Représentations de connaissance: de l'approximation à la confrontation

Jérôme Euzenat

Représentations de connaissance

de l'approximation à la confrontation

Mémoire présenté afin d'obtenir le diplôme

d'habilitation à diriger des recherches

Université Joseph Fourier — Grenoble 1

(version définitive : 24/04/99 15:18)

Jérôme Euzenat

INRIA Rhône-Alpes

655 avenue de l'Europe
38330 Montbonnot Saint Martin
Jerome.Euzenat@inrialpes.fr

PRÉAMBULE

Ce mémoire décrit une partie des recherches que j'ai menées ou encadrées entre 1992 et 1998 au sein du projet SHERPA de l'INRIA Rhône-Alpes dirigé par François Rechenmann. La présentation est organisée autour d'une préoccupation commune : comment rendre compte des relations que peuvent entretenir des représentations diverses d'une même situation. Cette situation peut être de nature extrêmement variée : les relations spatiales entre les différentes pièces d'un puzzle, l'organisation d'un ensemble d'individus en différentes classes ou les « états mentaux » d'un individu.

Il ne s'agit pas toujours d'un ensemble de recherches achevées, mais d'un état du travail en cours convergeant vers un but commun.

Motivations personnelles

Qu'est-ce qui motive mon travail ? Depuis dix ans que j'accomplis un travail de recherche, il est légitime que je m'interroge sur ce qui dirige ou ce qui structure mon activité en dépit de la diversité et l'apparente divergence des directions de travail liées au besoin d'obtenir des résultats tangibles sur des problèmes précis : représentation de connaissance par objets, systèmes de maintien du raisonnement, oubli, granularité temporelle et spatiale, analyse de données et apprentissage, rationalité dans les modèles de tâches, révision de bases de connaissance, construction collaborative de bases de connaissance, coopération entre agents, serveurs de connaissance et mémoire technique. Cette diversité est aussi présente au sein des applications traitées : conception de circuits, génétique moléculaire ou supervision de réseaux et de haut-fourneaux. Il est vrai que je m'intéresse à de nombreux champs d'activité sans autre motivation que la curiosité.

Mais, pour la plupart des thèmes que j'ai abordés, il existe une question qui les relie fondamentalement : c'est la question des contraintes que doivent satisfaire deux représentations d'une même situation. Ainsi, je ne m'intéresse pas tant aux modèles permettant de représenter la connaissance (bien qu'ayant travaillé sur l'implémentation et la sémantique de plusieurs d'entre eux : SHIRKA, IROISE, KOOL, SMECI, TROEPS) qu'aux relations qui peuvent exister entre deux représentations. Ces représentations peuvent s'exprimer dans un même système : c'est le cas des différentes vues du monde offertes par les systèmes de maintien du raisonnement, des points de vue de TROEPS ou de la granularité temporelle. Elles peuvent s'exprimer entre deux systèmes différents : c'est le cas lorsque deux agents se communiquent des informations ou lorsqu'une base de connaissance est construite collaborativement.

Bien entendu, énoncer des principes de rapports entre représentations sans se soucier du langage dans lequel la connaissance est représentée est assez peu utile. C'est pourquoi, le plus souvent, les rapports entre diverses représentations exprimées dans le même langage sont étudiés en se basant sur la sémantique de ce langage.

Motivations scientifiques

Les bases de connaissance — en tant que structure de stockage et de consultation mais surtout en tant qu'outil d'analyse exploratoire — peuvent être utilisées afin d'organiser la représentation d'un domaine particulier. En ce sens, la *représentation de la connaissance* est une activité de modélisation d'un domaine[1]. C'est dans ce cadre que se placent les travaux présentés ici. Le modèle peut être de nature très variée. Il n'est nécessairement ni adéquat ni complet, ce qui peut s'expliquer par le manque de connaissance sur le domaine mais surtout par la nécessité de simplifier le problème afin de pouvoir se concentrer (ou concentrer son algorithme) sur l'essentiel.

La construction d'un modèle implique trois entités : un modélisateur, le domaine modélisé et le modèle lui-même [Berthier 1994, Stefik 1995]. Le modèle est construit par un modélisateur pour un but précis. Le modélisateur va faire des choix en fonction de ce but.

Il est donc inévitable de trouver plusieurs représentations différentes d'un même domaine. Ces représentations peuvent être concurrentes ou des approximations les unes des autres. Elles peuvent aussi être décrites dans le même langage de représentation de connaissance ou dans des langages différents.

L'importance de pouvoir réaliser des abstractions de la situation à représenter apparaît nettement en ce qui concerne la connaissance des génomes par exemple : un génome et son support (l'acide nucléique) peuvent être représentés au niveau atomique, au niveau chimique, au niveau nucléique (moléculaire), à divers niveaux génétiques (cartes physiques, génétiques) ou au niveau chromosomique. Chaque niveau semble être une abstraction d'un autre (voir Figure 1).

Ces différentes abstractions de la même situation ont souvent pour but de simplifier la représentation d'un phénomène tout en en gardant certains traits saillants. À ce titre, l'abstraction est extrêmement utile — dans l'activité de modélisation comme en informatique — puisqu'elle va permettre :

- d'interagir avec les utilisateurs sans surcharger les représentations ;
- d'adopter une approche compositionnelle dans le développement des modèles [Cousot 1996] ;
- de simplifier les problèmes à traiter et donc de diminuer la charge de l'ordinateur [Giunchiglia& 1997, Nayak& 1995, Cousot 1996].

Il n'est donc pas question de supprimer les abstractions pour obtenir une abstraction ultime. Il faut que les différentes abstractions coexistent et concourent à la représentation [Stefik 1995]. Mais il ne faut pas se priver d'étudier comment ces abstractions peuvent être liées. En particulier, la problématique sous-jacente ici est celle de la construction d'une approximation par enrichissement progressif.

Il pourrait sembler que l'empilement naturel des niveaux de la Figure 1 part du niveau atomique pour construire le niveau fonctionnel. Or rien n'est aussi simple pour différentes raisons.

[1] Le terme consacré est l'« objet » d'une modélisation mais le lecteur comprendra que soit préféré « domaine » afin de ne pas confondre avec les représentations par objets.

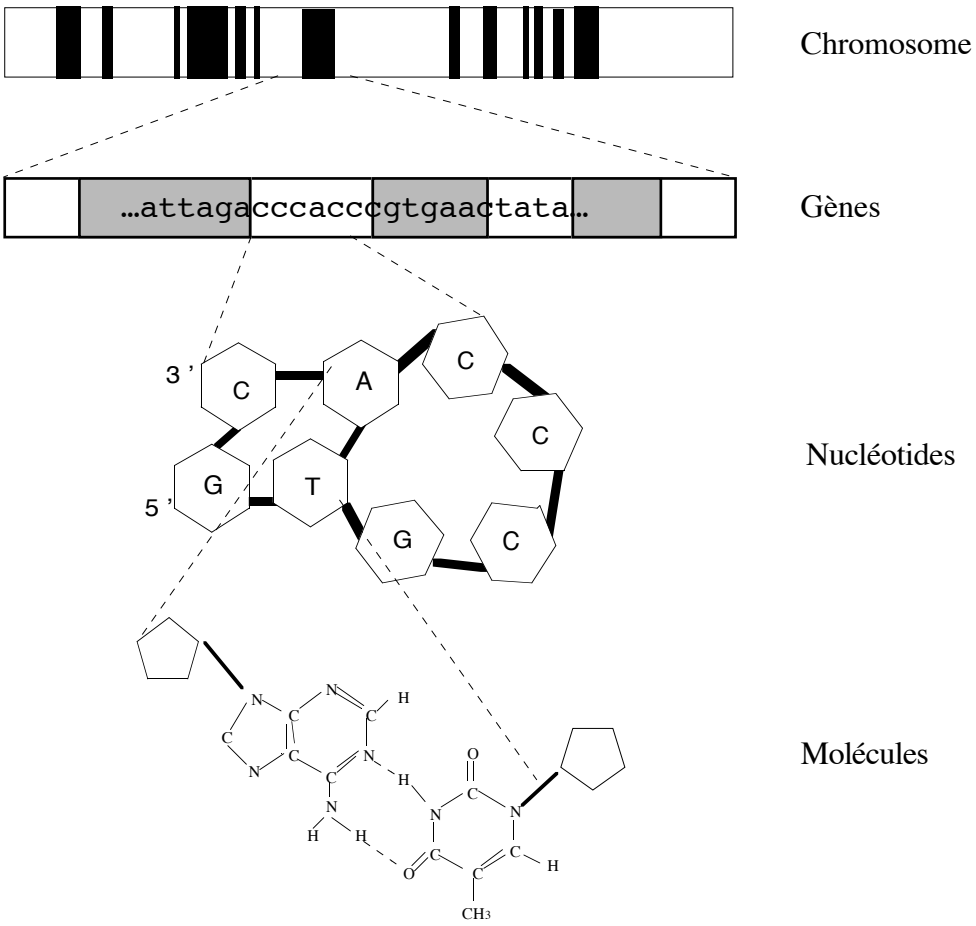


Figure 1 : Quelques niveaux de détails, entre l’atome et le noyau d’une cellule. Les bandes des chromosomes sont utilisées dans les cartes cytogénétiques pour localiser les gènes ; ceux-ci sont eux-mêmes localisés les uns par rapport aux autres dans des cartes génétiques ; ils peuvent être décomposés en une chaîne de nucléotides qui eux-mêmes sont des macromolécules composées d’atomes.

Tout d’abord, ce n’est pas ainsi que la connaissance se construit. Dans l’exemple de la Figure 1, des phénomènes ont d’abord été observés à un niveau génétique « théorique » (avec les lois de Mendel) ; le niveau chromosomique a ensuite pu être relié au niveau fonctionnel. Ainsi, de nombreuses manipulations sur le chromosome de la mouche à vinaigre (*Drosophila melanogaster*) permettent de connaître les transformations qu’elles entraînent sur le développement des mouches. Ce n’est que récemment que le niveau moléculaire a été relié au niveau chromosomique.

Ensuite, les niveaux décrits ne coïncident pas réellement (ils constituent deux approximations de la même chose plutôt qu’une approximation l’une de l’autre). Ainsi, l’énoncé d’un problème de repliement d’une molécule d’acide ribonucléique (ARN) en fonction d’une suite de lettres A, C, G et U telles que les A s’accolent aux U et les C aux G fait abstraction du niveau moléculaire où il existe des paires dites « non Watson-Crick ». Cet énoncé néglige aussi des cas particuliers où les nucléotides s’assemblent non par deux mais par trois. La solution théorique de l’énoncé a alors peu de chance d’être une solution au niveau moléculaire, mais elle en fournit une bonne approximation.

Enfin, l’incomplétude de la connaissance permet de faire plusieurs suppositions sur les niveaux les plus bas compatibles avec les observations.

Par ailleurs, à côté de l'organisation en niveaux, il en existe une autre en points de vue dépendants du centre d'intérêt de l'observateur. Par exemple, il est possible d'envisager une molécule sous le jour de sa chaîne de nucléotides (pour connaître la protéine en laquelle elle se transcrit) ou sous le jour de sa disposition spatiale (pour connaître les autres molécules sur lesquelles elle pourra se fixer). Il en est de même au niveau génétique : un gène peut être envisagé sous le jour de sa fonction (pour savoir à quoi il est utilisé dans un organisme), de son évolution (pour savoir quelles transformations il a subies) ou de ses interactions avec d'autres gènes (pour savoir quelles fonctions il est susceptible d'inhiber ou d'activer). Les points de vue décrits ne sont pas totalement indépendants, mais il est possible de ne s'intéresser qu'à l'un d'entre eux.

Bien entendu, la multiplicité des représentations possibles et la relation d'approximation d'une représentation au domaine représenté a été souvent observée [Aït-Kaci& 1993, Stefik 1995, Kayser 1997]. Mais c'est l'étude des *relations* entre ces différentes représentations qui est présentée ici. Mes travaux sur les « représentations de connaissance » sont présentés à travers leur point commun qu'est l'étude des contraintes que doivent satisfaire deux représentations du même domaine, qu'elles soient exprimées dans divers contextes (c'est-à-dire à partir d'hypothèses différentes), sous divers points de vue (c'est-à-dire en retenant des traits une organisation du domaine différents), diverses granularités (c'est-à-dire en retenant des objets en fonction de leur importance) ou par divers modélisateurs (qui doivent confronter leurs opinions). Ces contraintes se traduisent différemment suivant l'approche : opérateurs d'oubli, passerelles entre points de vues, opérateurs de conversion entre granularités ou opérateurs de révision de bases de connaissance.

Dans la démarche de développement de systèmes à base de connaissance dans laquelle je suis engagé, quelques notions très importantes apparaissent : modularité, intégration, incrémentalité. Ces notions sont exprimées dans les termes du génie logiciel et non dans ceux d'approximation ou de représentation. Cependant, il deviendra apparent à la lecture que ces notions, appliquées à la représentation d'un domaine, correspondent à la possibilité de développer des représentations partielles assemblables et superposables (modularité) ou à la possibilité de transformer une représentation en une représentation plus complète qu'elle approche (incrémentalité).

Le mémoire mettra en évidence ce qui est commun et ce qui sépare les différentes approches étudiées et mises en œuvre au cours de mes recherches. Il est clair que s'il était possible de donner les règles gouvernant l'assemblage de représentations modulaires, la représentation de connaissance, mais aussi les bases de données et les systèmes multi-agents pour ne citer que quelques domaines, pourraient rationaliser leurs méthodes de construction de manière significative.

Contribution originale

Au delà des différentes contributions que j'ai publiées, ce mémoire cherche à donner une vision d'ensemble de mon approche et à la présenter sous le jour de la notion d'approximation

entre représentations. La présentation est sans doute fragmentaire et ponctuelle mais tente de mettre clairement en évidence les intérêts et limites de cette approche.

Cadre de travail

Dans la plupart des travaux présentés ici, seules les représentations seront considérées ; il sera supposé qu'elles représentent toujours la même situation et qu'elles sont toujours exprimées dans le même langage. Cependant, chaque étude considérera un langage différent : représentation de connaissance par objets, algèbre de relations binaires…

Par ailleurs, il est supposé qu'il existe une relation d'approximation entre représentations : une représentation en approxime une autre si tout ce qui est représenté dans la première se trouve représenté dans la seconde. Pour chacun des langages considérés, il sera possible de définir la relation d'approximation de manière différente, mais elle sera toujours une donnée du travail présenté.

