ceci sera formalisé en adaptant la notion d'"ordre entre distances d'inférence" introduite dans [Touretzky, 86].

Dans ce qui suit, afin d'alléger les écritures, nous confondrons un sommet typique et la propriété qu'il représente.

Définition: Soient x, y et z trois sommets typiques d'un réseau $H = \langle X,\delta,\xi,\rho\rangle$. y hérite de la propriété x **par l'intermédiaire** de z si et seulement si z appartient à $\delta(x)$ et y appartient à $\hat{\delta}(z)$.

Règle: x l'emporte sur x' (non x) pour y (resp. x' l'emporte sur x) si et seulement si il existe z tel que y hérite de la propriété x (resp. x') par l'intermédiaire de z et, quel que soit z' tel que y hérite de la propriété x' (resp. x) par l'intermédiaire de z', z appartient à ö(z').

Exemples: 1. Reprenons (figure 4.79) l'exemple précédent.

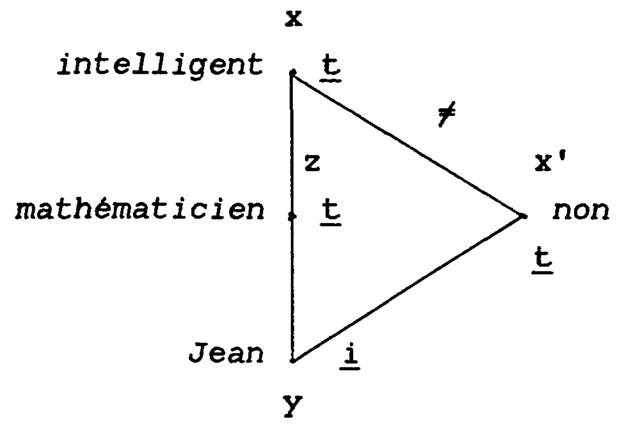


figure 4.79

On s'aperçoit que y hérite de x par l'intermédiaire de z alors qu'il hérite de x' par l'intermédiaire de y qui appartient à l'ensemble des descendants au sens strict de z. La propriété *Jean n'est pas intelligent* l'emporte donc sur la propriété *Jean est intelligent*

2. Soient les connaissances:

les Français sont des Européens

les Antillais sont des Français qui ne sont pas des Européens

les Martiniquais sont des Antillais

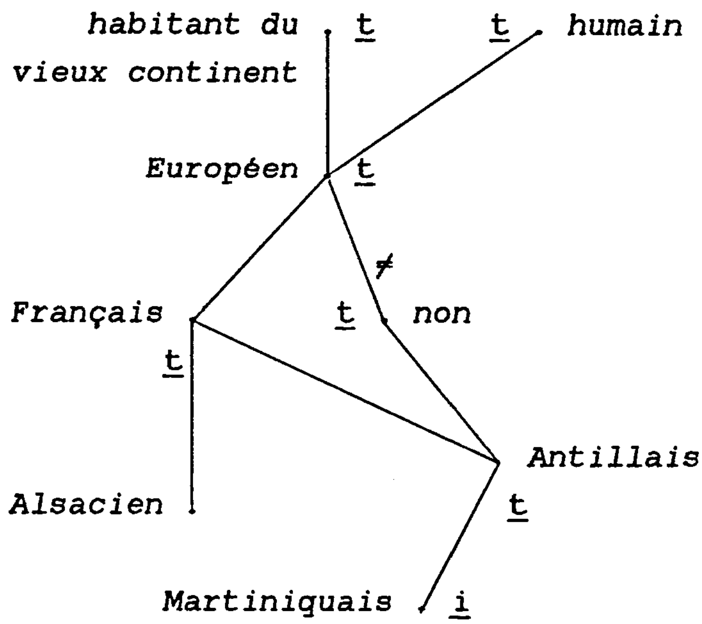
les Alsaciens sont des Français

les Européens sont des habitants du vieux continent

les Européens sont des humains

représentées dans le réseau de la figure 4.80.

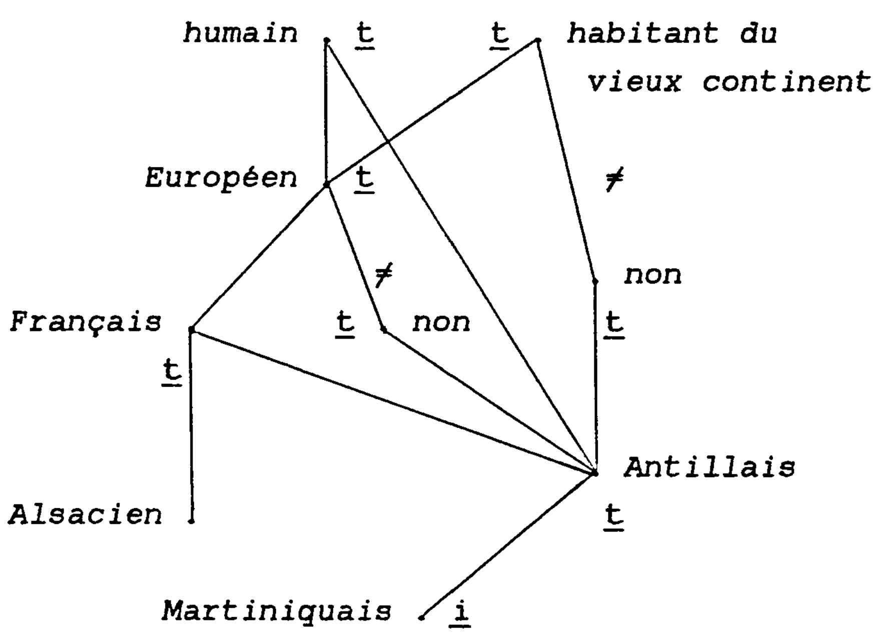
Il est clair qu'en appliquant la règle précédente on trouvera que les Martiniquais ne sont pas des Européens Mais cette règle ne nous permet pas d'interdire la déduction selon laquelle les Martiniquais seraient des habitants du vieux continent. C'est pourquoi, il faut donner une nouvelle règle:



---figure 4.80---

Règle: Si non x l'emporte sur x pour y, on ne peut pas dire si y hérite ou pas des ascendants de x.

On ne peut en effet conclure d'après le réseau précédent que les Martiniquais ne sont pas des habitants du vieux continent, parce



---figure 4.81---

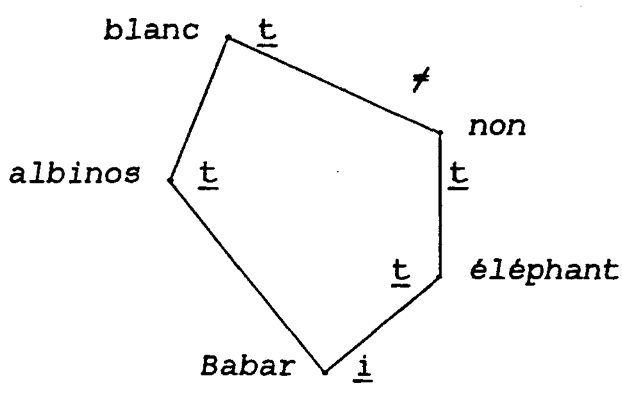
que de la même manière on pourrait conclure que les Martiniquais ne sont pas des humains. Si l'on veut faire les déductions correctes, il faut des liaisons plus directes, comme sur la figure 4.81.

Remarquons que l'application des deux règles précédentes ne nous met tout de même pas à l'abri des contradictions, comme pour l'exemple qui suit:

aucun éléphant n'est blanc

tous les albinos sont blancs

Babar est un éléphant albinos (figure 4.82)



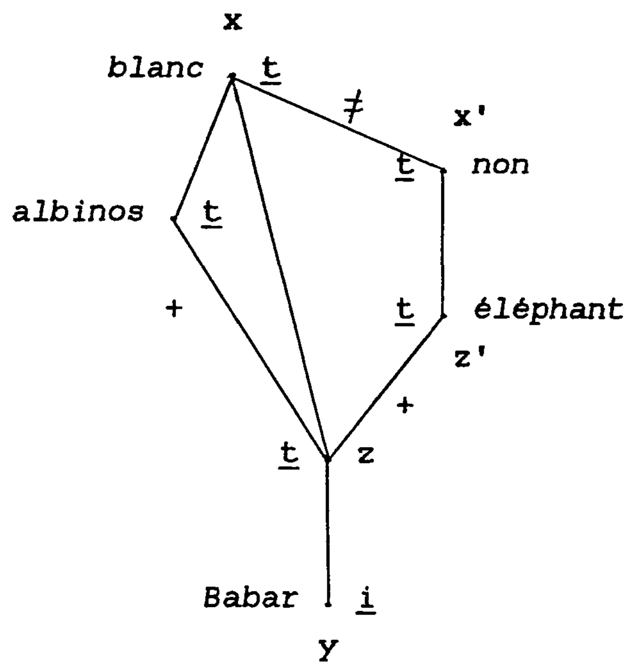
--------------------------figure 4.82--------------------------

Il n'y a aucune raison de privilégier la propriété *albinos* ou *éléphant* de *Babar*, et par conséquent de décider si *Babar* est *blanc* ou *non blanc*. Cela ne serait possible que si la première information était transformée en

aucun éléphant n'est blanc, *sauf ceux qui sont des albinos*

On aurait alors la représentation de la figure 4.83, et la propriété d'*être blanc* l'emporterait pour *Babar sur la propriété de ne pas être blanc*.

y hérite de x' par l'intermédiaire de z' et de x par l'intermédiaire de z qui est un descendant de z'.



---------------------------figure 4.83---------------------------

4.2.9 Le modèle mathématique

Finalement, la structure servant à représenter les connaissances sera un {p}-réseau (2.6.5) $H = \langle X,\delta,\xi,\rho\rangle$, l'application $\xi = \langle\xi^1,\xi^2\rangle$ vérifiant:

$$\xi^1 : X^1 \dashrightarrow S$$

où $S = \{\underline{t},\underline{i},\underline{p},\underline{j}\}$ est l'ensemble des sortes; X^1 n'étant pas égal à X uniquement dans le cas où l'on représente des connaissances incomplètes

$$\xi^2 : X^2 \dashrightarrow U$$

où U est un ensemble d'unités lexicales
ρ se réduisant à une seule application:

$$\rho : X \dashrightarrow \Xi^*$$

où $\Xi = \{\subset,+,\neq,|\}$ est un ensemble d'étiquettes d'arcs.

Remarquons que l'application ξ^2 est forcément définie pour tous les éléments de Ini(H), sans cela on ne voit pas quelle signification aurait le fait de descendre de tels sommets. De même cette application doit être définie sur tous les sommets x de H tels que $\xi^1(x) = \underline{p}$.

D'autre part, tout sommet terminal x de H devra être descendant d'au moins un sommet y tel que $\xi^2(y)$ soit définie (y pouvant être égal à x), à moins que l'on ne représente des connaissances incomplètes.

Les "sous-connaissances" seront représentées par un sous-réseau total H_1 de H (2.6.5) vérifiant les propriétés ci-dessus ainsi que la condition suivante: pour tout x appartenant à H_1, si $<y,x>$ est un arc de H étiqueté par $\neq$ ou par +, $<y,x>$ devra aussi être un arc de H_1. Sans cela il y aurait des sommets qui ne représenteraient plus rien dans la sous-structure, ou qui représenteraient une autre connaissance que dans la structure pleine.

4.2.10 <u>Problèmes linguistiques</u>:

Comme il a déjà été vu dans les exemples précédents, dans nos représentations les verbes sont placés aux sommets de graphes et ils ont pour successeurs leur sujet et leur complément (s'ils existent). Mais on ne sait pas toujours quelle étiquette lexicale doit être prise pour le verbe. Ainsi pour le verbe *avoir*.

On pourra représenter l'énoncé

Jean a un chien

par le graphe de la figure 4.84.

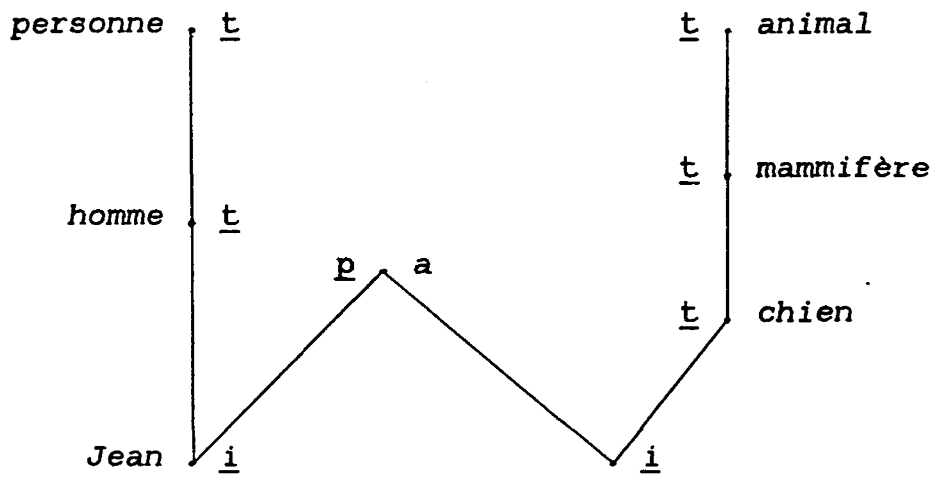
Ceci permettra d'effectuer certains raisonnemnts de manière immédiate. Par exemple de déduire du graphe que

Jean a un animal

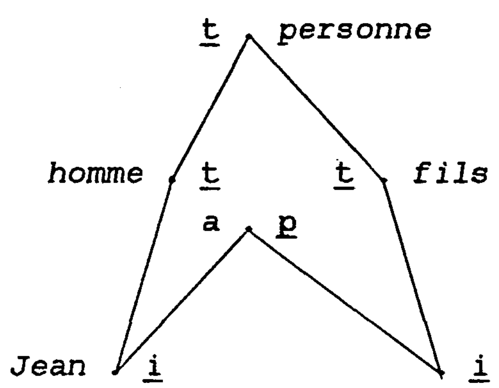
un homme a un mammifère

Supposons maintenant que l'on utilise un graphe semblable au précédent afin de représenter l'énoncé

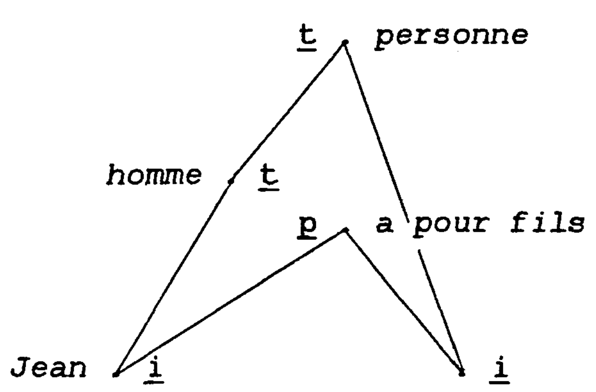
Jean a un fils (figure 4.85)



--------------------------<u>figure 4.84</u>--------------------------



--------------------------<u>figure 4.85</u>--------------------------



--------------------------<u>figure 4.86</u>--------------------------

De ce graphe on déduit que:

un homme a une personne

ce qui, bien évidemment, n'a aucune signification.

C'est pourquoi nous préfèrerons avoir la représentation de la figure 4.86, de laquelle on pourra déduire par exemple que:

Jean a pour fils une *personne*

ce qui est une déduction plus acceptable.

On voit pourquoi il est nécessaire de faire une étude linguistique sur le verbe *avoir*. Elle devrait permettre de déterminer dans quel cas on doit employer l'un ou l'autre type de représentation et, pourquoi pas, un éventuel troisième type.

4.3 L'utilisation de règles de réécriture

4.3.1 La déduction verbale

Afin de représenter les connaissances, il est fréquent de décomposer les verbes en "primitives sémantiques" ([Schank, 75], [Norman-Rumelhart, 75]). On parvient ainsi à une compréhension plus "profonde" des énoncés.

Il faut toutefois se rendre compte qu'en remplaçant un verbe par sa "signification", on risque toujours de faire des hypothèses arbitraires qui conduisent à des déductions illégitimes.

Selon [Norman-Rumelhart, 75] par exemple, on peut déduire de

x a acheté y

que *x possède y* (au moins pendant un certain intervalle de temps).

Une telle règle s'applique très bien à

Jean a acheté une voiture

mais elle produit de curieux résultats pour:

Jean a acheté l'arbitre

ou *Jean a acheté le silence de Pierre.*

C'est pourquoi nous allons apporter des restrictions aux décompositions verbales, en ne les autorisant qu'en fonction des arguments des verbes.

Dans le cas présent il faut que l'*acheteur* soit une *personne* et que l'*acheté* soit un *objet concret* qui ne soit pas une *personne*. On a alors à représenter la connaissance suivante:

Si une personne a acheté un objet concret qui n'est pas une *personne, alors la personne possède l'objet*

Une telle connaissance ne peut être représentée directement par un réseau. En effet, on a affaire à une implication portant sur des propositions alors que dans les réseaux les implications portent sur des objets typiques.

Soyons plus précis. Nous avons déjà donné des représentations d'implications qui portent sur des propositions comme:

si un homme aime Marie, alors il habite Marseille (4.2.4)

Toutefois, une telle connaissance peut être traduite en:

l'homme typique qui aime Marie est un *homme qui habite Marseille*

c'est-à-dire sous la forme d'une implication portant sur des objets typiques. Une telle traduction serait impossible pour la connaissance précédente. On pourrait bien dire que

la personne typique qui a acheté un objet concret qui n'est pas une personne est une personne qui possède un objet

mais alors on ne saurait pas que c'est du même *objet* qu'il s'agit. On perdrait une partie de l'information.

Plus généralement, il sera impossible de représenter par des réseaux toutes les implications dans lesquelles il y a une certaine forme de reprise anaphorique telle que deux objets liés par une proposition dans l'antécédent de l'implication le soient aussi dans le conséquent. C'est-à-dire les connaissances qui se traduiraient par des formules du calcul des prédicats telles que:

$$P(x,y) ==> Q(x,y)$$

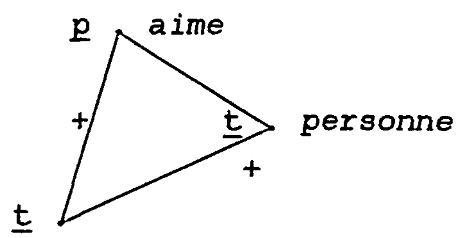
Par exemple:

si un homme habite une ville, alors il connaît cette ville

Toutefois, il est un cas particulier dans lequel il n'est pas nécessaire qu'il y ait deux propositions. Considérons en effet l'exemple suivant:

toute personne qui s'aime elle-même est narcissique

Il est impossible de représenter dans un réseau *la personne typique qui s'aime elle-même*. En effet, si on le fait selon la figure 4.87, on ne représente que *la personne qui aime toutes les personnes*.

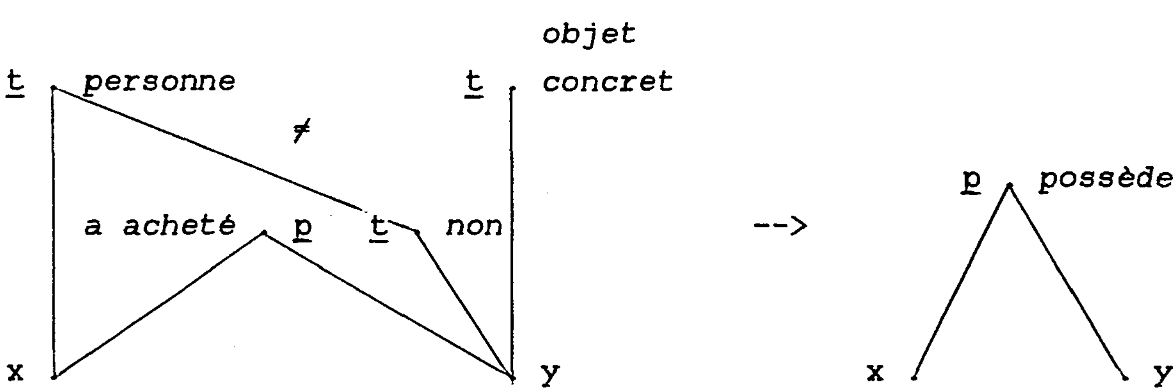


--------------------------figure 4.87--------------------------

On va donc employer, pour ces connaissances plus "complexes", des règles de réécriture qui seront pour nous des règles de réécriture de réseaux. Notons que, si l'on veut, on pourra représenter également par des règles de récriture les connaissances plus simples que nous savons déjà représenter.

4.3.2 Les règles

Reprenons le premier exemple du paragraphe précédent. On le représente par la règle schématisée en figure 4.88. Il s'agit d'une règle de réécriture de réseaux (chapitre 3).



--------------------------figure 4.88--------------------------

Remarquons tout d'abord que dans une telle règle, il n'y a ni éclatements ni fusions. On pourra donc la noter simplement $<H_1,H_2>$.

Notons que certains sommets de H_1 comme de H_2 ne sont pas sortés. Mais il est nécessaire que les sommets de H_2 qui ne sont pas sortés soient les correspondants de sommets de H_1. En effet, sans cela, le réseau construit après application de la règle contiendrait des sommets non sortés.

L'inverse n'est pas obligatoire: on peut avoir dans H_1 des sommets non sortés qui n'aient pas de correspondant dans H_2, comme on le voit sur l'exemple suivant (figure 4.89).

si une personne x a pour parent une personne y, et si y a pour père un homme z, alors x a pour grand-père z
(on note sur la figure certaines unités lexicales de manière abrégée)

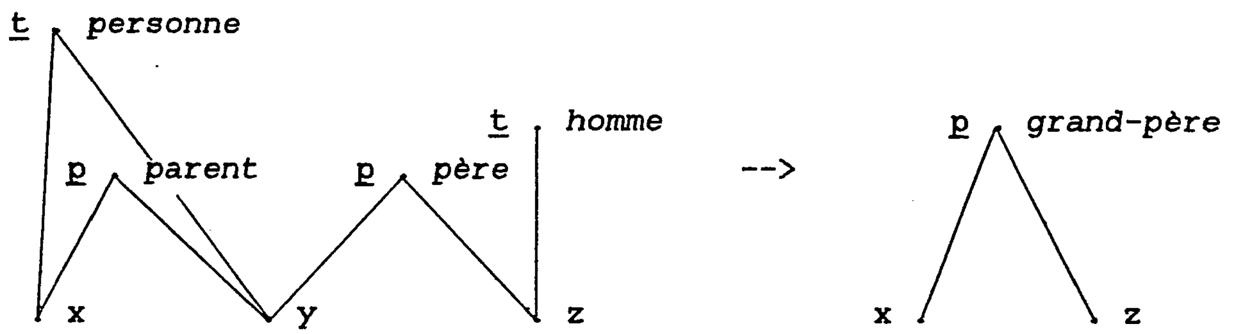


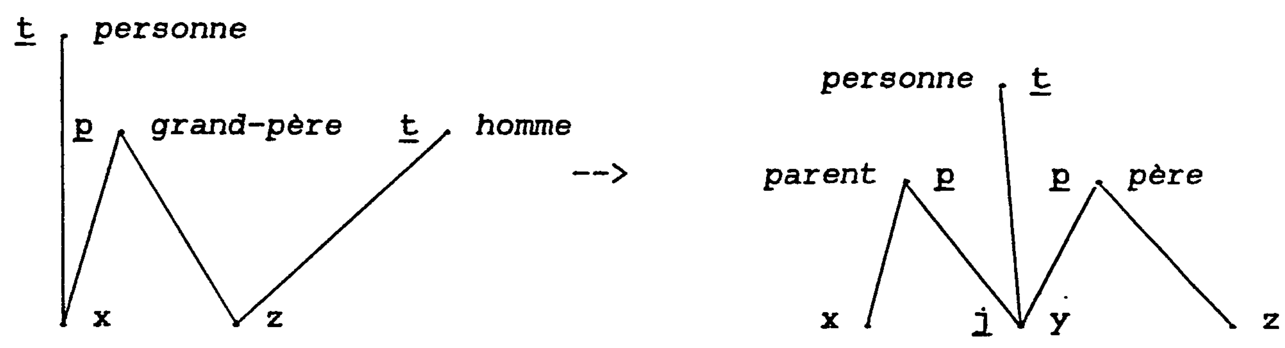
figure 4.89

Certaines règles pourront s'appliquer dans un sens ou dans l'autre. Ce n'est pas le cas de la règle de la figure 4.88, (on peut très bien possèder un objet sans jamais l'avoir acheté), mais c'est le cas pour la règle de la figure 4.89. Toutefois, il ne suffit pas d'inverser H_1 et H_2 pour obtenir la règle inverse. D'une part parce que les héritages de propriétés ne sont pas indiqués dans la partie droite d'une règle. La règle précédente inversée devrait s'exprimer:

si une personne x a pour grand-père un homme z, alors il existe une personne y telle que x a pour parent y et y a pour parent z
Cette fois l'héritage entre y et personne doit figurer dans la partie droite de la règle.

D'autre part, y doit être sorté dans H_1, sans cela on arriverai à une règle invalide. On donnera à y la sorte j, mais nous verrons que dans l'application de la règle cette sorte ne sera pas forcément conservée.

La règle inverse recherchée est en figure 4.90 et on donne



--------------------------figure 4.90--------------------------

la définition suivante:

Définition: Etant donnée une règle $<H_1,H_2>$, la **règle inverse** $<H_2',H_1'>$ est telle que:

H_2'est construit en rajoutant à H_2 tous les ascendants dans H_1 des sommets de sorte t de H_2 qui n'ont pas de sorte ainsi que tous les arcs de H_1 qui lient un sommet de sorte t et un sommet qui n'a pas de sorte;

H_1' est construit à partir de H_1 (1) en supprimant les arcs qui ont été rajoutés à H_2 pour former H_2' et les sommets typiques qui deviennent par cette opération terminaux, (2) en affectant la sorte j à tous les sommets de H_1 qui n'ont pas de sorte et qui ne sont pas dans H_2.

Définition: Une règle $<H_1,H_2>$ sera dite **réversible** si son inverse fait partie de l'ensemble des règles applicables.

4.3.3 L'application des règles

On dispose d'un réseau originel représentant des connaissances élémentaires, et d'un ensemble de règles applicables.

L'application d'une règle ne doit pas conduire à supprimer des informations du réseau originel. Le réseau de la partie droite de la règle s'ajoute par conséquent au réseau originel sans que la partie gauche soit supprimée. Il s'agit d'une application conservatrice de la règle (3.1.5).

Soit $<H_1,H_2>$ une telle règle. On cherche dans le réseau H représentatif des connaissances un sous-réseau qui soit image de

H_1 par un agrandissement (il s'agit d'une A-dérivation directe de réseaux au sens de 3.4.4). Et on rajoute à H les sommets et arcs de H_2 afin d'obtenir un réseau transformé. On remarque que les règles de réécriture sont libres (au sens de 3.2.3): il n'y a donc pas le risque d'introduire des circuits dans le réseau.

Exemple: On considère la règle (figure 4.91) représentative de la connaissance:

si une personne x a pour soeur une femme y, alors il existe une personne z telle que x et y aient pour parent z

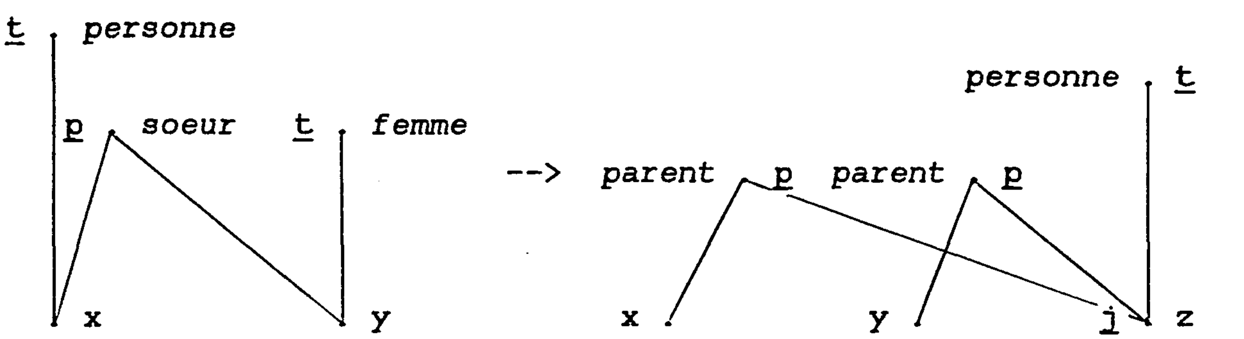


figure 4.91

On applique cette règle aux connaissances

Jean a pour soeur Marie

On obtient alors le réseau de la figure 4.92.

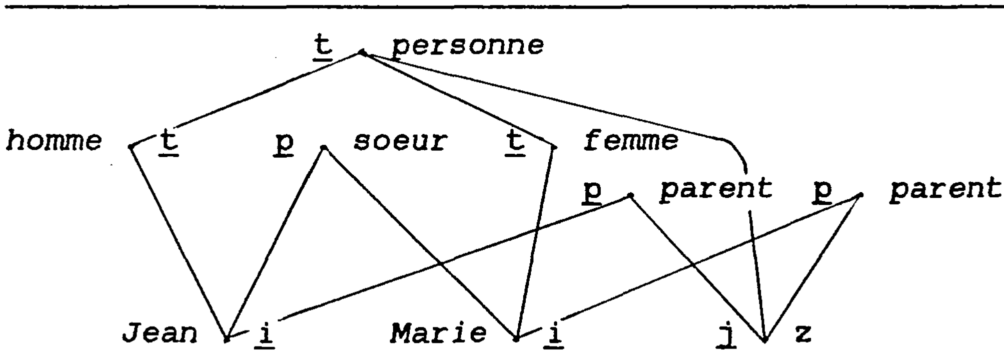


figure 4.92

Notons que si l'on cherche, en appliquant l'algorithme du paragraphe 4.2.5 de quels sommets quantifiés universellement dépend le sommet z de sorte j, on trouve un ensemble vide. On peut dans ces conditions remplacer la sorte j par i.

Ce remplacement ne sera pas possible si on applique la même règle de réécriture à la connaissance

chaque femme qui habite Trifouilly est la soeur de chaque homme qui habite Trifouilly

On obtient le réseau de la figure 4.93.

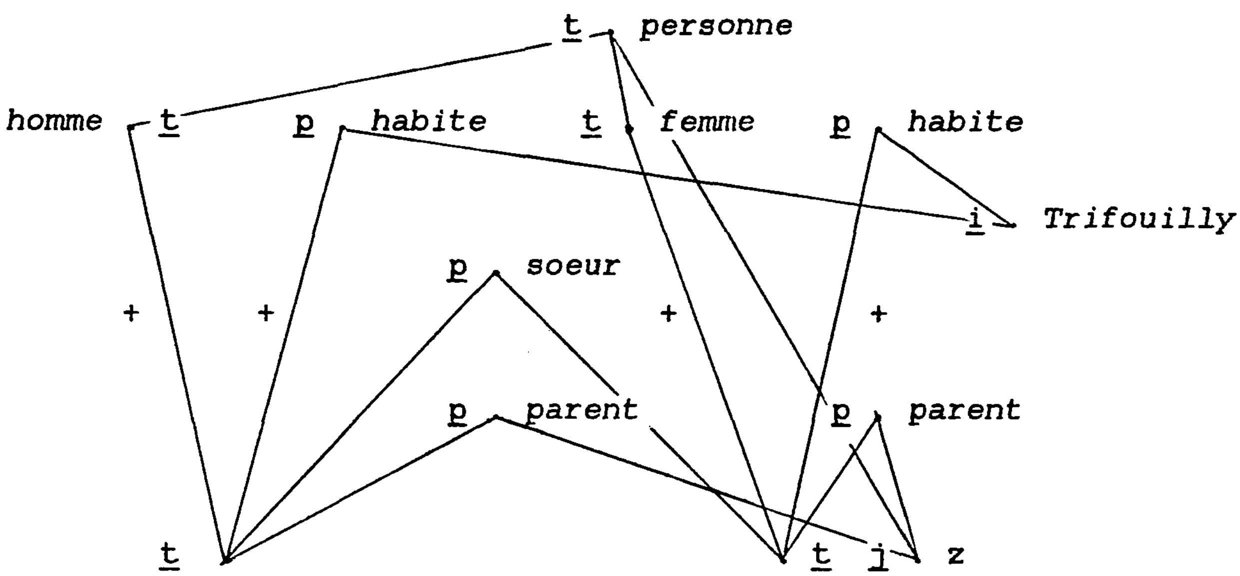


figure 4.93

Dans ce cas, le sommet z est un sommet qui dépend de 2 sommets typiques. Il doit donc garder la sorte j.

Quand on dispose d'un réseau H ainsi que d'un ensemble de règles de réécriture de réseaux, on a la possibilité (théorique) d'obtenir par dérivation toutes les connaissances possibles. Mais, ces connaissances ne sont pas en nombre fini. Il n'est donc pas possible d'effectuer toutes les dérivations. Et, de toutes façons, ce n'était pas le but recherché.

En réalité, on ne se servira des règles que lorsque l'on en aura besoin pour répondre à une question donnée, comme nous allons le voir dans le chapitre suivant.

5 L'interrogation

Le processus d'interrogation en langue naturelle peut, très grossièrement, être décrit de la manière suivante: un individu énonce une **question** et il attend de son interlocuteur (système informatique censé détenir certaines connaissances) une **réponse**, construite à partir de la question et des connaissances possédées par le système.

Disons clairement tout d'abord qu'il ne s'agit pas de prétendre traiter n'importe quelle question qui soit exprimable en une langue naturelle. Pour donner des exemples extrêmes, il va de soi que nous ne traiterons pas des questions telles que:

quelles sont les causes de la guerre de cent ans ?

comment fonctionne une voiture ?

ai-je intérêt à placer mon argent à la caisse d'épargne ou à acheter des actions Saint-Gobain ?

C'est ainsi que nous éliminons toutes les questions en *pourquoi* ou *comment* car la recherche de la réponse à de telles questions demande une subtilité de raisonnement trop grande.

Evidemment, si on a enregistré la connaissance

Jean a donné une gifle à Pierre parce que Pierre lui a pris son chewin-gum

on peut répondre simplement à la question

pourquoi Jean a-t-il donné une gifle à Pierre ?

Il reste que cette réponse pourra être contestée, certains préférant

parce qu'il était de mauvaise humeur

ou *parce que Pierre est moins fort que lui*

La manière dont les questions sont exprimées n'est pas le critère déterminant qui permet de dire si on peut y répondre ou pas. Tout dépend des données qui ont été enregistrées. Par exemple, on pourra plus probablement répondre à des questions telles que:

quelle est la capitale de la France ?

qu'à des questions telles que:

quelle est la racine carrée de 169 ?
parce qu'il paraît plus raisonnable d'enregistrer des données comme

la France a pour capitale Paris

que d'enregistrer toutes les connaissances de la forme:

169 a pour racine carrée 13

Notons que si on a enregistré la connaissance:

tout nombre réel a pour racine carrée un nombre réel

on obtient comme réponse à la question

169 a pour racine carrée un nombre réel

réponse qui risque de ne pas satisfaire l'interrogateur!

On ne traite donc pas ici les déductions qui font appel à une forme de raisonnement propre à un domaine donné, car on ne peut donner de solution générale à ce type de problèmes. L'interrogation reviendrait en ce cas à déclencher, en fonction de l'interprétation de questions, des actions qui n'auraient rien à voir avec la représentation des connaissances.

Nous limitons nos ambitions à l'extraction de connaissances enregistrées préalablement et aux déductions élémentaires sur ces connaissances. Il est clair que le type de connaissances que l'on pourra extraire et le type de déductions que l'on pourra effectuer dépendent du type de connaissances que l'on peut représenter et de la manière dont ces connaissances sont représentées.

On peut dire alors que le processus d'interrogation revient à extraire des connaissances possédées explicitement ou implicitement par un système. Dans ce contexte restrictif, une question peut être vue en réalité comme le condensé de deux questions. L'exemple le plus classique, énoncé notamment dans [Bonnet, 80], est fourni par la question

avez-vous l'heure ?

Celui qui la pose serait fort déçu de recevoir pour seule réponse oui. C'est parce que la question comporte une partie sous-entendue qui est:

si oui, dites moi quelle heure il est

De même la question:

qui a cassé le vase de Soissons ?

suppose qu'il a déjà été répondu à une question préliminaire:

quelqu'un a-t-il cassé le vase de Soissons ?

comme la question

quelle heure est-il ?

suppose que l'on sait que son interlocuteur possède une montre à l'heure.

Plus généralement, une question se décompose en ces deux parties:

1°/ *avez-vous les connaissances qui vous permettent de me répondre ?*

2°/ *si oui, donnez-moi ces connaissances.*

Remarquons toutefois qu'il est des questions qui se réduisent réellement à leur première partie, comme:

est-ce que Jean aime Marie ?

et qu'il en est d'autres que l'on peut considérer comme réduites à leur deuxième partie, par exemple:

qu'est-ce qu'un chien ?

bien que, dans ce dernier cas, on suppose que la personne à laquelle on s'adresse sait ce qu'est un *chien*.

Formellement, nous décrirons le processus d'interrogation en nous donnant une base de données textuelles ou une base de connaissances sous la forme d'une certaine structure et une question sous la forme d'une structure de même genre, mais "incomplète" (dans un sens qui est précisé dans les paragraphes qui suivent).

La première partie du processus d'interrogation consistera à chercher dans la structure représentative des données une (ou plusieurs) sous-structures à l'image de la question. On sait alors si on peut ou non fournir des réponses à la question. Dans le cas négatif, le processus s'arrête là.

Dans le cas positif, on peut passer à la deuxième partie qui consiste à "compléter" la question à l'aide d'éléments pris dans la base de données afin de construire une ou plusieurs réponses.

Par exemple, si on sait que

Jean aime Marie

et si on pose la question

qui aime Marie

cette question est décomposée en

quelqu'un aime-t-il Marie ?

question à laquelle il est répondu *oui* et en

si oui, qui ?

question à laquelle il est répondu Jean.

Toutefois, pour nous, la réponse à la première question ne se réduira pas à *oui* ou *non*. Nous considérons en effet que les données que nous enregistrons ne sont pas vraies ou fausses. Toutes les modalités sont possibles. Répondre à une question, cela revient alors à chercher dans les données une certaine structure, quelles que soient les conditions dans lesquelles cette structure est insérée.

Une telle conception risque de produire en certaines circonstances des réponses non attendues par celui qui pose la question. Par exemple la réponse

Paul croit que Jean aimait Marie en 1980

pour la question

est ce que Jean aime Marie ?

Mais nous pensons qu'il vaut mieux avoir trop de réponses que pas assez.

On aura la possibilité toutefois, si on le désire, d'éliminer certaines réponses dans le cas où l'on obtient des réponses contradictoires. Ceci nous permettra de traiter les exceptions aux règles générales.