Les travaux qui suivent seront donc présentés d'une manière simpliste par un ensemble de diagrammes mettant en jeu des ronds et des flèches, les premiers représentant les représentations et les secondes la relation d'approximation (des approximants vers les

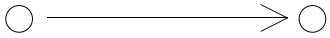


Schéma 1 : Approximation. La représentation à la source de la flèche approxime celle à l'extrémité. Tout ce qui est présent à la source est préservé à l'extrémité.

approximés).

L'existence d'une représentation, pour un langage donné, qui est celle qu'un modélisateur veut obtenir pour la situation à représenter est postulée. Cette représentation cible est figurée par un rond noir sur les schémas. Il est supposé que cette représentation, que l'on ne connaît pas, représente bien la situation en question (et qu'elle est donc consistante). On se place dans la

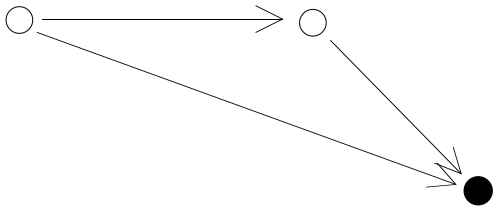


Schéma 2 : Représentation cible. Toutes les représentations intéressant le modélisateur cherchent à approximer cette représentation cible.

perspective où le modélisateur construit cette représentation en s'en approchant : les autres représentations considérées seront alors des approximations de la représentation cible.

Il est en général possible (et c'est en fait le cas dans la plupart des langages proposés ici) de considérer un objet initial (représenté ci-dessous par un rond cerclé de noir) qui est simplement défini parce qu'il peut approximer n'importe quelle autre représentation (et par conséquent la représentation cible). Cet objet est utile pour explorer la manière dont une représentation est construite à partir de rien. Cependant, il sera peu exploité dans la suite de l'exposé.

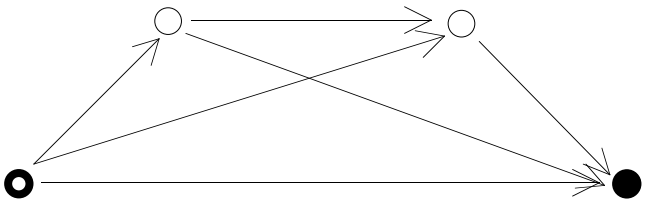


Schéma 3 : Représentation initiale. Toutes les représentations créées par le modélisateur sont approximées par une représentation vide. En procédant par incréments, le modélisateur peut enrichir la représentation de telle sorte qu'elle se rapproche de la représentation cible.

Par ailleurs, un point commun à l'ensemble des travaux présentés dans ce mémoire est qu'ils sont indicatifs[2] (ils précisent quelles sont *les* interprétations *possibles* des représentations utilisées) mais jamais impératifs (ils n'indiquent pas quelle est *l'*interprétation). C'est la différence entre une propriété faible et une propriété forte. Cette approche est sans doute due à la prise en compte de l'utilisateur (en général le modélisateur) dans l'ensemble des outils développés. Leur but n'est pas de se substituer au modélisateur mais de lui présenter le champ des possibles (en élaguant les impossibilités).

Contre-feu

Le lecteur averti aura reconnu, dans le cadre qui précède, les prémisses de la théorie des catégories [Barr& 1990, Pierce 1991]. Cependant, j'en resterai à ces considérations primitives. Il y a deux raisons à cela : tout d'abord, le travail présenté n'a pas été formellement reformulé dans la théorie des catégories (en particulier, les termes utilisés ne correspondent pas à leur acception catégorique) ; par ailleurs, il pourrait sembler que l'utilisation d'un tel formalisme est bien prétentieuse pour l'usage qui en est fait ici. Je reviendrai sur ce sujet dans la conclusion.

Plan du document

Le premier chapitre est principalement consacré à l'approximation entre deux représentations dans un formalisme à objets et des opérations permettant de passer d'une représentation à une représentation qu'elle approxime. Il présente ainsi presque toute la construction du système TROEPS au travers des systèmes classificatoires. Le second chapitre se concentre sur les représentations alternatives d'un même domaine sous le jour de taxonomies alternatives (les points de vue de TROEPS) et les problèmes que cela pose aux représentations par objets. Le troisième chapitre est consacré à un type particulier d'approximation puisqu'il s'agit de la représentation d'une même situation sous différentes granularités : il y a donc un ordre implicite entre les représentations dicté par leur résolution. Enfin, la dernière partie concerne la comparaison effective de plusieurs représentations afin de construire une représentation consistante et consensuelle. Pour cela elle détaille deux notions : la révision dans les bases de connaissance par objets (qui permet de résoudre les problèmes de consistance de manière

[2] J'utiliserais volontiers *normatif* au lieu d'indicatif, mais, pour certains, normatif est synonyme d'impératif.

minimale) et un protocole permettant à plusieurs intervenants d'intégrer des représentations alternatives d'un même domaine (construisant ainsi une base consensuelle).

Pour chacun des chapitres, le plan suivant sera observé :

— résumé du chapitre,
— sommaire,
— description du problème en fonction du cadre générique ci-dessus,
— une rapide perspective historique,
— parfois un bref rappel nécessaire à la compréhension de la suite,
— le corps du chapitre proprement dit,
— une conclusion et un aperçu des contributeurs sur le sujet traité.

Remerciements

Je voudrais remercier, en premier lieu, les étudiants que j'ai encadrés. J'ai appris avec eux la direction de la recherche. Merci donc, dans l'ordre d'apparition, à José Alejandro Quintero Garcia, Jérôme Gensel, Paule Merlin, Martin Strecker, Johannes Stein, Shong-Ye Tan, Petko Valtchev, Isabelle Crampé, Damien Raczy, Vincent Cligniez, Sylvain Chicois, Christophe Alemany, Loïc Tricand De La Goutte et Claire Lecourtier.

J'apprécie beaucoup les remarques et critiques des rapporteurs : Daniel Kayser, Hassan Aït-Kaci et Nicolaas Mars. Je les remercie d'avoir sacrifié une partie de leur temps pour se pencher avec indulgence sur mes activités.

Je remercie aussi vivement les membres du jury Grenoblois : Michel Adiba et Philippe Jorrand qui animent, de longue date, la communauté informatique grenobloise. Amedeo Napoli s'est montré très intéressé par ce projet de synthèse et en a été le premier lecteur et critique. Je le remercie de participer à ce jury.

Enfin, si j'ai appris beaucoup des étudiants, j'ai aussi appris de mon directeur de recherche, François Rechenmann. Ses conseils et son recul ont toujours été très utiles et appréciés.

Je dois remercier de nombreux collègues qui ont permis à mes travaux de progresser et, en premier lieu, ceux qui ont bien voulu éclairer ce mémoire de leur avis : Mathias Chaillot, Danielle Ziébelin, Bernard Euzenat, Patrice Uvietta, Roland Ducournau (qui me pardonnera, je l'espère de se retrouver dans l'index entre une mouche et un éléphant). Leurs suggestions m'ont beaucoup aidé à avancer (même celles que je n'ai pas suivies). Je n'allongerais pas cette liste en ajoutant ceux qui n'ont eu les soucis ni de lire le mémoire, ni d'être encadrés par moi, en particulier les différents membres du projet SHERPA.

Corinne Lachaize a joué son rôle de biologiste jusqu'au bout en examinant si mes assertions étaient biologiquement correctes. Christophe Chemla et Bernard Jacq ont développé la base de connaissance KNIFE et Françoise Husser et Pascal Gounet celle de SToRIA.

Ma gratitude plus personnelle va, bien entendu, à Jutta et Anton.

Enfin, les institutions ayant soutenu financièrement le bon déroulement de nos travaux doivent en être créditées : le GIP « Groupement de Recherches et d'Études sur les Génomes » (1993-1995), le ministère de la recherche pour l'action coordonnée-concertée « sciences de la vie » sur les interactions géniques, le SERICS (ministère de l'industrie) au travers du guichet « autoroutes de l'information » (SToRIA, 1996-1998), le CNRS qui a soutenu l'action « interactions géniques » par le biais de son programme génome (1998-1999) et enfin le Centre Jacques Cartier (1998). Bien entendu, mon employeur depuis 1992, l'INRIA, et toute l'infrastructure de l'INRIA Rhône-Alpes (sans oublier notre assistante Françoise De Coninck) ont contribué aux travaux présentés ici.

Jérôme Euzenat

Jerome.Euzenat@inrialpes.fr

SOMMAIRE

I. REPRÉSENTATION PAR OBJETS ET CLASSIFICATION

La notion d'approximation apparaît naturellement dans le cadre de la représentation de connaissance par objets. Ce chapitre considère la notion d'approximation dans le contexte de la classification (au sens large).

Les systèmes classificatoires (§I.1 et I.2) ont été développés pour plusieurs raisons : abstraire, dans des systèmes capables de classer des individus, cette unique fonction de classification, l'étudier théoriquement et comparer les différents types de systèmes classificatoires entre eux. Mais ce développement a eu des retombées dans le développement de systèmes de représentation de connaissance principalement.

Dans ce cadre, l'intégration, au sein d'un système de représentation de connaissance, de types abstraits représentant les types de base utilisés par la représentation de connaissance et de contraintes est présentée (§I.3). L'intérêt de les intégrer dans le cadre des systèmes classificatoires est que, les opérations nécessaires à la classification étant fournies, les opérations de classification sont alors données gratuitement.

En ce qui concerne la classification au sens de l'analyse de données et de l'apprentissage automatique (§I.4), le problème peut aussi être ramené à son expression en termes de systèmes classificatoires. Cependant, au delà de cette reformulation, les primitives nécessaires à cette classification et des algorithmes génériques ont été définis sur les constructions d'un système classificatoire. Ainsi, si les primitives (une mesure de dissimilarité) sont disponibles pour les types abstraits, un algorithme de classification sera disponible sur tout système classificatoire dont il est issu.

Problème : approximations dans une représentation par objets

Il est possible d'affiner une représentation par objets d'un domaine de multiples manières [Aït-Kaci& 1993]. Cela peut être fait en ajoutant des sous-classes à une classe qui n'en dispose pas, en ajoutant des attributs à une classe (ou à toute une taxonomie de classes), en attachant un objet — ou une classe — à une classe plus spécifique, en ajoutant des contraintes à un attribut de classe ou en mettant une valeur dans un attribut d'objet. Toutes ces opérations sont disponibles dans la plupart des représentations de connaissance par objets courantes. Il est possible d'en imaginer de plus complexes telles que l'ajout de nouvelles classes « au milieu » d'une taxonomie (c'est-à-dire comme classe intermédiaire entre une classe et sa super-classe) ou la construction d'une taxonomie à partir de la simple donnée des objets [Valtchev 1999b]. Les opérations présentées ici ont la propriété principale d'être monotones, c'est-à-dire qu'aucune d'entre elles ne remet en cause les inférences valides possibles avant leur application. Quelques unes sont présentées dans le diagramme ci-dessous. Un avantage de la représentation par objets est la facilité d'appréhension directe de la notion d'approximation.

La notion de système classificatoire présentée ci-après a été principalement développée pour rendre compte des diverses manières de classer les objets qui pouvaient exister, rendant par là compte des rapports entretenus par différentes représentations de la même situation. Cette notion est une vision extrêmement abstraite de l'activité de classification qui est destinée à cerner

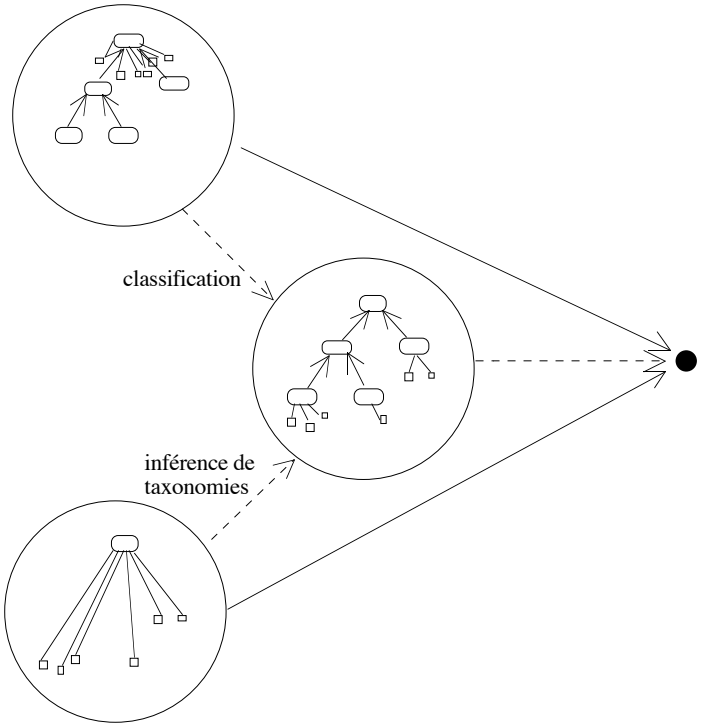


Schéma 4 : Inférence comme approximation. Ici le langage de représentation est celui d'une représentation de connaissance par objets avec des classes organisées en taxonomies et des objets attachés à ces classes. L'opération de classification permet d'obtenir une caractérisation plus précise de la position d'un objet. L'opération d'inférence de taxonomie permet d'obtenir une organisation plus détaillée du domaine. La représentation placée au centre est plus organisée que les deux autres car l'appartenance d'un objet à une classe y est plus précisément caractérisée.

l'espace des représentations possibles dans un langage permettant d'exprimer uniquement des taxonomies.

Ce premier chapitre est consacré aux aspects de l'approximation fortement liés aux représentations par objets. Il concernera principalement des notions taxonomiques telles que la classification, la catégorisation et l'inférence de taxonomie. La composition (de familles) d'objets sera aussi considérée dans le cadre des systèmes classificatoires et l'articulation de ces notions sera examinée.

Les problèmes soulevés dans ce chapitre peuvent être examinés dans le cadre d'une application dont le but est de modéliser les opérons (un ensemble de gènes qui concourent à la régulation d'un gène particulier chez les bactéries). Il peut être nécessaire de choisir le niveau de modélisation considéré comme primitif (faut-il partir des atomes, des nucléotides, des gènes ?). Les séquences — qui sont largement disponibles dans les banques de données — peuvent être considérées comme primitives dans ce modèle. Elles seront simplement introduites dans le système à partir d'un type de donnée abstrait. À partir de ce niveau minimum, il faut construire des objets plus élaborés. Ceux-ci peuvent être les gènes par exemple. Ils seront représentés en fonction de leur séquence, mais aussi d'autres informations comme leur taille, la protéine pour laquelle ils codent, etc. Il sera fait de même avec les opérons qui assembleront des gènes en tant que répresseurs et activateurs. Une fois cette structure décidée, il est possible de construire des taxonomies assemblant les objets créés en groupes, homogènes suivant un certain critère, qui seront appelés des classes. Pour cela, il est possible de créer les classes « à la main » et de les insérer dans la taxonomie à l'aide d'un mécanisme de catégorisation qui trouve la meilleure

place pour une classe. Une fois ceci fait les objets représentant des gènes ou des opérons individuels peuvent être attachés à des classes plus spécifiques en utilisant un mécanisme de classification. in, si d'emblée un ensemble représentatif d'objets est disponible, il est possible d'utiliser un mécanisme d'inférence de taxonomies capable de construire une taxonomie d'objets complète utilisant certains critères. Les systèmes de représentation de connaissance par objets offrent donc de nombreux services aux modélisateurs.

Ce chapitre envisage les différents services dans la perspective d'une construction progressive d'un modèle permettant d'obtenir des approximations de plus en plus précises. Il présente donc tout d'abord la notion de système classificatoire (§I.1) de manière générale (c'est-à-dire sous l'aspect taxonomique) avant de traiter de la construction de tels systèmes à l'aide de l'opération de produit (§I.2) couvrant ainsi l'aspect attributif. Ceci permet de construire des systèmes classificatoires à l'aide d'autres systèmes classificatoires et rend alors compte du principe de compositionalité. L'influence des composants sur les composés consistera ensuite en une restriction de l'espace des représentations possibles. Les principes développés ici sont appliqués au système TROEPS[3], tout d'abord au travers de l'intégration des types et des contraintes au sein du système (§I.3) puis au travers de l'introduction d'algorithmes d'inférence de taxonomie (§I.4).

Le chapitre suivant sera plus spécifiquement consacré à la notion de points de vue et, en particulier, à son intégration dans le domaine des systèmes classificatoires.

Perspective historique

Les formalismes de représentation de connaissance ont beaucoup évolué ces quinze dernières années. Les années 80 ont été simultanément l'occasion du développement d'environnements de représentation de connaissance d'envergure déployés dans de nombreuses applications [Filman 1988, Williams 1984, Ilog 1991, Rechenmann& 1988] et celle d'une étude théorique intensive et fertile de systèmes de représentation de connaissance idéaux [Napoli 1998, Levesque& 1987, Nebel 1990, Euzenat 1998d]. Ces travaux, ayant la même origine ont abouti au début des années 90 à une différenciation importante entre des systèmes non formalisés, relativement puissants et utilisés dans des applications et des systèmes formels très bien étudiés mais très peu implémentés et quasi inexistants dans les applications. Alors que les travaux théoriques auraient dû permettre un développement systèmes corrects et adéquats, il n'en est rien. D'une part, les systèmes implémentés se souciant un peu moins de leur formalisation sont utilisés dans des applications de plus en plus imposantes [Karp& 1995, MacGregor 1991] et, d'autre part, d'importantes critiques peuvent être faites à l'approche théorique (il est vrai que les travaux les plus connus ne se sont focalisés que sur un type de système) [Doyle& 1991, Davis 1991].

Le projet SHERPA se situe clairement dans la première tendance avec le système SHIRKA [Rechenmann& 1988] qui a connu de nombreuses applications [Euzenat& 1995]. Il fait partie cependant de ce que Roland Ducournau a appelé une « école française » [Ducournau 1998] de représentation de connaissance par objets qui se soucie de la cohérence des systèmes qu'elle

[3] Anciennement nommé TROPES mais dont le nom a été changé en 1998 pour des raisons juridiques.

implémente. Une des forces de SHIRKA était la disponibilité d'algorithmes de classification. Mais un problème s'est vite posé. En représentation de connaissance, la notion de classification était monopolisée par le formalisme des logiques terminologiques[4] dans lequel la classification ne peut être mise en œuvre qu'au sein de classes définitionnelles (c'est-à-dire des ensembles de conditions nécessaires et suffisantes). Ceci n'est pas le cas de la plupart des formalismes objets.

À partir de 1992, j'ai donc tenté de proposer un modèle permettant de rendre compte de l'activité des systèmes que le projet SHERPA implémentait mais aussi de le comparer avec les autres modèles proposés [Euzenat 1993a, 1994a]. La notion de système classificatoire est donc une tentative de justification de la classification sans condition suffisante. Elle a en partie été exploitée dans le développement de TROEPS [Capponi& 1995] et des algorithmes d'inférences de taxonomies [Valtchev& 1996].

Rappel : représentation de connaissance par objets

Ce qu'est un système de représentation de connaissance par objets classique, et surtout ce qui concerne la structure de la représentation, est brièvement rappelé ici. Une information plus complète peut être trouvée dans [Euzenat 1998b].

Un *objet* est un ensemble de couples attributs-valeurs associé à un identifiant. En général cette identification se fait à l'aide d'un nom (que celui-ci soit extérieur aux couples attributs-valeurs ou la valeur d'un attribut particulier). La valeur d'un attribut peut soit être un objet, soit être une valeur d'un type primitif du langage (par exemple, une chaîne de caractères ou un entier). Elle peut être connue ou pas. Souvent les attributs peuvent contenir une collection (par exemple, un ensemble) de valeurs.

Les objets sont regroupés en *classes* (dont ils sont alors *instance*). Les classes, à l'instar de celles des langages de programmation par objets (voir [Ducournau& 1998]), permettent de regrouper des traits communs à ces objets. Contrairement au modèle original des schémas ("frames" [Minsky 1974]), seules les classes associent des facettes (ou descripteurs) aux attributs et seul un ensemble restreint de *descripteurs* est autorisé et clairement identifié (même si celui-ci est parfois extensible par la programmation). Dans les systèmes les plus restreints, les facettes servent principalement à deux objectifs : préciser les valeurs d'attributs admissibles dans une classe (*typage*) et déclarer des mécanismes capables de déduire la valeur d'un attribut manquant (*inférence*). Les systèmes les plus riches offrent un ensemble plus complet de facettes permettant de spécifier des comportements lorsqu'une valeur est écrite ou lue, de gérer des relations entre objets ou d'assurer la liaison avec le monde extérieur. Seul l'aspect typage sera présenté ici.

Les facettes de typage permettent de restreindre les valeurs possibles d'un attribut en précisant les valeurs admissibles par une énumération (`domaine`) ou une réunion d'intervalles (`intervalles`), introduisant des restrictions (`sauf`) ou restreignant la cardinalité dans le cas de collections (`cardinal`). Lorsque la valeur d'un attribut peut être un autre objet, les restrictions

[4] Le terme « logique de descriptions » a été préféré à celui de « logique terminologique » par les chercheurs de ce domaine. J'estime que le premier est inadapté a des systèmes qui fonctionnent au mieux uniquement sur des « définitions ». J'utilise donc le second dans ce document.

```
<gène
  attributs = {
    <nom type = chaîne; constructeur = un;>,
    <synonymes type = chaîne; constructeur = ensemble;>,
    <protéine type = protéine; constructeur = un;>,
    <séquence type = séquence-adn; constructeur = ensemble;>,
    <taille-de-l-unité-de-transcription
           type = entier;
           constructeur = un;
           intervalles = { [10 +inf] };>
    };>

<gène-polarité-de-segment
  sorte-de = gène-structurel;
  attributs = {
    <protéine type = { protéine-structurelle };>};>
```

Figure 2 : Une classe et une sous-classe. Une classe est décrite par les ensembles de valeurs que peuvent prendre ses attributs (ici `nom`, `synonymes`, `protéine`, `séquence` et `taille-de-l-unité-de-transcription`). Ces ensembles de valeurs sont spécifiés par un constructeur (`un`, `liste` ou `ensemble`) et le type de la valeur qui peut être un type de base (`entier` ou `chaîne`) ou une autre classe (`protéine` ou `séquence-adn`). Les valeurs admises peuvent être précisées comme étant prises dans un ensemble d'intervalles (`{[10 +inf]}`) ou comme appartenant à une sous-classe (`protéine-structurelle`).

peuvent se faire en précisant les classes auxquelles l'objet doit appartenir (`type`). La Figure 2 présente deux définitions de classes.

Les classes sont organisées en une *taxonomie* par la relation de *spécialisation* (parfois nommée relation d'héritage ou de généricité). La représentation de connaissance par objets privilégie donc ce lien particulier entre les classes. La spécialisation est une relation d'ordre. Dans certains systèmes, le graphe de la réduction transitive de cette relation est restreint à un

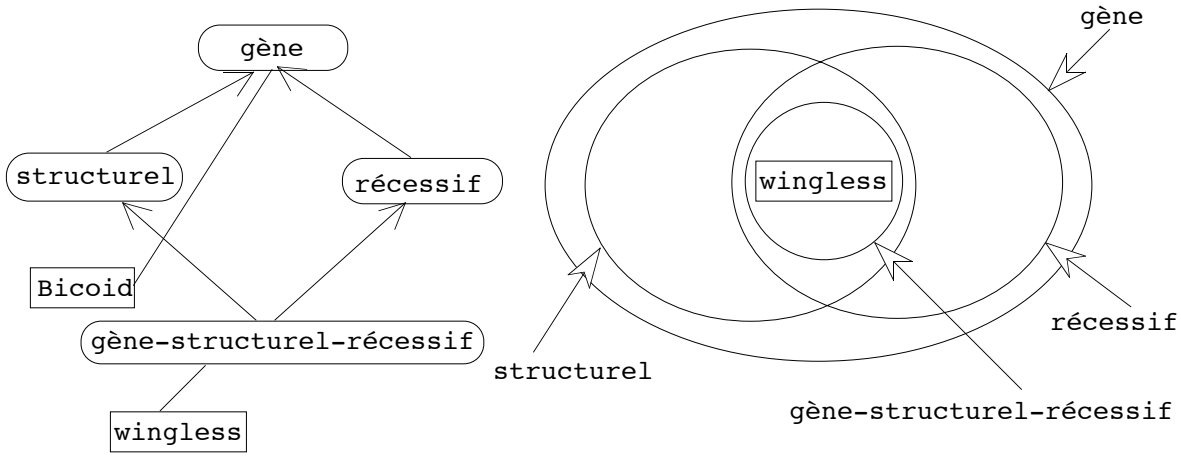


Figure 3 : Signification des relations entre classes. Les boîtes rectangulaires représentent les objets, les boîtes arrondies les classes. Les traits sans flèche représentent l'instanciation et les flèches vides la spécialisation. Le dessin de droite est une représentation ensembliste de la situation. Il faut noter que Bicoid n'y figure pas car s'il appartient à l'ensemble gène, rien n'est connu de son appartenance à un de ses sous-ensembles (il y a donc plusieurs interprétations possibles suivant les positions de Bicoid dans le diagramme).

arbre, contraint à être connexe ou à disposer d'une unique source (racine dans le cas d'un arbre). La sémantique de cette relation est celle de l'inclusion ensembliste : les individus

appartenant à l'interprétation d'une classe doivent appartenir à celle de ses super-classes. Les taxonomies peuvent donc être interprétées comme un ensemble de sous-ensembles imbriqués (voir Figure 3).

De cette interprétation de la relation de spécialisation, tout découle : plus une classe est spécifique, plus les domaines de ses attributs doivent être restreints ; si une classe est sous-classe de deux classes, les domaines de ses attributs sont forcément inclus dans (ou égaux à) l'intersection des domaines de ces deux classes ; si un objet est instance d'une classe, il l'est aussi de toutes ses super-classes, etc. Ceci permet de faciliter l'expression du domaine de valeur des attributs (qualifié d'*effectif*) en ne précisant dans les sous-classes que les restrictions par rapport à la super-classe (qualifié de domaine *exprimé*). Par exemple, s'il une contrainte sur la taille des gènes, celle-ci pèse aussi sur la taille des gènes récessifs : elle fait partie du domaine effectif de l'attribut `taille` de la classe `récessif`, mais il est inutile l'exprimer. Par contre, il n'est pas question d'introduire des exceptions dans les domaines de classes car cela permettrait inévitablement de définir « un éléphant comme un singe qui ne grimpe pas aux arbres » [Brachman 1985] (ou, pour poursuivre l'exemple, un ADN (acide désoxyribonucléique) comme une sorte d'ARN dans lequel l'uracile a été remplacé par la thymine).

Dans ce cadre (celui de la structure) certains problèmes n'apparaissent pas. C'est le cas des problèmes de conflits, dus à la *multi-généralisation*[5], c'est-à-dire le fait d'avoir plusieurs super-classes incomparables par la relation de spécialisation. Ces conflits ont été très étudiés dans le monde de la programmation par objets [Ducournau& 1995]. C'est aussi le cas des problèmes de non monotonie plus spécialement étudiés en intelligence artificielle. Ainsi, si l'utilisateur exprime que les `(gènes-)récessifs` ont la valeur de l'attribut `taille` comprise entre `10` et `20`, que les `(gènes-)structurels` ont la valeur de l'attribut `taille` comprise entre `100` et `300` et qu'il existe des gènes à la fois `structurels` et `récessifs`, il s'est tout simplement trompé quelque part.

I.1. Systèmes classificatoires

Le modèle des systèmes classificatoires permet d'établir un cadre unifié pour exprimer certaines des opérations utilisées dans une représentation par objets. À l'instar d'un système de typage, il ne constitue pas la sémantique du modèle de représentation par objets (pour une sémantique en théorie des modèles de TROEPS, voir §IV.2). Il constitue par contre un cadre indicatif permettant de circonscrire les solutions possibles de différentes opérations des systèmes de représentation par objets. Ceci permet de mettre en évidence les liens entre les diverses représentations au travers d'opérations transformant une représentation en une autre. Les systèmes classificatoires permettent donc de cadrer le fonctionnement des systèmes de représentation de connaissance mais non de spécifier complètement ceux-ci.

L'idée des systèmes classificatoires est très liée à l'activité de modélisation. Elle suppose qu'il existe une représentation cible (que le modélisateur veut obtenir). Cette représentation privilégiée correspond au rond noir des schémas et est représentée par l'interprétation réelle. Mais elle n'est pas mécaniquement appréhensible soit parce la machine ne dispose pas de toute

[5] Aussi nommé héritage multiple ou multi-spécialisation.

l'information, soit parce que le langage ne permet pas d'entrer dans tous les détails. Aussi, les systèmes classificatoires utilisent-ils la notion d'interprétation abstraite qui correspond à ce qui peut se calculer (et qui est donc accessible à la machine).