Dans ce qui suit, nous donnons des modèles formels et algorithmiques de l'interrogation qui correspondent à nos modèles de représentation des données textuelles et des connaissances.

Pour chaque type de question, on donne un formalisme mathématique défini en termes d'équations (qui s'appuient sur des morphismes de structures) et une description algorithmique. Nous présentons deux algorithmes principaux dont l'un est sous la forme d'un parcours classique de graphe sans circuit, et l'autre sous la forme d'une propagation de marqueurs binaires, comme suggéré par Fahlman [Fahlman, 79].

5.1 Cas élémentaire

Dans ce paragraphe nous reprenons, avec une formulation légèrement différente, l'ébauche introduite dans [Cori, 82b].

Nous supposons disposer d'une base de données textuelles représentée sous la forme d'un Σ-GOSME H = $\langle X,\delta,\sigma,\omega\rangle$ comme en 4.1.4.

5.1.1 La représentation des questions

Une question sera représentée par un Σ-GOSME H_1 obtenu après transformation des questions de la forme interrogative à la forme déclarative. A moins que la seule réponse attendue soit *oui* ou *non*, le texte déclaratif comportera des places vides, notées "()". L'énoncé déclaratif sera représenté par un Σ-GOSME qui sera en général non complet car aux places vides seront associées des sommets de H_1 qui ne recevront pas d'étiquette lexicale.

Les Σ-GOSME H et H_1 formeront l'**équation d'interrogation.** Les places vides seront assimilées aux **inconnues** de l'équation.

Exemples: Nous prenons comme base de données le texte suivant:

Jean présente Pierre à Marie. Pierre salue respectueusement Marie

Ce texte est représenté par le Σ-GOSME H = $\langle X,\delta,\sigma,\omega\rangle$ présenté en figure 5.1.

Question n°1: *Qui Jean présente-t-il à Marie ?*

Sous forme déclarative on obtient:

Jean présente () à Marie

ce qui sera représenté par le Σ-GOSME $H_1 = \langle X_1,\delta_1,\sigma_1,\omega_1\rangle$ défini par le tableau de la figure 5.2

Question n°2: *Pierre salue-t-il Marie ?*

x	a	b	c	d	e	f	g
$\delta(x)$	λ	ac	λ	be	λ	ce	f
$\sigma(x)$	i	p	i	p	i	p	p
$\omega(x)$	*Jean*	*présente*	*Pierre*	*à*	*Marie*	*salue*	*respectu-eusement*

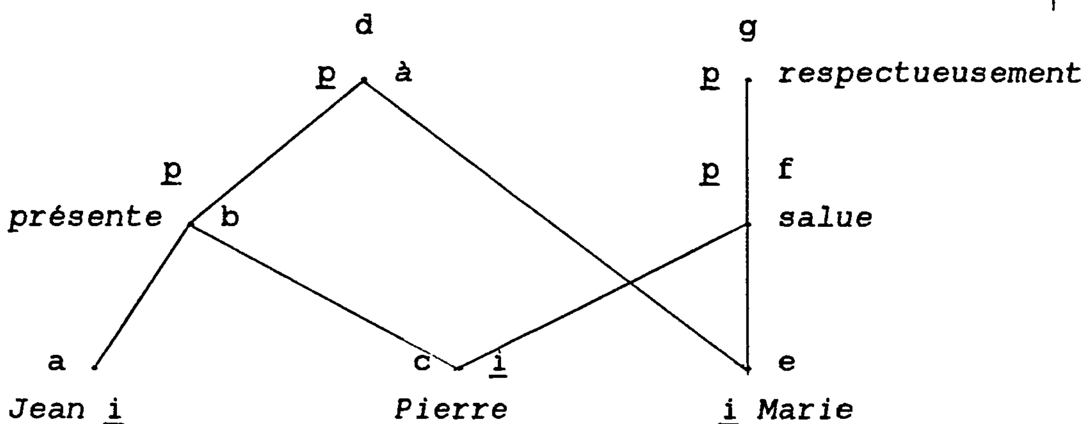


figure 5.1

x	a_1	b_1	c_1	d_1	e_1
$\delta_1(x)$	λ	a_1c_1	λ	b_1e_1	λ
$\sigma_1(x)$	i	p	i	p	i
$\omega_1(x)$	*Jean*	*présente*	*non défini*	*à*	*Marie*

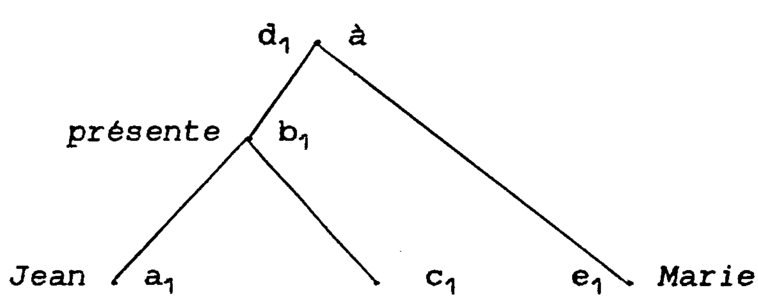


figure 5.2

x	a	b	c	d	e	f	g
$\delta(x)$	λ	ac	λ	be	λ	ce	f
$\sigma(x)$	i	p	i	p	i	p	p
$\omega(x)$	*Jean*	*présente*	*Pierre*	*à*	*Marie*	*salue*	*respectueusement*

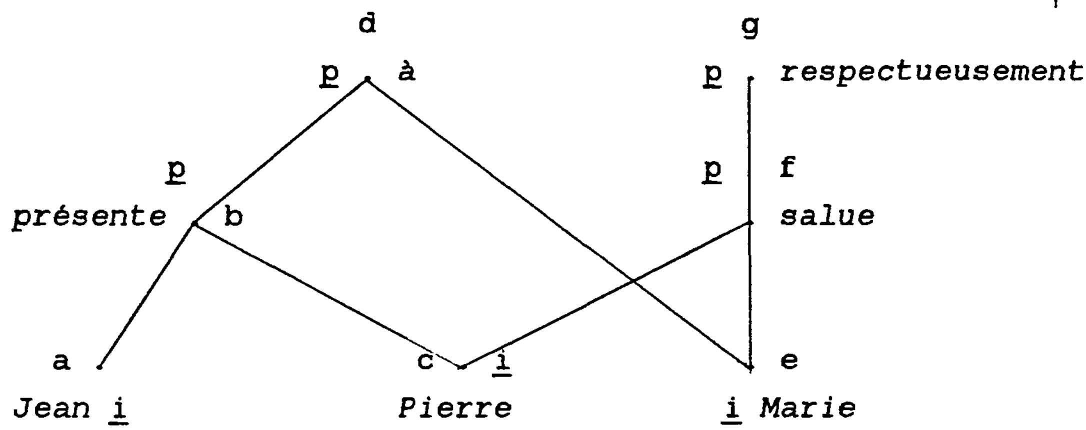


figure 5.1

x	a_1	b_1	c_1	d_1	e_1
$\delta_1(x)$	λ	a_1c_1	λ	b_1e_1	λ
$\sigma_1(x)$	i	p	i	p	i
$\omega_1(x)$	*Jean*	*présente*	*non défini*	*à*	*Marie*

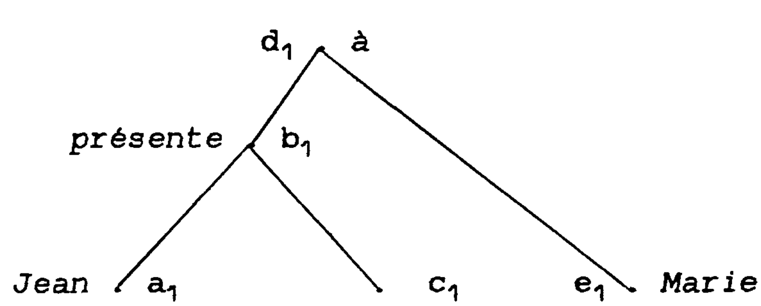


figure 5.2

La question est transformée en:

Pierre salue Marie.

Il n'y a pas de place vide, le Σ-GOSME représentatif (figure 5.3) est complet, l'équation d'interrogation est par conséquent sans inconnue.

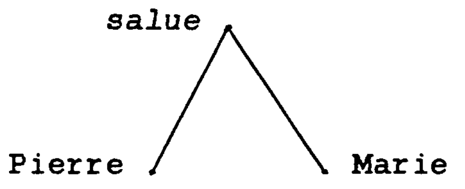


figure 5.3

x	a_1	b_1	c_1	d_1	e_1	e'_1	f_1
$\delta_1(x)$	λ	a_1c_1	λ	b_1e_1	λ	λ	e'_1
$\sigma_1(x)$	<u>i</u>	<u>p</u>	<u>i</u>	<u>p</u>	<u>i</u>	<u>i</u>	<u>p</u>
$\omega_1(x)$	*Jean*	*présente*	non défini	à	*Marie*	non défini	*salue*

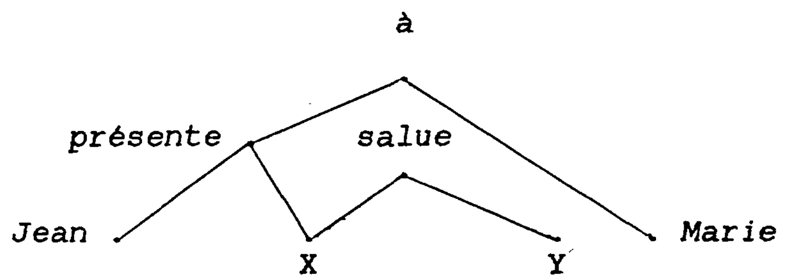


figure 5.4

<u>Question n°3</u>: On peut représenter des questions plus complexes que les deux précédentes, et notamment des questions qui ne s'expriment pas du tout simplement dans une langue naturelle.

Ainsi le Σ-GOSME de la figure 5.4 représente une question que l'on peut exprimer par:

Qui (qui salue qui ?) Jean présente-t-il à Marie ?

ou par: *Y a-t-il un individu X qui salue un individu Y et que Jean présente à Marie ?*

Remarquons que nous ne traitons pas pour l'instant des questions du type:

Que peut-on dire de Jean ?

Quel lien y a-t-il entre Jean et Marie ?
Que fait Jean à Pierre ?

5.1.2 <u>La recherche des réponses aux questions</u>

La réponse à la question sera forcément une sous-structure de H qui représente les données textuelles enregistrées. Comme cette réponse doit également être un texte, il faut qu'elle soit encore représentée par un Σ-GOSME. Ce texte devant être un "sous-texte" de H, il faut que la réponse soit représentée par un sous-Σ-GOSME total de H. Soit $H_1' = \langle X_1', \delta_1', \sigma_1', \omega_1' \rangle$ ce Σ-GOSME.

H_1' devra par ailleurs être "à l'image" de la question H_1. Plus précisément, il faut qu'à chaque syntagme de la question soit associé un syntagme de la réponse, et que les syntagmes soient combinés de la même manière dans la réponse que dans la question. A chaque arbre élémentaire de la question il faut par conséquent que soit associé un arbre élémentaire de la réponse, sauf si l'arbre élémentaire de la question est réduit à un point et ne reçoit pas d'étiquette lexicale. Autrement dit, si $\psi : X_1 \dashrightarrow X_1'$ est l'application qui assure la correspondance entre la question et la réponse:

$$\forall x \in X_1^2 \quad \psi\delta_1(x) = \delta_1'\psi(x)$$

ce qui est exactement la définition des Q-morphismes de graphes ordonnés (2.4.3).

De plus, il faut que les sommets de H_1' reçoivent la même sorte que les sommets de H_1 dont ils sont images, ainsi que la même étiquette lexicale lorsque le sommet de H_1 en a une. Autrement dit:

$$\forall x \in X_1 \quad \psi\sigma_1(x) = \sigma_1'\psi(x)$$

$\forall\ x \in X_1^2 \quad \psi\omega_1(x) = \omega_1'\psi(x)$

ce qui est la définition même d'un Q-morphisme de graphe multi-étiqueté (2.5.4).

Un tel morphisme sera, par définition, un Q-morphisme de Σ-GOSME.

Comme l'injection canonique de X_1' dans X détermine un morphisme de graphes entre H_1' et H, on peut en déduire que l'application ψ de X_1 dans X détermine un Q-morphisme de Σ-GOSME.

<u>Exemple</u>: La réponse à la question n° 3 (paragraphe précédent) est donnée par le tableau de la figure 5.5. Remarquons que l'application ψ n'est pas injective.

x	a_1	b_1	c_1	d_1	e_1	e_1'	f_1
$\psi(x)$	a	b	c	d	e	e	f
$\omega\psi(x)$	*Jean*	*présente*	*Pierre*	*à*	*Marie*	*Marie*	*salue*

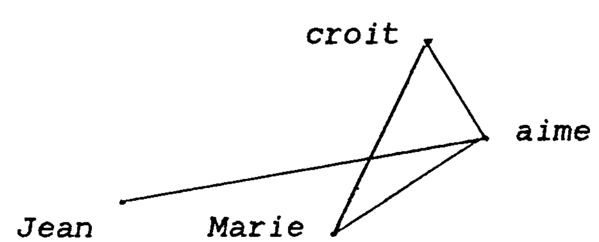
---------------------------<u>figure 5.5</u>---------------------------

La dernière ligne de ce tableau permet de voir par quelles variables sont instanciées les inconnues.

Soit maintenant le texte

Marie croit que Jean l'aime

représenté par le Σ-GOSME de la figure 5.6.



---------------------------<u>figure 5.6</u>---------------------------

Pour répondre à la question

Que croit Marie ?

on ne peut se contenter de déterminer l'application ψ. Cette application permettrait en effet d'instancier le sommet inconnu par l'étiquette lexicale *aime*, mais pas de trouver *qui aime qui*. Un sous-texte de H ne peut être représenté que par un sous-graphe total de H. Pour répondre complètement à la question, il faudra obtenir également tous les sommets de H qui descendent d'un sommet de $\psi(X_1)$. C'est-à-dire que la réponse sera donnée par le sous-GOSME total de H engendré par $\psi(X_1)$ (2.3.5).

Si l'on se donne enfin le texte

Henri, qui sait que Jean aime Marie, dit que Marie pense que Jean l'aime

représenté par le graphe de la figure 5.7, et la question

Que pense Marie ?

on ne peut se contenter de déterminer ψ et le sous-GOSME total de H engendré par $\psi(X_1)$.

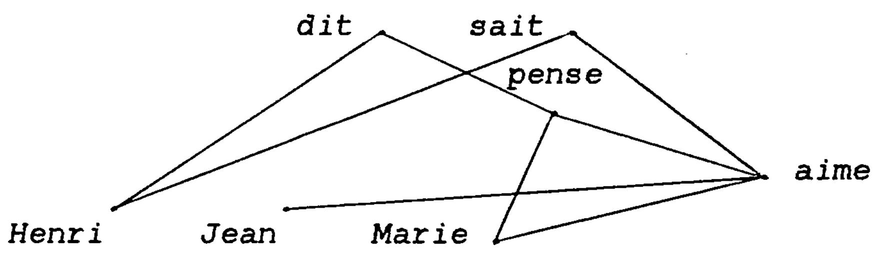


figure 5.7

Il faut de plus déterminer les conditions d'énonciation dans lesquelles s'insère cette première réponse. Cela sera effectué en construisant le plus petit sous-graphe total de H contenant les sommets initiaux du graphe construit à l'aide de l'application f. Plus précisément la réponse sera constituée par le sous-GOSME total de H englobant le sous-graphe de H engendré par $\psi(X_1)$ (2.3.5).

Dans le cas de notre exemple, la réponse sera le texte:

Henri dit que Marie pense que Jean l'aime.

Finalement, à tout Q-morphisme ψ de H_1 dans H sera associée une solution de l'équation. La solution sera le sous-Σ-GOSME H_1' de H englobant $\psi(X_1)$.

5.1.3 Algorithmes de recherche et construction des solutions

Nous donnons une description plus précise de l'algorithme ébauché dans [Cori, 84] qui est enrichi dans la présente version.

Les données de l'algorithme sont les Σ-GOSME H et H_1.

On peut supposer que le graphe sous-jacent à H_1 est connexe car, si ce n'était pas le cas, on pourrait le décomposer en ses composantes connexes; une question complexe serait alors décomposée en plusieurs questions simples et indépendantes. Il n'y aurait qu'à répéter autant de fois qu'il y aurait de composantes connexes le processus d'interrogation.

Définition: Etant donnés deux Σ-GOSME $H = \langle X,\delta,\sigma,\omega\rangle$ et $H_1 = \langle X_1,\delta_1,\sigma_1,\omega_1\rangle$ un élément y de X est un **Q-correspondant possible** d'un élément x de X_1 si et seulement si:

(i) $x \in X_1^2 \implies \begin{cases} \sigma_1(x) = \sigma(y) \\ \omega_1(x) = \omega(y) \\ |\delta_1(x)| = |\delta(x)| \end{cases}$

(ii) $x \in X_1 \setminus X_1^2 \implies \sigma_1(x) = \sigma(y)$

Si y est un Q-correspondant possible de x, on notera Q(x,y).

Propriété: Si $H_1 = \langle X_1,\delta_1,\sigma_1,\omega_1\rangle$ est un graphe connexe et si ψ est un Q-morphisme de Σ-GOSME de H_1 dans $H = \langle X,\delta,\sigma,\omega\rangle$ alors il existe suite $\langle Z_0,\ldots,Z_m\rangle$ de sous-ensembles de X_1, une suite d'applications $\langle \Psi_0,\ldots,\Psi_m\rangle$ avec

$\Psi_k : Z_k \dashrightarrow (X)$

et une suite $\langle x_1,\ldots,x_m\rangle$ de sommets de H_1 tous distincts, ces trois suites vérifiant les conditions suivantes:

(1) $Z_0 = \{x_1\}$

(2) $Z_m = \emptyset$

(3) $\forall\ k \in [m]\ x_k \in Z_{k-1}$

(4) $\forall\ k \in [m]\ Z_k = (Z_{k-1}\setminus\{x_k\}) \cup [\delta_1(x_k) \cup \delta_1^{-1}(x_k)]\setminus Y_{k-1}$

(On note $Y_{k-1} = \{x_1,\ldots,x_{k-1}\}$)

(5) $\Psi_0(x_1) = \{y \in X\ ;\ Q(x_1,y)\}$

(6) $\forall\ k \in [m]\ \forall\ z \in Z_k\ \forall\ x \in \Psi_k(z)\quad Q(z,x)$

(7) $\forall\ k \in [m]\quad \psi(x_k) \in \Psi_{k-1}(x_k)$

(8) Pour tout $k \in [m]$, pour tout $i \in [|\delta_1(x_k)|]$,

(8.1) $\delta_1^{(i)}(x_k) \in Y_{k-1} \implies \delta^{(i)}\psi(x_k) = \psi\delta_1^{(i)}(x_k)$

(8.2) $\delta_1^{(i)}(x_k) \in Z_{k-1} \implies \delta^{(i)}\psi(x_k) \in \Psi_{k-1}\delta_1^{(i)}(x_k)$

et $\Psi_k\delta_1^{(i)(x_k)} = \{\delta_{(i)}\psi(x_k)\}$

(8.3) $\delta_{1(i)}{}^{(x_k)} \notin Y_{k-1} \cup Z_{k-1} \implies \delta_{1(i)}{}^{(x_k)} \in Z_k$ et

$Q(\delta_{1(i)}{}^{(x_k)},\delta_{(i)}\psi(x_k))$ et $\Psi_k\delta_{1(i)}(x_k) = \{\delta^{(i)}\psi(x_k)\}$

(9) Pour tout $k \in [m]$, pour tout $u \in \delta_1^{-1}(x_k)$

(9.1) $u \in Y_{k-1} \implies \psi(u) \in \delta^{-1}\psi(x_k)$

(9.2) $u \in Z_{k-1} \implies$

$\Psi_{k-1}(u) \cap \delta^{-1}\psi(x_k) \neq \emptyset$ et $\Psi_k(u) = \Psi_{k-1}(u) \cap \delta^{-1}\psi(x_k)$

(9.3) $u \notin Y_{k-1} \cup Z_{k-1} \implies$

$u \in Z_k$ et $\Psi_k(u) = \{z \in \delta^{-1}\psi(x_k);\ Q(u,z)\}$

(10) $\forall\ k \in [m]\ \forall\ z \in Z_k\setminus[\delta_1(x_k) \cup \delta_1^{-1}(x_k)]\ \Psi_k(z) = \Psi_{k-1}(z)$

Remarquons que les conditions (6) et (7) entraînent:

$$\forall\ k \in [m]\quad \delta_1(x_k) \neq \lambda \implies |\delta_1(x_k)| = |\delta\psi(x_k)|$$

indépendamment du fait que ψ est un Q-morphisme, et donc permettent d'écrire la condition (8).

<u>Preuve</u>: H_1 étant un graphe fini connexe, il existe une suite, $\langle x_1,\ldots,x_m\rangle$ formée de tous les éléments de X_1 et où, pour tout i appartenant à [m],

$$x_i \in \delta_1(x_{i-1}) \quad \text{ou} \quad x_i \in \delta_1^{-1}(x_{i-1})$$

On pose:

$$Z_0 = \{x_0\}$$

et, pour tout k appartenant à [m],

$$Z_k = [\delta_1(Y_k) \cup \delta_1^{-1}(Y_k)]\setminus Y_k$$

$\forall\ z \in Z_k \cap \delta_1(Y_k)\quad \Psi_k(z) = \{\psi(z)\}$

$\forall\ z \in Z_k\setminus\delta_1(Y_k)\quad \Psi_k(z) =$

$$\{x \in X;\ Q(z,x) \text{ et } \forall\ y \in \delta_1(z) \cap Y_k\ x \in \delta^{-1}\psi(y)\}$$

La condition (1) est alors vérifiée de manière évidente et, comme $Y_m = X_1$, $Z_m = \emptyset$ (condition 2).

Comme x_k appartient à $\delta_1(x_{k-1})$ ou à $\delta_1^{-1}(x_{k-1})$ et pas à Y_{k-1}, il est clair que x_k appartient à Z_{k-1} (condition 3).

La condition (4) est vérifiée pour k = 1. Dans le cas où k est supérieur à 1, on peut écrire:

$Z_k = [\delta_1(Y_{k-1}) \cup \delta_1^{-1}(Y_{k-1}) \cup \delta_1(x_k) \cup \delta_1^{-1}(x_k)]\backslash Y_k$

ou: $Z_k = [\delta_1(Y_{k-1}) \cup \delta_1^{-1}(Y_{k-1})]\backslash Y_k \cup [\delta_1(x_k) \cup \delta_1^{-1}(x_k)]\backslash Y_k$

et, le graphe H_1 étant sans circuit, x_k ne peut appartenir ni à $\delta_1(x_k)$ ni à $\delta_1^{-1}(x_k)$, ce qui entraîne:

$Z_k = Z_{k-1} \cup [\delta_1(x_k) \cup \delta_1^{-1}(x_k)]\backslash Y_{k-1}$

Comme $\delta_1(Y_0) = \emptyset$, on a bien:

$\Psi_0(x_1) = \{y \in X;\ Q(x,y)\}$ (condition 5).

La condition (6) est vérifiée de manière évidente, de par la définition même de Ψ_k.

Afin de démontrer la condition (7) et la condition (8) nous allons démontrer d'abord:

(11) $\forall\ k \in [m]\quad \forall\ z \in Z_{k-1}\quad \psi(z) \in \Psi_{k-1}(z)$

Si k = 1, comme ψ est un Q-morphisme, on a bien:

$\psi(x_1) \in \Psi_0(x_1)$

Si k est supérieur à 1, deux cas se présentent:

- si $x_k \in Z_{k-1} \cap \delta_1(Y_{k-1})$ $\psi(x_k) \in \Psi_{k-1}(x_k)$;
- sinon, supposons que $\psi(x_k)$ n'appartient pas à $\Psi_{k-1}(x_k)$; cela entraîne:

$\exists\ y \in \delta_1^{-1}(x_k) \cap Y_{k-1}\quad \psi(x_k) \notin \delta\psi(y)$

ce qui est contradictoire avec le fait que ψ est un Q-morphisme.

La condition (3) et la condition (11) entraînent immédiatement la condition (7).

La condition (8.1) est une conséquence immédiate du fait que ψ est un Q-morphisme de GO.

De la condition (11) on déduit que, si $\delta_1^{(i)}(x_k)$ appartient à

Z_{k-1}, on a:

$$\psi\delta_1^{(i)}(x_k) \in \Psi_{k-1}\delta_1^{(i)}(x_k)$$

et, comme ψ est un Q-morphisme de GO,

$$\delta^{(i)}\psi(x_k) \in \Psi_{k-1}\delta_1^{(i)}(x_k)$$

ce qui démontre la condition (8.2).

Si $\delta_1^{(i)}(x_k)$ n'appartient ni à Z_{k-1} ni à Y_{k-1}, il doit évidemment appartenir à Z_k et vérifier les propriétés de la condition (8.3).

La condition (9.1) est vérifiée de manière évidente puisque ψ est un Q-morphisme.

Soit u appartenant à Z_{k-1}; d'après (11) $\psi(u) \in \Psi_{k-1}(u)$.

Or, $\psi(u) \in \delta^{-1}\psi(x_k)$. Donc:

$$\delta^{-1}\psi(x_k) \cap \Psi_{k-1}(u) \neq \emptyset$$

Si u appartient à $\delta(Y_k)$, il appartient forcément à $\delta(Y_{k-1})$ et

$$\Psi_{k-1}(u) = \{\psi(u)\}$$

Et comme $\psi(u)$ appartient à $\delta^{-1}\psi(x_k)$, on a bien:

$$\Psi_k\delta_1^{(i)}(x_k) = \{\delta^{(i)}\psi(x_k)\}$$

Si u n'appartient pas à $\delta(Y_k)$, il n'appartient pas non plus à $\delta(Y_{k-1})$ et donc appartient à $\delta^{-1}(Y_{k-1})$.

On en déduit:

$$\Psi_k(u) = \{x \in X;\ Q(u,x) \text{ et } \forall\, y \in \delta_1(u) \cap Y_k\ x \in \delta^{-1}\psi(y)\}$$

$$\Psi_k(u) = \{x \in X;\ Q(u,x) \text{ et } \forall\, y \in \delta_1(u) \cap Y_{k-1}\ x \in \delta^{-1}\psi(y) \text{ et } x \in \delta^{-1}\psi(x_k)\}$$

$$\Psi_k(u) = \Psi_{k-1}(u) \cap \delta^{-1}\psi(x_k)\}$$

ce qui démontre que la condition (9.2) est vérifiée.

Si u n'appartient ni à Z_{k-1} ni à Y_{k-1}, il appartient évidemment à Z_k et vérifie

$$\delta_1(u) \cap Y_k = \{x_k\}$$

ce qui entraîne

$$\Psi_k(u) = \{x \in X;\ Q(u,x) \text{ et } x \in \delta^{-1}\psi(x_k)\}$$

ce qui démontre la condition (9.3).

Soit $z \in Z_k\setminus[\delta_1(x_k) \cup \delta_1^{-1}(x_k)]$;

- si $z \in \delta_1(Y_k)$ alors $z \in \delta_1(Y_{k-1})$ et

$$\Psi_k(z) = \Psi_{k-1}(z) = \{\psi(z)\};$$

- sinon $z \in \delta_1^{-1}(Y_{k-1})$ et

$\Psi_k(z) = \{x \in X;\ Q(z,x)$ et $\forall\ y \in \delta_1(z) \cap Y_k \quad x \in \delta^{-1}\psi(y)\}$
Comme x_k n'appartient pas à $\delta_1(z)$, on a bien:
$\Psi_k(z) = \Psi_{k-1}(z)$.

Réciproque: Si $H_1 = <X_1,\delta_1,\sigma_1,\omega_1>$ est un Σ-GOSME connexe, si $H = <X,\delta,\sigma,\omega>$ est un Σ-GOSME, si ψ est une application de X_1 dans X telle qu'il existe une suite $<Z_0,\ldots,Z_m>$ de sous-ensembles de X_1, une suite d'applications $<\Psi_0,\ldots,\Psi_m>$ avec

$\Psi_k : Z_k \dashrightarrow (X)$

et une suite $<x_1,\ldots,x_m>$ de sommets de H_1 tous distincts, ces trois suites vérifiant les conditions (1) à (10) énoncées précédemment, alors ψ est un Q-morphisme de Σ-GOSME.

Afin de démontrer cette réciproque, on commence par démontrer deux lemmes.

Lemme 1: Pour $k \geqslant 1$, on a:

$$Z_k = [\delta_1(Y_k) \cup \delta_1^{-1}(Y_k)]\setminus Y_k$$

Preuve: On a $Y_1 = \{x_1\}$ et donc, d'après la condition (4),

$$Z_1 = [\delta_1(x_1) \cup \delta_1^{-1}(x_1)]\setminus Y_1$$

Supposons la propriété vraie pour k-1. La conditions (4) implique alors:

$$Z_k = [\delta_1(Y_{k-1}) \cup \delta_1^{-1}(Y_{k-1})]\setminus(Y_{k-1} \cup \{x_k\}) \cup [\delta_1(x_k) \cup \delta_1^{-1}(x_k)]\setminus Y_{k-1}$$

ce qui démontre la propriété pour k.

Lemme 2: $Y_m = X_1$

Preuve: D'après le lemme précédent,

$$[\delta_1(Y_m) \cup \delta_1^{-1}(Y_m)]\setminus Y_m = \emptyset$$

ce qui est une manière d'écrire que Y_m est une composante connexe du graphe dont il fait partie. Comme ce graphe est connexe, cela démontre le lemme.

Preuve de la réciproque: Soit x_k un élément quelconque de X_1 et soit y_k son image par ψ.

D'après les conditions (7) et (6) on a:

$$Q(x_k, y_k)$$

ce qui peut encore s'écrire:

(i) $x_k \in X_1^2 \implies \begin{cases} \sigma_1(x_k) = \sigma(y_k) \\ \omega_1(x_k) = \omega(y_k) \\ |\delta_1(x_k)| = |\delta(x_k)| \end{cases}$

(ii) $x_k \in X_1 \setminus X_1^2 \implies \sigma_1(x_k) = \sigma(y_k)$

Soient $\delta_1(x_k) = u_1 \ldots u_p$

et $\delta(y_k) = v_1 \ldots v_p$.

Pour un u_i donné, deux cas se présentent:

- si u_i appartient à Y_{k-1}, $\psi(u_i) = v_i$ d'après la condition (8.1);
- sinon, d'après la condition (8.2) ou la condition (8.3), on a: $\Psi_k(u_i) = \{v_i\}$, et comme $u_i = x_h$ avec $h>k$, on a, d'après les conditions (7) et (10) $\psi(u_i) = v_i$: c'est la seule valeur que peut prendre $\psi(u_i)$.

L'algorithme de recherche des réponses à une question est déduit de cette réciproque. Il consiste à partir d'un sommet x_1 de X_1 en posant $Z_0 := \{x_1\}$ et

$$\Psi_0(x_1) := \{y \in X;\ Q(x_1, y)\}$$

puis, tant que c'est possible, en commençant avec k = 1,

- à prendre $\psi(x_k)$ dans $\Psi_{k-1}(x_k)$ (selon la condition 7); il faut vérifier que la valeur prise pour $\psi(x_k)$ respecte bien les conditions (8) et (9);
- à construire alors Z_k selon les indications de la condition (4);
- à construire Ψ_k selon les indications des conditions (6), (8), (9) et (10);
- à prendre pour k:=k+1 un nouvel x_k dans Z_{k-1}.

Si l'on obtient un ensemble Z_k vide, on a trouvé une solution à l'équation. La réciproque que l'on vient de démontrer en fournit une preuve.

Toutefois, quand on prend une valeur pour $\psi(x_k)$ dans $\Psi_{k-1}(x_k)$, il se peut très bien qu'on laisse de côtés certaines possibilités. C'est pourquoi, en cas d'échec ou quand on trouve une solution, il faut pouvoir effectuer des retours en arrière afin d'examiner toutes les valeurs possibles de $\psi(x_k)$. Il faut donc mettre en pile les valeurs de Ψ_{k-1} qui n'ont pas été examinées et les examiner après retour en arrière.

Après avoir examiné toutes ces valeurs on sera sûr d'avoir trouvé toutes les solutions de l'équation: c'est la propriété directe qui en constitue la preuve.

Nous allons donner maintenant une description plus détaillée de l'algorithme.

H_1 est un Σ-GOSME en général non complet mais qui a toujours un sommet terminal x_1 auquel est associée une étiquette lexicale $\omega(x_1)$. On suppose en effet ne pas avoir à traiter des questions du genre: *Qui présente qui à qui?*

Le sommet x_1 servira de point de départ au parcours de H_1. Ce choix est effectué afin de réduire le nombre de possibilités au départ. Les correspondants possibles d'un sommet de H_1 sans étiquette lexicale seraient beaucoup trop nombreux et même les sommets représentant une proposition possédant la même étiquette lexicale peuvent se retrouver en grand nombre dans un Σ-GOSME représentant des données textuelles.

On cherche $\psi(x_1)$ dans les sommets terminaux de H. En effet, l'image d'un sommet terminal de H_1 auquel est associée une étiquette lexicale ne peut être qu'un sommet terminal de H. On trouvera soit 0 solution (cas où par exemple aucune personne ne s'appelle *Jean* alors que la question porte sur *Jean*), soit une solution, soit plusieurs solutions (plus d'une personne s'appelle *Jean*). $\psi(x_1)$ pourra donc être choisi dans une liste $(d_1 \ldots d_q)$.

On se donne:

- une pile P qui permettra de connaître à un moment donné les sommets de H_1 à parcourir, ainsi que les correspondants possibles de ces sommets dans H c'est-à-dire Z_k et Ψ_k;

- une pile Q qui permettra de stocker les solutions provisoires et définitives, c'est-à-dire de construire les Q-morphismes ψ pour les sommets $x_1,\ldots,x_k$.

P est une pile dont les éléments sont des piles α, chaque pile α étant composée de listes de la forme:

$(x\ a_1\ \ldots\ a_p)$

où x est un sommet de H_1 et les a_i sont des sommets de H, qui sont les correspondants possibles de x. $a_1,\ldots,a_p$ sont les éléments de $\Psi_k(x)$, ou plutôt les éléments de $\Psi_k(x)$ qui n'ont pas encore été examinés. En effet, dans P, on empile des piles α qui sont des applications partielles Ψ_k; ainsi peut-on prendre toutes les valeurs possibles de $\psi(x_k)$ dans $\Psi_{k-1}(x_k)$.

Q est une pile dont les éléments sont des couples (x a) où x est un sommet de H_1 et a un sommet de H : le correspondant choisi de x.

L'algorithme est le suivant:

{construction des applications ψ}
Début
{Initialisations:}
  P:= $(((x_1\ d_1\ \ldots\ d_q)))$
  Q:= λ
  fini:= **faux**
  retour_en_arrière:= **faux**
Répéter
{Progression de l'algorithme:}
  tant que non retour_en_arrière **faire**
    début
    α:= tête(P)
    si $\alpha = \lambda$ **alors**
      début
      imprimer (Q)
      retour_en_arrière:= **vrai**
      fin
    sinon si choix_suivant(α, Q, α', β, a) **alors**
        début
        empiler(Q, (tête (tête(α)), a))
        dépiler(P)

```
                    empiler(P, α')
                    empiler(P, β)
                    fin
                 sinon retour_en_arrière:= vrai
            fin
{Retour en arrière:}
     répéter
            dépiler(P)
            dépiler(Q)
     jusqu'à (nombre_d_éléments(tête(tête(P))) > 1) ou Q = λ
     si nombre_d_éléments(tête(tête(P))) = 1 alors fini:= vrai
            sinon retour_en_arrière:= faux
jusqu'à fini
fin
```

La fonction choix_suivant est une fonction booléenne qui cherche s'il est possible de trouver un correspondant dans H au prochain sommet de H_1 qui doit être examiné.

On se donne:

- la pile Q qui contient les correspondances déjà effectuées entre sommets de H_1 et sommets de H;

- la pile α qui propose les prochains sommets de H_1 à parcourir et un jeu de choix possibles pour les correspondants de ces sommets dans H. Le premier élément de la tête de α est le prochain sommet de H_1 à examiner.

Dans le cas où ce sommet a un correspondant, ce correspondant sera a.