Comme leur nom l'indique, les systèmes classificatoires focalisent sur la notion de classification. Mais la classification considérée est différente de la subsomption dans les logiques terminologiques car il ne s'agit pas d'une opération impérative destinée à décider si une expression est subsumée par une autre mais d'une opération indicative destinée à établir si ce que représente une expression peut être une spécialisation d'une autre [Napoli 1992]. En fait, plusieurs utilisations de la structure taxonomique (aussi appelée classification) seront présentées :

— la classification (appelée identification en analyse de données ou classement) consiste à trouver des classes auxquelles un individu peut appartenir ;
— la catégorisation (appelée test de subsomption dans les logiques terminologiques) consiste à trouver les classes dont une classe particulière peut être spécialisation ;
— l'inférence de taxonomie (appelée classification en analyse de données et classification conceptuelle en apprentissage) consiste à trouver une taxonomie organisant un ensemble d'objets.

La notion de système classificatoire est présentée ici à l'aide de ses trois composants : les catégories et les individus (§I.1.1), la relation de spécialisation (§I.1.2) et les contraintes applicables à la construction de taxonomies (§I.1.3).

I.1.1. Classification

La classification (ou identification) considère deux ensembles d'entités abstraites L_C et L_I dont les éléments sont respectivement nommés *catégories* et *individus*. Ces entités peuvent avoir de nombreuses interprétations. Les interprétations dépendront du domaine considéré (par exemple, il est possible que L_I soit interprété comme l'ensemble des entiers naturels et L_C comme l'ensemble des intervalles d'entiers). Certaines interprétations sont données dans les sections qui suivent. Un individu i peut être classé dans une catégorie c. La classification d'un tel individu peut être présentée comme un simple « étiquetage de l'individu ».

DÉFINITION (interprétations) : Les catégories sont interprétées comme des ensembles d'individus. Il existe deux interprétations d'une catégorie c :

— L'interprétation abstraite $I_A(c)$ est l'ensemble des individus (dans L_I) décrits par une catégorie c indépendamment de leur existence ou de leur représentation effective ;
— L'interprétation réelle $I_R(c)$ est l'ensemble des individus (dans L_I) dénotés par c dans le domaine modélisé.

Le fait que la catégorie doive effectivement représenter sa dénotation est exprimé par la contrainte d'inclusion des interprétations : $I_R(c) \subseteq I_A(c)$. ◊

Les catégories sont intuitivement interprétées comme des ensembles d'individus. Cependant, le langage dans lequel elles sont exprimées ne permet pas forcément de les définir exactement. C'est la raison pour laquelle deux interprétations sont utilisées : l'interprétation réelle correspond à l'ensemble des individus que la catégorie est censée représenter — que le

modélisateur peut représenter — et l'interprétation abstraite est l'ensemble des individus que la description de la catégorie peut représenter — ou qui tombent sous cette description. Bien entendu, les individus que la catégorie est censée représenter doivent être décrits par la catégorie ; l'interprétation réelle est donc contrainte à être incluse dans l'interprétation abstraite.

La dualité intension (ou compréhension)/extension n'est pas utilisée ici. En fait, la classe c est exprimée dans un langage en compréhension et il y correspond naturellement une extension (l'extension de la compréhension) qui est $I_A(c)$. Mais $I_R(c)$, qui correspond à l'intention du modélisateur, est aussi un ensemble en extension. Qui plus est, c'est ce dernier ensemble (pris comme l'ensemble des instances d'une classe) qui est dans le domaine des langages de programmation à objets considéré comme l'extension. Afin de ne pas surcharger une fois de plus les termes intension/extension, il a été préféré d'utiliser interprétation abstraite/réelle.

Par exemple, dans le langages des intervalles d'entiers, la catégorie c_2 telle que $I_R(c_2)=\{1, 2, 3, 4\}$ peut être représentée par l'intervalle [1 4] dont l'interprétation abstraite correspond à {1, 2, 3, 4}. Néanmoins, la catégorie c_1 telle que $I_R(c_1)=\{1,3,4\}$ ne peut être représentée exactement par un intervalle d'entiers. Par conséquent, c_1 sera exprimée par l'expression telle que $I_A(c_1)=\{1,2,3,4\}$: [1 4]. L'appartenance d'un entier à l'intervalle [1 4] est donc une condition nécessaire mais non suffisante d'appartenance à la catégorie c_1. De même, dans l'exemple de la Figure 2, les gènes de polarité de segment ne codent que pour des protéines structurelles mais pas pour toutes les protéines structurelles.

DÉFINITION (classification) : À partir d'un ensemble de catégories C (tel que $C \subseteq L_C$), l'opération de *classification* détermine l'ensemble de catégories $Cl(i,C)=\{c \in C; i \in I_A(c)\}$ dans lesquelles un individu i est classé (dans la suite, la notation $Cl(i)$ sera utilisée lorsqu'elle ne sera pas ambiguë). ◊

Un individu peut être classé dans une catégorie c. La classification a pour but de trouver les

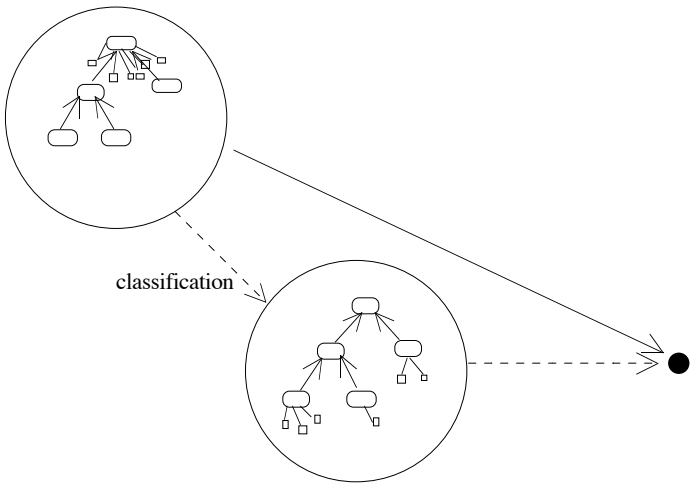


Schéma 5 : Classification comme approximation. À l'issue de la classification, si les individus sont attachés à des classes plus spécifiques que leurs classes initiales, la représentation initiale approxime la représentation résultante.

catégories dont l'interprétation réelle contient i. Cependant, comme l'interprétation réelle n'est pas mécaniquement accessible, classification retourne l'ensemble des classes dont

l'interprétation abstraite contient i. C'est pourquoi, c_1 et c_2 appartiennent toutes deux à *Cl(1)* et *Cl(2)*.

La notion abstraite de système classificatoire ne considère pas les conditions sous lesquelles un individu est classé dans une catégorie. Cependant, certains raffinements de cette notion (tels que les produits ou les projections qui seront présentés plus tard) détermineront ces conditions en fonction d'autres systèmes classificatoires.

La classification dans un système classificatoire permet de restreindre l'ensemble des positions possibles pour un individu particulier. Elle détermine donc un ensemble de représentations plus spécifiques.

I.1.2. Taxonomie

L'ensemble des catégories est habituellement organisé en une taxonomie.

DÉFINITION (taxonomie) : Une *taxonomie* $\langle C \leq \rangle$ par rapport à une interprétation I est composée d'un ensemble de catégories C et d'une relation d'ordre partiel $\leq$ (nommée *relation de sous-catégorisation*) sur ces catégories, tels que la propriété d'*inclusion des extensions* soit satisfaite :

$$c' \leq c \Rightarrow I(c') \subseteq I(c) \qquad \Diamond$$

Par la suite, $\leq$ sera confondue avec son graphe sur C. Si c et c' sont telles que $c \leq c'$, c sera une *sous-catégorie* de c' et c' sera une *super-catégorie* de c. La relation définie par $c_1 \leq c_2$ et $c_3=[2\ 2] \leq c_2$ est une taxonomie pour I_R car $I_R(c_1)=\{1,3,4\}$, $I_R(c_2)=\{1,2,3,4\}$ et $I_R(c_3)=\{2\}$. C'est aussi une taxonomie pour I_A parce que $I_A(c_1)=I_A(c_2)=\{1,2,3,4\}$ et $I_A(c_3)=\{2\}$. La relation définie par $c_2 \leq c_1$ et $c_3 \leq c_1$ est une taxonomie pour I_A mais pas pour I_R parce que les deux contraintes d'inclusions des extensions ne sont pas satisfaites.

La notion de taxonomie est très utile en tant que représentation de propriétés abstraites des catégories : celles qui permettent de considérer des catégories plus générales et des catégories plus spécifiques. Ceci est pris en compte dans l'implémentation d'algorithmes de classification.

I.1.3. Catégorisation

La *catégorisation* (nommée aussi classification de classes) permet de construire une taxonomie. Elle détermine par incrément les relations entre les catégories nouvellement ajoutées à C et celles précédemment dans C. La catégorisation construit donc $\leq$. Ceci est obtenu à l'aide du critère de sous-catégorisation.

DÉFINITION (système classificatoire) : Un *système classificatoire* $\langle C \leq \ll \rangle$ sur un couple de langages L_C et L_I est défini par un ensemble de catégories (C), une relation de sous-catégorisation ($\leq$) et un critère de sous-catégorisation ($\ll$), tels que :

- $\langle L_C \ll \rangle$ est une taxonomie pour l'interprétation abstraite,
- $\langle C \leq \rangle$ est une taxonomie pour l'interprétation réelle, et
- $\forall c,c' \in C, c \leq c' \Rightarrow c \ll c'$. $\Diamond$

Le critère de sous-catégorisation diffère de $\leq$: en fait, quand $\ll$ est utilisé, $\leq$ est en construction et ne s'applique qu'à C. Souvent, le critère de sous-catégorisation est un ordre « naturel » sur L_C (le sous-typage ou l'inclusion ensembliste par exemple). Intuitivement, c'est la contrainte la plus relâchée qui doit être satisfaite par la relation de sous-catégorisation. Une catégorie ne peut être insérée dans le graphe de $\leq$ si le critère de sous-catégorisation n'est pas respecté (c'est-à-dire si la dernière condition de la définition n'est pas satisfaite).

Dans l'exemple ci-dessus, cela signifie que la catégorie c_4=[2 4] ($I_R(c_4)$={2,3,4}) peut être introduite dans la taxonomie de telle sorte que $c_4 \leq c_2$ parce que [2 4]$\subseteq$[1 4]. Contraindre la relation de sous-catégorisation à satisfaire un ordre qui, (1) s'applique à L_C tout entier et (2) satisfait la propriété d'inclusion des extensions pour l'interprétation abstraite, assure durant la construction de $\leq$ que l'inclusion des extensions peut être satisfaite.

La catégorisation consiste à construire la taxonomie en ajoutant une catégorie après l'autre. Afin que la taxonomie construite respecte les contraintes données ci-dessus, les ensembles suivants sont définis :

DÉFINITION (catégories plus générales, plus spécifiques) : L'ensemble des *catégories plus générales* (resp. *plus spécifiques*) qu'une catégorie c par rapport à $\ll$ est : $MGC(c) = \{c' \in C; c \ll c'\}$ (resp. $MSC(c) = \{c' \in C; c' \ll c\}$). ◊

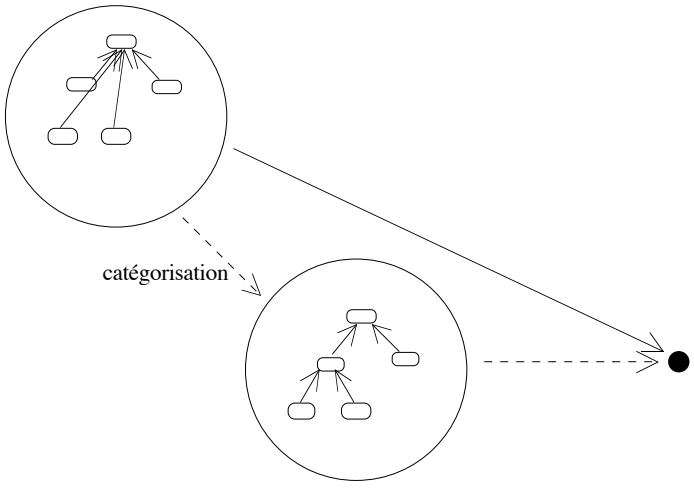


Schéma 6 : Catégorisation comme approximation. À l'instar de la classification, si, à l'issue de la catégorisation, certaines classes viennent spécialiser certaines sous-classes de leur super-classe initiale, la représentation initiale approxime la représentation résultante.

L'insertion d'une nouvelle catégorie satisfait le critère de sous-catégorisation seulement si la nouvelle catégorie est une sous-catégorie de catégories plus générales et une super-catégorie de catégories plus spécifiques. La catégorisation permet donc de déterminer les positions précises que la nouvelle catégorie peut occuper dans le graphe de sous-catégorisation.

Par exemple, $MSC(c_4)=\{c_3\}$ et $MGC(c_4)=\{c_1,c_2\}$ mais seulement c_2 peut être retenue pour l'insertion de c_4 parce que l'interprétation réelle de c_4 n'est pas incluse dans celle de c_1.

DÉFINITION (catégorisation) : L'opération de *catégorisation* détermine, pour une catégorie c, l'ensemble de ses catégories les plus générales et l'ensemble de ses catégories les plus spécifiques. ◊

L'intérêt de cette catégorisation est de pouvoir placer de nouvelles classes dans la taxonomie. La proposition suivante garanti que, pour peu que l'ordre partiel soit respecté, l'opération de catégorisation indique bien des positions acceptables.

PROPOSITION : L'insertion d'une catégorie c dans une taxonomie, de sorte que

1) $\forall c', c \leq c' \Rightarrow c' \in MGC(c)$,
2) $\forall c', c' \leq c \Rightarrow c' \in MSC(c)$ et
3) $\leq$ reste un ordre partiel,

résulte en une taxonomie. ◊

L'opération de catégorisation ainsi définie va permettre de déterminer l'ensemble des situations dont la situation présente est une approximation (au sens de l'ajout d'une classe).

Jusqu'à présent les taxonomies étaient modifiées de telle sorte que la taxonomie résultante soit approximée par la taxonomie initiale. C'est l'approche utilisée en général par un modélisateur qui construit son modèle par addition. Il est possible de modifier les taxonomies dans l'autre sens. Ainsi, la restriction d'une taxonomie supprime certaines catégories sans modifier l'ordre alors que la dégénérescence supprime certaines relations de sous-catégorisation (tout en garantissant que le résultat est encore un ordre partiel).