D'autre part, on détermine:

- la pile α modifiée, les choix ultérieurs s'étant restreints. La nouvelle valeur de la pile α est contenue dans α';

- la pile β qui marque l'avancée dans le parcours de H_1 en donnant les prochains sommets de ce graphe à parcourir ainsi que les correspondants possibles de ces sommets dans H.

fonction choix_suivant(α, Q, a, α', β)
{α ne peut être une pile vide}

```
début
(x x¹ x² ... xᵖ):= tête(α)
i:= 0
répéter
     β:= queue(α)
     i:= i+1
     résultat:= vérification(β, Q, x, xⁱ)
jusqu'à (i=p) ou résultat
si résultat alors
     début
     a:= xⁱ
     α':= queue(α)
     empiler(α', (x xⁱ⁺¹ ... xᵖ))
     choix_suivant:= vrai
     fin
sinon choix_suivant:= faux
fin
```

La fonction vérification est une fonction booléenne qui vérifie si il est possible d'associer au sommet x de H_1 le sommet x^i de H. Pour répondre à la question, il faut connaître les choix déjà effectués, qui sont contenus dans la pile Q et les prochains choix possibles qui sont contenus dans la pile β. Dans le cas où la réponse est positive, la pile β sera modifiée, le choix de x et de x^i permettant d'envisager une avancée dans le parcours de H_1 et de H. β sera donc un paramètre d'entrée et de sortie de la fonction.

```
fonction vérification(β, Q, x, a)
début
soit δ₁(x) = y₁...y_q
soit δ₁⁻¹(x) = {v₁,...,v_r}
si |δ₁(x)| = 0 alors résultat:= vrai sinon
     si |δ₁(x)| ≠ |δ(a)| alors résultat:= faux sinon
          début
          résultat:= vrai
          {examen des descendants immédiats de x et a}
```

```
soit δ(a) = b1...bq
j:=0
tant que résultat et (j<q) faire
si ω1(yj) défini et (ω1(yj) ≠ ω(bj)) alors
résultat:= faux sinon
    début
    j:=j+1
    si correspondant(Q, yj, z) alors
        début
        si z ≠ bj alors résultat:= faux
        fin
    sinon si liste_de_correspondants(β, yj, l) alors
            si bj ε l alors
            remplacer dans β (yj c1 ... cr) par (yj bj)
            sinon résultat:= faux
        sinon empiler(β, (yj bj))
    fin
{examen des ascendants immédiats de x et a}
k:=0
tant que résultat et (k<r) faire
    début
    k:=k+1
    si correspondant(Q, vk, w) alors
        début
        si w ∉ δ⁻¹(a) alors résultat:= faux

        fin
    sinon si liste_de_correspondants(β, vj, l) alors
            si δ⁻¹(a) ∩ l ≠ ∅ alors
            remplacer dans β (vk d1 ... dr)
              par (vk dk1 ... dks) où
                    (dk1 ... dks) = δ⁻¹(a) ∩ l
            sinon résultat:= faux
        sinon
        début
        soit (z1 ... zt) la liste composée des
        éléments de δ⁻¹(a) tels que ω(zi) = ω1(vj)
        si t = 0 alors résultat:= faux
```

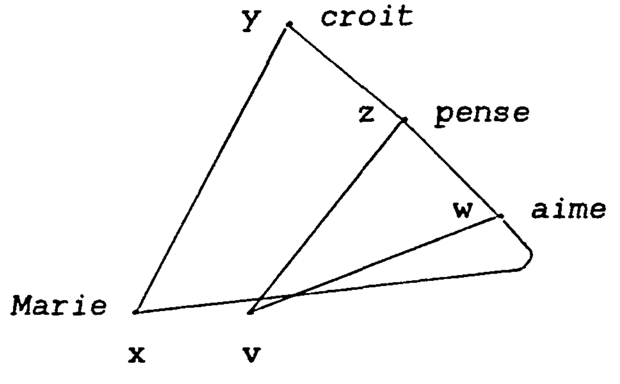
sinon empiler(β, (v_k z_1 ... z_t))

fin

fin

fin

L'appel de la fonction booléenne correspondant(Q, y_j, z) permet de savoir si dans la pile Q se trouve un couple (y_j b), auquel cas la valeur de b est mise dans z.
L'appel de la fonction booléenne liste_de_correspondants(β, v_j, l) permet de savoir si dans la pile β se trouve une liste telle que (v_j z_1 ... z_r), auquel cas la valeur (z_1 ... z_r) est mise dans l.



s	v	w	x	y	z
δ(s)	λ	vx	λ	xz	vw
ω(s)	-	*aime*	*Marie*	*croit*	*pense*

figure 5.8

Exemple: On se donne la question

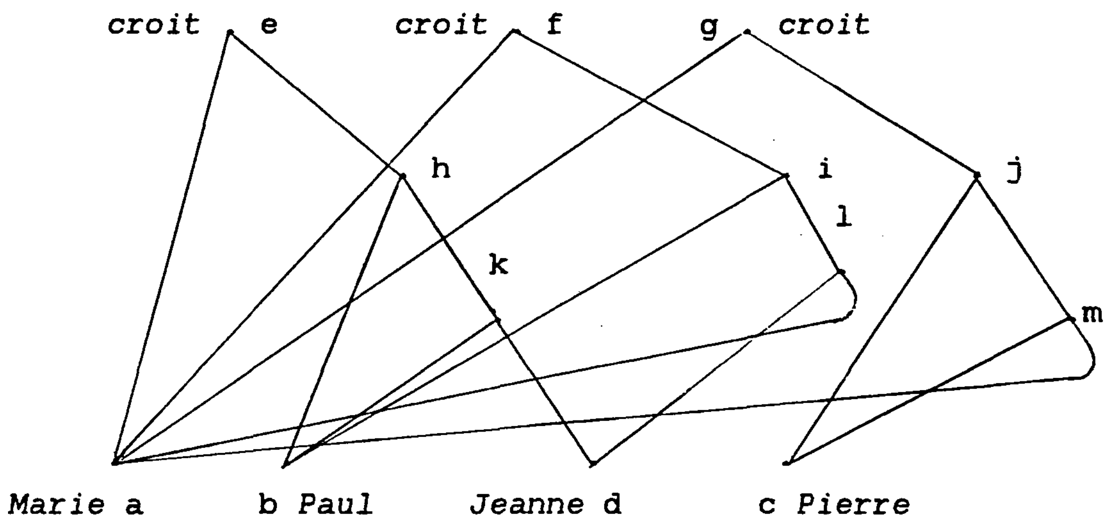
Marie croit-elle que quelqu'un pense qu'il l'aime ?

représentée par le Σ-GOSME H_1 de la figure 5.8
ainsi que les données:

Marie croit que Paul pense qu'il aime Jeanne

Marie croit que Paul pense que Jeanne aime Marie

Marie croit que Pierre pense qu'il aime Marie représentées par le Σ-GOSME H de la figure 5.9.



x	a	b	c	d	e	f	g	h	i	j	k	l	m
δ(x)	λ	λ	λ	λ	ah	ai	aj	bk	bl	cm	bd	da	ca
ω(x)	*Marie*	*Paul*	*Pierre*	*Jeanne*	*croit*			*pense*			*aime*		

---------------------------figure 5.9---------------------------

Les contenus successifs des piles P et Q sont les suivants:

Q	P
	((x a))
(x a)	((y e f g) (w l m)) ((x))
(x a) (y e)	((z h) (w l m)) ((y f g) (w l m))
	((x))
(x a)	((y f g) (w l m)) ((x))
(x a) (y f)	((z i) (w l m)) ((y g) (w l m)) ((x))
(x a) (y f) (z i)	((v b) (w l)) ((z) (w l m))
	((y g) (w l m)) ((x))
(x a) (y f)	((z) (w l m)) ((y g) (w l m)) ((x))
(x a)	((y g) (w l m)) ((x))
(x a) (y g)	((z j) (w l m)) ((y) (w l m)) ((x))
(x a) (y g) (z j)	((v c) (w m)) ((z) (w l m))

	((y) (w 1 m)) ((x))
(x a) (y g) (z j) (v c)	((w m)) ((v) (w m)) ((z) (w 1 m)) ((y) (w 1 m)) ((x))
(x a) (y g) (z j) (v c) (w m)	() ((w)) ((v) (w m)) ((z) (w 1 m)) ((y) (w 1 m)) ((x))
(valeur à imprimer)	
(x a) (y g) (z j) (v c)	((w)) ((v) (w m)) ((z) (w 1 m)) ((y) (w 1 m)) ((x))
(x a) (y g) (z j)	((v) (w m)) ((z) (w 1 m)) ((y) (w 1 m)) ((x))
(x a) (y g)	((z) (w 1 m)) ((y) (w 1 m)) ((x))
(x a)	((y) (w 1 m)) ((x))
	((x))

Remarquons que la tête de pile P se trouve située à gauche, alors que la tête de pile Q se trouve située à droite, afin que l'on puisse voir que les choix éventuels qui sont en tête de pile P deviennent des choix effectifs et passent donc en tête de pile P.

Une fois l'application ψ déterminée il reste, pour construire la réponse à la question, à trouver quel est le sous-graphe total englobant le sous-graphe de H engendré par $\psi(X_1)$. Cela consiste à:

(1) construire l'ensemble Y constitué par les sommets x de $\psi(X_1)$ tels que $\delta_1^{-1}(x) \cap \psi(X_1) = \emptyset$

(2) trouver tous les ascendants de ces sommets qui appartiennent à Ini(H); autrement dit construire l'ensemble $Z = \mathrm{Ini}(H) \cap Y$

(3) trouver tous les descendants des sommets de Z; la réponse sera représentée par le sous-Σ-GOSME de H engendré par $\hat{\gamma}(Z)$.

5.2 <u>Les questions comportant des modalités</u>

Avec le formalisme introduit dans la partie précédente, il a déjà été possible de répondre à des questions comportant des modalités du type *Que pense Marie ?* Néanmoins, il reste des cas où ce formalisme s'avère inopérant. Si par exemple on a la donnée suivante:

Paul croit que peut-être Pierre aime Marie

représentée par le Σ-GOSME H de la figure 5.10,

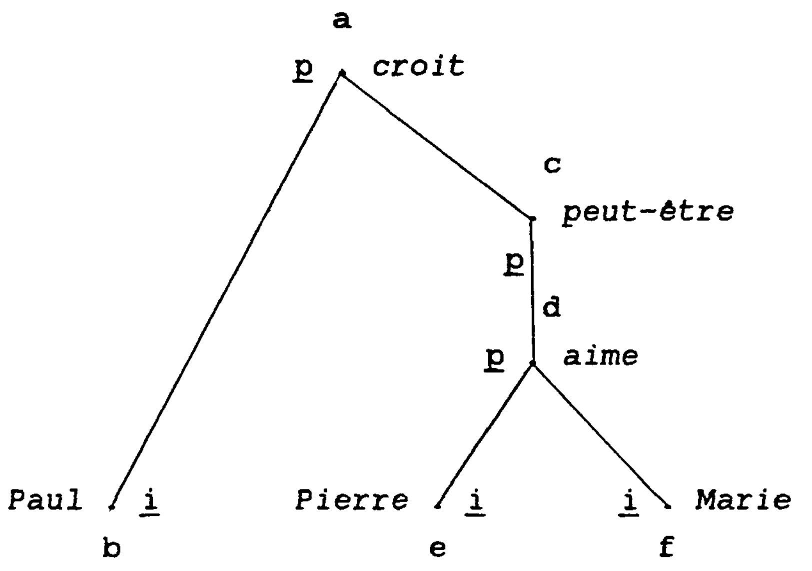


figure 5.10

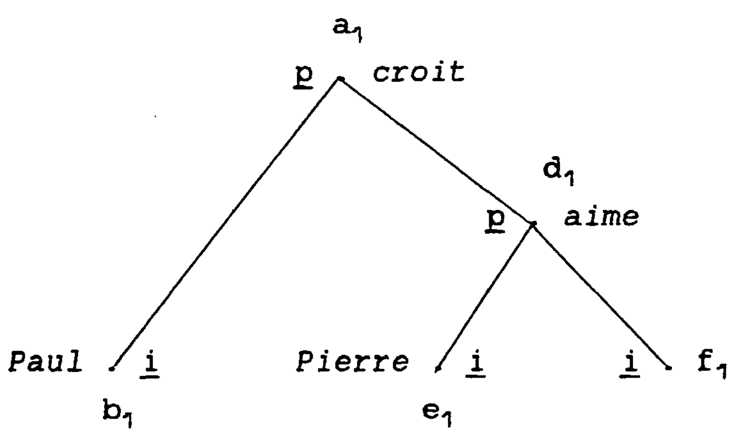


figure 5.11

il paraît naturel de pouvoir répondre à la question

qui Paul croit-il que Pierre aime ?

représentée par le Σ-GOSME H_1 de la figure 5.11. On en est pourtant incapable parce que l'image de l'arc $<a_1, d_1>$ de H_1 dans H ne peut être qu'un chemin: <a, c, d>.

C'est pourquoi il faudra que l'application ψ soit un morphisme plus large de structures, un agrandissement de GO (2.4.4), morphisme qui permet justement que l'image d'un arc soit un chemin. Mais, si l'on s'en tenait là, il deviendrait possible de répondre positivement à certaines questions alors qu'on ne le voudrait pas. Sur notre exemple, on répondrait positivement à la question

est-ce que Paul croit Pierre ?

représentée par le Σ-GOSME de la figure 5.12.

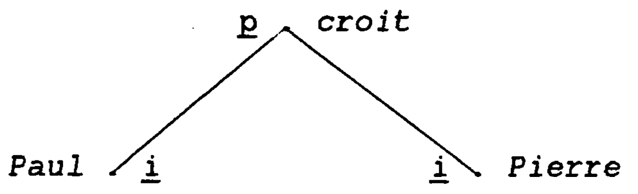


figure 5.12

C'est pourquoi l'application ψ devra finalement être un agrandissement de GOME vérifiant de plus la condition suivante:

$$\forall\, x,y \in X_1 \quad y \in \delta(x) \text{ et } \sigma(x) = \underline{i} \implies \psi(y) \in \delta\psi(x)$$

Autrement dit, si l'argument d'un verbe dans une question est un nom, alors dans la réponse l'argument correspondant du verbe est nécessairement le même nom. En revanche si l'argument d'un verbe est une proposition, cette proposition n'est pas forcément l'argument direct du verbe dans la réponse.

Sur le plan algorithmique, il est possible de reprendre l'algorithme décrit dans le paragraphe 5.1.3, moyennant quelques modifications: quand on examine un sommet de sorte p dans H_1 dont un ascendant immédiat ou un descendant immédiat est un sommet de sorte p, le correspondant de l'ascendant immédiat ou du descendant immédiat de ce sommet dans H peut être un ascendant ou un descendant au sens large du correspondant de ce sommet dans H.

Plus précisément, dans la fonction vérification,

- quand on examine les descendants immédiats de x, on ne se demande pas uniquement si le jième descendant b_j de a, correspondant éventuel de x, est tel que $\omega_1(y_j)$ est différent de $\omega(b_j)$, mais s'il existe une suite $b_j^1,\ldots,b_j^s$ d'éléments de $\hat{\delta}(\beta_j)$ tels que $\omega(b_j^k) = \omega_1(y_j)$. Auquel cas, on pourra, si nécessaire, empiler dans β $(y_j\ b_j^1 \ldots b_j^s)$ ou une sous-suite de cette suite;

- quand on examine les ascendants immédiats de a, $(z_1 \ldots z_t)$ est la liste des éléments de $\hat{\delta}^{-1}(a)$ tels que $\omega(z_i) = \omega_1(v_j)$.

5.3 Intrication avec les connaissances élémentaires

5.3.1 La représentation des questions

Si l'on dispose de connaissances enregistrées sous la forme d'un réseau H, une question à propos de ce réseau sera également représentée par un réseau H_1 obtenu après transformation de la question de la forme interrogative à la forme déclarative, et donc par une équation $<H,H_1>$.

Soit par exemple la représentation (figure 5.13) de l'énoncé:

Tous les cinéphiles parisiens ont vu tous les films de Woody Allen

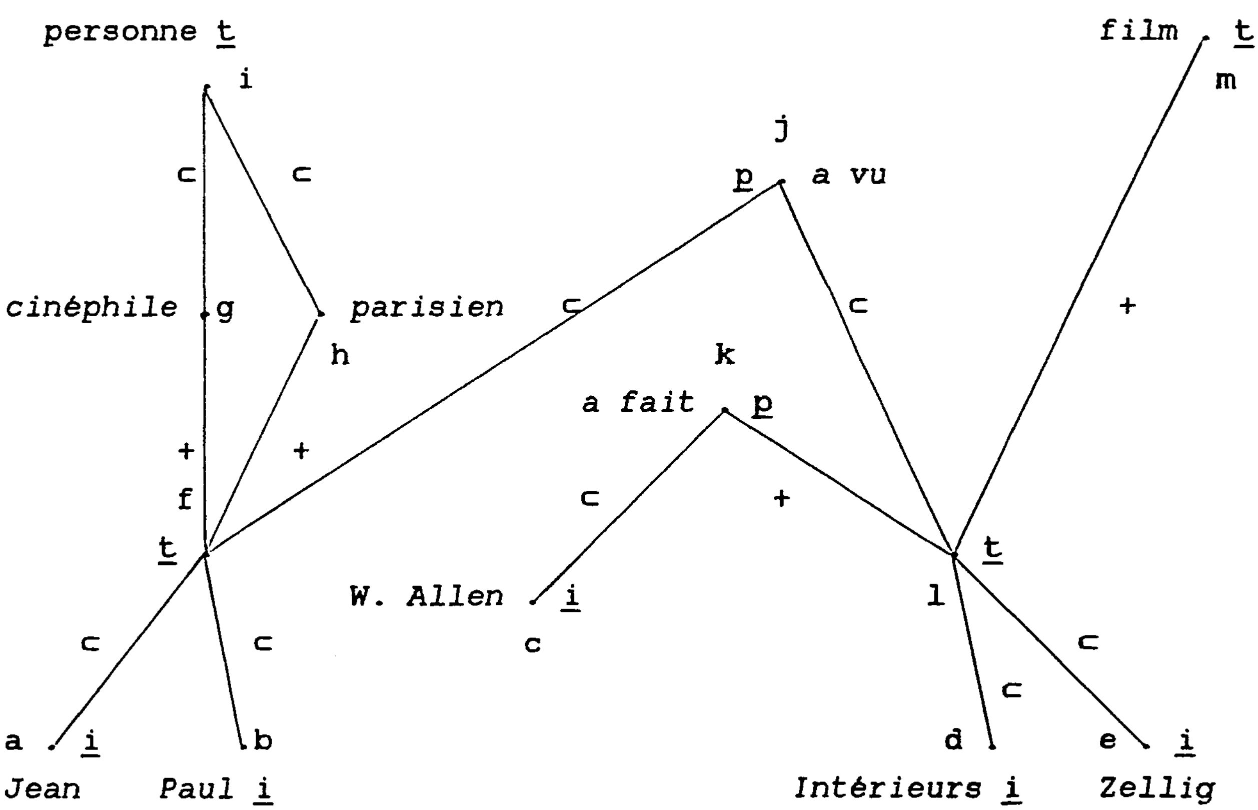


figure 5.13

et soit la question:

Quelle personne a vu Intérieurs ?

La mise sous la forme déclarative de cette question n'est pas évidente; on peut en effet envisager trois hypothèses:

(1) *une personne a vu Intérieurs*

(2) toute personne a vu Intérieurs

(3) *des personnes ont vu Intérieurs*

correspondant chacune a un réseau différent (figure 5.14).

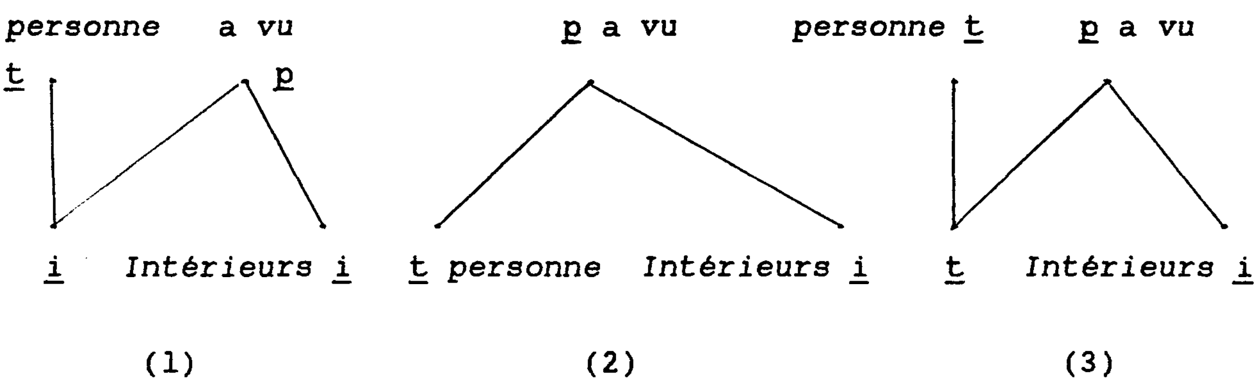


figure 5.14

Remarquons que les réseaux correspondant aux formes (1) et (3) ne diffèrent que par la sorte d'un sommet: i ou t. C'est pourquoi le réseau H_1, représentatif de la question, sera un réseau de la forme (1) ou (3), dans lequel simplement la sorte de ce sommet inconnu ne sera pas spécifiée (figure 5.15). On verra dans le paragraphe qui suit comment le cas correspondant à l'hypothèse (2) ne sera pas oublié dans la recherche des réponses aux questions.

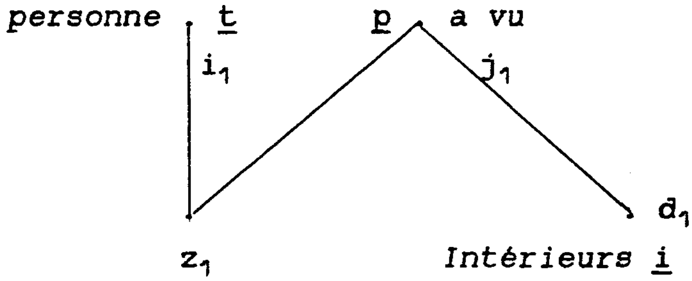


figure 5.15

5.3.2 La recherche des réponses aux questions

On se place dans le cas où il n'y a des modalités ni dans les questions, ni dans les textes enregistrés. On suppose d'autre part qu'il n'y a pas d'adverbes modifiant les adjectifs dans les questions, ni de sommets typiques complexes (absence des étiquettes d'arcs + et $\neq$). Ces deux derniers cas seront traités ultérieurement.

La réponse à une question sera une connaissance extraite du réseau représentatif des connaissances enregistrées, par exemple:

Jean a *vu* *Intérieurs*.

Mais, on s'aperçoit que cette connaissance ne figure pas telle quelle dans le réseau H. Dans ce réseau, en effet, les liens entre le sommet *a vu* et les sommets *Jean* et *Intérieurs* ne sont plus de simples arcs, mais ce sont des chemins, images des arcs du réseau H_1. C'est pourquoi l'application ψ de H_1 dans H devra encore être un agrandissement.

De plus, pour avoir le droit de donner cette réponse, il faut vérifier que

Jean est une *personne*

c'est-à-dire que dans le réseau H, le sommet *Jean* descend du sommet *personne*.

Ces contraintes sont vérifiées dans le cas où il y a un agrandissement de H_1 dans H.

Toutefois, notons qu'il est nécessaire que *Jean* et *Intérieurs* descendent de deux descendants immédiats du sommet *a vu* qui soient dans cet ordre de succession. Autrement dit que

$a\ \theta_j\ d$

alors qu'une telle contrainte serait inutile pour les descendants de *personne*.

Plus généralement, une telle contrainte est nécessaire pour uniquement les descendants des sommets de sorte $\underline{p}$.

On voit donc que toute application ψ fournissant une solution de l'équation devra être un agrandissement de réseaux (2.6.5) qui conserve les sortes et les unités lexicales.

Dans ce cas, il y a 3 applications ψ_1, ψ_2 et ψ_3 qui fournissent une solution, applications que nous regroupons dans le tableau de la figure 5.16.

x	d_1	i_1	j_1	z_1
$\psi_1(x)$	d	i	j	a
$\psi_2(x)$	d	i	j	b
$\psi_3(x)$	d	i	j	f

--------------------------<u>figure 5.16</u>--------------------------

L'image du sommet x de H_1 dont la sorte est indéterminée est, dans le premier ou le deuxième cas un sommet de sorte <u>i</u> représentant *Jean* ou *Paul*, alors que dans le troisième cas c'est un sommet typique qui ne reçoit pas d'étiquette lexicale et qui peut être décrit comme étant le cinéphile parisien typique.

Notons que si l'on avait comme donnée

toute personne a vu Intérieurs

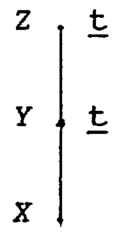
la même question étant posée, on trouverait bien une réponse puisque dans un agrandissement l'image d'un arc peut être un chemin de longueur nulle. Alors que si l'on avait représenté la question par un réseau de type (2) de la figure 5.14, on n'aurait pas pu trouver les réponses déterminées par ψ_1, ψ_2 ou ψ_3 car le sommet personne ne descend pas de a vu.

5.3.3 <u>Algorithmes de recherche et construction des solutions</u>

Puisqu'il n'y a pas d'adverbes modifiant les adjectifs dans les questions , puisque les questions ne sont pas modalisées et qu'elles ne comportent pas non plus d'objets typiques complexes, le rang du graphe représentatif d'une question sera au maximum égal à 1. En effet, les graphes de rang supérieur à 1 contiendraient des parties telles que celle de la figure 5.17.

On pourrait alors interpréter les questions correspondantes de la manière suivante:

y a-t-il un (resp. des) X qui, tout en vérifiant d'autre propriétés, est un (resp. sont des) Y, et tous les Y sont-ils des Z ?



--------------------------figure 5.17--------------------------

questions bien évidemment décomposables en deux questions indépendantes.

Remarquons d'autre part que tout sommet initial de la question a une étiquette lexicale, puisque l'on admet pas les questions telles que

une personne est un quoi ?

ou *quel lien y a-t-il entre Jean et Marie ?*

et que tout sommet terminal est successeur d'au moins un sommet initial qui admet une étiquette lexicale.

Enfin, tous les sommets typiques de la question admettent une étiquette lexicale.

L'algorithme de recherche des solutions sera décrit sous la forme d'un déplacement de marqueurs booléens. Mais, auparavant, on définit cet algorithme de manière formelle en se donnant 20 applications qui servent à décrire les différentes étapes du processus de déplacement des marqueurs.

On pose $X_1 = U_1 \cup V_1 \cup W_1$ avec

$U_1 = \{x \in X_1^2;\ \varepsilon_1^1 = \underline{t}$ ou $\varepsilon_1^1 = \underline{i}$ ou $\varepsilon_1^1 = \underline{j}\}$

$V_1 = X_1 \backslash X_1^2$

$W_1 = \{x \quad X_1;\ \varepsilon_1^1 = \underline{p}\}$

L'application ψ_1 est définie par

$\psi_1 :\ U_1 \dashrightarrow \mathcal{P}(X)$

$x \mapsto \{y \in X;\ \varepsilon^2(y) = \varepsilon^2(y)$ et $\varepsilon^1(y) = \varepsilon^1(y)\}$

(A tout sommet typique ou individuel de H_1 qui admet une étiquette lexicale, on associe tous les sommets de H qui admettent même sorte et même étiquette lexicale).

Pour les propriétés 1 à 14, on supposera que ψ est un agrandissement de réseaux de H_1 dans H qui conserve les sortes et les unités lexicales.

<u>Propriété 1</u>: $\forall x \in U_1 \quad \psi(x) \in \psi_1(x)$

<u>Preuve</u>: Immédiate, puisque ψ conserve les sortes et les unités lexicales.

Les applications ψ_2, ψ_3, ψ_4 et ψ_5 sont définies par:

$\psi_2 \ : \ U_1 \dashrightarrow \mathcal{P}(X)$

$\forall x \in \mathrm{Ini}(H_1) \quad x \mapsto \hat{\delta}\psi_1(x)$

$\forall x \in U_1 \setminus \mathrm{Ini}(H_1) \quad x \mapsto \psi_1(x)$

$\psi_3 \ : \ U_1 \cup V_1 \dashrightarrow \mathcal{P}(X_1)$

$x \mapsto \{y \in U_1;\ x \in \hat{\delta}_1(y)\}$

$\psi_4 \ : \ X \dashrightarrow \mathcal{P}(X_1)$

$x \mapsto \{y \in U_1;\ x \in \psi_2(y)\}$

$\psi_5 \ : \ V_1 \dashrightarrow \mathcal{P}(X)$

$x \mapsto \{y \in X;\ \psi_3(x) \subset \psi_4(y)\}$

<u>Propriété 2</u>: $\forall x \in V_1 \quad \psi(x) \in \psi_5(x)$

<u>Preuve</u>: x est nécessairement terminal dans H_1, par conséquent:

$\psi_3(x) = \delta^{-1}(x) \cap U_1$

Soit y appartenant à $\psi_3(x)$; comme ψ est un agrandissement,

$\psi(x) \in \hat{\delta}\psi(x)$

et, d'après la propriété 1

$\psi(x) \in \hat{\delta}\psi_1(y)$

D'où $\psi(x) \in \psi_2(y)$ et par conséquent $x \in \psi_4\psi(x)$.

On en déduit $\psi_3(x) \subset \psi_4\psi(x)$

ce qui implique $\psi(x) \in \psi_5(x)$.

Remarquons que la démonstration est valable y compris dans le cas où $\delta^{-1}(x) \cap U_1 = \emptyset$

L'application ψ_6 est définie par

ψ_6 : $U_1 \dashrightarrow \mathcal{P}(X)$

$x \mapsto \{y \in \psi_1(x);\ \psi_3(x) \subset \psi_4(y)\}$

<u>Propriété 3</u>: $\forall\, x \in U_1 \quad \psi(x) \in \psi_6(x)$

<u>Preuve</u>: On considère deux cas.

1er cas: $x \in \mathrm{Ini}(H_1)$; dans ce cas $\psi_3(x) = \{x\}$

D'après la propriété 1 $\hat{\delta}\psi(x) \in \psi_2(x)$;

on en déduit que $\psi(x) \in \psi_2(x)$ et,

$x \in \psi_4\psi(x)$

Donc $\psi_3(x) \subset \psi_4\psi(x)$ et $\psi(x) \in \psi_6(x)$.

2ème cas: $x \in \mathrm{Ter}(H_1)$; dans ce cas $\psi_3(x) = \{x\} \cup [\delta_1^{-1}(x) \cap U_1]$

Soit y différent de x appartenant à $\psi_3(x)$; en suivant la démonstration de la propriété 2, on trouve $\psi(x) \in \psi_2(y)$, ce qui permet d'en déduire:

$x \in \psi_4\psi(y)$

Si $y = x$, on parvient au même résultat en suivant la démonstration du 1er cas.

L'application ψ_7 est définie par

ψ_7 : $U_1 \cup V_1 \dashrightarrow \mathcal{P}(X)$

$\forall\, x \in U_1 \quad \psi_7(x) = \psi_6(x)$

$\forall\, x \in V_1 \quad \psi_7(x) = \psi_5(x)$

<u>Propriété 4</u>: $\forall\, x \in U_1 \cup V_1 \qquad \psi(x) \in \psi_7(x)$

<u>Preuve</u>: Résulte immédiatement des propriétés 2 et 3.

On définit les sous-ensembles W_1', W_1'' et W_1''' de la manière suivante:

$W_1' = \{x \in W_1;\ |\delta(x)| = 2\}$

$W_1'' = \{x \in W_1;\ \delta_1^{(1)}(x) \in \hat{\delta}_1(U_1)\}$

$W_1''' = \{x \in W_1';\ \delta_1^{(2)}(x) \in \hat{\delta}_1(U_1)\}$

Remarquons que l'on a: $\quad W_1 = W_1'' \cup W_1'''$

Les applications ψ_8, ψ_9, ψ_{10} et ψ_{11} sont définies par:

ψ_8 : $W_1'' \dashrightarrow \mathcal{P}(X)$

$x \mapsto \psi_7\delta_1^{(1)}(x)$

ψ_9 : $W_1''' \dashrightarrow \mathcal{P}(X)$

$x \mapsto \psi_7\delta_1^{(2)}(x)$

ψ_{10} : $W_1'' \dashrightarrow \mathcal{P}(X)$

$x \mapsto \{y \in X;\ \psi_8(x) \cap \hat{\delta}(y) \neq \emptyset\}$

ψ_{11} : $W_1''' \dashrightarrow \mathcal{P}(X)$

$x \mapsto \{y \in X;\ \psi_9(x) \cap \hat{\delta}(y) \neq \emptyset\}$

Propriété 5: $\forall\ x \in W_1''$ $\psi\delta_1^{(1)}(x) \in \psi_{10}(x)$

Preuve: $\psi_8(x) = \psi_7\delta_1^{(1)}(x)$

Soit z appartenant à $\psi_{10}(x)$; on a:

$\psi_{10}(x) = \{z;\ \psi_8(x) \cap \hat{\delta}(z) \neq \emptyset\}$

$\psi_{10}(x) = \{z;\ \psi_7\delta_1^{(1)}(x) \cap \hat{\delta}(z) \neq \emptyset\}$

Par conséquent: $\psi_7\delta_1^{(1)}(x) \subset \psi_{10}(x)$ et donc:

$\psi\delta_1^{(1)}(x) \in \psi_{10}(x)$

Propriété 6: $\forall\ x \in W_1'''$ $\psi\delta_1^{(2)}(x) \in \psi_{11}(x)$

Preuve: La preuve est analogue à celle de la propriété 5.

Propriété 7: Si $y \in \psi_{10}(x)$ (resp. $\psi_{11}(x)$) et $y \in \hat{\delta}(z)$ alors $z \in \psi_{10}(x)$ (resp. $\psi_{11}(x)$).

Preuve: Immédiate.

L'application ψ_{12} est définie de la manière suivante:

ψ_{12} : $W_1 \dashrightarrow \mathcal{P}(X)$

$x \in W_1'' \cap W_1''' \Longrightarrow \psi_{12}(x) = \{y \in X;\ Q(x,y)$ et $y \in \psi_{10}(x)$ et $y \in \psi_{11}(x)$ et $\delta^{(1)}(y) \in \psi_{10}(x)$ et $\delta^{(2)}(y) \in \psi_{11}(x)\}$

$x \in W_1''\backslash W_1''' \Longrightarrow \psi_{12}(x) = \{y \in X;\ Q(x,y)$ et $y \in \psi_{10}(x)$ et
et $\delta^{(1)}(y) \in \psi_{10}(x)\}$

$x \in W_1'''\backslash W_1'' \Longrightarrow \psi_{12}(x) = \{y \in X;\ Q(x,y)$ et $y \in \psi_{11}(x)$
et $\delta^{(2)}(y) \in \psi_{11}(x)\ \}$

<u>Propriété 8</u>: $\forall\ x \in W_1\quad \psi(x) \in \psi_{12}(x)$

<u>Preuve</u>: On considère trois cas.
1er cas: $x \in W_1'' \cap W_1'''$

Comme ψ est un agrandissement de {p}-réseaux, on a forcément $Q(x,\psi(x))$. De plus, on a:

$\psi\delta_1^{(1)}(x) \in \hat{\delta}\delta^{(1)}\psi(x)$

Or, la propriété 5 s'exprime: $\psi\delta_1^{(1)}(x) \in \psi_{10}(x)$
Il en résulte, d'après la propriété 7, que:

$\delta^{(1)}\psi(x) \in \psi_{10}(x)$

Et comme $\psi\delta_1^{(1)}(x) \in \hat{\delta}\psi(x)$, toujours d'après la propriété 3,

$\psi(x) \in \psi_{10}(x)$

On montre de même que $\delta^{(1)}\psi(x) \in \psi_{10}(x)$ et que $\psi(x) \in \psi_{10}(x)$
Dans les deuxièmes et troisièmes cas, la démonstration est analogue.

On définit les applications ψ_{13} et ψ_{14} de la manière suivante:

$\psi_{13} : W_1 \dashrightarrow \mathcal{P}(X)$
$\forall\ x \in W_1''\quad \psi_{13}(x) = \{y \in \psi_{10}(x);\ \exists\ z \in \psi_{12}(x)\quad y \in \hat{\delta}\delta^{(1)}(z)\}$
$\forall\ x \in W_1'''\backslash W_1''\quad \psi_{13}(x) = \{y \in X; \exists\ z \in \psi_{12}(x)\quad y \in \hat{\delta}\delta^{(1)}(z)\}$

$\psi_{14} : W_1' \dashrightarrow \mathcal{P}(X)$
$\forall\ x \in W_1'''\quad \psi_{14}(x) = \{y \in \psi_{11}(x);\ \exists\ z \in \psi_{12}(x)\quad y \in \hat{\delta}\delta^{(1)}(z)\}$
$\forall\ x \in W_1''\backslash W_1'''\quad \psi_{13}(x) = \{y \in X; \exists\ z \in \psi_{12}(x)\quad y \in \hat{\delta}\delta^{(1)}(z)\}$

<u>Propriété 9</u>: $\forall\ x \in W_1\quad \psi\delta_1^{(1)}(x) \in \psi_{13}(x)$

<u>Preuve</u>: Comme ψ est un agrandissement,

$\psi\delta_1^{(1)}(x) \in \hat{\delta}\delta^{(1)}\psi(x)$

D'après la propriété 8, on en déduit:

$\psi\delta_1^{(1)}(x) \in \hat{\delta}\delta^{(1)}\psi_{12}(x)$

et comme, d'après la propriété 5, si $x \in W_1''$ alors $\psi\delta_1^{(1)}(x) \in \psi_{10}(x)$, la propriété est bien démontrée.

<u>Propriété 10</u>: $\forall\ x \in W_1' \quad \psi\delta_1^{(2)}(x) \in \psi_{14}(x)$

<u>Preuve</u>: Analogue à la démonstration de la propriété 9.

Les applications ψ_{15}, ψ_{16}, ψ_{17} et ψ_{18} sont définies par:

ψ_{15} : $X \dashrightarrow \mathcal{P}(W_1)$
$\quad x \mapsto \{y \in W_1;\ x \in \psi_{13}(y)\}$

ψ_{16} : $X \dashrightarrow \mathcal{P}(W_1')$
$\quad x \mapsto \{y \in W_1';\ x \in \psi_{14}(y)\}$

ψ_{17} : $Ter(H_1) \dashrightarrow \mathcal{P}(W_1)$
$\quad x \mapsto \{y \in W_1;\ x = \delta_1^{(1)}(y)\}$

ψ_{18} : $Ter(H_1) \dashrightarrow \mathcal{P}(W_1')$
$\quad x \mapsto \{y \in W_1';\ x = \delta_1^{(2)}(y)\}$

<u>Propriété 11</u>: $\forall\ x \in Ter(H_1) \quad \psi_{17}(x) \subset \psi_{15}\psi(x)$

<u>Preuve</u>: Soit y appartenant à $\psi_{17}(x)$; on a: $x = \delta_1^{(1)}(y)$. Donc $\psi(x) \in \psi_{13}(y)$ et $y \in \psi_{15}\psi(x)$.

<u>Propriété 12</u>: $\forall\ x \in Ter(H_1) \quad \psi_{18}(x) \subset \psi_{16}\psi(x)$

<u>Preuve</u>: Analogue à la preuve de la propriété 11.

On définit l'application ψ_{19} de la manière suivante:

ψ_{19} : $\delta_1(W_1) \dashrightarrow \mathcal{P}(X)$
$\quad x \mapsto \{y;\ \psi_{17}(x) \subset \psi_{15}(y) \text{ et } \psi_{18}(x) \subset \psi_{16}(y)\}$

<u>Propriété 13</u>: $\forall\ x \in \delta_1(W_1)\ \psi(x) \in \psi_{19}(x)$

<u>Preuve</u>: Résulte immédiatement des propriétés 11 et 12.

On définit enfin l'application ψ_{20} de la manière suivante:

$\psi_{20} : X_1 \dashrightarrow \mathcal{P}(X)$

$\forall x \in \delta_1(W_1) \quad \psi_{20}(x) = \psi_{19}(x)$

$\forall x \in W_1 \quad \psi_{20}(x) = \psi_{12}(x)$

$\forall x \in (U_1 \cup V_1)\setminus\delta_1(W_1) \quad \psi_{20}(x) = \psi_7(x)$

<u>Propriété 14</u>: $\forall x \in X_1 \quad \psi(x) \in \psi_{20}(x)$

<u>Preuve</u>: Résulte immédiatement des propriétés 4, 8 et 13.

L'algorithme par déplacement de marqueurs booléens consiste à construire les différentes applications afin d'arriver à la construction de ψ_{20}.

Chaque sommet pourra recevoir 7 types de marqueurs booléens:

- T1, T2,...
- I1, I2,...
- P1, P2,...
- G1, G2,...
- D1, D2,...
- F1, F2,...
- C1, C2,...

(1) <u>le marquage des sommets typiques et individuels de H_1 et de H</u>:

Les p sommets typiques de H_1 sont marqués respectivement par T1, T2,..., Tp.

Les q sommets individuels de H1 qui reçoivent une étiquette lexicale sont marqués respectivement par I1, I2,...,Iq.

Chacun des sommets de U_1 reçoit donc un marqueur de type I ou T.

Dans H on marque tout sommet de sorte <u>t</u> ou de sorte <u>i</u> qui admet la même étiquette lexicale qu'un sommet de H_1 par le marqueur de son correspondant de H_1.

Ainsi, pour chaque sommet x de U_1 $\psi_1(x)$ est composé de l'ensemble des sommets de X qui reçoivent le même marqueur que x.

<u>Exemple</u>: Soit la question

Jean est-t-il un cinéphile parisien ?

représentée par le réseau H_1 de la figure 5.18

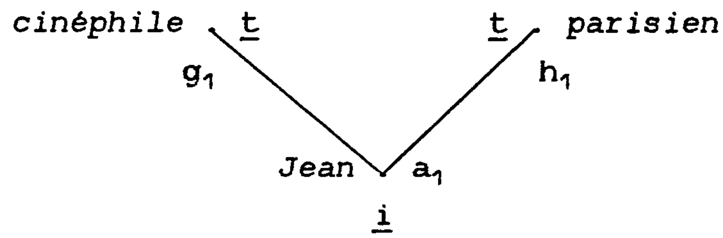


figure 5.18

et soit le réseau H de la figure 5.13.

Le sommet g_1 est marqué par T1, ainsi par conséquent que le sommet g de H.

Le sommet h_1 est marqué par T2, ainsi que le sommet h de H.

Le sommet a_1 est marqué par I1, ainsi que le sommet a de H.

(2) <u>le déplacement des marqueurs de type T</u>

Dans H tout descendant d'un sommet marqué par Ti et correspondant d'un sommet de $Ini(H_1)$ sera également marqué par Ti.

Pour x appartenant à U_1, tous les sommets de X qui reçoivent le même marqueur que x forment alors $\psi_2(x)$.

Dans H_1, tout successeur d'un sommet initial marqué par Ti sera également marqué par Ti.

ψ_3 décrit alors l'ensemble des marqueurs reçus par les sommets de H_1 qui ne sont pas de sorte $\underline{t}$.

ψ_4 associe à chaque sommet typique ou individuel de X l'ensemble des marqueurs qu'il reçoit, ou plutôt les sommets de U_1 d'où proviennent ces marqueurs.

Pour tout x appartenant à V_1, $\psi_5(x)$ est composé de tous les sommets de X dont l'ensemble des marqueurs contient l'ensemble des marqueurs associés à x.

De même, pour tout x appartenant à U_1, ψ_6 est composé de tous les sommets de X de même étiquette lexicale que x et dont l'ensemble des marqueurs contient l'ensemble des marqueurs associés à x.

A la suite de cette opération on peut répondre aux questions qui ne comportent aucune proposition, en vérifiant qu'à tout sommet de H_1 correspond un sommet de H recevant un ensemble de marqueurs dans lequel est inclus l'ensemble des marqueurs du sommet de H_1. Si le sommet de H_1 a une étiquette lexicale, alors il faut que le sommet de H correspondant possède la même étiquette lexicale. Dans le cas où la question comporte une inconnue, c'est-à-dire un sommet sans étiquette lexicale, le sommet de H associé au sommet inconnu de H_1 fournit une solution.

La solution est ainsi trouvée en associant à tout sommet x de H_1 un sommet de $\psi_7(x)$.

<u>Exemple</u>: On reprend l'exemple précédent. Dans H_1 le sommet a_1 est finalement marqué par I1, T1 et T2.

Dans H les sommets b et f vont être marqués par T1 et T2, alors que le sommet a va être marqué par I1, T1 et T2. En raison du marquage de ce dernier sommet, la réponse à la question sera bien positive.

(3) <u>le marquage des sommets pris dans des propositions</u>

Les r sommets de sorte <u>p</u> de H_1 sont marqués respectivement par P1, P2,...,Pr.

Chaque sommet de sorte <u>p</u> a au maximum deux successeurs. On suppose qu'il en a au moins un, excluant les questions de type *pleut-il ?* On marque le premier successeur d'un sommet marqué par Pi par Gi et le deuxième successeur, s'il y en a un, par Di.

A ce moment-là chaque sommet terminal de H_1 possède deux ensembles de marqueurs:

- un ensemble M_{ti} composé de marqueurs de type T et éventuellement d'un marqueur de type I; cet ensemble peut être vide;

- un ensemble M_{gd} composé de marqueurs de type G ou D; cet ensemble n'est pas vide dans le cas où la question comporte des propositions.

Pour chaque sommet terminal de H_1 tel que l'ensemble M_{ti} soit non vide, on cherche tous les sommets de H qui reçoivent un ensemble de marqueurs tel que M_{ti} soit inclus dans cet ensemble. Soit z un de ces sommets; pour chaque marqueur Gi (resp. Dj) de Y, z sera marqué par Fi (resp. Cj).

Cela revient, pour chaque sommet x de sorte p de H_1 marqué par Pi, à marquer tous les sommets de $\psi_8(x)$ par Fi et tous les sommets de $\psi_9(x)$ par Ci.

Exemple: Nous reprenons la question représentée par le graphe H_1 de la figure 5.15 avec le réseau H de la figure 5.13.

Dans H_1:
- le sommet i_1 est marqué par T1
- le sommet j_1 est marqué par P1
- le sommet d_1 est marqué par I1 et D1
- le sommet z_1 est marqué par T1 et G1.

Dans H:
- les sommets a,b,f,g,h,i sont marqués par T1
- le sommet d est marqué par I1.

Comme d est le seul sommet à être marqué par I1, il sera seul à être marqué par C1.

Tous les sommets marqués par T1, c'est-à-dire a, b, f, g, h et i seront marqués par F1.

(4) la propagation de ces marqueurs

Dans H, tous les ascendants des sommets marqués par Fi (resp. Cj) sont également marqués par Fi (resp. Cj).

Cela revient, pour chaque sommet x de W_1 marqué par Pi, à marquer par Fi tous les sommets de $\psi_{10}(x)$ et par Ci tous les sommets de $\psi_{11}(x)$.

Si un sommet x de sorte p de H est marqué à la fois par Fi et par Ci et que son premier successeur est marqué par Fi, son second successeur étant marqué par Ci, on vérifie si le sommet de H_1 marqué par Pi porte la même étiquette lexicale. Auquel cas, on marque le sommet x par Pi, on marque son premier successeur par Gi ainsi que tous les descendants de ce premier successeur à condition qu'ils soient déjà marqués par Fi, on marque son deuxième successeur par Di ainsi que tous les descendants de son deuxième successeur à condition qu'ils soient déjà marqués par Ci.

Si un sommet x de sorte p de H est marqué par Fi et que son premier successeur est marqué par Fi, on vérifie si le sommet de H_1 marqué par Pi n'a qu'un successeur ou si son deuxième successeur est non marqué par des marqueurs de type I ou T et s'il porte la même étiquette lexicale. Auquel cas, on marque le sommet x par Pi, on marque son premier successeur par Gi ainsi que tous les descendants de ce premier successeur à condition qu'ils soient déjà marqués par Fi, on marque son deuxième successeur (s'il en a un) par Di ainsi que tous les descendants de son deuxième successeur.

Si un sommet x de sorte p de H est marqué par Ci et que son second successeur est marqué par Ci, on vérifie si le sommet de H_1 marqué par Pi a un premier successeur non marqué avec des marqueurs de type I ou T et s'il porte la même étiquette lexicale. Auquel cas, on marque le sommet x par Pi, on marque son premier successeur par Gi ainsi que tous les descendants de ce premier successeur, on marque son deuxième successeur par Di ainsi que tous les descendants de son deuxième successeur à condition qu'ils soient déjà marqués par Ci.

Ainsi, à chaque sommet x de W_1, marqué par Pi, on associe tous les sommets de H qui forment $\psi_{12}(x)$: ce sont les sommets de H également marqués par Pi.

A x on associe également tous les sommets de $\psi_{13}(x)$ (ce sont les sommets de H marqués par Gi) et tous les sommets de $\psi_{14}(x)$ (les sommets de H marqués par Di).

<u>Exemple</u>: En reprenant l'exemple précédent, on commence par marquer le sommet j par Fl (en tant qu'ascendant par exemple de f) et tous les ascendants de d, c'est-à-dire j, k, l et m par Cl.

Le sommet k de H est un sommet de sorte <u>p</u> marqué par Cl, mais l'unité lexicale qui lui est associée est différente de celle qui est associée au sommet de H_1 marqué par Pl.

Le sommet j de H est un sommet de sorte <u>p</u> marqué par Fl et Cl et il lui est associé la même étiquette lexicale qu'au sommet de H_1 marqué par Pl. j va donc être également marqué par Pl, les sommets a et f de H vont être marqués par Gl, les sommets d, e et l étant pour leur part marqués par Dl.

On obtient alors des solutions à l'équation d'interrogation, puisque l'on peut faire correspondre trois sommets de H au sommet inconnu de H_1: les sommets a, b ou f, et que l'on a bien dans les trois cas un agrandissement de réseaux de H_1 dans H.

Remarquons que deux des sommets solutions sont des sommets individuels: on peut dire qu'ils fournissent une réponse de manière **extentionnelle**: *Jean* et *Paul*.

Le dernier sommet est typique; il fournit sa réponse de manière **intensionnelle**: il s'agit du *cinéphile parisien typique*. Mais, dans la mesure où il s'agit d'un sommet typique complexe, pour extraire cette réponse, on aura besoin de construire le sous-graphe définitoire du sommet typique f.

En 4.2.6 il a été donné un algorithme récursif permettant de construire le sous-graphe définitoire d'un sommet typique.

(5) <u>vérification que l'on a bien trouvé une solution</u>

On vérifie qu'à chaque sommet terminal de H_1 correspond un sommet de H qui reçoit tous les marqueurs reçus par le sommet de H_1.

Si x est un sommet de $Ter(H_1)$, les sommets de H qui reçoivent tous les marqueurs reçus par x sont les sommets de $\psi_{19}(x)$.

Nous allons donner un exemple montrant pourquoi cette vérification est indispensable. Soit la donnée

Jean est un homme qui connaît Marie

Marie est une personne qui connaît Jean

tout homme est une personne

représentée par le réseau H de la figure 5.19, et soit la question

Jean est-il un homme qui connaît une personne qui connaît Marie ?

représentée par le réseau H_1 de la figure 5.20.

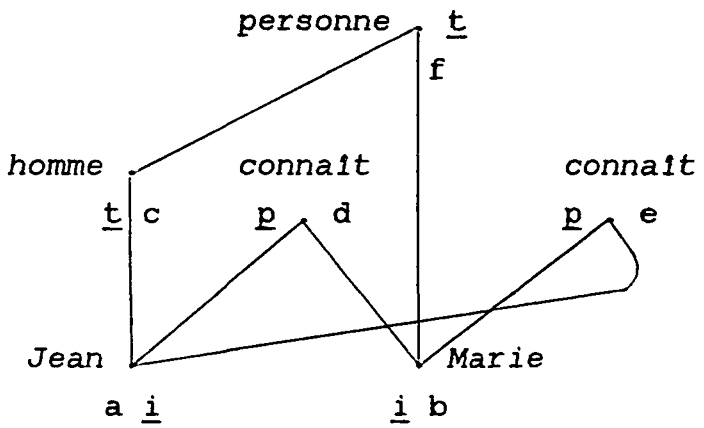


figure 5.19

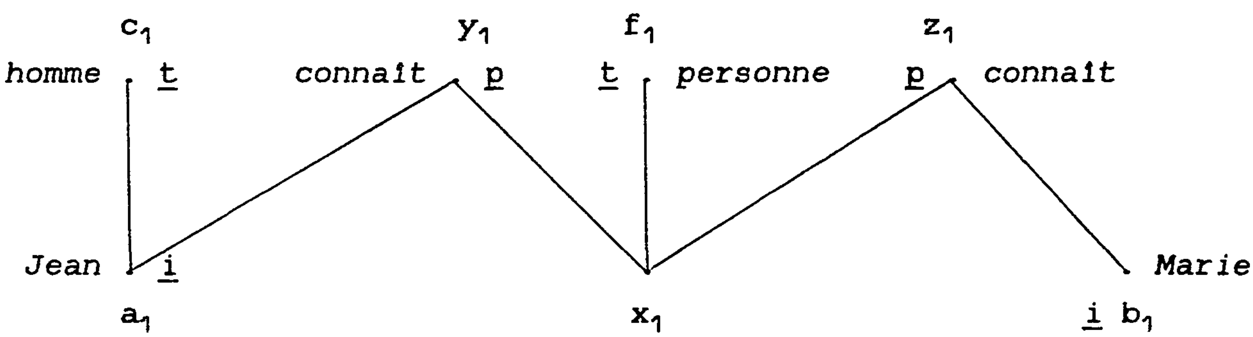


figure 5.20

Dans H_1,

- le sommet c_1 est marqué par T1
- le sommet f_1 est marqué par T2
- le sommet y_1 est marqué par P1

- le sommet z_1 est marqué par P2
- le sommet a_1 est marqué par I1, T1 et G1
- le sommet x_1 est marqué par T2, D1 et G2
- le sommet b_1 est marqué par I2 et D2.

Dans H, par conséquent,
- le sommet f est marqué par T2
- le sommet c est marqué par T1 et T2
- le sommet a est marqué par I1, T1 et T2
- le sommet b est marqué par I2 et T2.

On peut donc marquer a par F1, C1 et F2, ainsi que tous les ascendants de a, c'est-à-dire c, d, e et f.

On peut également marquer b par C1, F2 et C2, ainsi que tous les ascendants de b, c'est-à-dire d, e et f.

Finalement, le sommet d vérifie toutes les conditions qui lui permettent de recevoir les marqueurs P1 et P2.

Le sommet a va donc être marqué par G1 et G2, le sommet b étant marqué par D1 et D2.

Et pourtant, on ne doit pas pouvoir trouver de réponse positive à la question, comme il est évident intuitivement.

On vérifie qu'il n'y a pas de solution à l'équation en constatant qu'il n'y a pas dans H un sommet marqué de la même manière que le sommet x_1 de H_1, c'est-à-dire recevant les marqueurs D1 et G2.

Toute réponse à une question vérifiera les conditions suivantes:

- à tout sommet x de H_1 qui est successeur d'un sommet de sorte p correspond un sommet de H qui reçoit tous les marqueurs de type G ou D reçus par x;

- à tout sommet x de sorte p de H_1 marqué par Pi correspond un sommet de H qui reçoit le même marqueur;

- à tout sommet typique ou individuel x de H_1 qui n'est successeur d'aucun sommet de sorte p correspond un sommet qui reçoit tous les marqueurs de type I ou T reçus par x, et qui possède la même étiquette lexicale que x dans le cas où x possède une étiquette lexicale.

Dans ces conditions, le correspondant de chaque sommet x de H_1 appartient bien à $\psi_{20}(x)$.

Toutefois, notons bien que l'on n'a procédé que par conditions nécessaires; il reste donc à vérifier que chaque solution possible ainsi déterminée par le processus de déplacement des marqueurs booléens est bien une solution effective.

<u>Remarques</u>:

1. Lorsqu'une équation admet plusieurs solutions, une telle vérification permet de distinguer entre les différentes solutions.

Donnons un exemple; soient les connaissances

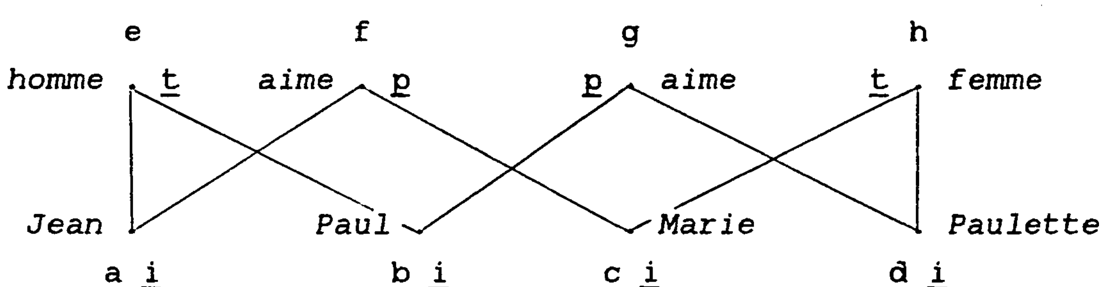
Jean aime Marie

Paul aime Paulette

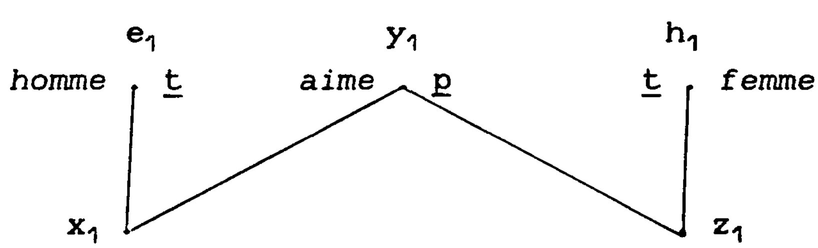
représentées par le réseau de la figure 5.21, et soit la question

quel homme aime quelle femme ?

représentée par le réseau de la figure 5.22



--------------------------<u>figure 5.21</u>--------------------------



--------------------------<u>figure 5.22</u>--------------------------

A la fin du processus, les sommets a et b de H recevront le marqueur G1, les sommets f et g recevront le marqueur P1, et les sommets c et d recevront le marqueur D1. Mais, la seule chose que l'on peut en déduire, c'est que:

Jean aime au moins une femme

Paul aime au moins une femme

il y a au moins un homme qui aime Marie

il y a au moins un homme qui aime Paulette

Le simple examen des marqueurs ne nous permet pas de savoir *qui aime qui*. Il faut donc rechercher quels sont les successeurs respectifs des sommets f et g afin de construire les deux solutions effectives sur les quatre possibles.

2. Il n'est pas exclu que la recherche d'une réponse fasse passer par un arc étiqueté par ≠. Cela signifie alors que la réponse comprend un adjectif modifié par un adverbe. Si l'on veut distinguer de telles réponses des réponses ordinaires, on procède comme suit: en phase (2) du processus de déplacement des marqueurs, si <x,y> est un arc étiqueté par ≠, si x est marqué par Ti, alors y est marqué par Si. Tout descendant d'un arc marqué par Si est également marqué par Si.

Ensuite on fait jouer aux marqueurs de type S le même rôle qu'aux marqueurs de type T, tout en ayant la possibilité de distinguer les réponses qui comprennent des S de celles qui comprennent uniquement des T.

5.4 Questions appelant des traitements particuliers

5.4.1 Questions avec des adverbes modifiant les adjectifs

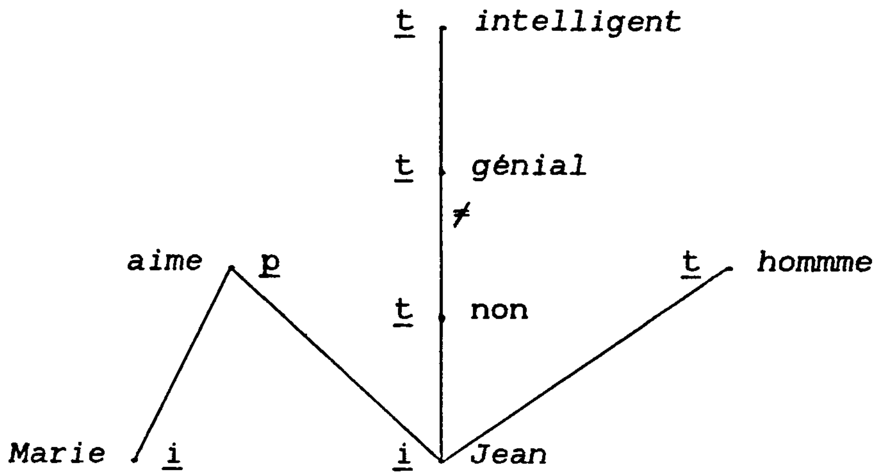
Lorsque la question comporte des adverbes modifiant les adjectifs, l'existence d'un agrandissement tel que défini précédemment ne suffit plus à déterminer qu'il y a une réponse à la question. Soient par exemple les connaissances suivantes:

Marie aime Jean

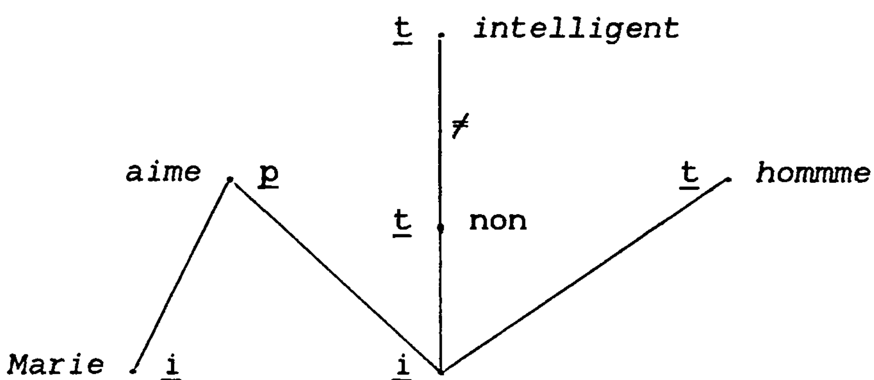
Jean est un homme qui n'est pas génial

si on est génial, on est intelligent

représentées par le réseau H de la figure 5.23



--------------------------figure 5.23--------------------------



--------------------------figure 5.24--------------------------

et soit la question

Marie aime-t-elle un homme qui n'est pas intelligent ?
représentée par le réseau H_1 de la figure 5.24.

Il est clair qu'il existe un agrandissement de H_1 dans H, et pourtant la réponse à la question ne peut être positive parce que le *non* porte uniquement sur le sommet qui lui est immédiatement ascendant. C'est pourquoi il faut ajouter la contrainte supplémentaire suivante à l'agrandissement ψ de H_1 dans H: tout arc de H_1 étiqueté par "≠" ne peut avoir pour image un chemin, mais un arc.

Autrement dit, l'agrandissement de réseaux défini en 5.3.2 doit de plus vérifier:

$\forall x,y \in X_1 \quad \delta_1^{(i)}(x) = y$ et $\rho_1^{(i)}(x) = \neq \Rightarrow$

$$\exists\ j \quad \delta^{(j)}\psi(x) = \psi(y) \text{ et } \rho^{(j)}\psi(x) = \neq$$

Il faut modifier également le processus de propagation des marqueurs booléens, en remarquant notamment que le rang de H_1 devient supérieur à 1.

Dans la phase (1), il ne suffira pas qu'un sommet typique de H admette la même étiquette lexicale qu'un sommet de H_1 pour recevoir le même marqueur Ti. Si le sommet correspondant de H_1 n'est ni initial, ni terminal, il faut vérifier que son ascendant immédiat possède le même marqueur que l'ascendant immédiat du sommet correspondant de H. Dès lors, il faudra marquer les sommets typiques de H en commençant par ceux qui correspondent à des sommets initiaux de H_1, puis en descendant dans H_1.

<u>Exemple</u>: Soit l'énoncé

Jean aime une femme belle et très intelligente

représenté par le réseau H de la figure 5.25 et soit la question

Jean aime-t-il une très belle personne ?

représentée par le réseau H_1 de la figure 5.26.

Les sommets d_1, f_1 et g_1 de H_1 reçoivent respectivement les marqueurs T1, T2, et T3. Si l'on affecte aux sommets typiques de H qui reçoivent la même étiquette lexicale que ces sommets de H_1 ces mêmes marqueurs, on confondra la femme belle et très intelligente avec *une femme intelligente très belle*. Avec la nouvelle méthode de marquage des sommets de H, le sommet g ne

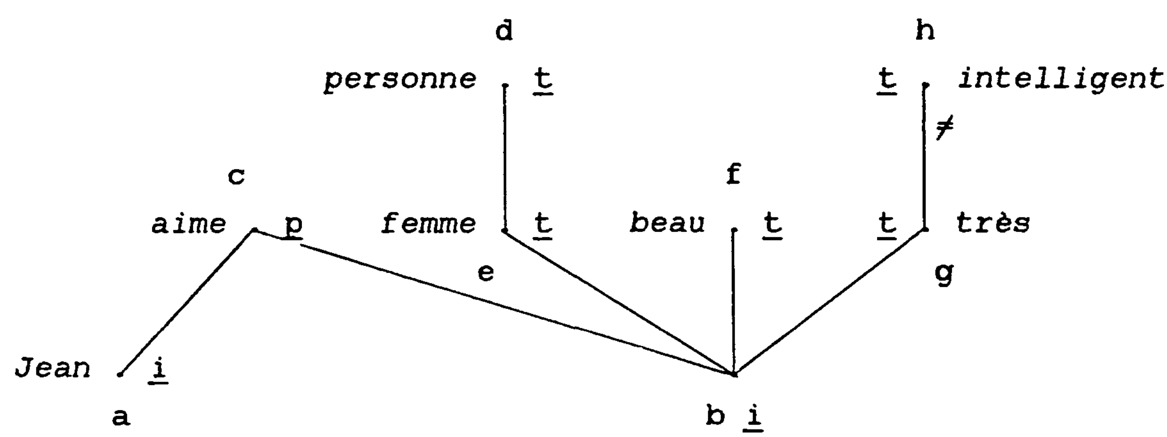
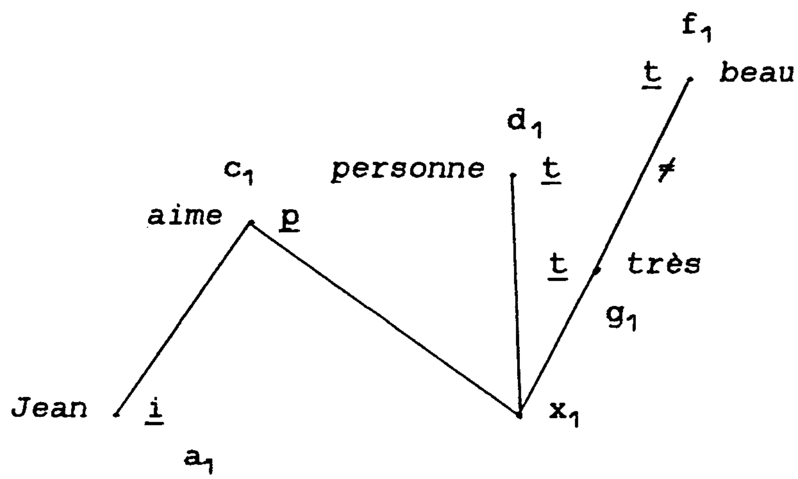


figure 5.25



---------------------------figure 5.26---------------------------

pourra recevoir le marqueur T3. En revanche, il recevrait ce marqueur si la question était

Jean aime-t-il une personne très intelligente ?

5.4.2 Questions portant sur des objets typiques complexes

Si une question porte sur des objets typiques complexes, deux problèmes se posent:

1. L'existence d'un agrandissement entre le graphe représentatif de la question et le graphe représentatif des connaissances n'atteste pas de l'existence d'une réponse à la question.

2. Il se peut très bien que l'objet typique complexe sur lequel porte la question ne soit pas représenté dans le graphe représentatif des connaissances, bien qu'il y ait une réponse à la question.

Montrons d'abord sur un exemple qu'il ne suffit pas qu'il y ait un agrandissement pour avoir une réponse; soient les connaissances:

Jean aime toutes les femmes belles et intelligentes

toute femme est une personne

représentées par le réseau H de la figure 5.27, et soit la question

qui aime toutes les belles personnes ?

représentée par le réseau H_1 de la figure 5.28.

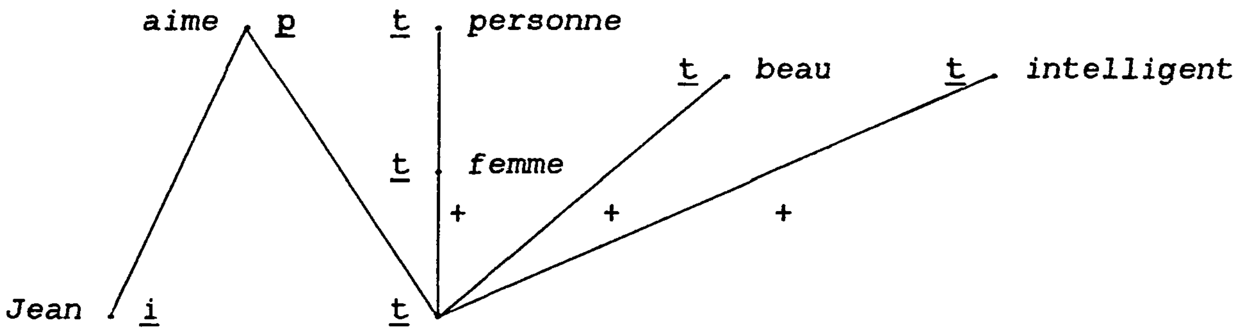


figure 5.27

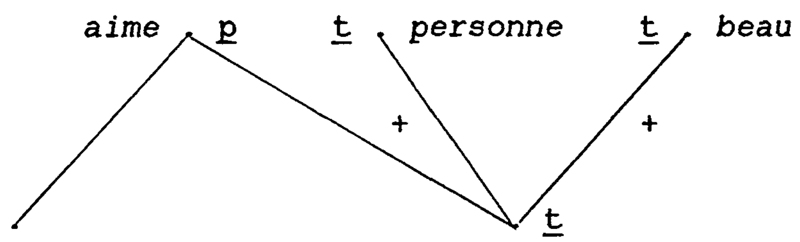


figure 5.28

Il y a bien un agrandissement de H_1 dans H, et pourtant, il n'y a pas de réponse à la question, et ce pour deux raisons. D'une part, l'amour de Jean ne porte pas sur la personne typique, mais sur la femme typique qui en descend, et d'autre part cet amour ne porte pas sur toutes les femmes belles, mais uniquement sur celles qui sont aussi intelligentes.

C'est pourquoi, il faudra retrouver dans la réponse à une question portant sur un objet typique complexe un sous-graphe définitoire du même objet typique que celui représenté dans la question. Il en était en réalité de même pour des questions portant sur des objets typiques simples, mais, dans ce cas, le sous-graphe définitoire était réduit à un point et la coincidence des sous-graphes définitoires était impliquée par l'existence d'un agrandissement qui conserve les étiquettes lexicales.

Afin de pouvoir traiter ce problème, nous allons faire l'hypothèse supplémentaire suivante: on suppose que tout sommet du graphe H qui est extrémité d'un arc étiqueté par + ne reçoit pas d'étiquette lexicale.

L'agrandissement ψ défini en 5.3.2 doit alors vérifier les conditions supplémentaires suivantes:

(i) à tout arc $<x,y>$ de H_1 doit correspondre un chemin $<x_0,x_1,\ldots,x_p>$ de H avec $p>0$, $x_0 = \psi(x)$, $x_p = \psi(y)$ et pour tout i appartenant à [p] il existe un arc $<x_{i-1},x_i>$ étiqueté par +;

(ii) si $<x_0,x_1,\ldots,x_p>$ est un chemin de H avec $p>0$, x_p étant l'image $\psi(y)$ d'un sommet de H_1, aucun des sommets $x_0,x_1,\ldots,x_{p-1}$ ne recevant une étiquette lexicale, et x_p recevant une étiquette lexicale, il faut que x_p soit l'image par ψ d'un sommet x de H_1, $<x,y>$ étant un arc de H_1 étiqueté par +;

(iii) à tout arc $<x,y>$ de H_1 extrait du sous-graphe définitoire d'un objet typique et non étiqueté par + doit correspondre un arc $<\psi(x),\psi(y)>$ de H.

Pour traiter l'aspect algorithmique de la recherche de la réponse à une question, nous allons envisager deux cas.

<u>ler cas</u>: Il n'y a pas de propositions dans la définition des objets typiques.

On se donne deux nouveaux type de marqueurs booléens:
N0, N1, N2,...
M0, M1, M2,...
et on ajoute T0 à l'ensemble ds marqueurs T1, T2,...

Comme indiqué en 5.3.3, les p sommets typiques de H_1 sont marqués respectivement par T1, T2,...,Tp, qu'ils admettent ou pas une étiquette lexicale. Les sommets de H qui recevront un marqueur de type Ti seront nécessairement des sommets qui admettent une étiquette lexicale. Il faut donc déterminer une procédure afin d'attribuer les marqueurs Ti correspondant à un sommet de H_1 qui n'admet pas d'étiquette lexicale à des sommets de H, sans quoi on ne pourra jamais trouver de réponse à la question.

Avant de passer à la phase (2), on effectue les opérations suivantes:

(1.1) On marque dans H_1 par Ni tous les successeurs y d'un sommet x marqué par Ti et tel qu'il existe un arc $<x,y>$ étiqueté par +.

Dans H on marque également par Ni tous les successeurs y d'un sommet x marqué par Ti et tel qu'il existe un arc <x,y> étiqueté par +. De plus, on marque par Ni tous les successeurs y d'un sommet x marqué par Ni et tel qu'il existe un arc <x,y> étiqueté par +.

(1.2) Tout sommet de H qui reçoit exactement le même ensemble $\{Ni_1,Ni_2,...,Ni_r\}$ de marqueurs qu'un sommet de H_1 marqué par Tj est marqué par Mj.

Soit y un sommet marqué par Mj (j>0) et <x,y> un arc de H étiqueté par +.

Si x n'admet pas d'étiquette lexicale, alors il est marqué par M0.

Si x admet une étiquette lexicale et qu'il ne reçoit aucun marqueur tel que Tk, alors il est marqué par T0.

(1.3) Comme en (1.1), on marque dans H par N0 tous les successeurs y d'un sommet x marqué par T0 et tel qu'il existe un arc <x,y> étiqueté par +. De plus, on marque par N0 tous les successeurs y d'un sommet x marqué par N0 et tel qu'il existe un arc <x,y> étiqueté par +.

Tous les sommets de H marqués par Mi et pas par N0 sont également marqués par Ti.

On peut alors passer à la phase (2) du processus.

Exemple: Soient les connaissances:

Jean aime toutes les petites fleurs

Paul aime toutes les fleurs rouges

Marie aime toutes les petites fleurs rouges

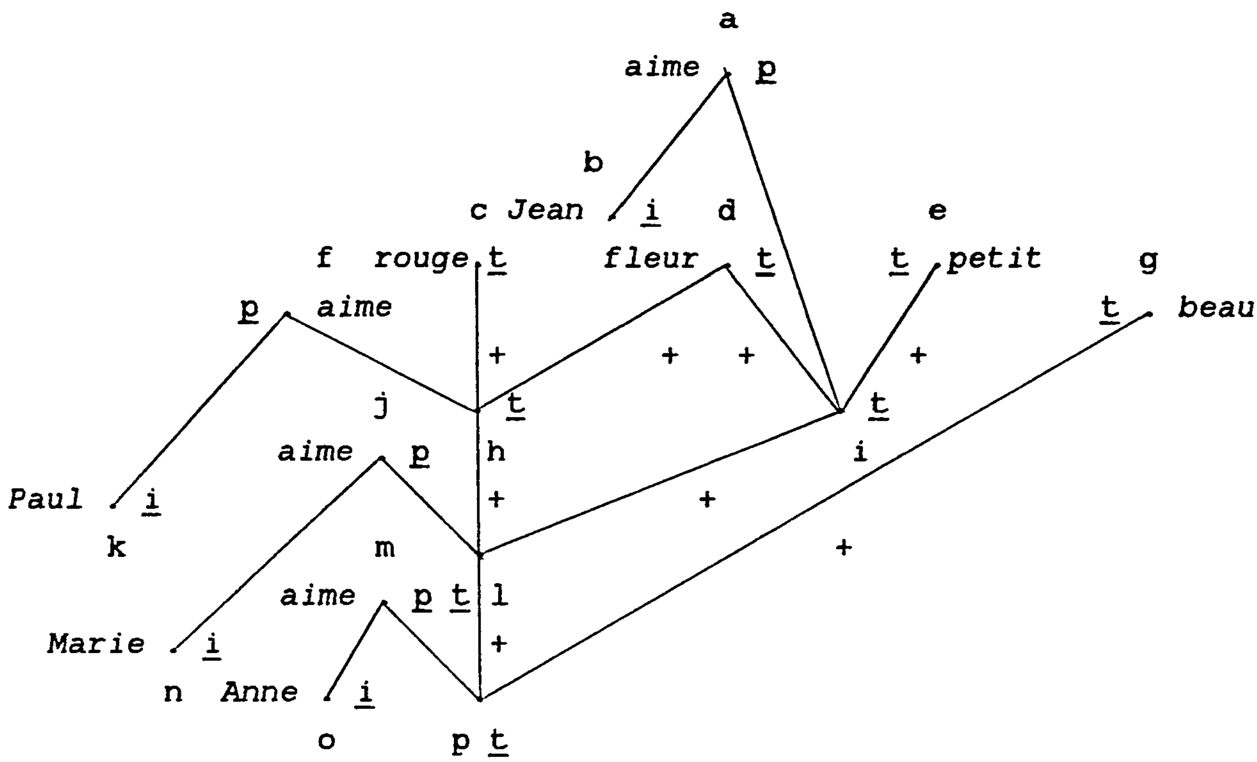
Anne aime toutes les belles petites fleurs rouges

représentées par le réseau H de la figure 5.29,
et soit la question

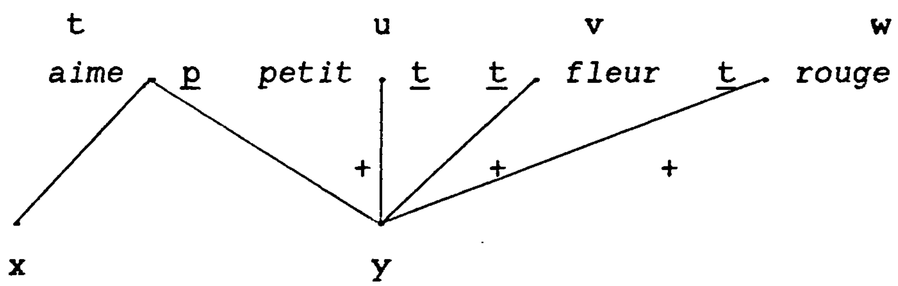
qui aime toutes les petites fleurs rouges ?

représentée par le réseau H_1 de la figure 5.30.

Les sommets u, v, w et y de H_1 sont respectivement marqués



---------------------------figure 5.29---------------------------



---------------------------figure 5.30---------------------------

par T1, T2, T3, T4.

Les sommets c, d et e de H sont respectivement marqués par T3, T2, T1.

Dans H_1, le sommet y est marqué par N1, N2 et N3.

Dans H:

- le sommet h est marqué par N2 et N3
- le sommet i est marqué par N1 et N2
- les sommets l et p sont marqués par N1, N2 et N3; ces deux sommets reçoivent par conséquent en plus le marqueur M4.