DÉFINITION (restriction et dégénérescence) : un ordre partiel $\leq$ sur un ensemble S peut être défini par le sous-ensemble O de $S \times S$ contenant tous les couples d'éléments en relation. La *restriction* de $\langle S\ O \rangle$ à l'ensemble $S' \subseteq S$ est $\langle S'\ O' \rangle$ tel que O' est le sous-ensemble de O dont les éléments sont éléments de $S' \times S'$. Une *dégénérescence* $\langle S\ O' \rangle$ de $\langle S\ O \rangle$ est telle que O' est un sous-ensemble de O clos par transitivité et réflexivité. ◊

La restriction et la dégénérescence conservent la propriété d'être une taxonomie. Elles permettent en fait de définir une notion d'approximation dans la construction de taxonomie. Par ailleurs, il existe dans tout système classificatoire une taxonomie terminale (c'est-à-dire dont toutes les autres sont une approximation) : $\langle L_C \ll \rangle$.

PROPOSITION :
1) $\langle L_C \ll \rangle$ est une taxonomie ;
2) toute restriction d'une taxonomie est une taxonomie ;
3) toute dégénérescence d'une taxonomie est une taxonomie. ◊

PROPOSITION : Une taxonomie $\langle C \leq \rangle$ dans un système classificatoire $\langle C \leq \ll \rangle$ est une dégénérescence de la restriction de $\langle L_C \ll \rangle$ à C. ◊

Par exemple, la taxonomie c_4=[2 4]≤c_3=[2 2]≤c_2=[1 4], $c_4 \leq c_2$ est une restriction de $\langle IN \subseteq \rangle$ (l'ensemble des intervalles d'entiers munis de l'inclusion d'ensembles) aux catégories mentionnées et la taxonomie $c_3 \leq c_2$, $c_4 \leq c_2$ est une dégénérescence de celle-ci supprimant $c_3 \leq c_4$.

Il faut noter que la propriété précédente est vraie de l'interprétation abstraite alors que $\langle C \leq \rangle$ est une taxonomie pour l'interprétation réelle. Cependant, c'est aussi une taxonomie pour l'interprétation abstraite par la définition des systèmes classificatoires (ceci est requis afin que le système puisse tirer avantage de l'interprétation abstraite dans les opérations de classification).

Naturellement, si la dégénérescence n'était pas considérée comme une opération de construction, ≤ serait la simple restriction de ≪ à C et l'insertion d'une catégorie dans C serait immédiate puisqu'il suffirait qu'elle soit sous-catégorie de toutes ses catégories plus générales et super-catégorie de toutes ses catégories les plus spécifiques. En général, cette contrainte ne sera pas requise.

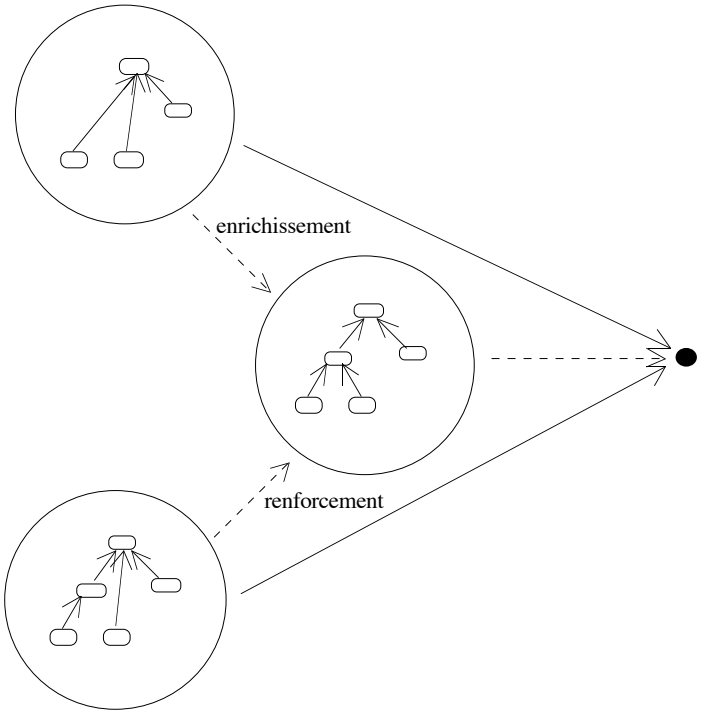


Schéma 7 : Approximations engendrées par les inverses de l'appauvrissement (enrichissement) et la dégénérescence (renforcement). Elles permettent d'ajouter des contraintes à la représentation en restant approximées par la représentation initiale.

Certaines propriétés des taxonomies ont été mises en évidence (exclusivité, exhaustivité…) et la manière dont elles sont conservées par les diverses opérations (restriction, dégénérescence mais aussi d'autres opérations qui seront vues plus bas) établie partiellement [Euzenat 1993a].

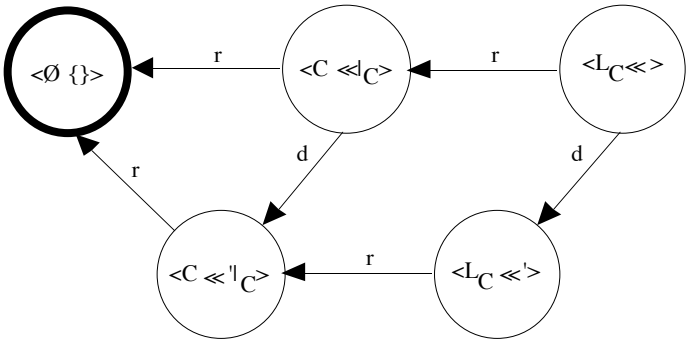


Figure 4 : Étant donné un langage de catégories L_C et un ordre partiel (≪) sur ce langage, un ensemble de taxonomies est engendré par l'application systématique de restrictions (r) et de dégénérescences (d) à partir de $\langle L_C \ll \rangle$ ($\emptyset \subseteq C \subseteq \ldots \subseteq L_C$). L'élément terminal de cet ensemble (celui qui ne peut pas être plus restreint ou dégénéré) est celui dont l'ensemble de catégories est vide. Pour construire une taxonomie, l'utilisateur commence en général avec l'élément terminal et utilise les flèches dans l'ordre inverse. Ces flèches correspondent alors aux morphismes du Schéma 7.

La Figure 4 décrit, de manière simpliste, l'ensemble des taxonomies qui peuvent être construites à partir d'un langage L_C et d'un critère de sous-catégorisation. Chaque taxonomie

peut être obtenue à partir de $\langle L_C \ll \rangle$ en appliquant les opérations de restriction et de dégénérescence. L'opération de catégorisation consistera à construire une taxonomie en ajoutant une catégorie après l'autre. Elle ne procède donc pas par restriction ou dégénérescence, mais commence avec une taxonomie vide $\langle \emptyset \ \{\} \rangle$ et procède par extension et enrichissement. Cependant l'espace de recherche est déjà figé, il correspond au graphe de la Figure 4 dont les arcs sont parcourus en sens inverse.

L'espace des taxonomies possibles est ainsi défini en fonction de la restriction et de la dégénérescence, mais peut aussi l'être en fonction de leur réciproque.

Ainsi, les trois éléments des systèmes classificatoires permettent respectivement de définir la classification, les taxonomies et la catégorisation. Il est alors possible de rendre compte dans ce cadre de manière purement taxonomique de certaines opérations des représentations par objets : la classification (d'un objet) et la catégorisation (l'ajout d'une classe dans une taxonomie). Dans la perspective considérée ici, ces opérations définissent la relation d'approximation entre diverses représentations, la plus générale pouvant être restreinte ou dégénérée en une plus spécifique (il est possible, par exemple, de montrer que la composition de restrictions est une restriction de même que celle de dégénérescences est une dégénérescence).

Il est notable que ce qui a été présenté ici fait la part belle à la notion de taxonomie et néglige totalement les attributs des représentations par objets. La section suivante réintègre les attributs dans ce tableau.

I.2. Produit de systèmes classificatoires

La section précédente a développé la notion de système classificatoire. Cette notion rend compte de la manière dont la classification et la catégorisation opèrent au sein d'ordres partiels sans considérer la justification de ces ordres partiels (les conditions pour appartenir à $Cl(i)$). Le produit de systèmes classificatoires permet d'analyser ces systèmes classificatoires en fonction de systèmes classificatoires plus petits. Ainsi, la notion générale de système classificatoire trouve une instanciation plus constructive et un principe de compositionalité peut intervenir dans la construction des représentations. La notion de produit permet de rendre compte de la composition d'objets et de la classification multi-points de vue (voir chapitre II).

Un produit de systèmes classificatoires est simplement défini ci-dessous de telle sorte que les deux ordres $\ll$ et $\leq$ soient les ordres produits, l'interprétation réelle soit incluse dans le produit cartésien des interprétations réelles initiales et le résultat de la classification des objets (produits) soit le produit de celui de leurs classifications initiales.

DÉFINITION (produit) : Soient $S_1,\ldots S_n$ des systèmes classificatoires (pour $j\in[1,n]$, $S_j=\langle C_j \leq_j \ll_j \rangle$ est défini sur les langages L_C^j et L_I^j), le *produit* systèmes classificatoires $S=\times_{j=1}^n S_j$ est défini par $\langle \times_{j=1}^n C_j \leq \ll \rangle$ sur les langages $L_C=\times_{j=1}^n L_C^j$ et $L_I=\times_{j=1}^n L_I^j$ tels que :

(1) $\forall c=(c_1,\ldots c_n)\in\times_{j=1}^n L_C^j, \forall c'=(c_1',\ldots c_n')\in\times_{j=1}^n L_C^j, c \ll c' \Leftrightarrow \forall j\in[1\ n], c_j \ll_j c_j'$

(2) $\forall c=(c_1,\ldots c_n)\in\times_{j=1}^n C_j, \forall c'=(c_1',\ldots c_n')\in\times_{j=1}^n C_j, c \leq c' \Leftrightarrow \forall j\in[1\ n], c_j \leq_j c_j'$

(3) $\forall i=(i_1,\ldots i_n)\in\times_{j=1}^n L_I^j, \forall c=(c_1,\ldots c_n)\in\times_{j=1}^n L_C^j, c\in Cl(i,C) \Leftrightarrow \forall j\in[1\ n], c_j\in Cl(i_j,C_j)$

(4) $\forall i=(i_1,\ldots i_n)\in\times_{j=1}^n L_I^j, \forall c=(c_1,\ldots c_n)\in\times_{j=1}^n L_C^j, i\in I_R(c) \Rightarrow \forall j\in[1\ n], i_j\in I_R(c_j)$ $\Diamond$

Il faut noter que les langages L_I et L_C sont définis explicitement : ce sont les produits des langages initiaux. Par ailleurs, la définition de produit définit implicitement I_A (en définissant *Cl*) mais ne fait que contraindre I_R. La quatrième clause oblige à satisfaire $I_R(c) \subseteq I_A(c)$ mais n'est pas suffisante pour garantir l'inclusion des extensions sur $\langle C \leq \rangle$ (il est en effet possible d'avoir pour c(=[1 2]×{a})≤c'(=[1 3]×{a, b}), $I_R(c)$={(1,a), (2,a)} et $I_R(c')$={(1,a), (2,b), (3,b)}).

La définition du produit est purement formelle car elle définit I_R comme étant inclus dans le produit ce qui n'est pas très utile dans une application réelle. Mais une fois caractérisé ce produit, il est possible de le transformer (par restriction et dégénérescence). Une autre définition, moins stricte, contraint le résultat à être $\langle C \leq \ll \rangle$ tel que $C \subseteq \times_{j=1}^{n} C_j$, les conditions 2 et 3 étant remplacées par :

$$\forall c = (c_1,\ldots c_n) \in \times_{j=1}^{n} C_j, \forall c' = (c'_1,\ldots c'_n) \in \times_{j=1}^{n} C_j, c \leq c' \Rightarrow \forall j \in [1\ n], c_j \leq_j c'_j$$

Cette dernière définition intègre la possibilité de dégénérer et restreindre le résultat dans la

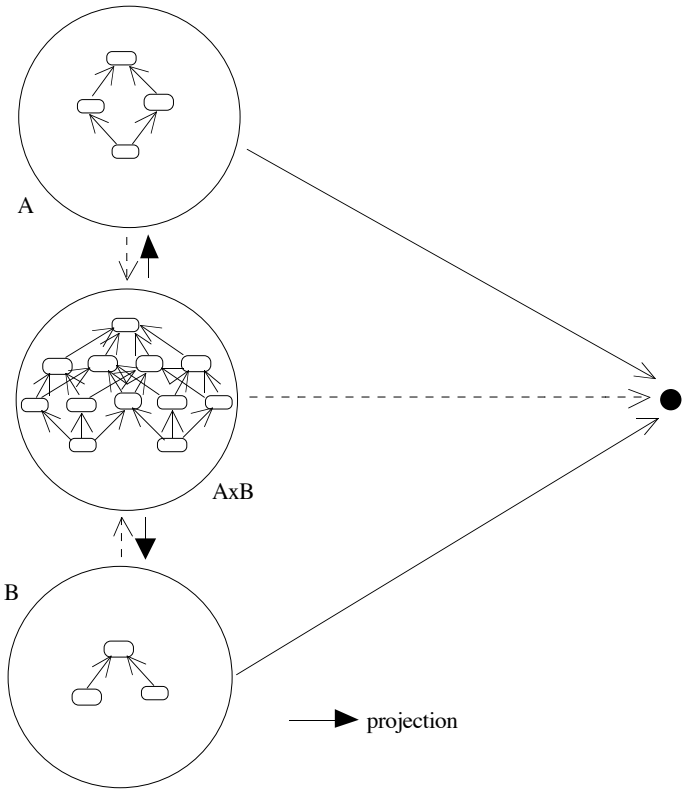


Schéma 8 : Construction de représentation par produit. À partir de deux systèmes classificatoires élémentaires, il est possible d'obtenir un système classificatoire produit reprenant les caractéristiques des deux premiers. Dans ce cas, le système produit est approximé par les deux systèmes initiaux, et tout ce qui est approximé par les deux l'est par le produit. Par ailleurs, il existe une paire de projections permettant de retrouver les systèmes classificatoires initiaux à partir du produit.

construction même du produit.

PROPOSITION : Un produit de systèmes classificatoires satisfaisant l'inclusion des extensions est un système classificatoire. ◊

Une manière de construire des systèmes classificatoires en fonction d'autres systèmes classificatoires ayant été définie, l'aspect attributif des objets doit être considéré.

DÉFINITION (produit disjoint) : Un produit *disjoint* de systèmes classificatoires est le produit de $S_1,\ldots S_n$ tel que pour tout j et k dans $[1\ n]$, $S_k \neq S_j$ ou $k=j$. ◊

Le système classificatoire vide, inutile dans la pratique, permet de définir très généralement les notions de projection associées au produit sans se soucier du cas où la projection est faite sur une dimension absente de l'objet considéré.

DÉFINITION (système classificatoire vide) : Le système classificatoire $\langle \emptyset\ \{\}\ \{\}\rangle$ est nommé *système classificatoire vide* et noté $\perp$. ◊

Le système classificatoire vide n'est pas associé à des langages de catégories ou d'individus (par convention). Une projection est une opération qui sélectionne un système classificatoire plus simple dans un produit de systèmes classificatoires. Ce système classificatoire peut être un produit plus simple. Si la sélection ne retient aucun facteur du produit initial, elle retourne le système classificatoire vide.

DÉFINITION (projection) : Une *projection* π est une fonction qui, à partir d'un produit de systèmes classificatoires $\times_{j=1}^{n} S_j$ retourne un autre système classificatoire S tel que :

— $S=\perp$, ou
— $S = \times_{j=1}^{m} S'_j (1 \leq m \leq n); \forall j \in [1\ m], (\exists k \in [1\ n]; S'_j = S_k) \wedge (\forall l \in [1\ m], S'_j = S'_l \Leftrightarrow j = l)$. ◊

Si $S=\perp$ pour tout produit de système classificatoire $\times_{j=1}^{n} S_j$, la projection est dite vide et si $m=1$, la projection est nommée projection unitaire et notée σ. L'opération de projection s'étend aux langages L_I et L_C, aux ensembles de catégories et aux relations de sous-catégorisation associées (c'est-à-dire que π est appliqué à tous les composants du système classificatoire et dénote donc les composants correspondants dans le système classificatoire résultant : si $S=\langle C \leq \ll\rangle$ sur L_I et L_C, $S'=\langle C' \leq' \ll'\rangle$ sur L'_I et L'_C et $S'=\pi S$, alors $\pi\ L_I = L'_I$, $\pi\ L_C = L'_C$, $\pi \leq = \leq'$, $\pi \ll = \ll'$ ces dernières expressions étant respectivement notées $\leq_\pi$ et $\ll_\pi$).

CONSÉQUENCE : Une projection π appliquée à un système classificatoire retourne un autre système classificatoire. ◊

CONSÉQUENCE : Un produit de n systèmes classificatoires définit implicitement 2^n projections (modulo l'ordre des facteurs dans les projections). ◊

DÉFINITION (orthogonalité) : Deux projections π_1 et π_2 sont dites *orthogonales* (ce qui est noté $\pi_1 \perp \pi_2$) si et seulement si leur composition appliquée à n'importe quel système classificatoire retourne le système classificatoire vide (pour tout produit de systèmes classificatoires S, $\pi_1\pi_2 S = \pi_2\pi_1 S = \perp$). ◊

Intuitivement, deux projections sont orthogonales si elles ne sélectionnent aucun facteur commun dans les produits.

DÉFINITION (produit homogène) : Un produit de systèmes classificatoires est dit *homogène* si les langages considérés $L_1^1,\ldots L_1^n$ sont les mêmes. Dans le cas contraire, le produit est dit *hétérogène*. ◊

La classification composite telle qu'elle a été définie dans TROEPS [Mariño 1991] peut être vue comme la classification dans un système classificatoire particulier. Ce système classificatoire est analysé récursivement en classifiant les valeurs de ses attributs dans leurs propres systèmes classificatoires. Ceci est intégré dans la notion de méronomie[6].

DÉFINITION (méronomie) : Une *méronomie* est un produit de systèmes classificatoires, éventuellement hétérogène. ◊

La notion de produit de système classificatoire oscille entre le produit cartésien (où le nommage des dimensions correspond à l'ordre, ce qui empêche la commutativité) et les enregistrements (où le nommage est réalisé par une étiquette ce qui permet de ne pas se soucier de l'ordre des dimensions). Cette position intermédiaire tient à ce que les systèmes classificatoires ont vocation à être utilisés à la fois pour construire les taxonomies et pour les utiliser ou les comparer, ce qui ne nécessite pas les mêmes contraintes (par exemple, afin de faire la fusion de deux bases de connaissance, on voudrait comparer deux classes dont les attributs n'ont pas les mêmes noms).

Ainsi, le produit peut faire référence plusieurs fois au même système classificatoire comme les objets peuvent avoir différents attributs avec le même type (c'est aussi le cas dans le produit cartésien traditionnel comme $\mathbb{R}^n$ ou dans le cas de la modélisation de l'opéron dans laquelle les gènes apparaissent comme répresseurs et activateurs). Qui plus est, les méronomies ne rendent pas seulement compte des attributs mais aussi des contraintes entre attributs (voir §I.3).

Le produit complet des taxonomies des composants sera la taxonomie maximale qui pourra être considérée dans la représentation du produit. Généralement, les systèmes de représentation de connaissance ne considèrent pas le produit complet des taxonomies comme la taxonomie du produit. Ils autorisent à sélectionner une partie de ce produit. Ceci peut être rendu à l'aide des opérations de restriction et de dégénérescence présentées plus haut.

Ainsi, à partir de la définition de système classificatoire totalement indépendant de la structure des individus considérés, il est possible de procéder à l'analyse de ces systèmes classificatoires en systèmes plus simples. Ceci s'applique aux systèmes de représentation de connaissance par objets (voir section suivante). Les avantages de cette approche sont principalement l'uniformité de la structure dans laquelle les opérations (comme la classification ou la catégorisation) sont appliquées : il s'agit toujours de systèmes classificatoires et certaines propriétés peuvent être propagées par ces opérations. Les algorithmes de classification sont, par exemple, applicables généralement à tout système classificatoire.

Les produits permettent d'imaginer plusieurs types d'algorithmes de classification : certains agissant d'abord sur une seule dimension (projection), d'autres agissant d'abord sur une classe.

---

[6] Le terme utilisé auparavant, « partonomie », tiré de [Tversky& 1984] qui le crédite à G. Miller, P. Johnson-Laird, Language and perception, Harvard university press, Cambridge (MA US), 1976, est abandonné pour des raisons de barbarisme.

L'un des intérêts de la construction par produit est de rendre compte de la possibilité d'ajouter un attribut à une famille d'objets sans rien perdre de l'ensemble initial. Pour cela il faut noter que les produits sont faits à plat : $\pi_B[(A\times B)\times C]=B$ (c'est-à-dire que le produit est associatif et commutatif). Les notions de produit et de projection permettent donc diverses opérations de complétion de la description d'une famille d'objets : l'ajout de nouveaux attributs. En effet, il est possible à tout moment d'ajouter un nouveau système classificatoire dans le produit. Le produit résultant est alors approximé par le produit initial (voir Schéma 8).

I.3. Intégration de types et de contraintes

Jusqu'à présent, les systèmes classificatoires sont des structures abstraites dont la décomposition s'exprime en termes de systèmes classificatoires. Ils ne définissent pas constructivement une structure sur laquelle faire des calculs. Le but des systèmes classificatoires étant de rendre compte du fonctionnement de systèmes de représentation de connaissance existants, il est nécessaire de leur donner un fondement à partir duquel les construire.

Le fondement nécessaire au système de représentation de connaissance TROEPS (développé au sein du projet SHERPA) est présenté ici. Ce fondement prend la forme d'un système de types abstraits permettant de prendre en compte des valeurs primitives (dit autrement, des systèmes classificatoires qui ne se décomposent pas en produits) et d'un système de contraintes. Dans les logiques terminologiques, par exemple, il prendrait la forme de taxonomies de concepts primitifs ; dans les graphes conceptuels, de taxonomies de types.

L'intérêt de ce qui est présenté ci-dessous est de considérer de manière modulaire la conception d'un système de représentation de connaissance sans que cela empêche de réaliser les inférences présentées dans les deux précédentes sections (classification, catégorisation, ajout d'attributs). Caractériser les types et les contraintes comme des systèmes classificatoires permet d'assurer la transition entre ces systèmes primitifs et les systèmes construits au sein de TROEPS.

I.3.1. Types

TROEPS considère des valeurs d'attributs qui ne sont pas des objets, mais requiert le typage de ces valeurs. Les valeurs doivent être décrites par un type abstrait de données [Capponi 1995]. Un type abstrait de données (ADT) T définit nécessairement un prédicat d'égalité entre valeurs du type ($x=_T y$ signifie que x et y sont la même valeur de l'ADT T), un prédicat de typage ($x\in T$ signifie que x est une valeur de l'ADT T). De surcroît, un ADT peut définir d'autres opérateurs sur ses valeurs : un ordre sur les types ordonnés ($x\leq_T y$ signifie que x est antérieur à y pour T), une succession pour les types discrets, etc. À partir de cette information, le système de type est capable de gérer correctement des valeurs et des expressions de type indépendamment de leur implémentation.

Par exemple, un utilisateur peut vouloir introduire un nouveau type abstrait `Date` dont les valeurs sont des triplets d'entiers : `(année, mois, jour)`. `Date` peut être explicitement déclaré comme un type discret et ordonné :

— L'utilisateur doit spécifier les deux prédicats requis $=_{\texttt{Date}}$ et $\in\texttt{Date}$.

— Puisque `Date` est ordonné, il doit spécifier la relation d'ordre entre deux dates $\leq_{\texttt{Date}}$.

— Puisque `Date` est discret (et totalement ordonné), l'utilisateur doit spécifier les opérations de successeur et de prédécesseur.

Le système de type fournit aussi les constructeurs de types polymorphes (dépendant de variables ou d'abstraction de types *T*) `liste` et `ensemble` qui permettront de considérer automatiquement les ADT *liste(T)* et *ensemble(T)*.

Jusqu'ici les ADT décrivent un ensemble de valeurs mais pas de taxonomie. Cependant, la description des entités TROEPS de restreindre la valeur des attributs dans les classes à l'aide de descripteurs de domaines : `domain` restreint les valeurs admissibles à faire partie d'un ensemble particulier, `sauf` à ne pas faire partie d'un ensemble particulier, `intervalles` à être compris dans une union d'intervalles de valeurs (pour les types ordonnés), `card` à avoir un cardinal compris entre deux bornes (pour les types collecteurs — `liste` et `ensemble`), etc. Pour prendre en compte ces expressions, le système de type procède à deux opérations principales :

— la *normalisation* qui établit une forme unique pour une expression de type particulière ;
— le *calcul de sous-typage* qui détermine si une expression de type est sous-type d'une autre.

La représentation interne (normalisée) des expressions de type dépend des caractéristiques de l'ADT considéré : par exemple, si le type `chaîne` (de caractères) est défini comme énuméré, les expressions de type seront des ensembles (ou une différence entre l'ADT et des valeurs à exclure) alors que, puisque `Date` est décrit comme discret, elles seront des unions d'intervalles de dates fermés et disjoints. De la même façon, les opérations de normalisation et de calcul de sous-typage dépendent uniquement des opérations procurées par l'ADT ($\in_T$, $=_T$ et $\leq_T$). Par exemple, le calcul de sous-typage pour les dates revient à la vérification de l'inclusion entre deux unions d'intervalles fermés.

Les expressions de types construits avec le même constructeur à partir du même ADT peuvent être organisées dans un treillis dont la relation d'ordre est la relation de sous-typage, dont l'élément maximal est l'ensemble des valeurs de l'ADT et l'élément minimal l'ensemble vide (normalisé).

Ces types sont les systèmes classificatoires élémentaires sur lesquels repose TROEPS et sur lesquels des systèmes classificatoires plus élaborés peuvent être construits par produit, etc. Pour un ADT *T*, L_I correspond à l'ensemble des valeurs du type (reconnues par $\in_T$) et L_C correspond à l'ensemble des classes d'équivalence des expressions de type pour la normalisation. L'ensemble des catégories *C* considérées à un instant sera le résultat de la normalisation des expressions de types communiquées au système de type par TROEPS augmenté de ceux qui sont nécessaires à maintenir la structure de treillis [Valtchev 1999a]. Le critère de sous-catégorisation est la relation de sous-typage sur l'ensemble des éléments de L_C alors que la relation de sous-catégorisation est la relation de sous-typage restreinte à *C*. Les interprétations réelles et abstraites coïncident et s'appliquent aux expressions de type normalisées. Ainsi, le système de type est capable d'associer à un type primitif *T* le comportement d'un système classificatoire c'est-à-dire de calculer le critère et la relation de sous-catégorisation.

Par exemple, le type `Date` détermine un système classificatoire dont L_I est l'ensemble des

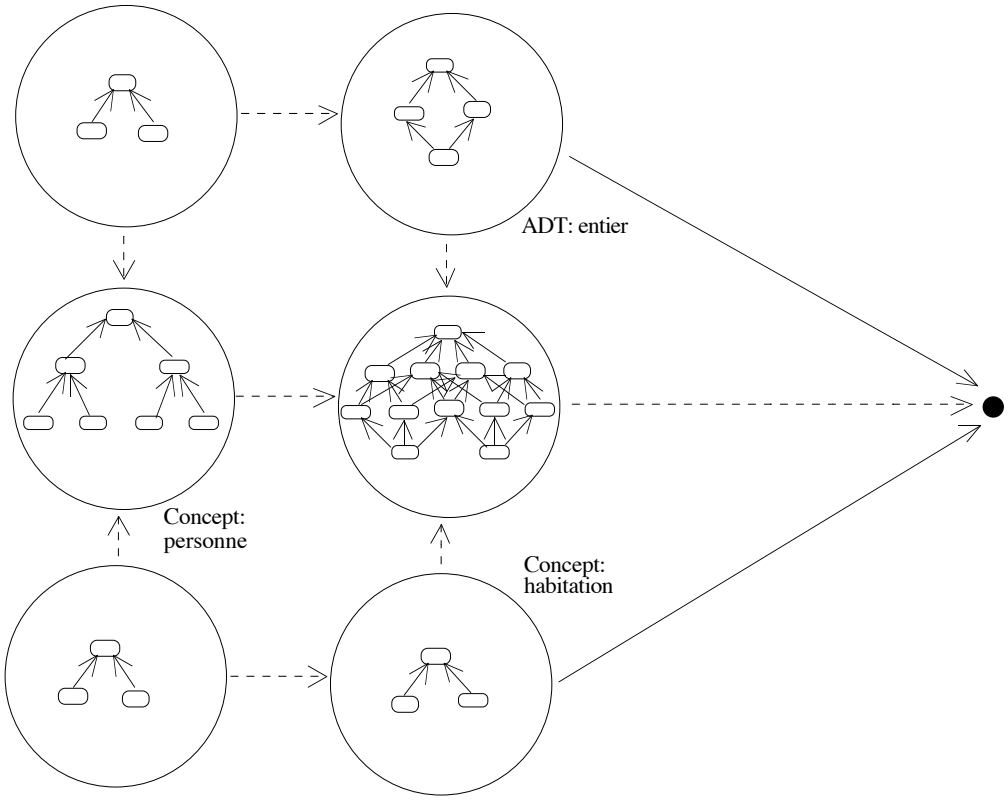


Schéma 9 : Produit adapté aux types et concepts. Un concept est construit par produit à partir de certains systèmes classificatoires existants qui peuvent être des concepts (autres produits de systèmes classificatoires) ou des types de données (systèmes classificatoires primitifs).

jours et L_C est l'ensemble des unions d'intervalles fermés et ordonnés (interprété comme l'ensemble des parties de L_I). Le critère de sous-catégorisation est le sous-typage et l'opération de normalisation assure que la taxonomie est exactement la restriction de $\langle 2^{L_I} \subseteq \rangle$ aux ensembles nécessaires pour produire une structure de treillis à partir des expressions de type utilisées.