Du coup, l, h et i reçoivent le marqueur M0 etg reçoit le marqueur T0.

p reçoit donc le marqueur N0.

L est par conséquent le seul sommet de H à recevoir le marqueur T4.

On peut alors passer aux phases (2) (3) et (4) du processus qui nous permettront effectivement de trouver que seule *Marie* est solution de la question.

2ème cas: Il y a des propositions servant à définir des objets typiques.

La partie du sous-graphe définitoire contenant des propositions doit alors se retrouver telle quelle dans H. On peut dire alors que le morphisme de H_1 dans H concernant cette partie n'est plus un agrandissement. Dans ce cas, il n'est plus du tout intéressant de procéder par déplacement de marqueurs booléens. On utilise alors un algorithme de parcours de H qui reprend l'algorithme de construction du sous-graphe définitoire d'un objet typique.

Examinons maintenant le cas où l'objet typique complexe sur lequel porte la question n'est pas représenté dans le graphe représentatif des connaissances. Il est possible qu'il y ait tout de même une réponse à la question.

En effet, la combinaison distinguée considérée dans la question peut très bien n'avoir pas été considérée dans la représentation des connaissances. Reprenons l'exemple des figures 5.27 et 5.28 "à l'envers". Supposons que l'on ait les connaissances

Jean aime toutes les belles persones

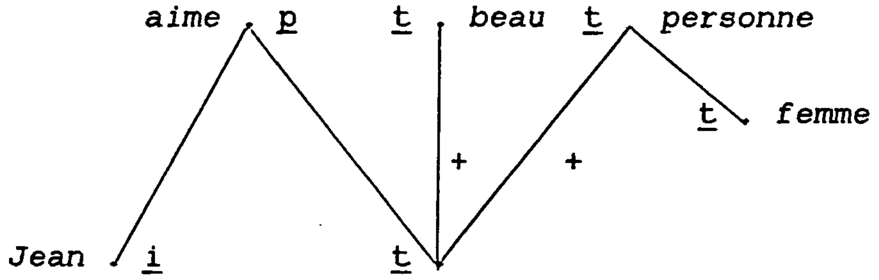
toute femme est une personne

représentées par le réseau de la figure 5.31 et la question

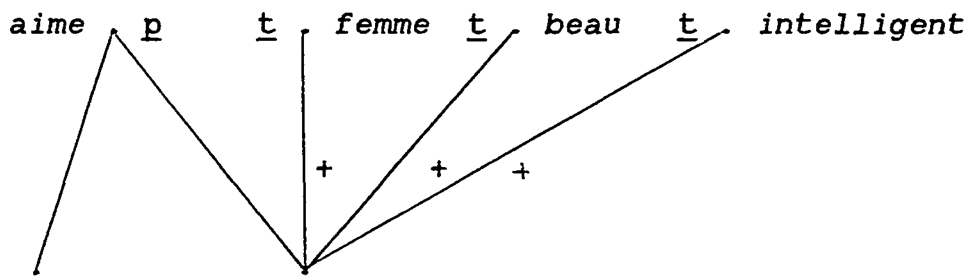
qui aime toutes les femmes belles et intelligentes ?

représentée par le réseau de la figure 5.32. Il y a bien une réponse, mais on ne peut la trouver.

C'est pourquoi il faut, avant d'appliquer les algorithmes



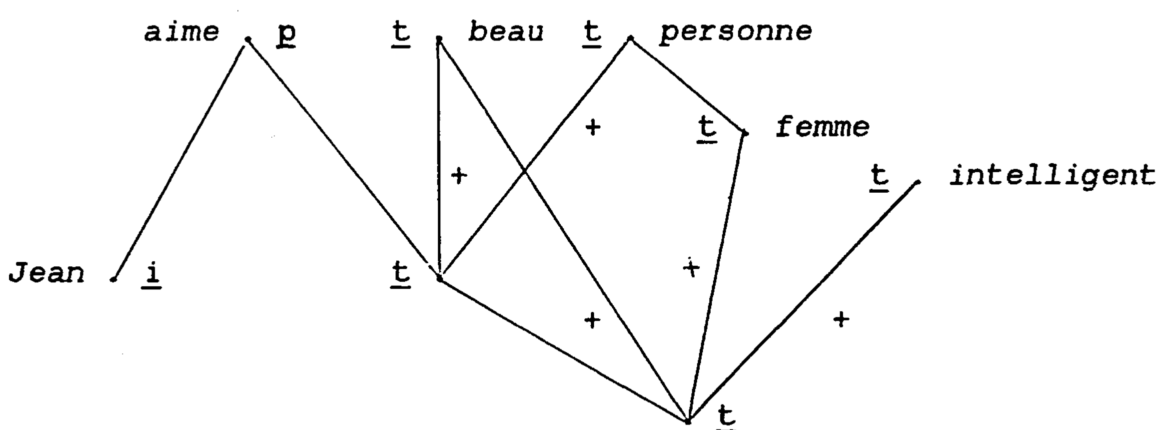
--------------------------figure 5.31--------------------------



--------------------------figure 5.32--------------------------

d'interrogation, intégrer au graphe représentatif des connaissances les combinaisons de notions présentes dans la question et non dans ce graphe, en n'oubliant surtout pas d'appliquer les règles de cohérence énoncées aux paragraphes 4.2.2 et 4.2.6.

Cela donne, sur notre exemple, un nouveau graphe H' obtenu par la modification du graphe H (figure 5.33) et on obtient la



--------------------------figure 5.33--------------------------

réponse à notre question.

Afin de montrer comment s'effectue la construction du graphe H', nous allons nous placer dans le cas où l'objet typique complexe n'est pas défini à l'aide de propositions.

On dispose d'un suite $e_1,\ldots,e_p$ de p étiquettes lexicales correspondant aux p notions constituant la combinaison à représenter.

Soient $x_{i_1},\ldots,x_{i_q}$ les sommets typiques de H qui reçoivent respectivement l'étiquette lexicale $e_{i_1},\ldots,e_{i_q}$ (on suppose que deux sommets typiques distincts de H reçoivent deux étiquettes lexicales différentes).

H' est construit en adjoignant à H les sommets x_i qui n'y sont pas (un sommet x_i a pour sorte $\underline{t}$ et pour étiquette lexicale e_i) ainsi qu'un sommet x de sorte $\underline{t}$. On aura par conséquent dans H' une suite de sommets $x_1,\ldots,x_p$. Chacun de ces sommets est relié au sommet x par un arc étiqueté par +.

La construction des arcs nécessitée par la règle de cohérence peut encore une fois être effectuée par un déplacement en parallèle de marqueurs booléens.

On se donne deux nouvelles séries de marqueurs:

A1, A2,...

B1, B2,...

(1) A chacun des sommets x_i de H' on associe le marqueur Ai.

(2) Tout ascendant de sorte $\underline{t}$ d'un sommet marqué par Ai est également marqué par Ai.

(3) Pour tout sommet u marqué par Ai et tout arc <u,v> de H' étiqueté par + et tel que v ne soit marqué par aucun Aj, on marque v par Bi à condition qu'il n'y ait pas d'arc <w,v> étiqueté par + et tel que w ne soit marqué par aucun Ak.

Pour tout sommet u marqué par Bi et tout arc <u,v> de H' étiqueté par + et tel que v ne soit marqué par aucun Aj, on marque v par Bi à condition qu'il n'existe pas un arc <w,u> étiqueté par + et tel que w ne soit marqué par aucun Aj ou Bk.

Tout sommet u de H' ($u \neq x$) marqué par au moins un Bi devra être tel qu'il y ait un arc joignant u à x étiqueté par $\subset$.

Exemple: On se donne les connaissances suivantes:

tout chien est un animal

tout caniche est un chien

Paul craint tous les chiens méchants

Jean craint tous les animaux sauvages

ainsi que la question:

qui craint tous les caniches sales et méchants ?

On rajoute au graphe H d'origine représentant les connaissances les sommets x et y ainsi que les arcs <e,x>, <c,x> et <y,x> tous les trois étiquetés par + (figure 5.34).

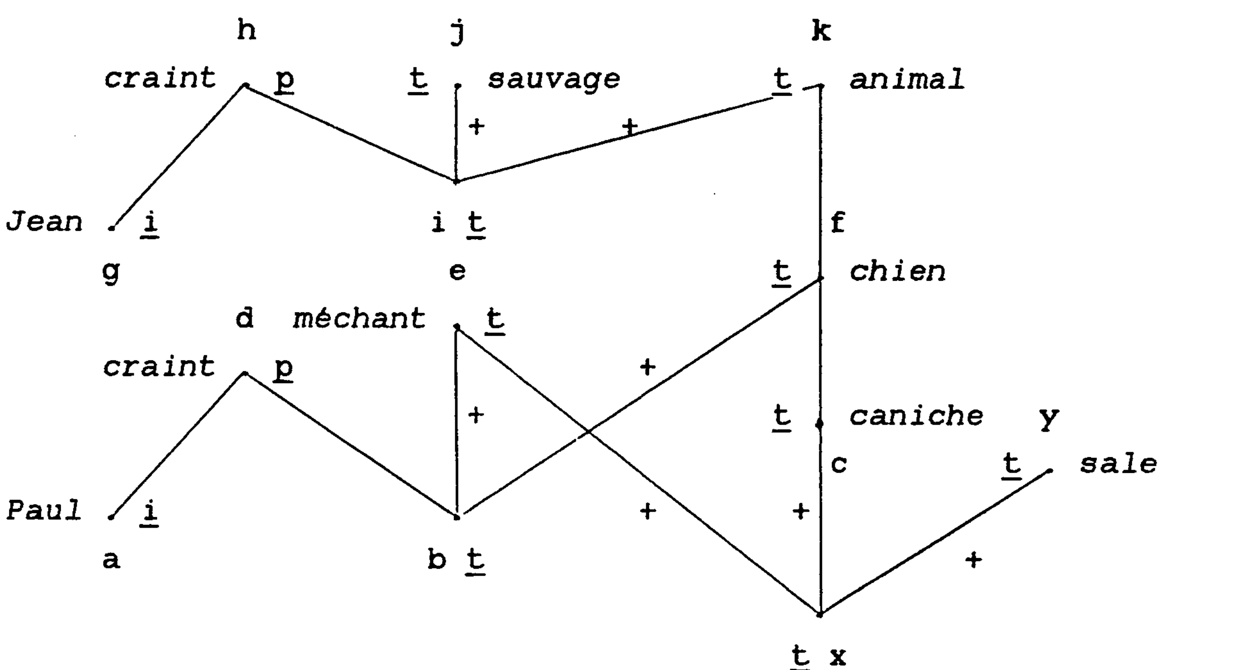


figure 5.34

Les sommets e, c et y sont respectivement marqués par A1, A2 et A3.

Les sommets f et k sont marqués par A2.

Les sommets b et x sont marqués par B1 et B2.

Le sommet x est marqué par B3.

Il faut relier le sommet b à x par un arc marqué par ⊂.

5.4.3 Les exceptions

S'il y a des contradictions dans la base de connaissances la recherche de la réponse à une question peut produire plusieurs réponses contradictoires; autrement dit, il peut y avoir plusieurs agrandissements qui répondent à la question. On peut fournir à l'interrogateur les différentes réponses possibles et le laisser se débrouiller avec. Mais on peut également sélectionner de manière algorithmique certaines réponses parmi les réponses trouvées (dans certains cas une réponse unique) en appliquant les règles définies au paragraphe 4.2.7.

On suppose que la seule modalité possible est une modalité de type *non*.

On se donne 6 nouvelles sortes de marqueurs booléens:

S1, S2,...
R1, R2,...
Q1, Q2,...
W1, W2,...
U1, U2,...
V1, V2,...

C'est la phase (2) du déplacement des marqueurs booléens (5.3.2) qui est modifiée.

(1.1) Si x est un sommet marqué par Ti, <x,y> un arc étiqueté par $\neq$, (et par conséquent y un sommet d'étiquette lexicale *non*) on marque y par Si.

Si x est un sommet marqué par Ti, <x,y> un arc étiqueté par $\subset$, on marque y par Ri.

Si x est un sommet marqué par Si, <x,y> un arc étiqueté par $\subset$, on marque y par Qi.

(2.1) Dans H, si x est un sommet marqué par Ti, <x,y> un arc étiqueté par $\subset$, on marque également y par Ti sauf s'il est déjà marqué par Qi.

Si x est un sommet marqué par Si, <x,y> un arc étiqueté par $\subset$, on marque également y par Si sauf s'il est déjà marqué par Ri.

(2.2) Si x est un sommet marqué par Ti, <x,y> un arc étiqueté par $\neq$, (et par conséquent y un sommet d'étiquette lexicale *non*), <y,z> un arc étiqueté par $\subset$, on marque z par Wi, sauf s'il est déjà marqué par Si.

On marque par Ui x et tous les sommets w déjà marqués par Ti et tels que <w,u> soit un arc étiqueté par $\subset$, u étant marqué par Ui; on marque par Vi tous les sommets v tels qu'il existe un arc <u,v>, u étant marqué par Ui.

(2.3) Si <u,v> est un arc étiqueté par $\subset$, si u est marqué par Wi, v est également marqué par Wi, sauf s'il est déjà marqué par Vi.

Finalement, si un sommet est marqué par Ti et pas par Si ni Wi, on peut en déduire qu'il vérifie bien la propriété recherchée.

S'il est marqué par Si et pas par Ti, on peut dire qu'il ne vérifie pas la propriété.

S'il est marqué par Ti et Si ou Ti et Wi, on ne peut pas conclure.

Exemple: Soient les connaissances (figure 5.35)

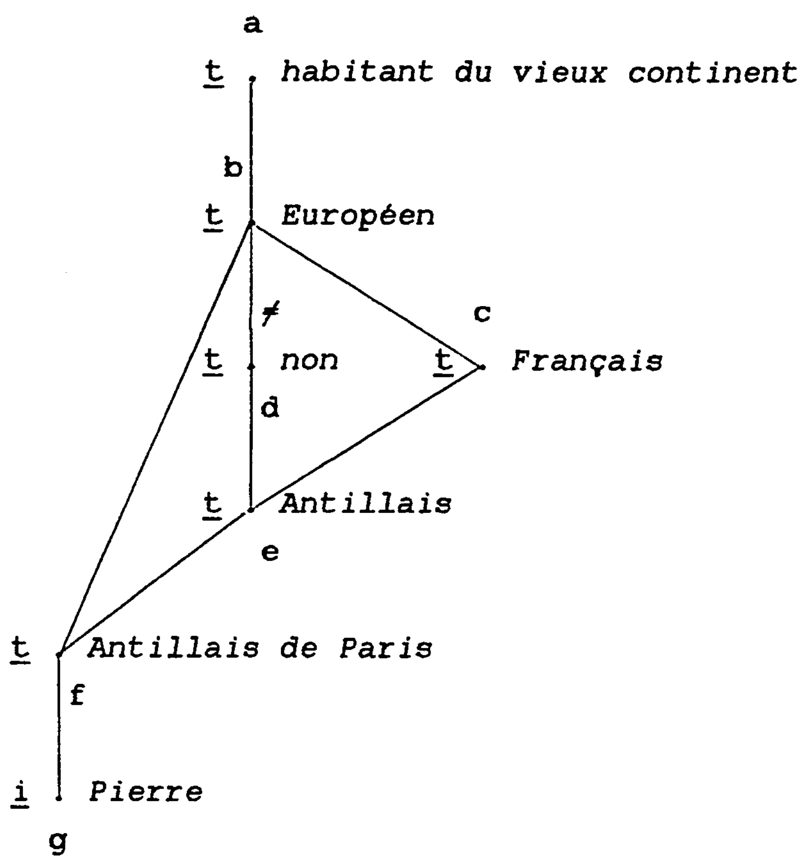
les Européens sont des habitants du vieux continent
les Français sont des Européens
les Antillais sont des Français non européens
les Antillais de Paris sont des Antillais européens
Pierre est un Antillais de Paris

et les questions (figure 5.36)

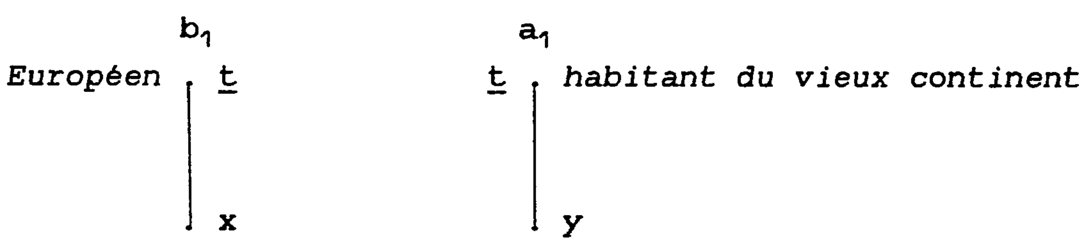
qui est européen ?
qui est un habitant du vieux continent ?

On traite les deux question en parallèle.
Dans H_1, on marque a_1 et y par T2, b_1 et x par T1.
Dans H, on marque a par T2 et b par T1.

On marque d par S1.
On marque b par R2, c et f par R1, e par Q1.
On marque c, f, g par T1, b, c, e, f et g par T2, e par S1.
b est marqué par U1, a et b sont marqués par U2.
c et f sont marqués par V1 et V2, b par V2.
e est marqué par W2, mais pas f et par conséquent pas g.



<u>figure 5.35</u>



<u>figure 5.36</u>

Finalement on peut conclure que les sommets b, c, f et g répondent aux deux question, que le sommet a répond à la deuxième question; quant au sommet e, il répond de manière négative à la deuxième question puisqu'il est marqué par S1, mais on ne peut rien conclure pour la deuxième question puisqu'il reçoit à la fois les marqueurs T2 et W2.

5.5 L'utilisation des règles de réécriture

Les connaissances sont représentées, dans le cadre du modèle plus élaboré de 4.3, par un réseau H et un ensemble de règles de réécriture de réseaux $\{<G_i,D_i>; i\in[n]\}$.

La stratégie employée est une stratégie **à profondeur variable** dans un sens voisin de celui employé dans [Coulon-Kayser, 81]. Cela signifie qu'on ne fait appel aux règles (qui effectuent une recherche à un plus grand niveau de profondeur) qu'en cas de nécessité.

De plus, l'appel aux règles ne provoque pas la construction de toutes les connaissances possibles à partir des connaissances de base. Cela serait beaucoup trop coûteux de faire toutes les transformations possibles sur le réseau avant de chercher à extraire la réponse à une question. Ce qui est effectué, c'est une transformation **virtuelle** du réseau. La question est modifiée selon les règles de réécriture. On n'effectue pas des transformations sur le réseau d'origine qui reste intact.

Pour répondre à une question représentée par un réseau H_1:

(1) On cherche dans le réseau si la réponse s'y trouve déjà (comme indiqué en 5.4) en formant l'équation $<H,H_1>$.

(2) Si non, on cherche à appliquer à la question les règles obtenues à partir des règles $<G_i,D_i>$ en échangeant G_i et D_i. Cette application se fait d'une manière non conservatrice, c'est-à-dire que la sous-structure de H_1 à l'image de D_i est supprimée avant d'être remplacée par un réseau construit à l'image de G_i. Il s'agit donc d'une QR-dérivation directe de réseaux (3.4.4).

(3) Dans le cas où il est possible d'appliquer une règle de réécriture et par conséquent de construire un réseau transformé H_2, on forme une nouvelle question $<H,H_2>$.

(4) Ce processus est répété autant de fois qu'il est nécessaire.

Nous ne donnons pas dans ce cadre de stratégie dans le choix des règles et des retours en arrière éventuels. Il suffit de s'inspirer des nombreuses stratégies qui existent pour les systèmes experts.

Un tel processus peut demander une combinatoire importante. Mais on ne peut échapper à une telle combinatoire chaque fois qu'on fait appel à des règles.

Nous allons décrire le processus d'interrogation sur deux exemples délibérément choisis très simples.

1. On part des connaissances suivantes:

Jean est né en France

Pierre a tué Jean

constituant le réseau H originel représenté en figure 5.37.

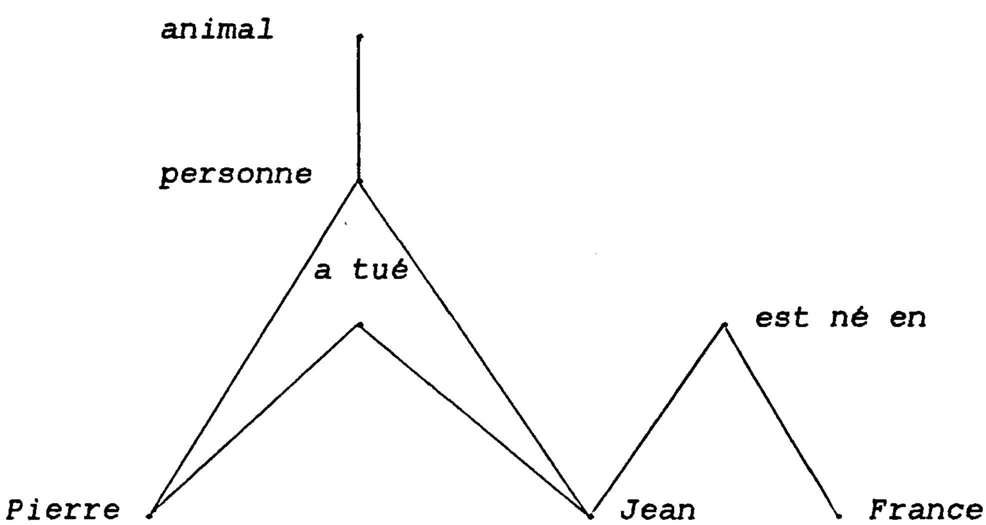


figure 5.37

et des règles:

(R1) *si une personne est née en France, alors elle est française*

(R2) *si on a tué un animal, alors il est mort*

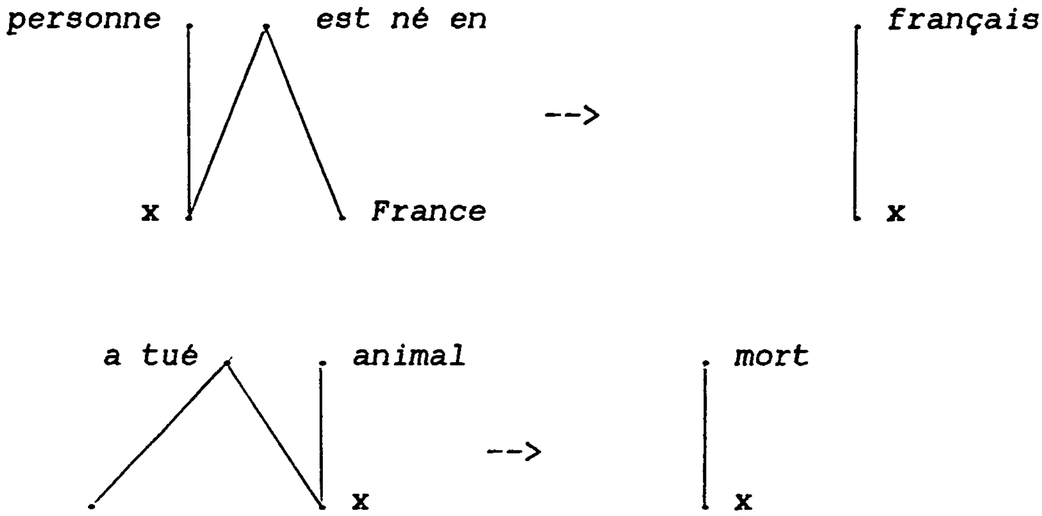
schématisées en figure 5.38.

La question posée sera la suivante:

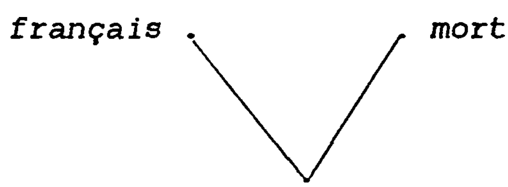
Quels Français sont morts ?

Elle est représentée par le réseau de la figure 5.39.

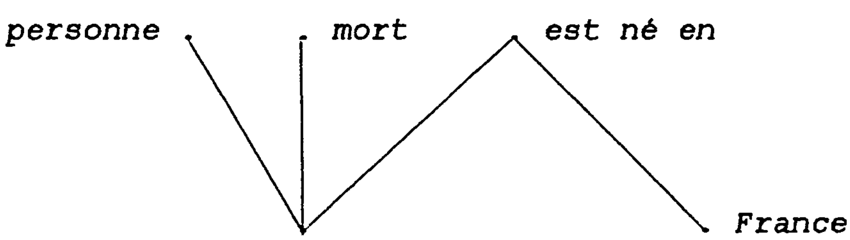
Il est clair qu'on n'en trouve pas la réponse immédiatement. L'application de la règle R1 produit la question schématisée en figure 5.40, et l'application de R2 produit celle de la question schématisée en figure 5.41 qui est une question à laquelle on



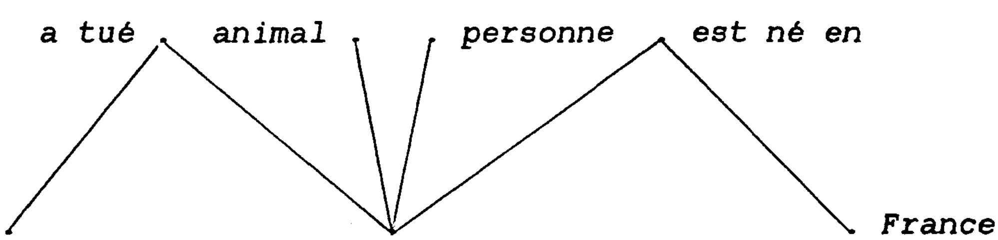
---------------------------figure 5.38---------------------------



---------------------------figure 5.39---------------------------



---------------------------figure 5.40---------------------------



---------------------------figure 5.41---------------------------

peut répondre directement.

2. On se donne les connaissances suivantes:

Marie est une personne

tout homme est une personne

toute personne a pour père un homme

représentées par le réseau H de la figure 5.42,

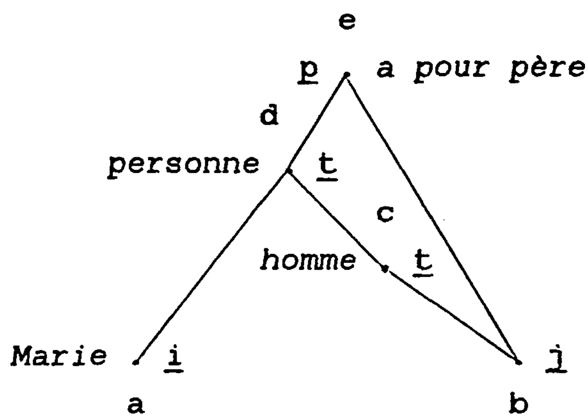


figure 5.42

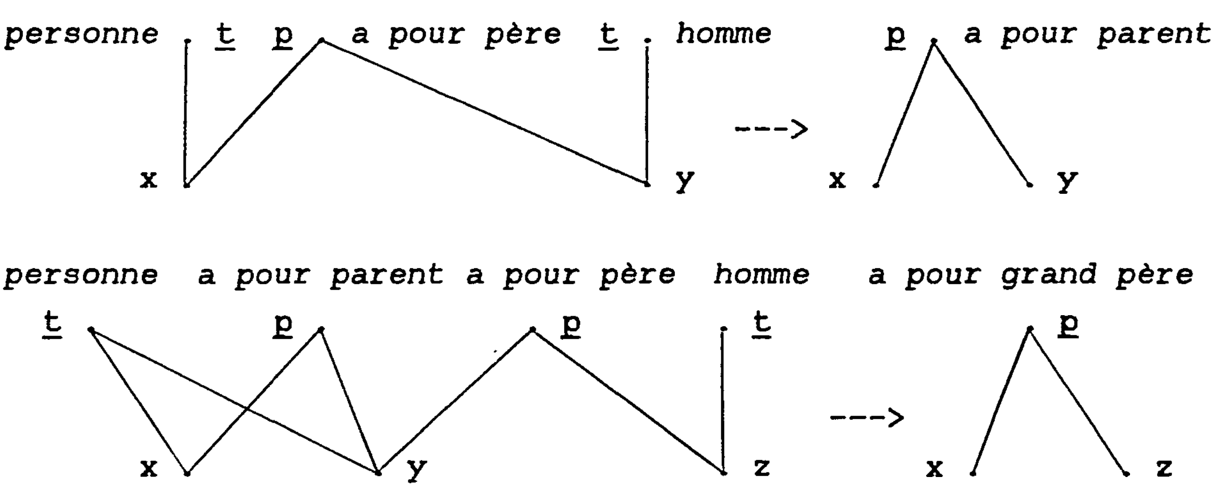


figure 5.43

ainsi que les règles

(R1) *si une personne x a pour père un homme y, alors x a pour parent y*

(R2) *si une personne x a pour parent une personne y et si y a pour père un homme z, alors x a pour grand-père z*
schématisées en figure 5.43.

On pose la question

Marie a-t-elle un grand-père ?

représentée par le réseau H_1 de la figure 5.44.

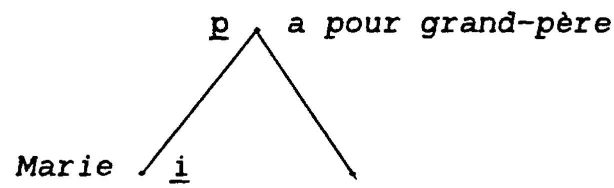


figure 5.44

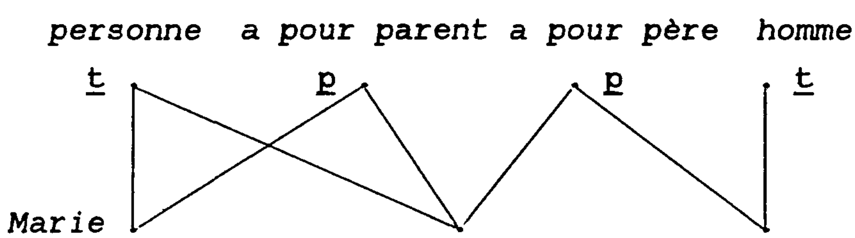


figure 5.45

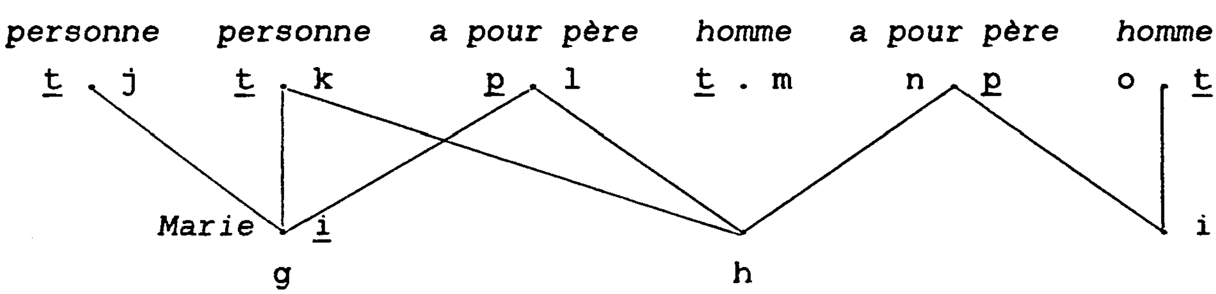


figure 5.46

x	g	h	i	j	k	l	m	n	o
$\psi(x)$	a	b	b	d	d	e	c	e	c

figure 5.47

L'application à cette question de la règle R2 produit la question transformée H_2 représentée en figure 5.45 et l'application à H_2 de la règle R1 produit la question H_3 représentée en figure 5.46, question à laquelle il existe une réponse, comme l'indique l'application ψ de la figure 5.47.

6 La construction de la représentation

6.1 Un langage quasi-naturel pour la description des connaissances

6.1.1 Les objectifs

Il s'agit de faire en sorte qu'un expert en un quelconque domaine, ignorant tout de la représentation interne sous forme de réseau, puisse communiquer au système les connaissances qu'il possède. Ces connaissances doivent alors être exprimées en un langage accessible à tous, et le réseau doit être construit automatiquement à partir de l'analyse de ce langage.

Dans la mesure où, dans les réseaux sémantiques, les connaissances sont structurées, une telle construction automatique est plus difficile à réaliser que pour les bases de connaissances formées de règles indépendantes: dans ce dernier cas, il suffit de traduire chaque connaissance en une règle.

Le plus confortable serait évidemment d'exprimer les connaissances en langue naturelle. Mais cela supposerait:

- l'existence d'un système d'analyse qui permette de traduire toutes les formes syntaxiques et de résoudre toutes les ambiguïtés;
- l'existence d'un système de représentation des connaissances qui permette de représenter tous les types de connaissances, sans aucune limite: les limites du système de représentation se reflètent forcément dans le langage de description des connaissances.

C'est pourquoi nous proposons d'exprimer les données en un **langage quasi-naturel** (ou LQN) qui soit un sous-ensemble très restreint de la langue. La syntaxe de ce langage sera contrôlée, on pourra par conséquent en donner une traduction dans le système de représentation des connaissances décrit au chapitre 4. Un tel langage a le même usage que par exemple le "jargon" de KLONE [Brachman-Schmolze , 84].

Toutefois, le langage qui va être décrit ci-après ne reprend qu'une partie de la représentation des connaissances introduite précédemment. Notre objectif en effet dans cette partie est la description de **méthodes** de construction de réseaux à partir de données en langage quasi-naturel, méthodes qui s'appuient sur des règles de réécriture de graphes.

6.1.2 <u>Les contraintes</u>

Il y a deux sortes de contraintes:

(1) des contraintes afin d'avoir un langage quasi-naturel non ambigu,

(2) des contraintes afin de construire un système qui soit suffisamment simple.

(1) Les déterminants sont classés en deux catégories. D'une part l'ensemble constitué par

tout, les, tous les, chaque,...

qui permettent d'affecter des propriétés à un objet typique, comme par exemple dans la phrase:

les coquelicots sont rouges.

On s'interdit d'utiliser à cette fin l'article défini *le* (ou *la*) comme dans

le chat aime le poisson

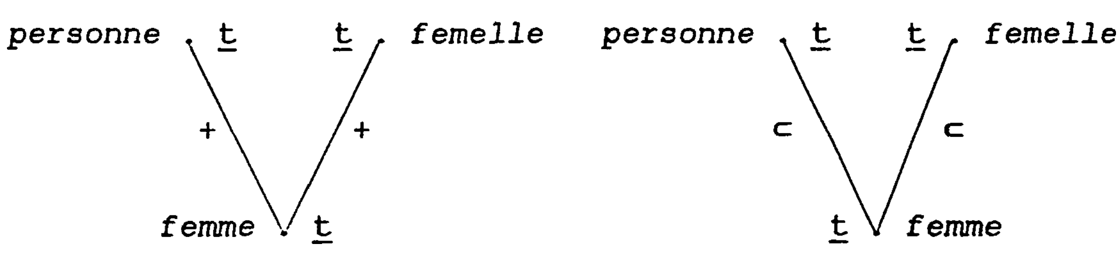
D'autre part les articles indéfinis permettent de donner la définition d'un objet typique. Par exemple:

une *femme est une personne femelle*

recevra une représentation différente de

toute femme est une personne femelle.

Dans le premier cas on définit l'objet typique *femme* alors que dans le deuxième cas on ne fait qu'en donner des propriétés, ce qui se traduit par les deux structures représentatives distinctes de la figure 6.1.



<u>figure 6.1</u>

Les articles indéfinis permettent également de marquer la quantification existentielle d'un objet, comme dans

Jean aime une femme

toute personne possède une tête

Dès lors, l'énoncé

tous les garçons aiment une fille

recevra l'interprétation

pour chaque garçon il existe une fille que le garçon aime

Si on veut indiquer que *tous les garçons aiment la même fille*, on dira plutôt:

il y a une fille que tous les garçons aiment.

Les énoncés ambigus tels que

tout homme qui possède un ordinateur qui fonctionne et qui est intelligent sait programmer

recevront arbitrairement une interprétation unique. Ici, c'est l'*homme* qui est *intelligent*, et pas l'*ordinateur*.

(2) Un nom propre représente un objet et un seul du monde étudié.

Les adjectifs représentent une notion à part entière (au même titre que les noms communs). Par conséquent, si l'on sait que tout *éléphant est un animal*, on en déduit que un *petit éléphant est un petit animal*.

Les seuls temps pris en compte sont le présent et le passé composé lorsque ces temps décrivent l'état du monde à un moment donné.

Il n'y a pas de conditions d'énonciation, ni de modalités, donc pas de verbe dont l'objet est un autre verbe.

Les liens anaphoriques sont interdits. Les seuls pronoms par conséquent autorisés sont je et tu. Ils sont analysés comme jouant le même rôle que les noms propres. Et l'article défini *le* (ou *la*) indique qu'il y a un seul objet du type donné dans le monde. Par exemple: *l'informatique*. Un tel groupe nominal joue encore le même rôle qu'un nom propre.

Les seules négations possibles sont celles qui portent sur les adjectifs, comme dans:

toute *tulipe est non noire*.

Enfin, il n'y a pas de compléments circonstanciels, ni groupes prépositionnels.

On voit que ces contraintes sont très fortes. Elles se justifient parce qu'il s'agit pour l'instant d'un système expérimental.

6.1.3 La grammaire du langage quasi-naturel

Notre objectif, dans le présent chapitre, étant de décrire des méthodes de construction de réseaux, il ne nous a pas paru nécessaire de nous donner une grammaire complexe. La grammaire du langage quasi-naturel est une grammaire hors contexte qui a été construite pour avoir la propriété d'être LALR (voir [Aho-Ullman, 77]). Cela permet d'en faire une analyse ascendante sans retour en arrière.

Ce choix a été effectué afin que l'analyse de textes en LQN soit déterministe (ce qui n'est pas vrai dans le cas de grammaires hors contexte plus générales), qu'il y ait indépendance entre les diverses règles et qu'il y ait une indépendance entre l'analyse d'une phrase et la construction de la structure représentative qui lui correspond (ce qui n'est pas vrai dans le cas des ATN). Pourtant, dans le système expérimental que nous avons réalisé, nous avons préféré, pour des raisons pratiques, utiliser un ATN (voir chapitre 7).

La grammaire utilisée est notée $G = \langle V_T, V_A, \mathcal{R}, PHR \rangle$ où V_T est un vocabulaire terminal, V_A est un vocabulaire auxiliaire, $\mathcal{R}$ est un ensemble de règles et PHR est l'axiome de la grammaire. On note $V = V_T \cup V_A$.

La grammaire de ce langage très restreint est entièrement donnée en annexe au présent chapitre.