Les travaux sur les types dans TROEPS sont allés encore plus loin [Capponi 1995], puisqu'ils ont permis la définition d'un type d'enregistrement [Cardelli& 1991] pour chaque classe (et un enregistrement — « record » — pour chaque objet) ces types s'exprimant aussi sous forme de systèmes classificatoires. Mais ce n'est pas le propos d'en parler ici.

Grâce à la notion de produit de systèmes classificatoires, tous les types décrits par un ADT peuvent être pris en compte dans un concept TROEPS.

I.3.2. Contraintes

Une contrainte est interprétée dans le cadre des problèmes de satisfaction de contraintes (CSP) [Mackworth 1977]. Un CSP considère un ensemble de variables $V=\{v_1,\ldots v_n\}$ avec un ensemble de domaines de valeurs associées $D=\{d_1,\ldots d_n\}$ et un ensemble de relations $R=\{r_1,\ldots r_m\}$. Une relation est interprétée comme un sous-ensemble du produit cartésien des domaines de ses variables ($d_{i_1}\times\ldots\times d_{i_p}$). Une contrainte $r_1(v_{i_1},\ldots v_{i_p})$ est une relation liant des variables de V qui est satisfaite si le produit des valeurs des variables appartient à la relation. Un

CSP est posé par un ensemble de telles contraintes et il est résolu par une affectation à chaque variable d'une valeur dans son domaine satisfaisant toutes les contraintes.

En TROEPS des contraintes peuvent être définies et appliquées aux champs d'un objet (il est considéré qu'appliquer une contrainte à des attributs d'objets sans relation n'est pas nécessaire — ou, autrement dit, que ces objets doivent entretenir une relation). Au sein d'une classe TROEPS, une contrainte est une assertion signifiant qu'un ensemble d'attributs satisfont la contrainte en question. Les variables sont donc des attributs (ou des chemins permettant d'accéder à un attribut dans un objet référencé), les domaines sont les expressions de type associées à ces attributs et les relations sont des contraintes exprimées (en compréhension) dans un langage de définition de contraintes classique [Gensel 1995a, b]. Ces contraintes sont des conditions nécessaires au même titre que les contraintes de typage. Elles doivent donc être satisfaites par les objets appartenant à la classe sur laquelle elles sont posées. Le langage de description de contraintes et ses facilités de résolution ne seront pas détaillés ici (voir [Gensel 1995b]).

Afin de garantir le sous-typage des attributs contraints d'une classe vers ses sous-classes, les contraintes doivent être héritées. Ainsi, l'ensemble des contraintes portant sur une classe est l'union des contraintes portant sur ses super-classes et de ses contraintes propres. Cependant, ce mécanisme d'héritage peut être gênant lorsque des contraintes redondantes (ou en relation de généralité) sont définies dans les super-classes. Une contrainte ajoutée sur une classe est dite redondante par rapport à un ensemble de contraintes si elle peut être obtenue par simplification ou dérivation de ce dernier. En fait, la redondance peut être définie comme l'inclusion entre les extensions des relations. Ainsi, si deux prédicats n-aires P et Q définis sur le même produit cartésien $d=d'_1\times\ldots\times d'_n$ sont tels que l'ensemble des n-uplets satisfaisant P est un sous-ensemble des n-uplets satisfaisant Q alors une relation de sous-typage $\leq_d$ entre P et Q (telle que $P\leq_d Q$) peut être considérée sur d. Cette relation permet d'organiser les contraintes en une taxonomie.

TROEPS est capable de tester la satisfaction des contraintes par un n-uplet de valeurs, mais il ne peut comparer les extensions des prédicats (simplement parce qu'elles peuvent être infinies) ni leurs intensions. Il doit donc être informé des relations de sous-catégorisation entre contraintes. L'utilisateur peut fournir cette information au système en déclarant une taxonomie de contraintes (que TROEPS ne maintient pas automatiquement : l'utilisateur est donc libre d'organiser les contraintes comme il l'entend).

Un système classificatoire lié aux contraintes est ainsi défini : une contrainte particulière r sur les variables $v'_1,\ldots v'_p$, dont les domaines sont $d'_1,\ldots d'_p$, est facilement introduite dans un système classificatoire dont L_I est le produit cartésien des domaines $d=d'_1\times\ldots d'_p$ et L_C est l'ensemble des prédicats qui reconnaissent des sous-ensembles de d (sous-ensembles de L_I). C est simplement l'ensemble des prédicats exprimés (il contient donc r). À l'instar des types, les interprétations réelles et abstraites coïncident et ont pour domaine d. Le critère de sous-catégorisation est interprété comme l'inclusion de l'extension des prédicats (qui n'est pas accessible par le programme) et la relation de sous-catégorisation est sa restriction à C (exprimée par $\leq_d$). Ainsi, il est possible d'utiliser ce système classificatoire dans les opérations de classification : même si le système utilise un algorithme de satisfaction de contraintes classique

pour déterminer si r est satisfait par un n-uplet d'attributs (c'est-à-dire si un individu appartient à l'interprétation abstraite d'une catégorie), le comportement abstrait du système classificatoire subsiste.

I.3.3. Conclusion

Les facilités d'extension du système de représentation de connaissance TROEPS ont été brièvement présentées. Les définitions d'ADT sont relativement simples par rapport aux expressions de types qu'elles autorisent. Ainsi TROEPS peut manipuler des expressions externes au langage de représentation de connaissance de manière minimale et sans avoir à accéder à la structure des valeurs. Les types et les contraintes ont été caractérisés en tant que systèmes classificatoires. Des systèmes classificatoires plus élaborés tels que les concepts de TROEPS sont justifiés uniquement parce qu'ils reposent sur la combinaison (par produit, restriction et dégénérescence) de ces systèmes classificatoires plus simples. Ainsi, grâce au formalisme des systèmes classificatoires, TROEPS est capable de réaliser la classification, la catégorisation, etc.

Le problème de l'extensibilité due au système de type a été considéré par d'autres dans le cadre des logiques terminologiques [Baader& 1991, Borgida& 1992, Gaines 1993], mais l'extensibilité des mécanismes considérés n'est pas aussi générale que celle des systèmes classificatoires.

Par contre, et principalement en raison de la spécification de la subsomption sous forme de déduction naturelle [Napoli 1998], l'extensibilité par le biais de constructeurs (correspondant en partie ici aux constructeurs et descripteurs) est plus générale dans les logiques terminologiques.

I.4. Inférence de taxonomies

Jusqu'à présent la contribution des systèmes classificatoires a été mise en évidence dans la manière d'intégrer, autour de la notion fondamentale de classification, les différents éléments qui ont toujours été présents dans les systèmes de représentation de connaissance. Cette dernière partie est consacrée à la fonction d'inférence de taxonomie (aussi dénommée classification en analyse de données et "clustering" ou classification conceptuelle en apprentissage).

Il est possible d'interpréter les systèmes classificatoires d'une manière immédiate en termes d'ensembles. Cela correspond à l'interprétation fournie dans [Leuschner 1991] de l'opération de classification en analyse de données qui construit une taxonomie à partir d'un ensemble d'individus. Cette taxonomie est une réduction (restriction et/ou dégénérescence) « adéquate » du treillis de l'ensemble des parties de l'ensemble d'individus. Dans cette interprétation, L_I est l'ensemble des individus, L_C est l'ensemble de ses parties (2^{L_I}) et $\ll$ est la relation d'inclusion ensembliste ($\subseteq$). Par conséquent $C \in Cl(i)$ si et seulement si $i \in C$. $\langle C \leq \rangle$ est alors construit par restriction et dégénérescence de $\langle 2^{L_I} \ll \rangle$. L'opération de classification peut être ascendante (ou par agglomération) en construisant des classes rassemblant de plus en plus d'individus, soit descendante (ou par partition) en partitionnant les classes en de plus petits ensembles. Il serait sans doute possible de raffiner cette caractérisation à l'aide des treillis de Galois.

Maintenant, le problème a été seulement circonscrit et deux questions se posent : que constitue une réduction adéquate de ≪ et comment en obtenir une ? Encore une fois, l'idée ici n'est pas de donner une réponse unique mais de délimiter un ensemble de représentations

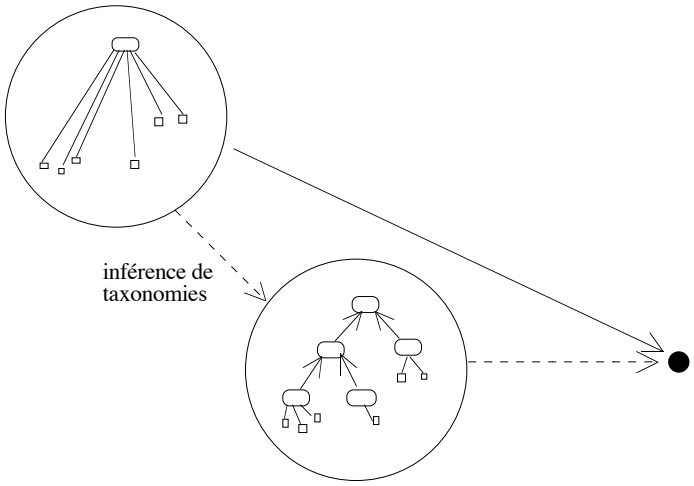


Schéma 10 : L'inférence de taxonomie comme approximation. La construction de taxonomie là où elle n'existait pas préserve ce qui existait : les objets et leur appartenance à la classe racine. Une fois la taxonomie construite, la représentation est plus précise puisqu'elle fait intervenir des classes supplémentaires et que les objets sont attachés à des classes plus spécifiques (préservant leur attachement initial). La représentation initiale approxime donc la représentation obtenue lors de l'inférence de taxonomies.

approximées par une représentation initiale et acceptables pour certains critères. Dans le cadre de la classification (resp. de la catégorisation), le critère d'acceptabilité, représenté par l'interprétation abstraite (resp. le critère de sous-catégorisation), est donné dans la structure de système classificatoire. En ce qui concerne l'inférence de taxonomie, il n'existe pas de tel critère dans le système classificatoire. Aussi, faut-il concevoir une mesure de dissimilarité entre individus d'un système classificatoire afin de délimiter les taxonomies les plus acceptables.

DÉFINITION (dissimilarité, distance, distance arborée, normalisation) : Soit un ensemble C, une (mesure de) *dissimilarité* est une fonction $d : C \times C \longrightarrow R^+$ respectant les propriétés suivantes :

— $d(x, y)=0 \Leftrightarrow x=y$ (minimalité)
— $d(x, y)=d(y, x)$ (symétrie)

une *distance* est une mesure de dissimilarité respectant en plus :

— $d(x, y)+d(y,z)\geq d(x, z)$ (inégalité triangulaire)

lorsque C est muni d'une structure d'arbre, une *distance arborée unitaire* est la distance définie par [Barthélémy& 1988] :

— $d(x, y)$=le nombre d'arêtes dans l'arbre entre x et y

une telle mesure est dite *normalisée* si son co-domaine est l'intervalle [0 1]. ◊

Le résultat de cette première réflexion a amené à la conception de T-TREE [Euzenat 1993c] : un simple algorithme d'inférence de taxonomie ascendant intégré au sein de TROEPS. Ce système implémentait des algorithmes de type « plus proches voisins ». La distance utilisée ne prenait cependant en compte que des attributs numériques, d'où l'idée de développer une notion de dissimilarité plus générale qui puisse s'appliquer à TROEPS et à son système de types extensible.

Elle pourrait alors prendre en compte des objets, des valeurs et des ensembles ou des listes de ceux-ci. Le point commun entre ces notions étant le système classificatoire, il devenait naturel de développer une notion de distance pour les systèmes classificatoires. Cela pose cependant quelques problèmes :

— il faut tout d'abord savoir comment définir une telle dissimilarité et comment la justifier ;
— si elle doit s'appliquer à tout système classificatoire, il faut qu'elle ait une pertinence pour les types et les contraintes qui viennent d'être présentés ;
— enfin, il faut l'articuler avec les notions des systèmes classificatoires.

C'est l'objet des trois sections suivantes.

I.4.1. Dissimilarité dans une taxonomie

Il est nécessaire de développer une mesure de dissimilarité permettant de mesurer la différence entre les classes auxquelles sont attachés les objets. Telle que présentée plus haut, la notion principale disponible au sein des systèmes classificatoires est la taxonomie. Aussi, lorsque la question se pose de décider d'une structure sur laquelle appuyer la dissimilarité, la taxonomie vient naturellement à l'esprit.

Certains considèrent qu'il n'est pas légitime de réduire les objets aux classes auxquelles ils sont attachés lors de l'inférence de taxonomie et préfèrent considérer plutôt la structure interne des objets et les autres objets avec lesquels ils sont en relation. C'est ce qui sera fait lors de la constitution d'une taxonomie (voir §I.4.4). Mais quelle est alors l'utilité de la construction d'une taxonomie si, à son tour, elle ne rassemble pas dans les classes les objets similaires ? Il faut par ailleurs noter la réduction de coût apportée par cette méthode qui ne considère que la place occupée dans la taxonomie et non la structure interne des objets (dont les valeurs peuvent faire référence à d'autres objets, etc.).

Si cet argument permet de parler de la similarité, il ne dit rien de la mesure de similarité. Elle est définie ici comme la longueur normalisée du plus court chemin entre les classes des deux objets dans la taxonomie [Valtchev& 1996]. La normalisation étant la longueur maximale entre deux classes au sein de la taxonomie.