Les éléments du vocabulaire terminal sont des classes lexicales telles que subs (pour "substantif"), vt (pour "verbe transitif"), npr (pour "nom propre") ou bien tout (qui regroupe tous les déterminants dont on a considéré qu'ils avaient la même fonction que *tout*), qui, que, ou, et, etc... On suppose qu'il existe un analyseur morphologique susceptible de découper une suite de caractères et d'attribuer à chaque unité la classe qui lui correspond. A noter les classes etqui, etque ou estun que l'on attribue aux suites "et qui", "et que" ou "est un" chaque fois qu'elles apparaissent plutôt qu'une suite de deux classes, ceci afin que la grammaire soit effectivement LALR.

L'analyse morphologique doit fournir une liste L de couples $<u_1,v_1>,\ldots,<u_m,v_m>$ où chaque u_i est une unité lexicale et chaque v_i un élément du vocabulaire terminal.

On sépare les éléments du vocabulaire terminal en deux catégories. Il y a d'une part les éléments qui apportent une information uniquement grammaticale, tels que tout, qui, un,..., pour lesquels $u_i = v_i$. D'autre part les éléments tels que subs, verb, adj,... auxquels doit être associée une unité lexicale dans le but de traduire l'expression du LQN en la représentation de la connaissance sous-jacente. Une telle unité lexicale est bien entendu inutile si l'on veut uniquement faire l'analyse syntaxique du texte en LQN.

On dira que les premiers éléments appartiennent à V_{T1} et les seconds à V_{T2}.

L'unité lexicale associée à un élément de V_{T2} sera une "forme standard". Par exemple un adjectif au masculin singulier, ou un verbe à la troisième personne du singulier du présent de l'indicatif.

6.2 Une grammaire de graphe correspondant au langage quasi-naturel

6.2.1 Les structures manipulées

Il s'agit de {p}-réseaux $H = <X,\delta,\xi,\rho>$ comme introduits en 4.2.9, mais les étiquetages ne sont pas exactement les mêmes. L'application $\xi = <\xi^1,\xi^2,\xi^3>$ vérifie:

$$\xi^1 : X^1 \dashrightarrow S$$

où $S = \{\underline{t},\underline{i},\underline{p},\underline{j},\underline{x}\}$ est l'ensemble des sortes; X^1 n'est pas forcément égal à X

$$\xi^2 : X^2 \dashrightarrow U$$

où U est un ensemble d'unités lexicales (que l'on suppose fournies par l'analyse morphologique).

$$\xi^3 : X^3 \dashrightarrow \mathbb{N}^+$$

ρ se réduisant à une seule application:

$$\rho : X \dashrightarrow \Xi^*$$

où $\Xi = \{\subset, +, \neq\}$ est un ensemble d'étiquettes d'arcs.

On a un étiquetage supplémentaire, ξ^3, constitué par des nombres entiers strictement positifs.
D'autre part, on a la sorte supplémentaire $\underline{x}$ qui est une sorte indéterminée.

Ces structures, toutefois, ne sont que provisoires. Les structures construites finalement n'utiliseront jamais la sorte $\underline{x}$ et l'étiquetage ξ^3 aura un domaine de définition vide. Il s'agira donc bien de structures telles que celles définies en 4.2.9.

6.2.2 <u>Les règles de réécriture de graphes</u>

A chaque fois que l'on effectue une réduction, on appelle une "action sémantique", comme cela se fait en compilation. Mais, dans notre cas, chaque action sémantique est une règle de réécriture de graphes, et plus précisément une règle de réécriture de réseaux $\langle H_1, H_2, \zeta \rangle$. A chaque règle de la grammaire hors contexte doit donc correspondre une règle de réécriture de réseaux. Cette règle est telle que $H_1 = \langle X_1, \delta_1, \xi_1, \rho_1 \rangle$ et $H_2 = \langle X_2, \delta_2, \xi_2, \rho_2 \rangle$ sont deux $\{\underline{p}\}$-réseaux et ζ une application de X_1 dans X_2. En effet, aucun sommet ne sera supprimé par l'application des règles de réécriture.

Si la règle de la grammaire syntaxique correspondant à la règle de réécriture de graphes est la suivante:

$$g \dashrightarrow d_1 d_2 \ldots d_p$$

où g est un élément de V_A et où $d_1, \ldots, d_p$ sont des éléments de V, soit $d_{i_1} \ldots d_{i_q}$ la suite des éléments de $V_A \cup V_{T2}$ extraite de $d_1 \ldots d_p$. On a:

$$X_1 = \{x_1, \ldots, x_q\}$$

tous les x_i étant distincts.

Le réseau H_1 vérifie d'autre part la propriété particulière suivante:

$\forall\ x \in X_1 \quad \delta(x) = \lambda$

Cela signifie que le graphe sous-jacent à H_1 est composé de q sommets isolés.

D'autre part, les applications ξ_1 et ξ_2 sont telles que:

ξ_1^1 est non définie; autrement dit: $X_1^1 = \emptyset$

ξ_2^1 est définie sur X_2^1 qui contient $X_2 \backslash f(X_1)$

X_1^1 est vide car, comme nous le verrons par la suite, le choix de l'application d'une règle sera déterminée uniquement par le troisième étiquetage. Les sortes de tous les sommets créés lors d'une transformation sont précisées. Ainsi, les sortes de tous les sommets d'un graphe représentatif de connaissances seront connues.

Les seuls sommets de X_2^1 qui ne sont pas dans $X_2 \backslash f(X_1)$ sont ceux qui sont images par ζ de plusieurs sommets distincts de X_1^1. En effet, les sortes des sommets ne sont pas modifiées par les transformations. Mais il faut bien que soient connues les sortes des sommets qui doivent fusionner lors d'une transformation.

ξ_1^2 et ξ_2^2 ont un domaine de définition vide; dans une extension de ce modèle, X_2^2 pourrait ne pas être vide. Un sommet créé par l'application d'une règle de réécriture pourrait en effet recevoir une pseudo-étiquette lexicale telle que *temps*.

ξ_1^3 est définie sur X_1 et vérifie:

$$\forall\ j \in [q] \quad \xi_1^3(x_j) = p - i_j + 1$$

ξ_2^3 est définie sur un ensemble $X_2^3 = \{y\}$ composé d'un seul sommet (qui est un sommet terminal de H_2) et vérifie:

$$\xi_2^3(y) = 1$$

sauf si g est égal à PHR, l'axiome de la grammaire. Auquel cas, X_2^3 est vide.

ρ_1 et ρ_2 sont partout définies; mais il est clair que:

$\forall\ x \in X_1 \quad \rho_1(x) = \lambda$

<u>Exemples</u>: 1. La règle syntaxique n° 8 est la suivante:

GN --> GN1 GRL

Il lui correspond la règle de réécriture de graphes $<H_1,H_2,\zeta>$ représentée en figure 6.2.

H_1

x	a_1	b_1
$\delta_1(x)$	λ	λ
$\varepsilon_1^1(x)$	non définie	
$\varepsilon_1^2(x)$	non définie	
$\varepsilon_1^3(x)$	2	1
$\rho_1(x)$	λ	λ
$\zeta(x)$	a_2	b_2

H_2

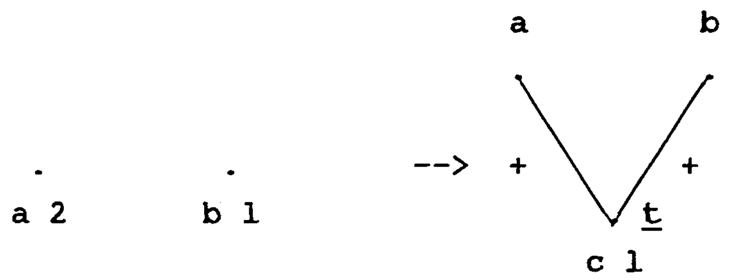
x	a_2	b_2	c_2
$\delta_2(x)$	c	c	λ
$\varepsilon_2^1(x)$	non définie		$\underline{t}$
$\varepsilon_2^2(x)$	non définie		
$\varepsilon_2^1(x)$	non définie		1
$\rho_2(x)$	+	+	λ

-------------------------<u>figure 6.2</u>-------------------------

On peut interpréter cette règle en disant que le sommet a_1 est associé à l'élément GN1 de V, que b_1 est associé à GRL et que c_2 est associé à GN.

Nous présentons les exemples d'une manière plus intuitive, par des schémas dans lesquels les sommets de H_2 reçoivent le même nom que le sommet de H_1 dont ils sont images par ζ. Cela donne, pour cet exemple-ci, le schéma de la figure 6.3.

Si un sommet est image par ζ de deux sommets de X_1, son nom est la concaténation des deux noms de ces sommets. Par exemple ab est image à la fois de a et de b, comme dans l'exemple qui suit.



-------------------------figure 6.3--------------------------

2. La règle syntaxique n° 6 est la suivante:

PHR --> npr GV

La règle de réécriture de graphes qui lui correspond est représentée en figure 6.4.

2 1 i

. . --> .

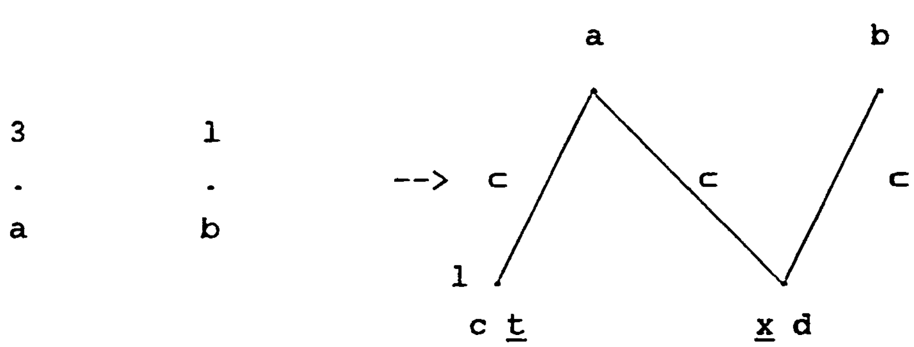
a b ab

-------------------------figure 6.4--------------------------

Cette règle est une **règle terminale** de la grammaire de graphes en ce sens qu'on ne pourra plus appliquer d'autres règles après l'avoir appliquée. Nous allons le voir dans le paragraphe qui suit.

3. A la règle syntaxique n° 29:

GV --> vt un GN



-------------------------figure 6.5--------------------------

correspond la règle représentée en figure 6.5.

4. A la règle syntaxique n° 23:

REL --> qui GV

correspond la règle de réécriture de graphes schématisée en figure 6.6.

. --> .

a l a l

figure 6.6

Une telle règle n'aura aucune action sur la construction du réseau.

En annexe, on donne toutes les règles de réécriture de réseaux.

6.2.3 L'enchaînement des règles

L'analyse morphologique fournit la liste

$L = \langle\langle u_1, v_1\rangle, \ldots, \langle u_m, v_m\rangle\rangle$

où chaque u_i est une unité lexicale (forme standard) et chaque v_i un élément du vocabulaire terminal.

L'analyse syntaxique s'effectue par avancée dans la liste L ("shift") et par réduction ("reduce"). Elle peut être schématisée en une pile P, initialement vide et par une variable entière, i, initialement nulle. A chaque fois que l'on avance dans la liste L, on incrémente i de 1 et on empile l'élément v_i de V_T dans P. Quand on effectue une réduction correspondant à une règle syntaxique telle que $A \dashrightarrow \alpha$, on dépile P de $|\alpha|$ éléments et on empile A. Lorsqu'il y a reconnaissance, la pile P doit contenir uniquement PHR, l'axiome de la grammaire, et i doit être égal à m.

La construction du réseau H représentatif de la connaissance apportée par le texte s'effectue parallèlement à l'analyse syntaxique. Elle requiert l'existence d'une pile, Q, qui a toujours (sauf au moment de la reconnaissance) la même longueur que P et donc qui est initialement vide. Initialement, le réseau H peut déjà contenir des connaissances. Il est modifié au fur et à mesure de l'analyse.

Quand on avance dans la liste L, pour un i donné:

- si $v_i \in \{vt, vi, adv\}$ on crée un sommet x qui reçoit pour étiquettes:

$\varepsilon^1(x)$ = p pour vi et vt, $\varepsilon^1(x)$ = t pour adv,

$\varepsilon^2(x) = u_i$ dans les deux cas,

$\varepsilon^3(x)$ n'étant pas défini.

Ce sommet n'est relié à aucun autre sommet du réseau en construction. On empile le nom du sommet ainsi créé dans Q;

- si $v_i \in \{subs, npr, adj\}$ on vérifie s'il existe un sommmet de H qui reçoit déjà u_i comme deuxième étiquette. Sinon, on crée un sommet qui reçoit pour étiquettes:

$\varepsilon^1(x)$ = t pour subs et adj, $\varepsilon^1(x)$ = i pour npr,

$\varepsilon^2(x) = u_i$ dans les deux cas,

$\varepsilon^3(x)$ n'étant pas défini.

Dans les deux cas, on empile le nom du sommet dans Q;

- si v_i est une des autres catégories, c'est-à-dire s'il appartient à V_{T1}, on n'effectue aucune action, si ce n'est d'empiler dans Q le symbole "£".

Quand on effectue une réduction, la règle syntaxique correspondant à la réduction (règle numéro j) est de la forme:

$$g \dashrightarrow d_1 d_2 \ldots d_p$$

On retire p éléments du haut de la pile P comme de la pile Q. Mais, auparavant, on étiquette les sommets correspondant aux p éléments de la pile Q de la manière suivante: si le k ième élément retiré est le nom d'un sommet, on étiquette ce sommet par k (troisième étiquetage). Si c'est le symbole "£", on ne fait aucune action.

Puis, on applique la règle de réécriture de graphe n° j, selon des procédures qui vont être décrites en 6.2.4. Si cette règle est terminale on vérifie simplement que la pile Q est vide. Sinon on empile le nom du sommet dont le troisième étiquetage est 1. Et on supprime ce troisième étiquetage.

Au moment où l'on effectue une des réductions numéro 1, 2, 5, 25 ou 30 on remplace toutes les sortes $\underline{x}$ présentes dans le réseau en construction par la sorte $\underline{j}$. Les objets quantifiés existentiellement dépendent bien d'un objet quantifié universellement.

En revanche, si l'on effectue une des réductions 3, 4 ou 6 $\underline{x}$ est remplacé par $\underline{i}$. Il n'y a pas d'objet quantifié universellement dont dépendent les objets quantifiés existentiellement.

Comme toutes les règles terminales provoquent un remplacement de $\underline{x}$ par une autre sorte, il est clair que $\underline{x}$ ne figurera pas en tant qu'étiquette du réseau finalement construit.

6.2.4 <u>L'application des règles</u>

Soit R = $<H_1,H_2,\zeta>$ une règle de réécriture de graphes à appliquer telle que $X_1 = \{x_1,\ldots,x_q\}$.

Au moment où on doit appliquer cette règle, il existe dans H, par construction, un ensemble unique de sommets $\{y_1,\ldots y_q\}$, tous distincts et vérifiant:

$$\forall\ i \in [q] \quad \xi_1^3(x_i) = \xi^3(y_i)$$

Il existe donc forcément un QR-morphisme ψ et un seul de H_1 dans H qui soit un morphisme de GME, et ce QR-morphisme a la propriété d'être injectif.

Si l'on veut appliquer la règle de transformation, il faut toutefois que les conditions relatives aux transformations de graphes étiquetés (3.4.1) soient vérifiées, de même que les conditions relatives aux graphes ordonnés (3.3.1) pour les sommets de sorte $\underline{p}$; il faut enfin ne pas risquer d'introduire de cricuits lors de l'application de la règle.

ψ étant injective, ψ' l'est aussi (propriété 3.1.3). Donc $Cl_2(x)$ se réduit, pour tout x, à une classe composée d'un seul élément. Par conséquent la condition (i) de 3.4.1 est vérifiée pour les 3 étiquetages.

La condition (ii) de 3.4.1 est évidemment vérifiée pour le troisième étiquetage, puisque ε_1^3 est définie sur tout X_1. Elle est également vérifiée pour le premier étiquetage puisque si deux sommets x et y de H_1 ont une image commune z par ε_2^1, alors z appartient à X_2^1. Enfin, elle est vérifée pour le deuxième étiquetage parce que les règles de réécriture de graphes que nous avons sont telles que si deux sommets de H doivent fusionner, alors au moins un des deux ne possède pas d'étiquette lexicale.

Les conditions (i) et (iii) de 3.3.1 sont vérifiées parce qu'il n'y a pas de suppression de sommets par les transformations.

γ étant injective, la condition (iv) est vérifiée.

Enfin, les conditions (ii) et (v) sont vérifiées pour les sommets de sorte p parce que l'on n'empile jamais dans Q le nom d'un sommet de sorte p qui n'est pas terminal dans H.

Comme toutes les règles ne sont pas libres, il y a le risque d'introduire des circuits en appliquant une règle. Toutefois, toutes les règles non libres ne sont pas susceptibles d'introduire des circuits. Les seules règles qui risquent d'introduire des circuits sont les règles 1, 3 et 4. Elles correspondent à l'introduction de connaissances circulaires du genre:

toute chose est un machin
tout machin est un truc
tout truc est une chose

Il faut donc vérifier, avant d'appliquer les règles en question que l'on n'introduit pas de circuit.

Cette vérification s'effectue très simplement en recherchant, lorsque l'on veut établir une liaison du sommet x au sommet y de H, s'il n'existe pas déjà un chemin de x à y.

C'est dans ce seul cas qu'une règle sémantique ne peut être appliquée, alors que la réduction syntaxique correspondante est appelée.

<u>Exemple</u>: La phrase exprimée en langage quasi-naturel est la suivante:

Jean aime une *très belle femme qui habite Paris*

L'analyse morphologique fournit la liste:
<npr,*Jean*>,<vt,*aime*>,<un,*un*>,<adv,*très*>,<adj,*beau*>,<subs,*femme*>,
<qui,*qui*>,<vt,*habite*>,<npr,*Paris*>

pour i = 1 à i = 5 on crée les sommets a, b, c, et d et on obtient le réseau provisoire, fait uniquement de sommets isolés, de la figure 6.7.

---

a	b	c	d
Jean . <u>i</u>	*aime* . <u>p</u>	*très* . <u>t</u>	*beau* . <u>t</u>

--------------------------<u>figure 6.7</u>--------------------------

pile P	pile Q
npr	a
npr vt	a b
npr vt un	a b £
npr vt un adv	a b £ c
npr vt un adv adj	a b £ c d

pour i = 5 on effectue les réductions numéros 15, 16 et 12

réduction n° 15 (et n° 12): action nulle

réduction n° 16: l'action correspondante consiste à relier par un arc les sommets c et d (figure 6.8).

---

a	b	d
Jean . <u>i</u>	*aime* . <u>p</u>	*beau* . <u>t</u>
		≠
		c *très*

--------------------------<u>figure 6.8</u>--------------------------

pour i = 6 on crée le sommet e, puis on effectue la réduction 11 qui conduit à créer un nouveau sommet, f, qui est de sorte t mais ne reçoit pas d'unité lexicale et à le relier aux sommets c et e (figure 6.9).

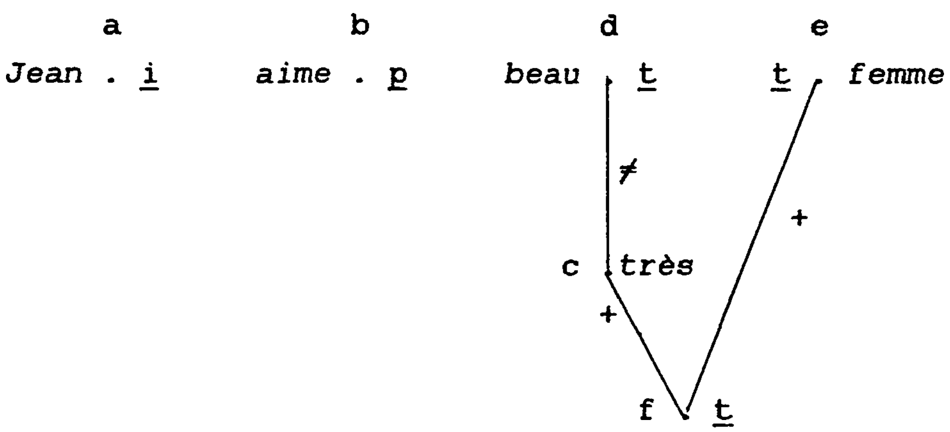


figure 6.9

pile P	pile Q
npr vt un adv GAD1	a b £ c d
npr vt un GAD1	a b £ c
npr vt un GAD	a b £ c
npr vt un GAD subs	a b £ c e
npr vt un GN1	a b £ f

Pour i = 7 à i = 9 on crée les sommets g et h. Les dernières réductions sont toutes effectuées pour i = 9

npr vt un GN1 qui	a b £ f £
npr vt un GN1 qui vt	a b £ f £ g
npr vt un GN1 qui vt npr	a b £ f £ g h
npr vt un GN1 qui GV	a b £ f £ i
npr vt un GN1 REL	a b £ f i
npr vt un GN1 GRL	a b £ f i
npr vt un GN	a b £ j
npr GV	a k
PHR	

réduction 31: création d'un sommet, établissement de 2 liaisons (figure 6.10).

réduction 23: aucune action sur le réseau, mais il y a une action sur la pile

réduction 17: action nulle

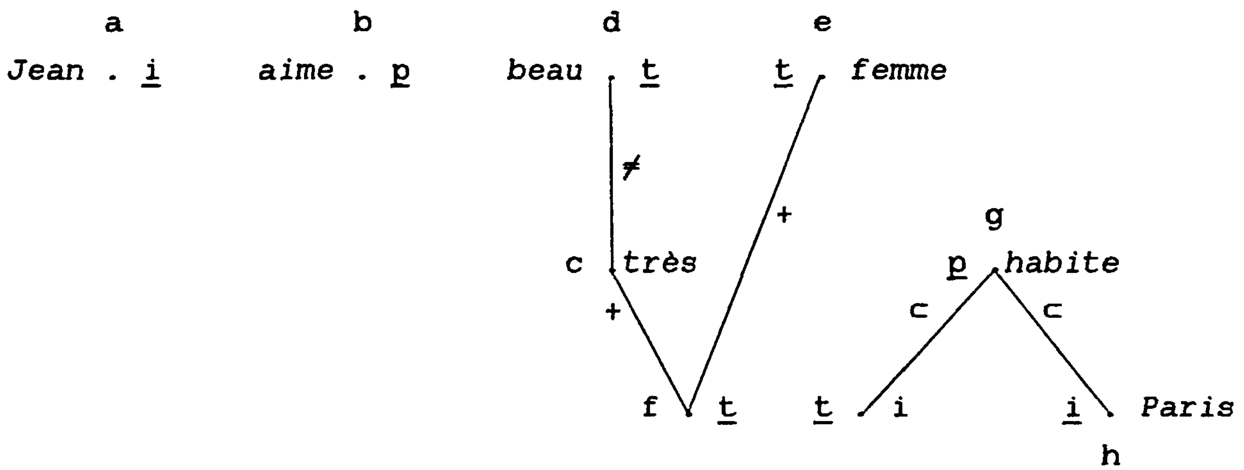


figure 6.10

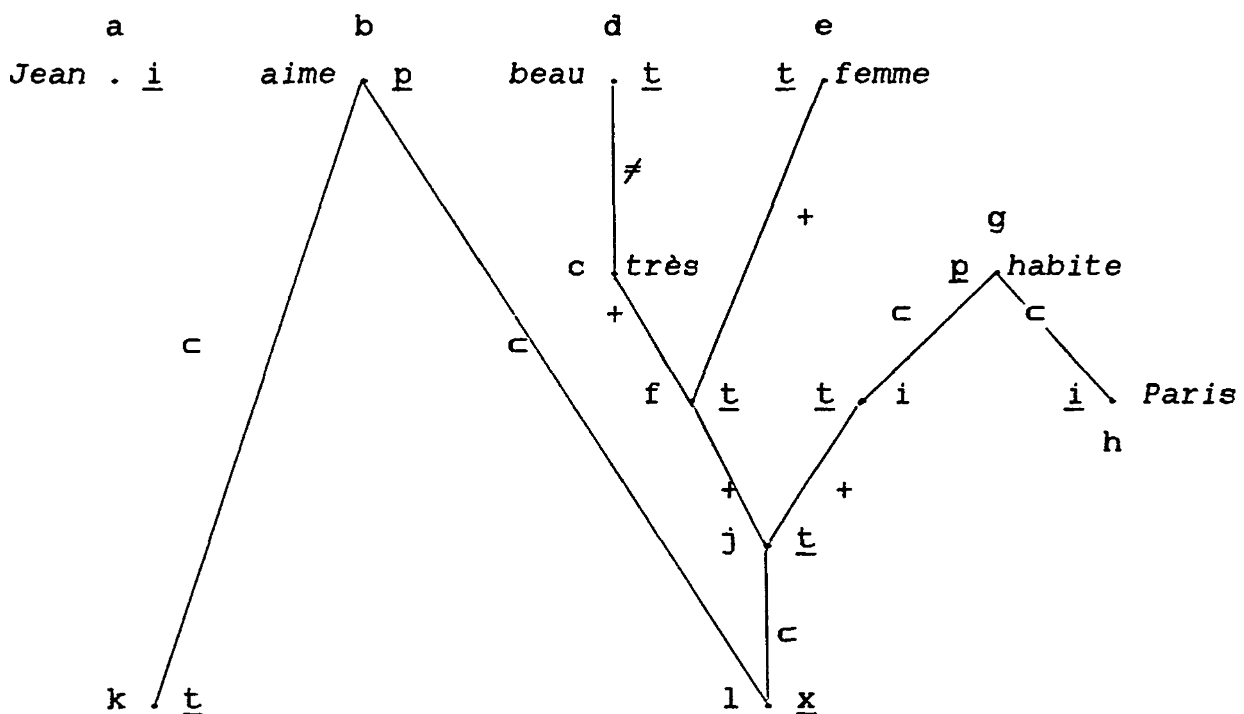
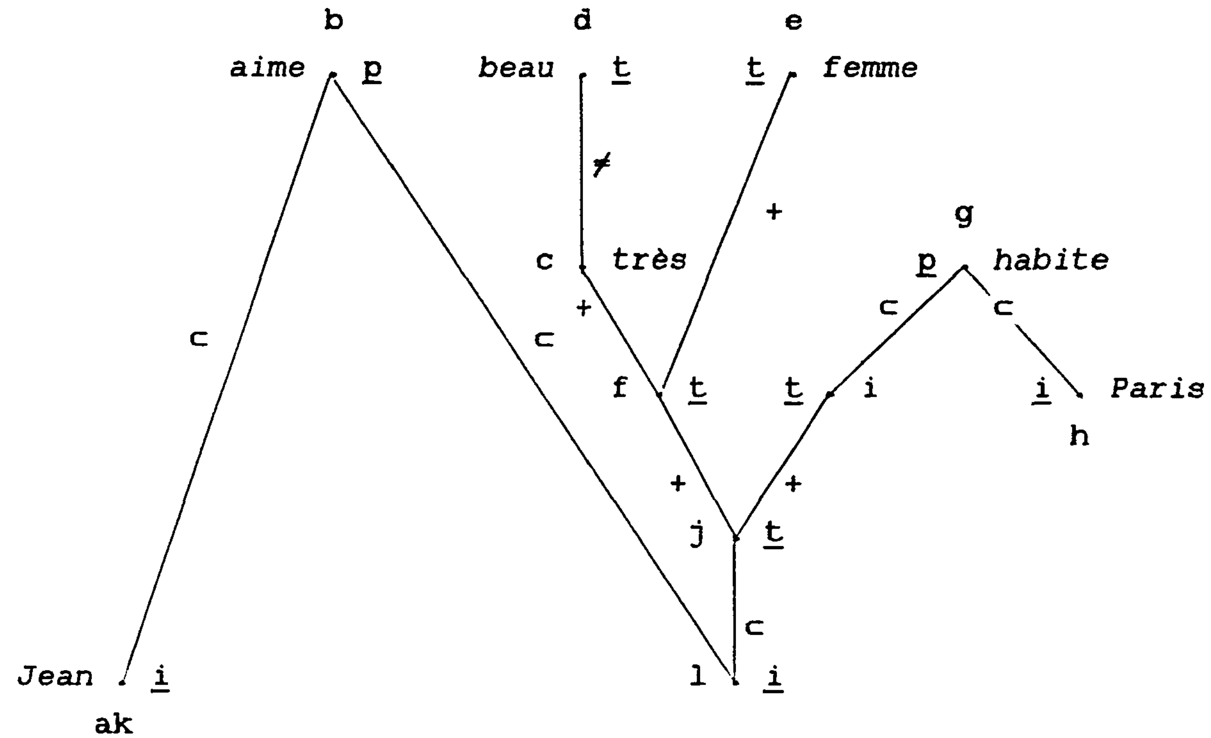


figure 6.11

réduction 8: création du sommet j de sorte <u>t</u> et établissement de deux liaisons

réduction 29: création du sommet k de sorte <u>t</u> et du sommet l de sorte <u>x</u> (figure 6.11).

réduction 6: l'action, terminale, crée le réseau définitif de la figure 6.12.



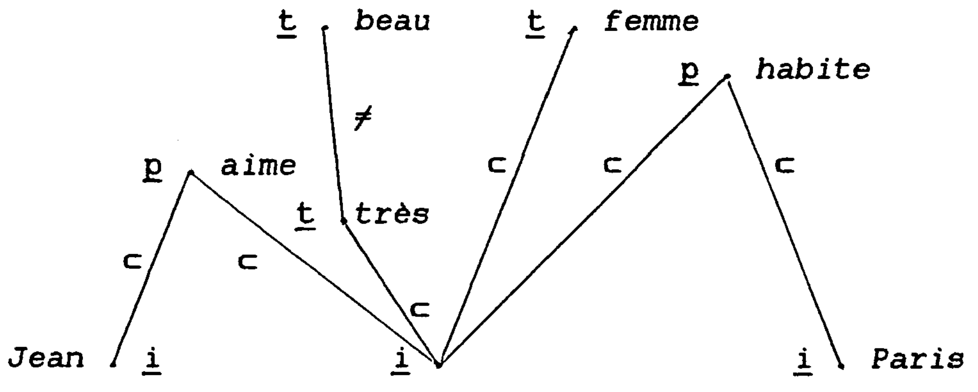
<u>figure 6.12</u>

On s'aperçoit que la sorte <u>x</u>, en raison de l'exécution de cette dernière réduction, est remplacée par la sorte <u>i</u>.

<u>Remarque</u>: La structure construite après analyse d'une phrase en LQN est un sous-réseau total (2.6.4) du réseau représentatif des connaissances. En effet, les seuls sommets de sorte <u>p</u> de cette structure sont introduits à l'occasion de cette construction et tous leurs successeurs appartiennent à cette sous-structure.

6.3 <u>Les simplifications</u>

On constate que le réseau ainsi obtenu est beaucoup plus complexe que le réseau que nous construirions intuitivement selon les indications du chapitre 4 (figure 6.13).



--------------------------figure 6.13--------------------------

Afin de parvenir tout de même à de telles structures, nous nous donnons des **règles de simplification** qui sont des règles de réécriture de graphes d'un nouveau type.

6.3.1 Les règles de simplification

Il s'agit de règles de réécriture de réseaux que l'on peut noter $\langle H_1,H_2\rangle$ puisqu'elles ne suscitent pas de fusion de sommets. On a:

$$H_1 = \langle X_1,\delta_1,\xi_1,\rho_1\rangle$$
$$H_2 = \langle X_2,\delta_2,\xi_2,\rho_2\rangle$$

Les règles de simplification sont présentées schématiquement en figure 6.14.

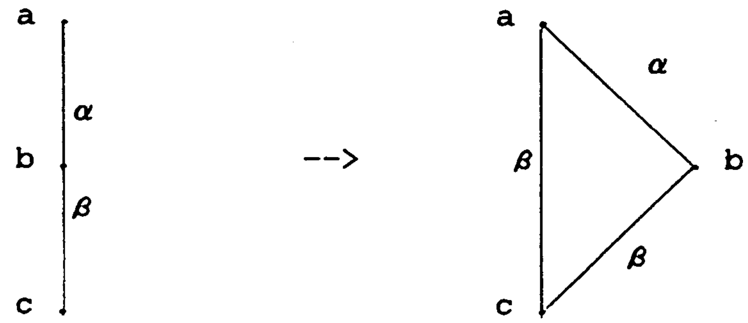
ξ_1^1 et ξ_2^1 ont un domaine de définition vide parce que les sortes n'interviennent pas du tout dans les simplifications.

ξ_1^2 et ξ_2^2 ont toutes les deux pour domaine de définition {b} et sont telles que:

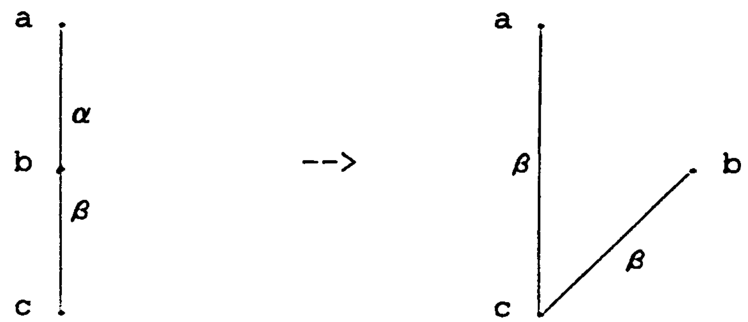
$$\xi_1^2(b) = \xi_2^2(b) = \pi$$

parce que l'on ne supprime un sommet intermédiaire dans la liaison entre deux sommets que si ce sommet intermédiaire ne possède pas d'étiquette lexicale.

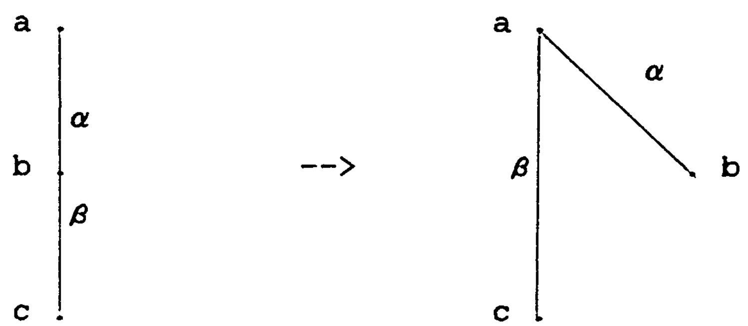
règle n°1 :



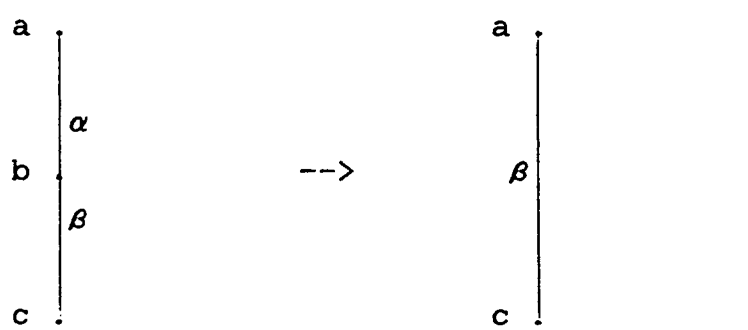
règle n°2 :



règle n°3 :



règle n°4 :



---------------------------<u>figure 6.14</u>---------------------------

Les étiquettes d'arcs, α ou β sont égales soit à +, soit à $\subset$. Cela veut dire qu'en réalité il y a 16 règles de réécriture, chacune des règles 1 à 4 représentant en fait 4 règles.

On constate que les parties gauches des règles sont toutes identiques (aux étiquettes d'arcs près). Dès lors, si on peut appliquer une règle de simplification, on peut également, indifféremment, en appliquer 3 autres. Mais, cela ne sera pas le cas, car on va donner des conditions restrictives d'application de ces règles qui détermineront laquelle des 4 règles applicables à un moment donné il faudra appliquer.

6.3.2 <u>L'application des règles</u>

Pour pouvoir appliquer une règle de simplification à un réseau $H = \langle X,\delta,\xi,\rho\rangle$ il faut qu'il existe une application ψ de $\{a,b,c\}$ dans X telle que, si $\psi(a) = x$, $\psi(b) = y$ et $\psi(c) = z$, y ne possède pas d'étiquette lexicale et il existe dans H un arc $\langle x,y\rangle$ étiqueté par α et un arc $\langle y,z\rangle$ étiqueté par β (ce qui correspond aux R-morphismes de GME conservant l'étiquetage des arcs). Il s'agira donc d'une R-dérivation directe de réseaux.

Toutefois, le choix de la règle qui va être appliquée lorsqu'il existera un tel R-morphisme sera déterminé par les conditions qui suivent:

- la règle 1 sera appliquée si y a un autre prédécesseur que x et un autre successeur que z;
- la règle 2 sera appliquée si y a un autre prédécesseur que x et pas d'autre successeur que z;
- la règle 3 sera appliquée si y a un autre successeur que z et pas d'autre prédécesseur que x;
- la règle 4 sera appliquée si y n'a pas d'autre successeur que z et pas d'autre prédécesseur que x.

<u>Exemple</u>: On part du réseau obtenu par la construction de l'exemple précédent et qui est représenté en figure 6.12.

Si on veut appliquer une des règles de manière à ce que x = c, y = f et z = j, c'est la règle n° 2 qu'il faut appliquer

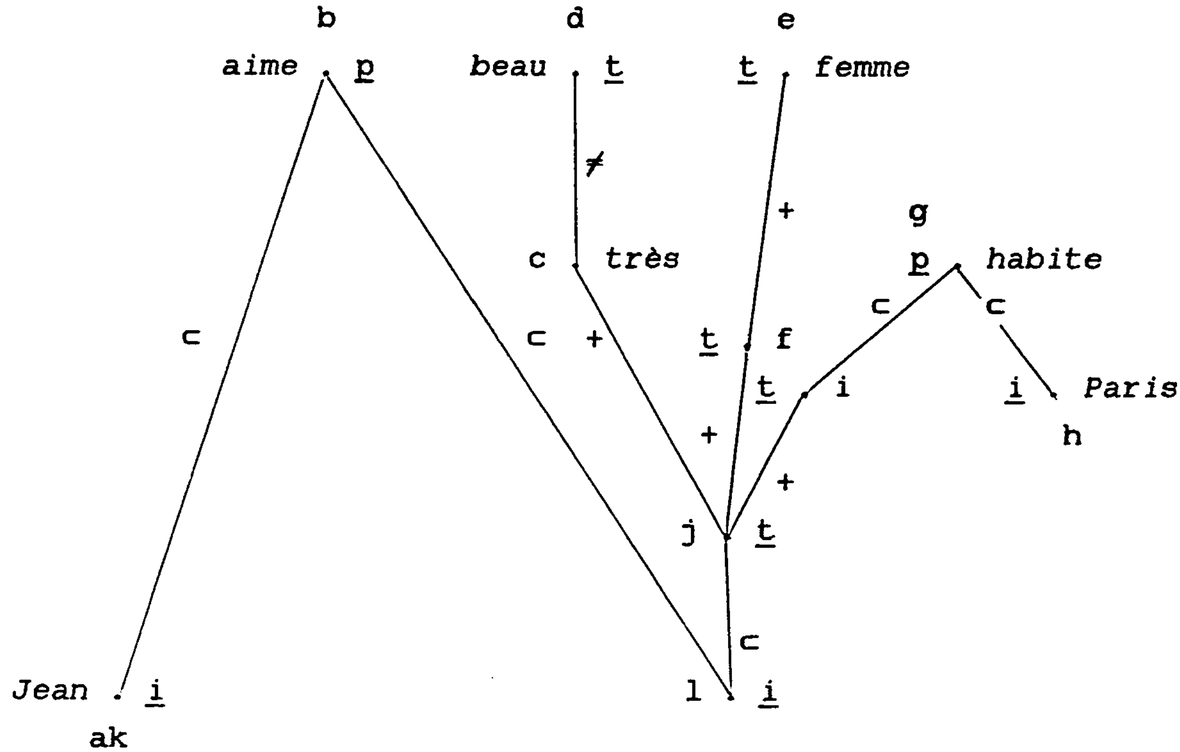


figure 6.15

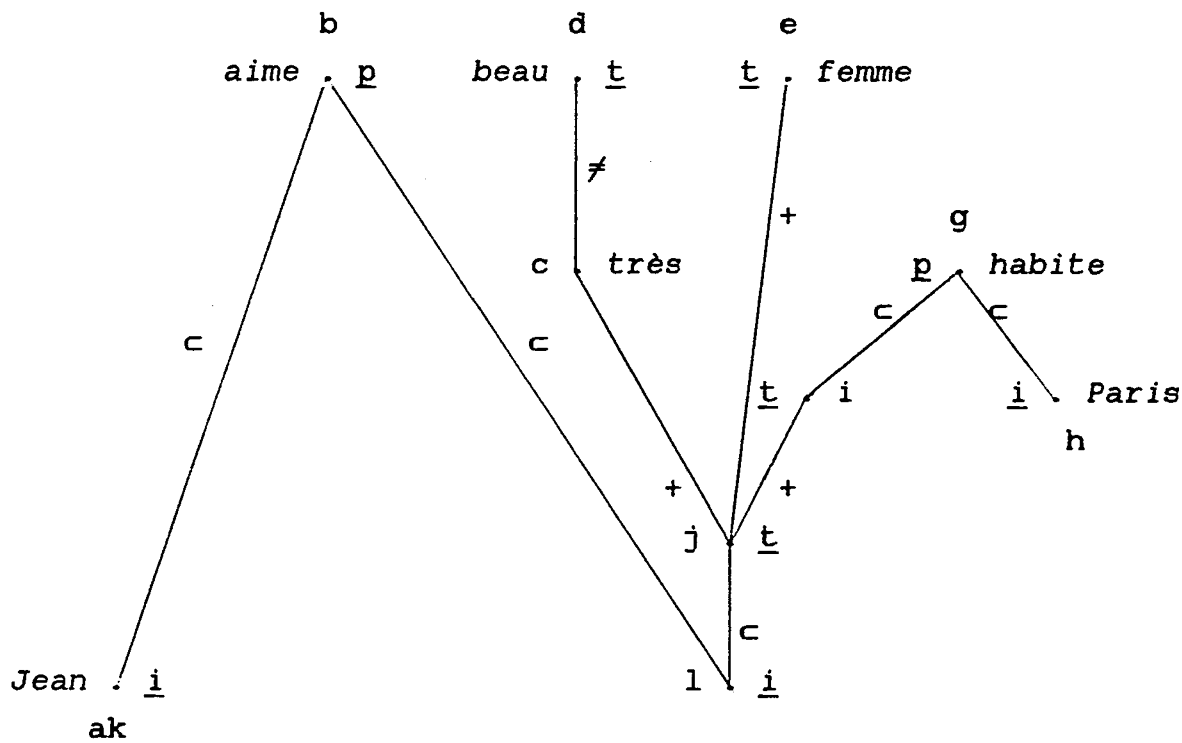


figure 6.16

et on obtient le réseau transformé de la figure 6.15.

Si on applique de nouveau une règle de simplification au réseau obtenu par transformation pour $x = e$, $y = f$ et $z = j$, c'est la règle n° 4 qu'il faut appliquer, et on obtient le réseau de la figure 6.16.

Remarque: On applique les règles de simplification uniquement au sous-réseau de H qui vient d'être construit et qui en est un sous-réseau total. Sans cela on risque de détruire des connaissances représentées, notamment après l'application des règles de cohérence (6.3.3).

Lemme 1: Soit $H = \langle X, \delta, \xi, \rho \rangle$ un réseau et x et y deux éléments de X^2. Soit $\langle x_0, x_1, \ldots, x_{p-1}, x_p \rangle$ un chemin de H tel que p soit un nombre strictement positif, $x_0 = x$, $x_p = y$ et:

$$\forall\ i \in [p-1] \qquad x_i \in X \backslash X^2$$

Si H' est le transformé de H par une règle de simplification (dans les conditions énoncées ci-dessus), alors il existe dans H' un chemin $\langle y_0, y_1, \ldots, y_{q-1}, y_q \rangle$ tel que $y_0 = x$, $y_q = y$, q étant un nombre strictement positif égal à p ou à p-1, l'arc $\langle y_{q-1}, y_q \rangle$ recevant la même étiquette que l'arc $\langle x_{p-1}, x_p \rangle$ de H et:

$$\forall\ i \in [q-1] \qquad x_i \in X \backslash X^2$$

Preuve: Si aucun sommet $x_1, \ldots, x_{p-1}$ n'est image de b par ψ, alors aucun arc constituant le chemin $\langle x, \ldots, y \rangle$ de H ne peut être détruit dans la transformation.

Sinon, il existe i appartenant à [p-1] tel que $\psi(b) = x_i$. Il y a trois possibilités:

(1) $\psi(a) = x_{i-1}$ et $\psi(c) = x_{i+1}$

Quelle que soit la règle appliquée, il existera dans H' un arc $\langle x_{i-1}, x_{i+1} \rangle$ (application des règles de réécriture dans le cas où il n'y a pas de fusion de sommets 3.1.4); il existera bien un chemin entre x et y, et si $x_{i+1} = y$, l'arc $\langle x_{i-1}, y \rangle$ recevra bien la même étiquette, β, que l'arc $\langle x_i, y \rangle$ de H.

(2) $\psi(a) \neq x_{i-1}$

Soit $v = \psi(a)$; x_i a un autre prédécesseur que v, c'est donc soit la règle 1, soit la règle 2 qu'il faut appliquer. Mais, que ce soit l'une ou l'autre de ces règles, l'arc $<x_i,x_{i+1}>$ est conservé, avec son étiquette, et $<x_{i-1},x_i>$ n'est pas touché dans la transformation.

(3) $\psi(a) \neq x_{i+1}$
Soit $w = \psi(c)$; on ne peut appliquer que la règle 1 ou la règle 3. Dans les deux cas le chemin de H est conservé dans H'.

Définition: Soit H un réseau obtenu par construction à partir d'un texte écrit en LQN. H' est un réseau simplifié de H si et seulement si il est obtenu par une suite (éventuellement vide) d'applications de règles de simplification et s'il est tel qu'on ne peut lui appliquer aucune des règles de simplification.

Lemme 2: Soit $H = <X,\delta,\xi,\rho>$ un réseau et x et y deux éléments de X^2. Soit $<x_0,x_1,\ldots,x_{p-1},x_p>$ un chemin de H tel que p soit un nombre strictement positif, $x_0 = x$, $x_p = y$ et:

$$\forall\ i \in [p-1] \qquad x_i \in X \backslash X^2$$

Si H' est un réseau simplifié de H alors il possède un arc $<x,y>$ qui a pour étiquette l'étiquette de $<x_{p-1},y>$ dans H.

Preuve: D'après le lemme 1, tout réseau dérivé de H admettra un chemin $<x,y_1,\ldots,y_{q-1},y>$ avec $1 \leqslant q \leqslant p$.

Si q est différent de 1, alors on peut appliquer au moins une des règles de simplification.

Lemme 3: Etant donné un réseau H obtenu par construction après analyse d'une phrase en LQN, tout sommet de H qui ne possède pas d'étiquette lexicale a au plus un successeur.

Preuve: Cela se voit en examinant les différentes règles de réécriture servant à construire le réseau.

Propriété: Etant donné un réseau H obtenu par construction après analyse d'une phrase en LQN, il existe un et un seul réseau H' simplifié de H.

Preuve: Soit x un sommet de H qui possède une étiquette lexicale. On a $\delta(x) = x_1 \dots x_m$.

D'après le lemme 3, pour tout i de [m] il existe un chemin et un seul $\langle x_0^i, x_1^i, \dots, x_{p-1}^i, x_p^i \rangle$ de H tel que p soit un nombre positif ou nul, $x_0^i = x$, $x_1^i = x_i$ et $x_p^i = y_i$ soit ou bien un élément de X^2, ou bien un élément de Ter(H) et:

$\forall$ j $\in$ [p-1] $\quad x_j^i \in X \backslash X^2$

D'après le lemme 2, le réseau H' sera obtenu en supprimant tous les sommets non terminaux de H qui ne possèdent pas d'étiquette lexicale et en posant pour un élément x qui possède une étiquette lexicale:

$\delta'(x) = y_1 \dots y_p$

Il est clair qu'un tel réseau est unique pour un H donné.

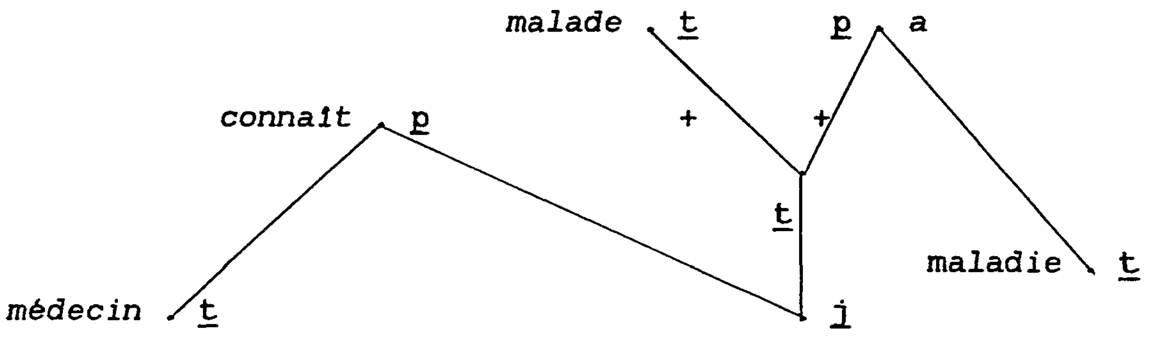
Exemple: Le réseau simplifié du réseau de la figure 6.12 est bien le réseau de la figure 6.13.

Remarque: Il est un cas où, en appliquant une règle de simplification, on introduit une erreur quant à la liaison entre un objet quantifié existentiellement et l'objet typique dont il dépend. Plus précisément c'est dans le cas où les sommets x, y et z, images de a, b et c par ψ sont tels que $\xi^1(x) = \underline{p}$, $\xi^1(z) = \underline{j}$, où le deuxième successeur, w, de x est tel que $\xi^1(w) = \underline{t}$ et où l'arc <x,y> est étiqueté par +.

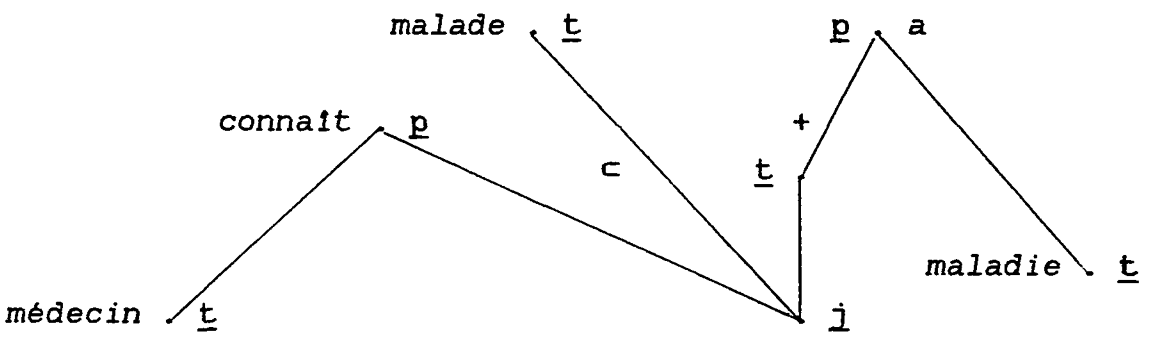
Ce cas se retrouve par exemple lorsque l'on analyse la phrase en LQN:

chaque médecin connaît un malade qui a toutes les maladies dont le réseau représentatif, avant simplification, est en figure 6.17.

Si on appliquait sans restriction les règles de simplification, on obtiendrait une structure dans laquelle le sommet de sorte $\underline{j}$ dépenderait, selon l'algorithme de 4.2.5, des deux objets typiques *médecin* et *maladie*. Avec la restriction à l'application des règles on obtient pour réseau simplifié le réseau de la figure 6.18.



-------------------------figure 6.17-------------------------



-------------------------figure 6.18-------------------------

Cette restriction à l'application des règles de simplification ne change rien au résultat fondamental énoncé plus haut.

6.3.3 L'application des règles de cohérence

Comme la structure construite par les règles de réécriture de réseaux, puis simplifiée par les règles de simplification est insérée dans un réseau de connaissances déjà existant, il faut respecter certaines règles de cohérence. Ces règles ont été énoncées en 4.2.2 et en 4.2.6 et un algorithme de propagation de marqueurs permettant d'assurer le respect des règles de cohérence a été donné en 5.4.2.

Nous allons simplement donner ici un exemple montrant la succession des procédures à mettre finalement en oeuvre pour la mise à jour d'un réseau à partir d'un texte en langage quasi-naturel.

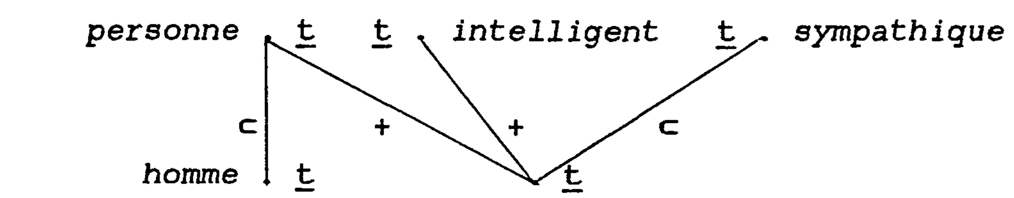


figure 6.19

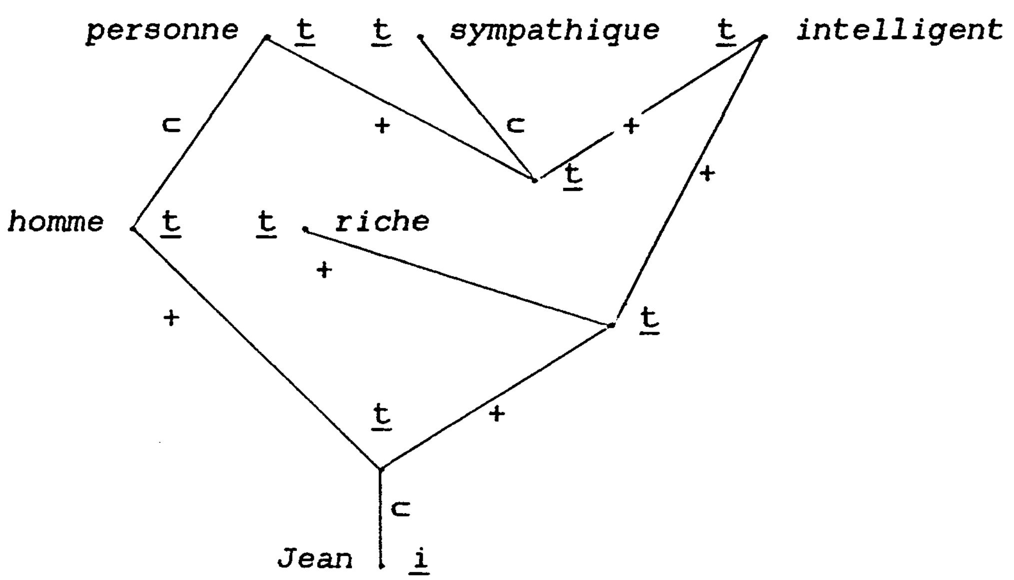


figure 6.20

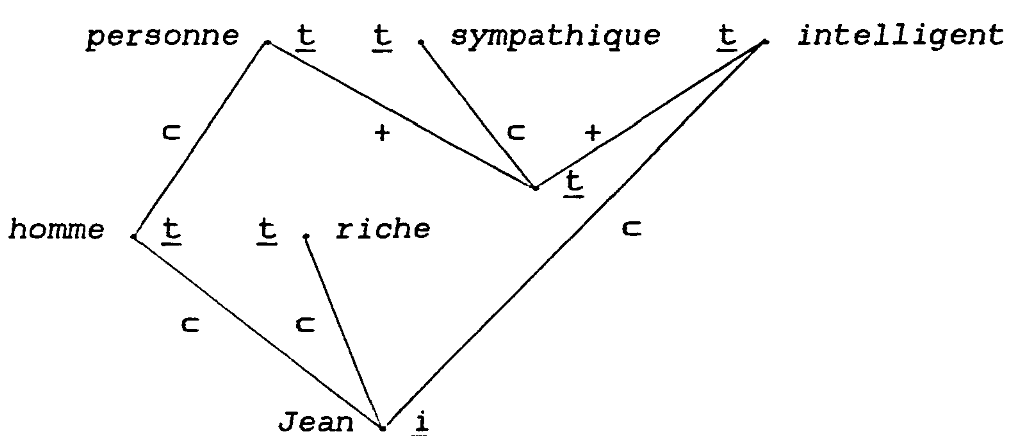


figure 6.21

Exemple: Supposons que l'on ait déjà (figure 6.19) un réseau représentatif des connaissances:

Tout homme est une personne

toutes les personnes intelligentes sont sympathiques
Si l'on veut ajouter la connaissance

Jean est un homme riche et intelligent
on obtiendra par construction le réseau de la figure 6.20, par simplification le réseau de la figure 6.21, et par application des règles de cohérence le réseau de la figure 6.22.

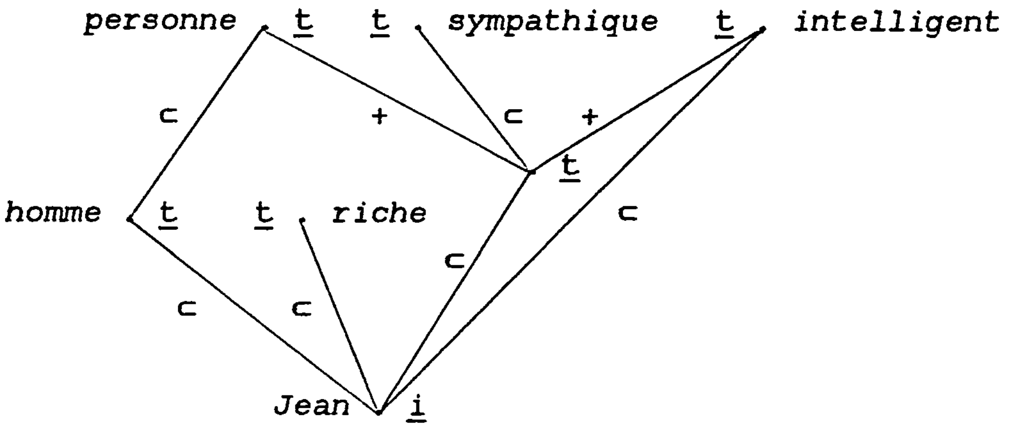


figure 6.22

6.4 La suppression de données

Il est spécialement difficile, avec les réseaux sémantiques, de supprimer des connaissances du système après qu'elles lui aient été fournies. En effet même si, comme c'est le cas ici, on introduit les différentes connaissances de manière indépendante, ces différentes connaissances sont arrangées entre elles: à chaque fois que l'on introduit une connaissance nouvelle, le système la structure avec les connaissances anciennes, des déductions sont effectuées. Lorsqu'il faut oublier une connaissance, il y a par conséquent d'autres connaissances qu'il faut oublier avec.

Certaines connaissances ne pourront pas être supprimées du réseau représentatif des connaissances; ce sont les connaissances telles que:

une femme est une personne femelle
En effet, si on avait introduit auparavant une autre connaissance telle que

les femmes sont intelligentes

on ne saurait distinguer cette connaissance de

les personnes femelles sont intelligentes

et par conséquent on ne saurait à quelle entité affecter la propriété d'être intelligent après dissociation des sommets *femme* et *personne femelle*.

Dans d'autres cas, on risque de ne pas vraiment réussir à supprimer du réseau la connaissance que l'on veut supprimer. Reprenons l'exemple du paragraphe précédent. Supposons que l'on veuille retirer du réseau la connaissance que l'on vient d'y introduire, c'est-à-dire:

Jean est un homme riche et intelligent

Il est tout à fait possible de supprimer les arcs liant les sommets *homme*, *riche* et *intelligent* au sommet *Jean*. Mais il restera encore la liaison entre le sommet *personne intelligente* et le sommet *Jean*, liaison qui a été obtenue précédemment par application de la règle de cohérence. Il faudrait par conséquent, à chaque fois que l'on supprime des liaisons, appliquer les règles de cohérence "à l'envers" afin de vérifier si l'existence des arcs qui partent d'un sommet représentant un objet typique complexe est justifiée.

Par ailleurs, le fait d'admettre que des connaissances contradictoires puissent coexister au sein d'un même réseau - afin de permettre de traiter les exceptions - interdit de supprimer les connaissances déduites d'une connaissance que l'on veut supprimer. Si l'on veut par exemple supprimer d'un réseau la connaissance

tous les Antillais sont des Européens

même sachant que

tous les Antillais sont des Français

il n'y a pas de raison de supprimer la connaissance

tous les Français sont des Européens

De même, si l'on veut supprimer la connaissance

il y a une personne qui aime Marie

cela ne conduit pas forcément à supprimer la connaissance

il y a un homme qui aime Marie

ou la connaissance

Jean aime Marie

même si l'on sait, directement ou indirectement, que

Jean est une personne

En effet, qu'est-ce qui nous permet d'affirmer a priori qu'il n'y a pas d'exceptions à la règle

tout *homme* est *une personne* ?

Et pourquoi faudrait-il supprimer une connaissance dont il n'est pas dit explicitement qu'il faut la supprimer?

Dans ce qui suit nous allons décrire une méthode de suppression des connaissances ayant les caractéristiques suivantes:

- la connaissance à supprimer doit être exprimée en langage quasi-naturel;
- c'est cette connaissance et pas une connaissance déduite qui est supprimée;
- on supprime des connaissances dans un réseau auquel les règles de cohérence n'ont pas encore été appliquées.

Afin de supprimer des connaissances du réseau, on applique à ce réseau des règles de réécriture qui sont construites automatiquement à partir de la connaissance à supprimer exprimée en langage quasi-naturel.

Dans ce but, on remplace d'abord la grammaire du LQN par une grammaire équivalente: celle obtenue en supprimant la règle 2 et en remplaçant dans les règles 5 et 6 GV par les différentes parties droites des règles 27, 28, 29, 30 et 31.

A chacune des nouvelles règles syntaxiques ainsi obtenues on associe des nouvelles règles de réécriture de réseaux par composition des règles de réécriture de réseaux précédentes. Par exemple à la règle syntaxique

PHR --> tout GN vt un GN

obtenue à partir des règles 5 et 29 on associe la règle de réécriture schématisée en figure 6.23.

A l'aide de ces règles on construit deux réseaux, H_1 et H_2, de la manière suivante: on se donne la connaissance à supprimer sous la forme d'une expression en LQN. L'analyse de cette expression conduit à la construction du réseau H_1, conformément

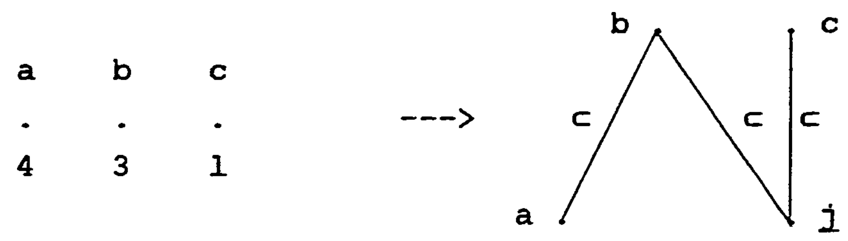


figure 6.23

aux procédures indiquées en 6.2.3 et 6.2.4, mais indépendamment du réseau existant. On applique les règles de simplification au réseau ainsi obtenu, mais pas les règles de cohérence.

Le réseau H_2 est obtenu de la même manière, mais on s'arrête avant d'appliquer la règle terminale.

On obtient ainsi une règle de réécriture de graphes que l'on peut noter $<H_1,H_2>$ puisqu'elle n'implique pas de fusion de sommets. Cette règle est une **règle de suppression**. On applique cette règle au réseau représentatif des connaissances en une QR-dérivation. Cette application a lieu éventuellement plusieurs fois, en tout cas tant que c'est possible, puisque la connaissance à supprimer figure peut-être plusieurs fois dans le réseau.

Exemple: On reprend l'exemple du paragraphe précédent, en s'arrêtant au réseau obtenu avant l'application des règles de cohérence (figure 6.21). Si l'on veut supprimer de ce réseau la connaissance

Jean est riche et intelligent

on construit automatiquement la règle de suppression schématisée

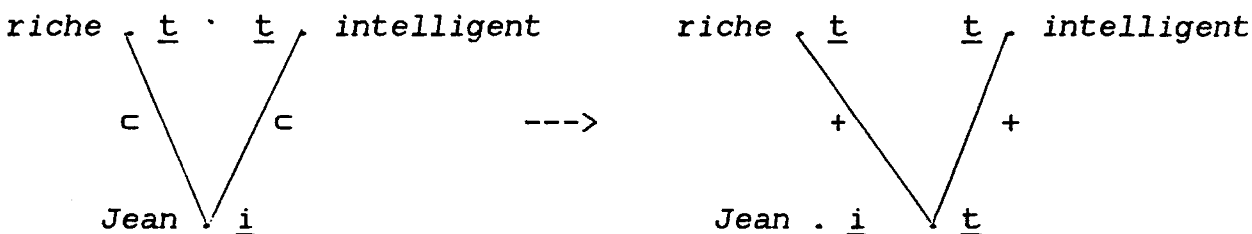


figure 6.24

en figure 6.24.

L'application de cette règle au réseau de la figure 6.22 donne le réseau transformé de la figure 6.25.

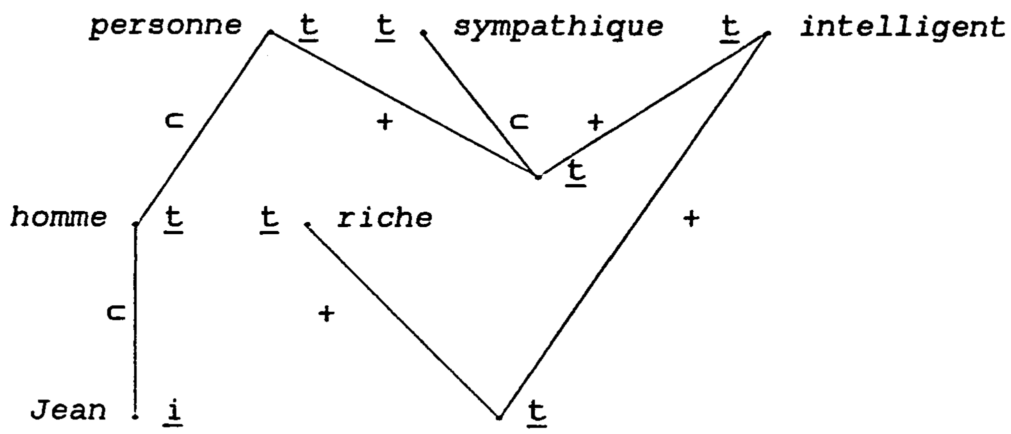


figure 6.25

Remarquons qu'après l'application d'une règle, il peut y avoir des sommets qui restent dans le réseau bien que ne servant plus à rien dans la représentation des connaissances. Ici, le sommet *riche* et le sommet *riche et intelligent* ne sont plus d'aucune utilité.

6.5 La construction automatique de règles de réécriture

La construction automatique de règles de réécriture telles celles qui ont été introduites en 4.3 ne pose guère de problèmes nouveaux par rapport à la construction automatique du réseau. Il faut simplement étendre légèrement la grammaire syntaxique en y substituant l'axiome REG à l'axiome PHR et en y adjoignant les règles:

REG --> si PHR alors PHR

REG --> si PHR REG

On ajoute donc si et alors au vocabulaire terminal.

On doit également pouvoir reconnaître des symboles tels que x, y ou z qui joueront exactement le même rôle syntaxique que les noms propres mais dont les sommets représentatifs, à la différence des sommets représentant les noms propres, ne seront pas sortés. De plus, lorsque l'on effectue les réductions 4 et 6, si c'est un symbole tel que x ou y qui joue le rôle de npr, au lieu de remplacer la sorte indéterminée x par i, on la remplace par j.

On construit alors deux réseaux, H_1 et H_2, en appelant les actions sémantiques définies précédemment, ainsi que les règles de simplification et les règles de cohérence si nécessaire.

H_1 est construit au fur et à mesure de la reconnaissance des différents PHR, tant qu'on ne rencontre pas alors.

H_2 est construit en même temps que le dernier PHR est reconnu.

Exemple: Reprenons un exemple traité en 4.3.2. Si l'on veut construire automatiquement la règle représentative de la connaissance

si une personne x a pour grand-père un homme z, alors il existe une personne y telle que x a pour parent y et y a pour parent z

il faut d'abord la traduire en LQN:

si x est une personne
si x a pour grand-père z
si z est un homme
alors x a pour parent une personne qui a pour père z

On vérifie qu'on obtient bien par construction automatique la règle de la figure 4.90.

Annexe 1 : la grammaire du langage quasi-naturel

G = $<V_T, V_A, \mathcal{R}, PHR>$ avec

V_T = {tout, un, subs, estun, ilyaun, npr, et, adj, adv, etqui, etque, qui, que, vt, vi, est}

V_A = {PHR, GN, GD, GV, GN1, GRL, GAD, GAD1, REL, PHR1} et l'ensemble $\mathcal{R}$ des règles est composé de:

1	PHR	-->	tout GN GD
2	PHR	-->	un subs estun GN
3	PHR	-->	ilyaun GN
4	PHR	-->	npr GD
5	PHR	-->	tout GN GV
6	PHR	-->	npr GV
7	GN	-->	GN1
8	GN	-->	GN1 GRL
9	GN1	-->	subs
10	GN1	-->	subs GAD
11	GN1	-->	GAD subs
12	GAD	-->	GAD1
13	GAD	-->	GAD GAD1
14	GAD	-->	GAD et GAD1
15	GAD1	-->	adj
16	GAD1	-->	adv GAD1
17	GRL	-->	REL
18	GRL	-->	GRL etqui GD
19	GRL	-->	GRL etque PHR1
20	GRL	-->	GRL etqui GV
21	REL	-->	qui GD
22	REL	-->	que PHR1
23	REL	-->	qui GV
24	PHR1	-->	un GN vt
25	PHR1	-->	tout GN vt
26	PHR1	-->	npr vt
27	GV	-->	vi
28	GV	-->	vt
29	GV	-->	vt un GN
30	GV	-->	vt tout GN
31	GV	-->	vt npr
32	GD	-->	estun GN
33	GD	-->	est GAD

Annexe 2: la grammaire de graphes

1

b

. . --> c

a 2 b 1 a

2 3 1 $\underline{i}$

. . --> .

a b ab

3

a

. --> c

a 1 $\underline{i}$

4

b

. . --> c

a 2 b 1 a

5 2 1 $\underline{i}$

. . --> .

a b ab

6 2 1 $\underline{t}$

. . --> .

a b ab

7, 9, 12, 15, 17, 21, 22, 23, 32, 33

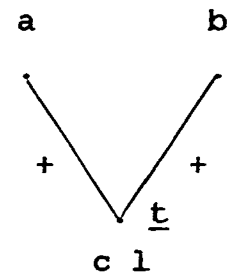
1
.
a

--> . a

8, 10, 11, 13

. .
a 2 b 1

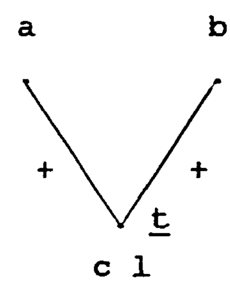
-->



14, 18, 19, 20

. .
a 3 b 1

-->



16

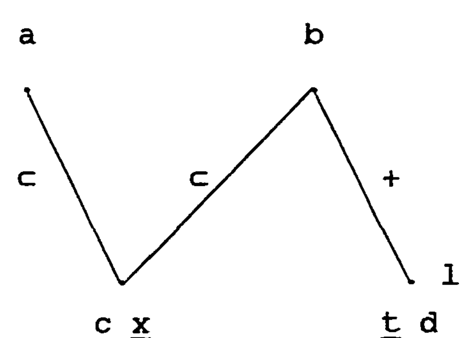
. .
a 2 b 1

-->

24

2 1
. .
a b

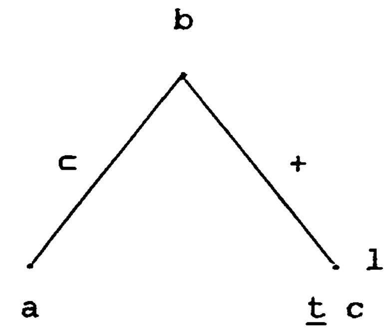
-->



25, 26

2   1
.   .
a   b

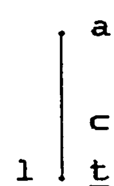
-->



27

.
a 1

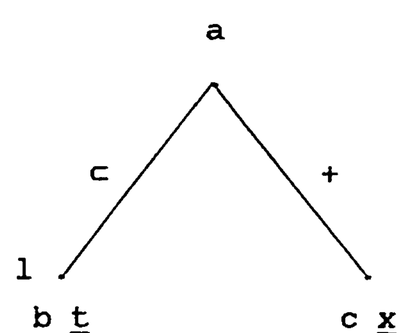
-->



28

1
.
a

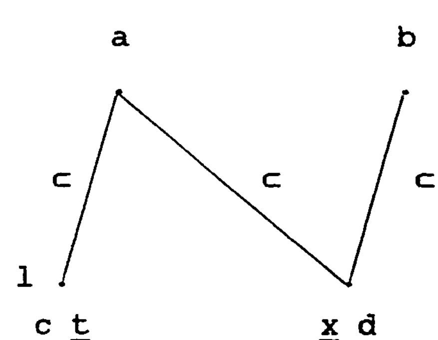
-->



29

3   1
.   .
a   b
.

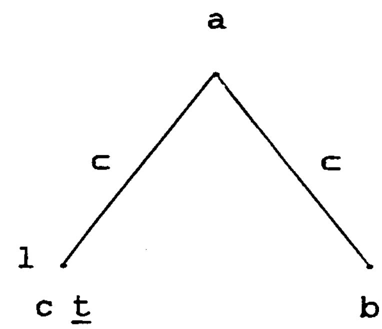
-->



30

3	1
.	.
a	b

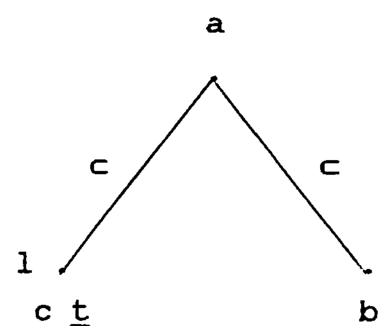
-->



31

2	1
.	.
a	b

-->



7 La réalisation du système informatique

Rappelons tout d'abord que l'objectif en écrivant des programmes a été uniquement d'**expérimenter** les outils théoriques que nous avons définis. C'est pourquoi le système que nous avons construit n'a pas été conçu afin d'être performant. Nous avons uniquement porté notre attention sur la programmation des parties originales de notre travail. Pourtant, en l'absence d'"environnements logiciels pour le traitement des langues naturelles" (voir [de Boissieu, 86]), nous avons dû "bricoler" des modules de liaison indispensables. Le module d'analyse morphologique et le module de génération des réponses en langue naturelle sont ainsi "sacrifiés". De la même manière, le temps d'accès au dictionnaire n'est pas optimisé.

Les programmes ont été écrits en LISP, plus précisément en MULISP 85, sur un micro-ordinateur de type "PC-AT".

Le problème que nous traitons est d'une part celui de la construction ou de la mise à jour d'un réseau représentatif de connaissances élémentaires exprimées en langage quasi-naturel, d'autre part celui de l'extraction de connaissances selon des questions exprimées également en langage quasi-naturel.

Le schéma fonctionnel de notre système est présenté en figure 7.1.

7.1 Le découpage de la suite de caractères entrée

En entrée, on a soit une affirmation, soit une question. Il s'agit donc, dans les deux cas d'une suite de caractères.

Dans un premier temps, il faut regrouper par segments des sous-suites de cette suite. On construit une suite de couples, chaque couple étant composé de:

- un mot sous sa **forme standard**, c'est-à-dire pour les noms communs le singulier, pour les adjectifs le masculin singulier et pour les verbes la troisième personne du singulier du présent de l'indicatif;
- une **catégorie grammaticale** au sens très large de ce terme,

phrase (suite de caractères entrée de manière interactive)

algorithme de découpage ← dictionnaire

suite d'unités lexicales mises sous forme standard et à chacune desquelles est associée une catégorie syntaxique

Analyseur syntaxique ← ATN

liste d'actions

exécution des actions

le réseau

une question

algorithmes d'interrogation

une liste de solutions

synthèse

réponses en langage quasi-naturel

--------------------------figure 7.1--------------------------

puisque cela peut aussi bien être la catégorie "verbe transitif" associée à la forme standard *mange* que la catégorie "tout" associée à la forme standard *chaque*. A une forme standard donnée ne pourra être associée qu'une catégorie grammaticale au maxixmum, ce qui est bien entendu une restriction fort importante.

7.1.1 Le prétraitement

On remplace toutes les lettres majuscules par des minuscules. On supprime tous les signes de ponctuation, sauf le tiret et l'apostrophe, et on les remplace par des blancs. L'apostrophe est remplacée par un tiret (en raison de l'usage du langage LISP). Toute suite de blancs est ramenée à un blanc. Le symbole de fin de données est soit le point, soit le point d'interrogation. Le point est purement et simplement supprimé, alors que le point d'interrogation est remplacé par la suite "int+", afin de distinguer la question

Jean aime Marie ?

de l'affirmation correspondante.

7.1.2 La recherche dans le dictionnaire

Le dictionnaire est une suite de triplets de la forme suivante:

<mot catégorie grammaticale [forme standard] >

Si la forme standard est omise, cela signifie que le mot est donné sous sa forme standard.

Des mots composés, c'est-à-dire comportant un tiret interne, figurent dans le dictionnaire.

Le découpage de l'entrée s'effectue en premier lieu selon les blancs. Si on obtient ainsi un segment avec un tiret interne, deux cas se présentent:

(1) le segment figure dans le dictionnaire en tant que mot; dans ce cas, il n' y a aucune difficulté;

(2) le segment n'y figure pas; dans ce cas, on regarde si la première partie, précédant le tiret peut être reconnue; ainsi le découpage de la phrase

la chauve-souris mange-t-elle?

pourra-t-il être le suivant:

(la chauve-souris mange t elle int+)

à condition que les mots "chauve-souris", "mange", "t" et "elle" figurent dans le dictionnaire. Si "chauve" et "souris" figurent également dans le dictionnaire, cela ne changera rien au découpage.

L'analyse morphologique est très rudimentaire.
Soit $x_1x_2...x_{p-1}x_p$ une suite de caractères à reconnaître qui ne figure pas dans le dictionnaire.

Si, dans le dictionnaire figure la suite $x_1x_2...x_{p-1}$ avec pour catégorie grammaticale substantif (commun) et que x_p = s, alors il y a reconnaissance.

Si, dans le dictionnaire figure la suite $x_1x_2...x_{p-1}$ avec pour catégorie grammaticale adjectif et que x_p = s ou x_p = e alors il y a reconnaissance.

Si, dans le dictionnaire figure la suite $x_1x_2...x_{p-2}$ avec pour catégorie grammaticale adjectif et que x_{p-1} = e et x_p = s alors il y a reconnaissance.

Si, dans le dictionnaire figure la suite $x_1x_2...x_{p-2}$ avec pour catégorie grammaticale verbe et que x_{p-1} = n et x_p = t alors il y a reconnaissance.

Afin d'effectuer cette analyse, les mots sont rangés dans le dictionnaire dans l'ordre inverse de l'ordre alphabétique. Ainsi, si l'on effectue une recherche séquentielle dans le dictionnaire, à partir du moment où il y a échec dans la recherche directe il n'est pas nécessaire de revenir en arrière dans le parcours du fichier pour envisager l'une des hypothèses qui précèdent.

Donnons un exemple; si la suite recherchée est "chiens" et que, dans le dictionnaire figurent, dans cet ordre, les mots:

chiffon
chienne
chiendent
chien
chat

on constatera l'échec de la recherche directe en examinant le mot "chienne". A ce moment là, on commencera, dès ce mot et en continuant de parcourir le fichier, la recherche avec "analyse morphologique" et on reconnaît le mot "chien" au pluriel. Si le mot "chien" ne figurait pas dans le dictionnaire, on s'arrêterait au mot "chat" en constatant l'échec définitif de la reconnaissance.

L'analyse morphologique ne fonctionne donc que pour des cas très simples de flexion. Dans les autres cas, il faut mettre toutes les formes dans le dictionnaire. Par exemple, on a les entrées "beau", "beaux", "bel" et "belle" qui toutes admettent la même forme standard "beau". Il n'est en revanche pas nécessaire d'avoir une entrée "belles".

Remarquons alors que pourront être reconnus comme adjectifs de forme standard "beau" les suites de caractères suivantes:

beaus
beaue
beaues
beauxs
beauxe
beauxes
bels
bele
beles
bellee
bellees.

Mais nous considérons que, compte tenu des objectifs poursuivis, une telle liberté n'est pas vraiment gênante.

7.3 l'analyse syntaxique

7.3.1 L'ATN

L'analyse syntaxique est effectuée au moyen d'un ATN [Woods, 70].

Si l'on a choisit un tel outil, c'est en raison de sa puissance et de sa commodité d'emploi. La classe des grammaires LR est trop restreinte, y compris pour des systèmes expérimentaux comme le nôtre.

Les actions associées à chaque transition vont consister à mettre à jour le graphe représentatif des connaissances, ou à construire une question à partir de son expression en langage quasi-naturel.

Toutefois, l'analyse n'étant plus déterministe, on ne peut exécuter les actions au fur et à mesure du parcours de l'ATN. En effet, si on parcourait l'ATN avec des retours en arrière, il faudrait pouvoir revenir sur les actions effectuées. Et, en cas de parcours en parallèle (solution que nous avons choisie), on ne voit pas comment effectuer toutes les actions possibles en même temps.

C'est pourquoi les actions consistent à construire une liste d'instructions qui seront exécutées uniquement en cas de reconnaissance.

De cette manière, il n'est pas nécessaire de se placer dans une situation différente selon que l'on tape une nouvelle connaissance ou une question. On peut très bien décider en fin de reconnaissance si ce qui a été tapé est une question ou une connaissance à rajouter au réseau (par exemple en détectant un point d'interrogation en fin de données).

Dans le cas où plusieurs listes d'instructions sont construites à travers l'analyse d'une phrase, il faut pouvoir effectuer un choix. Deux solutions sont possibles:

- demander de manière interactive à l'utilisateur du programme ce qu'il avoulu dire;
- prendre une solution arbitrairement.

L'ATN utilisé dans le cadre de cette réalisation est construit d'une telle manière que ce cas ne puisse se présenter.

7.3.2 L'interpréteur d'ATN

Il y a un parcours en parallèle de l'ATN comme dans [Kochut, 83].

Soit $x_1 \ldots x_p$ la suite à analyser.

On construit au fur et à mesure de la lecture de cette suite des ensembles de configurations S_0, $S_1, \ldots, S_p$. S_k est l'ensemble construit après lecture de k éléments de la suite.

Une configuration est un triplet composé d'un état de l'ATN, d'une pile qui mémorise les différents appels de sous-automates, et d'une pile de listes de registres avec les contenus de ces registres.

On se donne au départ une configuration unique dans S_0. Cette configuration est composée de:

- l'état initial de l'ATN;
- une pile vide;
- une pile composée d'un seul élément qui est la liste vide.

Selon les arcs qui partent d'un état donné de l'ATN, on effectue deux opérations:

- une opération de fermeture qui peut rajouter des configurations dans un état donné;
- une opération de construction qui permet d'initialiser un ensemble de configurations S_{k+1} à partir des configurations qui se trouvent dans S_k.

L'opération de fermeture est effectuée pour les arcs de type PUSH, de type POP ou de type JUMP. Pour les arcs PUSH on effectue un empilement dans les deux piles, alors que pour les arcs POP on dépile.

L'opération de construction est effectuée pour les arcs de type CAT. Les niveaux d'empilement restent alors les mêmes.

Avec une telle méthode, on n'est obligé de garder en mémoire simultanément que deux ensembles de configurations.

7.4 La construction de la représentation

7.4.1 La représentation interne des graphes

Les sommets sont créés au fur et à mesure que l'on en a besoin. Ce sont des atomes de la forme:

s-1, s-2,...

Quand certains sommets sont supprimés (notamment en raison des simplifications) on les met de côté afin de les réutiliser par la suite. C'est pourquoi, à chaque instant, on dispose de la liste des sommets disponibles, en même temps que du numéro du dernier sommet qui a été créé.

A chacun de ces atomes on associe, sous la forme de liste de propriétés:

- une sorte; dans le cas où, dans la question, la sorte n'est pas définie, on associe la sorte Q;
- une étiquette lexicale; dans le cas où l'étiquette lexicale n'est pas définie, on lui donne la valeur V-;
- la liste des successeurs du sommet;
- la liste des étiquettes d'arcs qui descendent du sommet;
- une liste des prédécesseurs du sommet (dont l'ordre est indifférent);
- une liste des étiquettes d'arcs dont le sommet est extrémité (l'ordre en est fixé par l'ordre des éléments de la liste des prédécesseurs).

De plus, à chaque unité lexicale est associée, comme liste de propriétés, la liste des sommets du réseau représentatif des connaissances et de la question qui admettent cette étiquette lexicale.

Chaque graphe est représenté par la double liste de ses sommets initiaux et de ses sommets terminaux.

7.4.2 <u>Les actions élémentaires</u>

Les actions définies au chapitre précédent sont décomposées en actions élémentaires:

(1) **Rajouter** un sommet à un graphe. Le sommet est défini par sa sorte et son étiquette lexicale. Il sera forcément, du moins dans un premier temps, initial et terminal.

(2) **Rajouter** un sommet à un graphe **si** il n'y figure pas déjà. On n'effectue l'opération (1) qu'après avoir vérifié s'il n'y a pas déjà dans le graphe un sommet qui possède cette même sorte et cette même étiquette lexicale.

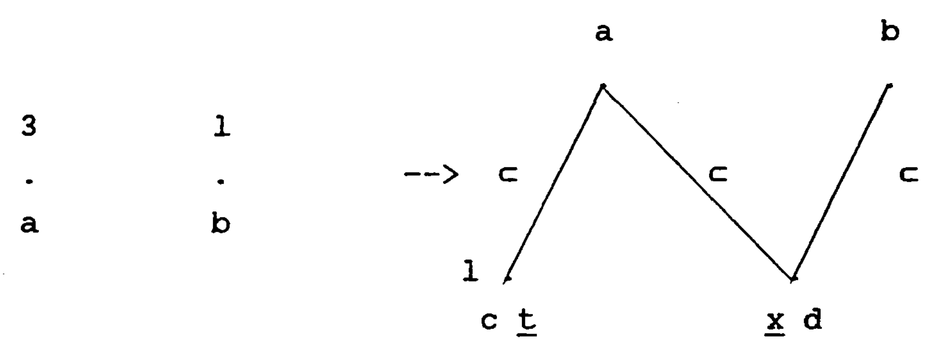
(3) **Relier** deux sommets. Etant donnés deux sommets x et y, on construit un arc d'origine x et d'extrémité y. Si x a déjà d'autres successeurs, y est rajouté en fin de liste.

(4) **Fusionner** deux sommets. Etant donnés deux sommets x et y, on

construit un sommet qui a même sorte et même étiquette lexicale que x, qui a pour liste de successeurs la concaténation de la liste des successeurs de x et de celle de y, qui a pour liste de prédécesseurs également la concaténation des listes de prédécesseurs de x et de y.

On se sert d'une pile contenant des sommets qui a un usage analogue à celui de la pile définie en 6.2.3, et éventuellement, si nécessaire, de variables temporaires telles que TEMP, TEMP1, etc.

Ainsi, une règle de réécriture de graphes telle que la règle numéro 29 (figure 7.2) se décompose ici en:



-------------------------<u>figure 7.2</u>-------------------------

- rajouter un sommet d de sorte indéterminée et sans unité lexicale;
- relier la tête de pile à d en étiquetant l'arc par $\subset$;
- dépiler;
- rajouter un sommet c de sorte <u>t</u> et sans unité lexicale;
- relier la tête de pile à c en étiquetant l'arc par $\subset$;
- relier la tête de pile à d en étiquetant l'arc par $\subset$;
- dépiler;
- empiler c.

c et d sont conservés momentanément dans les variables temporaires.

En réalité, les actions sont plus complexes parce qu'on applique en même temps et dès que c'est nécessaire les règles de simplification (6.3.1).

7.5 <u>L'interrogation</u>

On a un réseau H représentatif de connaissances et un réseau H_1 qui représente une question.

L'objectif est de construire une liste de solutions.

On construit cette liste en parallèle, au fur et à mesure du parcours de H_1.

Chaque solution, provisoire ou définitive, est composée d'une liste de triplets (x y lc) où x est un sommet de H_1, y est le correspondant de x dans H pour cette solution et lc est une liste de chemins: les chemins d'origine y et qui sont images dans l'agrandissement des arcs qui ont pour origine x. Si on ne sait pas quel chemin correspond à un arc donné, on met le symbole + à sa place.

A chaque fois qu'on examine un nouveau sommet de H_1, à partir de chaque solution de la liste provisoire on construit n nouvelles solutions, n pouvant être égal à zéro.

A la fin du processus on a une liste composée de p solutions dans lesquelles tous les sommets de H_1 ont un correspondant et où tous les arcs de H_1 ont un correspondant, c'est-à-dire qu'on ne trouve plus le symbole +.

Si p = 0, on ne peut fournir de réponse à la question.

<u>Remarque</u>: Dans certains cas, alors qu'il n'y a qu'un agrandissement de H_1 dans H, on trouvera plusieurs solutions distinctes avec notre algorithme de recherche.

Par exemple, si on a enregistré les connaissances (figure 7.3)

Jean est un *homme*

Jean est un *professeur*

tout *homme* est une *personne*

tout *professeur* est une *personne*

Jean aime Marie

et que l'on pose la question (figure 7.4)

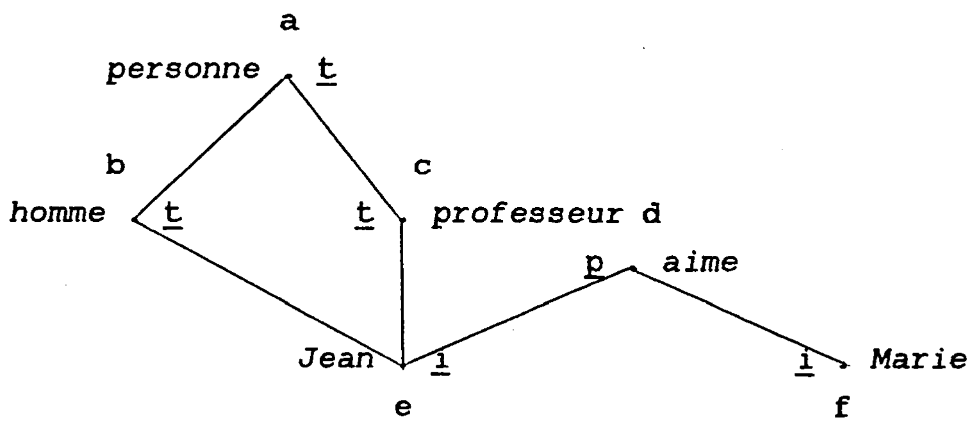


figure 7.3

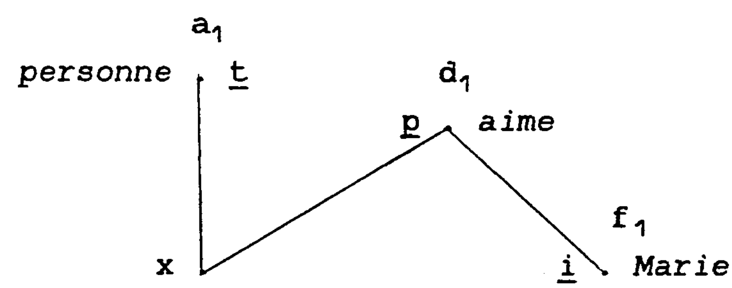


figure 7.4

y a-t-il une personne qui aime Marie ?
on aura plusieurs solutions. Ce sont les solutions qui associent à l'arc $<a_1,x>$ soit le chemin <a,b,e>, soit le chemin <a,c,e>. On fournira les différentes réponses correspondantes à l'interrogateur, car elles marquent qu'il y a plusieurs cheminements logiques pour trouver une même réponse.

7.6 La synthèse des réponses

Il ne s'agit pas, dans le cadre de ce travail, de construire des phrases en langue naturelle. Mais, de présenter les réponses aux question, qui sont sous la forme de listes de triplets, d'une manière qui soit compréhensible.

7.6.1 L'interprétation des sommets

Aux sommets x de sorte i ou t va être associée une interprétation que l'on note int(x). Cette interprétation sera constituée d'une liste d'unités lexicales.

Les sommets de sorte i qui reçoivent une étiquette lexicale recevront pour interprétation la liste constituée par cette étiquette lexicale.

Aux sommets de sorte i qui ne reçoivent pas d'étiquette lexicale seront associées des étiquettes lexicale fictives, conçues de telle manière qu'elles soient toutes distinctes. Ce seront les étiquettes lexicales:

X1, X2,...

Ainsi, si l'on a la connaissance

Jean aime une femme qui habite Paris

et que l'on pose la question

Qui Jean aime-t-il ?

nous choisissons de répondre par

Jean aime X1

en ajoutant toutes les propriétés qui définissent X1, c'est-à-dire:

X1 est une femme

X1 habite Paris .

Remarquons que le fait d'ajouter ces propriétés peut introduire de nouveaux objets individuels sans étiquette lexicale, dont il faudra également donner la définition. Ainsi, si l'on avait plutôt la connaissance:

Jean aime une femme qui possède une belle voiture

les propriétés qui définissent X1 seraient les suivantes:

X1 est une femme

X1 possède X2

X2 est une voiture

X2 est beau .

Les sommets x de sorte t qui reçoivent une étiquette lexicale $\xi^2(x)$ ont pour interprétation la liste constituée par cette étiquette lexicale précédée de *chaque*.

Afin de définir l'interprétation des autres sommets typiques on définit une interprétation simplifiée, notée $int_1(x)$, qui sera, pour les sommets qui ont une étiquette lexicale, constituée uniquement par cette étiquette lexicale.

On considère alors tous les ascendants immédiats $y_1,\ldots y_p$ de x tels que l'arc $<y_i,x>$ soit étiqueté par +. Les sommets y_i sont soit de sorte t, soit de sorte p et il y a au moins un sommet y_i qui est de sorte t. On suppose que les sommets $y_1,\ldots,y_q$ sont de sorte t et que les sommets $y_{q+1},\ldots,y_p$ sont de sorte p. On définit l'interprétation de ces derniers sommets sous la forme d'une proposition relative à x. Cette proposition est notée $int_2(y_i,x)$ et elle a pour valeur, selon les cas:

(a) si $\delta(y_i) = z$, *qui* $\xi^2(y_i)$

(b) si $\delta(y_i) = xz$, *qui* $\xi^2(y_i)$ int(z)

(b) si $\delta(y_i) = zx$, que int(z) $\xi^2(y_i)$.

On définit l'interprétation de x comme étant constituée par la concaténation de tous les $int_1(y_i)$ pour les y_i de sorte t et de tous les $int_2(y_j)$ pour les y_j de sorte p, ces derniers étant séparés par des *et* s'il y en a plus qu'un.

int(x) = *chaque* $int_1(y_1)$ ... $int_1(y_q)$ $int_2(y_{q+1},x)$ et ... et $int_2(y_p,x)$

Remarquons que l'interprétation d'un sommet dont l'étiquette lexicale est un adverbe s'obtient de manière légèrement différente car, bien qu'un tel sommet reçoive une étiquette lexicale, il faut concaténer cette étiquette à l'interprétation de son ascendant immédiat.

7.6.2 Les suites lexicales associées à une réponse

On sépare en deux parties les connaissances dont on veut obtenir l'expression en un langage quasi-naturel.

(1) On considère tous les sommets x de H qui sont images d'un sommet de sorte p de H_1; ces sommets représentent les véritables propositions de la réponse. L'expression de ces propositions est la suivante, selon les deux cas possibles:

(a) $\delta(x) = x_1x_2$ (verbes employés transitivement)
$int(x_1)$ $\xi^2(x)$ $int(x_2)$

(b) $\delta(x) = x_1$ (verbes employés intransitivement)
$int(x_1)$ $\xi^2(x)$

On remarque que cette expression suppose que soit connue également int(z) pour les sommets z de sorte $\underline{j}$. Cette interprétation est obtenue de manière similaire à celle des sommets de sorte $\underline{t}$ qui n'admettent pas d'étiquette lexicale. On considère tous les ascendants immédiats $y_1,...y_p$ de x, mais en excluant le sommet x représentant la proposition que l'on cherche à exprimer. Sans cela on aurait des réponses telles que:

tout enfant a *un chien que tout enfant* a

On définit l'interprétation de z comme étant constituée par la concaténation de tous les $int_1(y_i)$ pour les y_i de sorte $\underline{t}$ et de tous les $int_2(y_j,z)$ pour les y_j de sorte $\underline{p}$, ces derniers étant séparés par des et s'il y en a plus qu'un.

int(x) = *un* $int_1(y_1)$ ... $int_1(y_q)$ $int_2(y_{q+1},x)$ et ... et $int_2(y_p,x)$

(2) Tout arc de H_1 a pour image un chemin de H. On considère tous les arcs de H qui font partie d'un de ces chemins. Parmi ces arcs on sélectionne les arcs <x,y> étiquetés par $\subset$ et qui vérifient:

$\varepsilon^1(x) = \underline{t}$

$\varepsilon^1(y) = \underline{t}$ ou $\varepsilon^1(y) = \underline{i}$

A chacun de ces arcs on fait correspondre la suite d'unités lexicales:

int(y) est [un] $int_1(x)$

l'existence de *un* dépendant de la présence ou pas d'un nom dans $int_1(x)$.

<u>Remarques</u>: 1. Un problème spécifique se pose dans le cas où il existe dans le réseau H des sommets typiques qui possèdent une étiquette lexicale, mais qui sont malgré cela défini par certains de leurs ascendants immédiats, du style:

une *femme est* une *personne femelle* .

Dans le cas où un arc <x,y> étiqueté par + figure dans la réponse à une question, nous préfèrerons ressortir toute la définition plutôt que simplement la relation *est* (un) qui lie x à y.

2. Il existe des réponses auxquelles on ne peut associer de suite lexicale de la manière décrite ci-dessus.

Donnons un exemple; si on a représenté la connaissance

Jean est un homme

par le réseau H de la figure 7.5 et que l'on pose la question

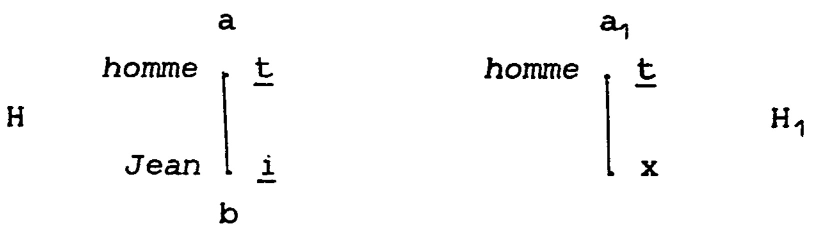


figure 7.5

qui est un homme ?

représentée par le réseau H_1 de la même figure, on obtient deux réponses à la question. Ce sont les agrandissements qui associent à a_1 le sommet a et à x soit le sommet b, soit le sommet a. La réponse, dans ce dernier cas, n'est pas fausse; elle pourrait être traduite par

tout homme est un homme .

On comprend toutefois qu'il ne soit pas nécessaire de la fournir à un utilisateur du système.

Annexe 1: exemples de connaissances exprimées en langage quasi-naturel et "comprises" par le système

Tout mammifere est un animal.
Toute personne est un mammifere.
Tout homme est une personne.
Les femmes sont des personnes.
Un garcon est un jeune homme.
Tout enfant est une personne.
Jean est un homme qui aime une tres belle femme qui habite Paris.
Paul connait une femme qui a un enfant qui possede un chien et que Jean deteste.
Paul est un garcon.
Luc est un homme laid qui connait tous les ministres.
Luc est tres peu intelligent.
Max est un ministre intelligent.
Les femmes intelligentes aiment les hommes sympathiques.
Marie est une femme intelligente.
Pierre est un homme sympathique.
Les mercedes sont des grosses voitures.
Les grosses voitures sont cheres.
Toute voiture est un objet.
Jean possede une mercedes.
Les chiens aboient.
Tout chien est un mammifere.
Tout mammifere possede une tete.
Paris est une belle ville.
Medor est un chien qui habite Paris et qui deteste les enfants.
Jean-Pierre est un enfant.

Annexe 2: la représentation interne de ces connaissances

```
S-2  T  ANIMAL  (S-1)  (:)
S-7  T  JEUNE  (S-6)  (+)
S-10  P  AIME  (S-9 S-14)  (: :)
S-12  T  BEAU  (S-11 S-15)  (<> :)
S-13  P  HABITE  (S-14 S-15)  (: :)
S-17  P  CONNAIT  (S-16 S-24)  (: :)
S-18  P  A  (S-24 S-22)  (: :)
S-19  P  POSSEDE  (S-22 S-21)  (: :)
S-23  P  DETESTE  (S-9 S-24)  (: :)
S-26  T  LAID  (S-25)  (:)
S-28  P  CONNAIT  (S-25 S-29)  (: :)
S-31  T  INTELLIGENT  (S-30 S-32 S-36)  (<> : +)
S-33  P  AIME  (S-36 S-39)  (: :)
S-34  T  SYMPATHIQUE  (S-39)  (+)
S-40  T  GROS  (S-45)  (+)
S-42  T  CHER  (S-45)  (:)
S-43  T  OBJET  (S-41)  (:)
S-44  P  POSSEDE  (S-9 S-46)  (: :)
S-47  P  ABOIE  (S-20)  (:)
S-48  P  POSSEDE  (S-1 S-50)  (: :)
S-49  T  TETE  (S-50)  (:)
S-51  T  VILLE  (S-15)  (:)
S-53  P  HABITE  (S-52 S-15)  (: :)
S-54  P  DETESTE  (S-52 S-8)  (: :)
S-1  T  MAMMIFERE  (S-3 S-20)  (: :)
S-6  T  GARCON  (S-16)  (:)
S-9  I  JEAN  NIL  NIL
S-14  I  V-  NIL  NIL
S-11  T  TRES  (S-14)  (:)
S-15  I  PARIS  NIL  NIL
S-16  I  PAUL  NIL  NIL
S-24  I  V-  NIL  NIL
S-22  I  V-  NIL  NIL
S-21  I  V-  NIL  NIL
S-25  I  LUC  NIL  NIL
S-29  T  MINISTRE  (S-32)  (:)
S-30  T  PEU  (S-27)  (<>)
S-32  I  MAX  NIL  NIL
S-36  T  V-  (S-37)  (:)
S-39  T  V-  (S-38)  (:)
S-45  T  V-  (S-35)  (:)
S-41  T  VOITURE  (S-45)  (+)
S-46  I  V-  NIL  NIL
S-20  T  CHIEN  (S-21 S-52)  (: :)
S-50  J  V-  NIL  NIL
S-52  I  MEDOR  NIL  NIL
S-8  T  ENFANT  (S-22 S-55)  (: :)
S-3  T  PERSONNE  (S-4 S-5 S-8)  (: : :)
S-27  T  TRES  (S-25)  (:)
S-37  I  MARIE  NIL  NIL
S-38  I  PIERRE  NIL  NIL
S-35  T  MERCEDES  (S-46)  (:)
S-55  I  JEAN-PIERRE  NIL  NIL
S-4  T  HOMME  (S-6 S-9 S-25 S-39)  (+ : : +)
S-5  T  FEMME  (S-14 S-24 S-36)  (: : +)
```

Annexe 3: exemples de questions "comprises"

par le système et les réponses fournies

Medor est-il un animal?

reponse numero 1
CHAQUE MAMMIFERE EST UN ANIMAL
CHAQUE CHIEN EST UN MAMMIFERE
MEDOR EST UN CHIEN

Qui deteste Jean-Pierre ?

reponse numero 1
MEDOR DETESTE CHAQUE ENFANT
JEAN-PIERRE EST UN ENFANT

Y a-t-il un animal qui aboie et qui habite une belle ville?

reponse numero 1
CHAQUE CHIEN ABOIE
MEDOR HABITE PARIS
PARIS EST UN VILLE
PARIS EST BEAU
CHAQUE MAMMIFERE EST UN ANIMAL
CHAQUE CHIEN EST UN MAMMIFERE
MEDOR EST UN CHIEN

Qui Jean deteste-t-il ?

reponse numero 1
JEAN DETESTE X1
X1 EST UN FEMME
X1 A X2
PAUL CONNAIT X1
X2 EST UN ENFANT
X2 POSSEDE X3
X3 EST UN CHIEN

Jean aime une femme ?

reponse numero 1
JEAN AIME X1
X1 EST UN FEMME
X1 EST TRES BEAU
X1 HABITE PARIS

Quelles personnes Marie aime-t-elle?

reponse numero 1
CHAQUE INTELLIGENT FEMME AIME CHAQUE SYMPATHIQUE HOMME
CHAQUE HOMME EST UN PERSONNE
MARIE EST INTELLIGENT FEMME

reponse numero 2
CHAQUE INTELLIGENT FEMME AIME CHAQUE SYMPATHIQUE HOMME
CHAQUE HOMME EST UN PERSONNE
MARIE EST INTELLIGENT FEMME
PIERRE EST SYMPATHIQUE HOMME

Qui possede un objet cher ?

reponse numero 1
JEAN POSSEDE X1
CHAQUE VOITURE GROS EST CHER
CHAQUE VOITURE EST UN OBJET
CHAQUE MERCEDES EST UN VOITURE GROS
X1 EST UN MERCEDES

Quelles personnes connaissent une personne ?

reponse numero 1
PAUL CONNAIT X1
CHAQUE HOMME EST UN PERSONNE
PAUL EST UN GARCON
CHAQUE FEMME EST UN PERSONNE
X1 EST UN FEMME
UN GARCON EST UN HOMME JEUNE
JEAN DETESTE X1
X1 A X2
X2 EST UN ENFANT
X2 POSSEDE X3
X3 EST UN CHIEN

Qui connait tous les ministres ?

reponse numero 1
LUC CONNAIT CHAQUE MINISTRE

Luc est-il intelligent?

reponse numero 1
LUC EST TRES PEU INTELLIGENT

Y a-t-il un animal qui possede une tete et qui possede une voiture ?

reponse numero 1
CHAQUE MAMMIFERE POSSEDE UN TETE
JEAN POSSEDE X1
CHAQUE MERCEDES EST UN VOITURE GROS
X1 EST UN MERCEDES
CHAQUE MAMMIFERE EST UN ANIMAL
CHAQUE PERSONNE EST UN MAMMIFERE
CHAQUE HOMME EST UN PERSONNE
JEAN EST UN HOMME

Quels animaux aboient ?

reponse numero 1
CHAQUE CHIEN ABOIE
CHAQUE MAMMIFERE EST UN ANIMAL
CHAQUE CHIEN EST UN MAMMIFERE

reponse numero 2
CHAQUE CHIEN ABOIE
CHAQUE MAMMIFERE EST UN ANIMAL
CHAQUE CHIEN EST UN MAMMIFERE
X1 EST UN CHIEN
X2 POSSEDE X1
X2 EST UN ENFANT
X3 A X2
X3 EST UN FEMME
JEAN DETESTE X3
PAUL CONNAIT X3

reponse numero 3
CHAQUE CHIEN ABOIE
CHAQUE MAMMIFERE EST UN ANIMAL
CHAQUE CHIEN EST UN MAMMIFERE
MEDOR EST UN CHIEN

Bibliographie

A.V. AHO et J.D. ULLMAN

1977 : Principles of Compiler design, Addison Wesley.

J.F. ALLEN

1983 : "Maintaining Knowledge about Temporal Intervals", Communications of the ACM 26 (11).

M.A. ARBIB et Y. GIVE'ON

1968 : "Algebra Automata I : Parallel Programming as a Prolegomena to the Categorical Approach", Information and Control 12 .

Y. BAR-HILLEL

1970 : Aspects of language, The Magnes Press, Jerusalem.

C. BERGE

1970 : Graphes et Hypergraphes, Dunod.

H. BESTOUGEFF et G. LIGOZAT

1985 : "Parametrized abstract objects for linguistic information processing", Proceedings of the Second ACL European Chapter Conference, Genève.

M.-J. BLOSSEVILLE et R. BOUILLOUX

1980 : "Méthodologie de construction d'une base de données à partir d'un texte la décrivant", Thèse de 3° cycle, Université Paris 7.

D.G. BOBROW et T. WINOGRAD

1977 : "An Overview of KRL, A Knowledge Representation Language", Cognitive Science 1 (1).

A. de BOISSIEU

1986 : "Environnement logiciel pour le traitement des langues naturelles. Application au traitement des données temporelles en français", Thèse de 3ème cycle, Université Paris 7.

A. BONNET

1980 : "Les grammaires sémantiques, outil puissant pour interroger les bases de données en langage naturel", RAIRO Informatique, vol. 14, n°2.

R. J. BRACHMAN

1979 : "On the Epistemological Status of Semantic Networks". Associative networks. Representation and Use of Knowledge by Computers, Edited by N.V. Findler, Academic Press.

R. J. BRACHMAN et J. G. SCHMOLZE

1984 : "An Overview of the KL-ONE Knowledge Representation System", FLAIR Technical Report n°30, Palo Alto.

N. CHOMSKY

1957 : Syntactic structures, Mouton, La Haye.

A. COLMERAUER

1977 : "Programmation en logique du premier ordre", Actes journées "La compréhension", IRIA.

1979 : "Un sous-ensemble intéressant du français", RAIRO Informatique théorique, vol 13, n°4.

M. CORI

1980 : "Structures hiérarchiques et opérateurs typés: algorithmes de transformations", Thèse de 3° cycle, Université Paris 7.

1982a : "Description d'une classe de grammaires de graphes sans circuit", RAIRO Informatique théorique, vol. 16, n°1.

1982b : "L'interrogation : ébauche d'une représentation formelle par des équations", Mathématiques et Sciences humaines, n°77.

1982c : "Une représentation de connaissances compatible avec une représentation d'énoncés", Informatique linguistique Recueil d'articles du groupe PITFALL 1975-1982 J.P. Desclés et M. Santha éditeurs, Université Paris 7.

1984 : "Interrogation de données textuelles à l'aide d'équations", TA Informations, vol. 25, n°1, 1984.

1985 : "Un système de représentation des connaissances à deux vitesses", Colloque Cognitiva, Paris.

M. CORI et P. MIAUX

1983 : "Un système questions-réponses expérimental", Université Paris 7, Département de recherches linguistiques.

D. COULON et D. KAYSER

1981 : "Variable-depth natural language understanding", IJCAI 7.

J. COURTIN

1977 : "Algorithmes pour le traitement interactif des langues naturelles", Thèse d'état, Grenoble.

A. CULIOLI

1982 : "The Role of Metalinguistic Representation in Syntax", XIIIth International Congress of Linguists, Tokyo.

A. CULIOLI et J.-P. DESCLES

1975 : "Considérations sur un programme de traitement automatique des langues et du langage", Informatique et Sciences humaines, Colloque C.N.R.S., Marseille.

J.-P. DESCLES

1980 : "Opérateurs/Opérations : Méthodes intrinsèques en informatique fondamentale. Applications aux bases de données et à la linguistique", Thèse d'état, Université René Descartes.

H. EHRIG

1979 : "Introduction to the algebraic theory of graph grammars" Graph Grammars and Their Application to Computer Science and Biology, Lecture Notes in Computer Science n° 73.

H. EHRIG, M. PFENDER et H.J. SCHNEIDER

1973 : "Graph Grammars: an Algebraic Approach", Proceedings of 14th Annual I.E.E.E. Symposium on Switching and Automata Theory, Iowa City.

D.W. ETHERINGTON et R. REITER

1983 : "On Inheritance Hierarchies With Exceptions", Proc. AAAI-83, Washington.

S.E. FAHLMAN

1979 : NETL A system for representing and using real-world knowledge, The MIT Press.

C.J. FILLMORE,

1968 : "The case for case", Universals in linguistic theory, Bach et Harms (eds), Holt, Rinehart and Winston, Inc., Chicago.

Ch. FROIDEVAUX

1985 : "Exceptions dans les hiérarchies sorte-de", 5ème congrès Reconnaissance des formes et Intelligence artificielle, Grenoble.

E. GENOUVRIER et J. PEYTARD,

1970 : Linguistique et enseignement du français, Larousse.

D.G. HAYS

1962 : "Automatic language-data processing" Computer

Applications in the Behavorial Sciences, H. Borko (ed.), Prentice Hall, Englewood Cliffs, New Jersey.

G.G. HENDRIX
1979 : "Encoding Knowledge in Partitioned Networks". Associative networks. Representation and Use of Knowledge by Computers, Edited by N.V. Findler, Academic Press.

K. KOCHUT
1983 : "Towards the Elastic ATN Implementation" The design of interpreters, compilers, and editors for Augmented Nteworks, edited by L. Bolc, Springer Verlag.

J.-L. LAURIERE
1982 : "Représentation et utilisation des connaissances", première et deuxième parties, TSI vol.1, n° 1 et n° 2.

J. LYONS
1970 : Linguistique générale, Larousse.

D. MCDERMOTT et J. DOYLE
1980 : "Non-Monotonic Logic I", Artificial Intelligence 13.

J.R. MCSKIMIN et J. MINKER
1979 : "A predicate calculus based semantic network for deductive searching". Associative networks. Representation and Use of Knowledge by Computers, Edited by N.V. Findler, Academic Press.

M. MINSKY
1975 : "A framework for representing knowledge", The Psychology of Computer Vision P. Winston (ed.), McGraw-Hill.

D.A. NORMAN et D.E. RUMELHART
1975 : Explorations in cognition, W.H Freeman and Company, San Francisco.

J.L. PFALTZ et A. ROSENFELD

1969 : "Web Grammars", Proceedings International Joint Conference on Artificial Intelligence, Washington.

M.R. QUILLIAN

1968 : "Semantic memory", Semantic Information Processing M. Minsky (ed.), MIT Press.

R.B. ROBERTS et I.P. GOLDSTEIN

1977 : "The FRL manual", AI Memo n°409, MIT Artificial Intelligence Laboratory, Cambridge.

G. SABAH, M. RADY, L. SOQUIER et J.B. BERTHELIN

1981: "Un système modulaire de compréhension d'histoires racontées en français", T.A. Informations, n°2.

R. C. SCHANK

1975 : Conceptual information processing, North-Holland.

S.C SHAPIRO

1979 : "The SNePS Semantic Network Processing System". Associative networks. Representation and Use of Knowledge by Computers, Edited by N.V. Findler, Academic Press.

1980 : Book Reviews, AJCL vol. 6, n° 3-4.

J.F. SOWA

1984 : Conceptual structures: Information Processing in Minds and Machines, Addison-Wesley.

L. TESNIERE

1959 : Eléments de syntaxe structurale, Klincksieck, Paris.

D. S. TOURETZKY

1986 : The Mathematics of Inheritance Systems, Pitman/Morgan Kaufmann.

G. VEILLON

1970 : "Modèles et algorithmes pour la traduction automatique", Thèse d'état, Grenoble.

W.A. WOODS

1970 : "Transition Network Grammars for Natural Language Analysis", Communications of the ACM, vol. 13, n° 10.

1975 : "What's in a Link: Foundations for Semantic Networks" Representation and Understanding: Studies in Cognitive Science, edited by D.G. Bobrow and A.M. Collins, Academic Press.

L.A. ZADEH

1983 : "Commonsense Knowledge Representation Based on Fuzzy Logic", IEEE Computer 16 (10).

Table

1987 COR