DÉFINITION (dissimilarité topologique) : Soient x et $y \in C$, $dist(x,y)$ le nombre minimal d'arcs entre x et y, une *dissimilarité topologique* entre x et y est définie par :

$$\delta(x,y) = \min_{c; x \leq c \wedge y \leq c} (dist(x,c) + dist(y,c))$$

La dissimilarité topologique est normalisée (s'il existe au moins deux individus) par la dissimilarité maximale entre deux individus dans le système classificatoire :

$$\bar{\delta}(x,y) = \frac{\delta(x,y)}{\max_{x',y' \in C} \delta(x',y')} \qquad \Diamond$$

Il faut noter que les x et y ci-dessus ne sont pas des objets mais des classes. En effet, pour que la mesure soit valable pour les objets comme pour les classes, les objets sont considérés

comme des classes n'ayant qu'un seul élément attaché. Ainsi, le seul moyen d'avoir $d(x,y)=0$ est que $x=y$.

PROPOSITION : Si la taxonomie est un arbre ou un treillis, la dissimilarité topologique est une distance. Si c'est un arbre, il s'agit d'une distance arborée. ◊

Cette dernière proposition est intéressante parce que beaucoup d'algorithmes de classification considèrent des distances et que les taxonomies dans TROEPS sont des arbres et celles des ADT (ou des contraintes) des treillis.

I.4.2. Passage aux types et constructeurs

La mesure définie plus haut coïncide, sur plusieurs sortes de types introduits dans TROEPS, avec des mesures classiquement utilisées en analyse de données.

Ainsi, pour un domaine de valeur nominal (nommé énuméré en TROEPS, c'est-à-dire sans plus de structure que d'être un ensemble) la distance normalisée classique est celle qui retourne 0 lorsque les éléments sont identiques et 1 lorsqu'ils sont différents. Le système classificatoire correspondant à ce genre de domaine D est celui où L_C est l'ensemble des parties du domaine. Ainsi, pour deux valeurs quelconques v et v', le plus court chemin passe par $\{v, v'\}$. La valeur est alors égale à 2 (mais comme ceci est vrai pour tout couple de valeurs différentes, une fois normalisée, la dissimilarité vaut 1).

De même, lorsque le domaine D est ordinal et fini, les catégories peuvent être des intervalles. Ainsi la taxonomie est faite de tous les intervalles possibles et les plus courts chemins entre deux valeurs doivent atteindre l'intervalle dont les deux valeurs à comparer sont les bornes. Ainsi, $\delta(v,v') = 2.|\,ord(v) - ord(v')\,|$ (où ord est une fonction donnant le rang d'un élément). Ceci est normalisable à l'aide des valeurs les plus grandes et les plus petites du domaine.

Des problèmes apparaissent lorsque les domaines sont non bornés (les valeurs effectivement exprimées dans l'environnement peuvent alors être prises comme bornes), denses ou continus (par convention la distance euclidienne normalisée comme précédemment est utilisée) ou que le langage ne considère plus des intervalles mais des unions d'intervalles comme domaines. Petko Valtchev [1999b] a récemment généralisé cette mesure à la notion d'étendue relative d'un domaine qui mesure le rapport entre l'étendue du domaine et celle du type considéré et se démarque légèrement des systèmes classificatoires.

En général, soit parce que ces mesures classiques se calculent plus rapidement de manière directe que par la considération des taxonomies, soit parce qu'une dissimilarité particulière est préférables, la mesure de dissimilarité peut être redéfinie dans l'ADT en y ajoutant l'opérateur correspondant.

Enfin, sur la base de ces dissimilarités associées à un système classificatoire, Petko Valtchev [Valtchev& 1997] a défini la dissimilarité sur les listes ou ensembles d'éléments de ces systèmes classificatoires. Elle est exprimée en fonction d'un couplage (qui associe des éléments d'un ensemble avec des éléments d'un autre ensemble) maximal (associant le nombre maximal d'éléments) entre deux ensembles (resp. listes) minimisant la somme des dissimilarités entre objets couplés. Dans le cas des listes, ce couplage doit bien entendu respecter l'ordre. Ces

mesures sont donc fondées sur la notion de système classificatoire ; elles s'appliquent donc aux listes et ensembles d'objets et elles construisent une mesure de dissimilarité.

I.4.3. Agrégation d'objets

Une mesure de dissimilarité permet maintenant de comparer divers objets inscrits dans une taxonomie. Mais s'il faut construire une taxonomie, c'est-à-dire comparer des objets qui ne sont pas inscrits dans une telle taxonomie, il est nécessaire de déterminer une dissimilarité qui ne va plus dépendre de la position dans la taxonomie mais des valeurs d'attributs (c'est-à-dire des facteurs d'un produit de systèmes classificatoires).

TROEPS dispose de trois sortes d'attributs :

propriété qui fait référence à de simples valeurs.

composant qui fait référence à des objets qui composent l'objet courant. Cette relation ne peut être circulaire.

lien qui permet d'exprimer des relations — autre que la composition — entre divers objets et autorise donc la circularité dans ces relations.

Seuls les deux premiers aspects seront abordés ici parce que l'aspect circulaire n'est pas pris en compte dans la notion de produit de systèmes classificatoires et parce que le travail à ce sujet n'est pas abouti. Cependant, il faut noter que dans le cadre de l'inférence de taxonomie, des solutions ont été proposées [Bisson 1995, Valtchev 1998].

La dissimilarité utilisée par les algorithmes d'inférence de taxonomies sera donc une fonction agrégeant, possiblement de manière pondérée, les dissimilarités en fonction des différents facteurs du produit de systèmes classificatoires. Elle doit s'appliquer aux propriétés comme aux composants. Elle est définie ainsi :

DÉFINITION (dissimilarité construite) : Soit un produit de n systèmes classificatoires, soient o et $o' \in L_I$ dans ce produit. Si *aggr* est une fonction d'agrégation et λ_i le poids associé au facteur i (isolé par la projection π_i), une dissimilarité entre o et o' est définie par :

$$d(o,o') = \underset{i=1}{\overset{n}{aggr}}\, \lambda_i . \bar{\delta}(\pi_i(o),\pi_i(o')) \qquad \Diamond$$

La d'agrégation peut être une somme (ce qui ramène à une mesure de type Manhattan, ou une distance de type Minkowski — qui peut être la distance euclidienne). Bien entendu, la mesure obtenue restera alors une mesure de dissimilarité.

Le cas où certaines valeurs ne sont pas connues est traité de manière non totalement satisfaisante en reportant tout sur les autres poids (ceci pose problème lorsque aucune valeur n'est connue). D'autres solutions guère plus satisfaisantes consistent à utiliser une valeur pour la dissimilarité : en général le maximum sinon la moyenne des valeurs (et si aucune valeur n'est disponible l'attribut est ignoré).

I.4.4. Construction de taxonomies

Dans TROEPS, en l'absence de circularité, les concepts sont virtuellement structurés en un graphe de dépendance (suivant que l'un est le type d'un attribut de l'autre) qui est un graphe

orienté sans circuit dont les feuilles sont les ADT. Puisque ces ADT sont déjà pourvus d'une taxonomie, il est aisé d'établir la dissimilarité entre les éléments de l'ADT. Ainsi, les concepts ne dépendant que d'ADT (munis d'une taxonomie) peuvent à leur tour être munis d'une taxonomie et ce processus peut être itéré.

Un algorithme complet d'inférence de taxonomie pour TROEPS peut donc être décomposé ainsi [Valtchev& 1996] :

`Build` est un parcours post-fixé du graphe de dépendances qui appelle l'algorithme `Cluster` sur un concept, uniquement lorsque tous ses successeurs dans le graphe disposent d'une taxonomie ; il construit donc une taxonomie pour chaque concept ;

`Cluster` appliqué à un concept lance `Partition` sur l'ensemble des objets du concept puis sur chaque classe ainsi construite jusqu'à ce que les classes ne soient plus partitionnables ; il construit donc une taxonomie sur les objets d'un concept ;

`Partition` est un algorithme qui à partir d'un ensemble d'objets en développe une partition en fonction de la dissimilarité entre ces objets ; il permet donc d'obtenir une division des objets attachés à une classe en sous-ensembles disjoints destinés à devenir eux aussi des classes.

Il existe de nombreuses méthodes pour obtenir une partition en accord avec une dissimilarité. Certaines ont été implémentées et intégrées dans le cadre de TROEPS : une version modifiée de CLUSTER [Michalski& 1983] et les algorithmes classiques de plus proches voisins ("Single linkage/multiple linkage") [Day& 1984] qui agglomèrent au lieu de partitionner.

La classe obtenue n'est qu'un ensemble d'objets. Il faut donc une opération de généralisation pour décrire les classes obtenues. Elle n'est pas détaillée ici (voir [Valtchev 1999b]).

Conclusion

Cette première partie n'a considéré que d'une manière très lointaine les relations entre diverses représentations. Il est donc nécessaire de rappeler que la notion d'approximation est essentielle dans la possibilité de disposer de plusieurs représentations du même domaine. Cette approximation peut être mise en œuvre de manière volontaire en ayant plusieurs niveaux de résolution des représentations ou de manière implicite lors de la construction d'une représentation complétée par incrément. L'introduction d'un formalisme modulaire tel que les systèmes classificatoires est un premier pas en ce sens : cela permet de délimiter l'espace des représentations approximées dans le cadre d'un langage précis; mais cela permet aussi de définir les transformations de cet espace (ajout d'attributs à l'aide du produit, ajout de classes par catégorisation…).

La contribution principale de l'ensemble des travaux présentés ici est de pouvoir appréhender de manière homogène l'ensemble des composantes entrant dans les représentations par objets. Exprimer les problèmes en termes abstraits tels que l'autorise la notion de système classificatoire a donc permis de réels progrès dans l'intégration de plusieurs notions au sein des représentations de connaissance par objets. L'intérêt de l'approche est qu'une fois une notion caractérisée en tant que système classificatoire, il est possible de l'utiliser dans la construction de nouveaux systèmes classificatoires et de disposer d'algorithmes s'appliquant à tout système classificatoire. L'implémentation du système TROEPS est fondée sur ces notions. Elle permet de

développer des algorithmes de classification, de catégorisation et d'inférence de taxonomie indépendants du type d'objet considéré.

Le sujet des systèmes classificatoires n'est pourtant pas épuisé. En effet, le chapitre suivant les présentera dans la prise en compte d'un des aspects les moins classiques du système TROEPS : la notion de point de vue qui permet de considérer simultanément deux représentations concurrentes d'un même domaine.

Mes travaux sur ces systèmes classificatoires se sont clos en 1995 par manque de disponibilité. L'une des notions faisant défaut à l'approche est un constructeur permettant d'exprimer la récursivité (ou plus simplement le fait qu'un objet puisse être « construit » à partir de lui-même).

Crédits

La notion de système classificatoire constitue ma contribution propre. Bien que ces travaux n'aient pas donné lieu aux développements que j'aurais aimé poursuivre, ils ont été un aiguillon pour que Roland Ducournau développe ses propres travaux sur des « systèmes classificatoires » tout différents (correspondant à une approche plus classique, et plus proche des objets, de la sémantique des représentations de connaissance) [Ducournau 1996]. À leur tour, ces travaux ont permis d'améliorer la sémantique et l'implémentation de TROEPS.

L'intégration de types et de contraintes au sein des systèmes classificatoires est le résultat de la collaboration avec Cécile Capponi et Jérôme Gensel alors doctorants au sein du projet SHERPA. Elle a bénéficié du travail de DEA de Shong-Ye Tan [1993] que j'ai encadré. Enfin, les travaux sur la catégorisation ont été principalement développés par Petko Valtchev, en thèse sous ma direction.

II. POINTS DE VUE ET REPRÉSENTATION PAR OBJETS

La représentation d'organisations alternatives de la connaissance n'a pas été considérée dans le chapitre précédent. Ce second chapitre montre quels problèmes l'introduction de taxonomies alternatives pose à la représentation de connaissance par objets (§II.1) et comment ils sont résolus dans le système TROEPS.

Le système TROEPS est original puisqu'il présente, en particulier, une notion de points de vue (§II.2) permettant d'organiser le même ensemble d'objets sous plusieurs taxonomies simultanément. Cela conduit à une réorganisation de la représentation de connaissance puisque les objets ne dépendent plus d'une unique classe à laquelle ils sont attachés définitivement. Cette réorganisation introduit deux niveaux, nommés ici ontologique et taxonomique (§II.1), contribuant à résoudre les problèmes d'évolution, d'interprétation des classes, de nommage et d'identité posés aux langages de programmation d'une part et aux représentations de connaissance d'autre part.

La notion de point de vue autorise l'expression de différentes taxonomies sur le même ensemble d'objets mais ne permet pas leur confrontation. Cette confrontation est partiellement autorisée par la notion de passerelle dénotant (pour simplifier) l'équivalence entre classes de différents points de vue. Comme souvent en représentation de connaissance, cette notion peut être utilisée comme une contrainte ou un mécanisme d'inférence. En tant que contrainte, elle permet de mettre en évidence l'inconsistance entre deux points de vue. En tant que mécanisme d'inférence, deux algorithmes d'inférence de passerelles permettent de confronter différents points de vue obtenus par classification automatique (§II.3). Le chapitre IV présentera des manières plus générales de confronter les points de vue et les autres éléments de TROEPS.

Problème : intégration de points de vue dans les objets

L'idée de diviser une représentation en modules représentant chacun un point de vue peut apporter beaucoup. Elle permet de construire une représentation relativement complexe à partir de briques de base indépendantes les unes des autres. L'un des avantages du point de vue est de présenter une version simplifiée de la connaissance. Cela permet de diminuer la charge pour l'utilisateur comme pour un programme.

Un premier problème consiste donc à décider de ce qui va être entendu par point de vue dans un cadre particulier. Dans les représentations par objets, il est possible de considérer diversement cette notion mais ici un point de vue sera une taxonomie sur un ensemble d'objets particulier : il peut donc y avoir plusieurs taxonomies sur le même ensemble d'objets. Par ailleurs, les points de vue peuvent ne pas être totalement indépendants, ce qui est rendu en TROEPS par la notion de passerelle permettant d'indiquer qu'une conjonction de classes sous un ensemble de points de vue est plus spécifique qu'une classe sous un autre point de vue.

Appliqué à l'exemple biologique introduit plus haut, ce travail s'applique naturellement à l'organisation d'un ensemble de gènes, par exemple, suivant leur fonction (il y aura alors des gènes dont le produit intervient dans la structure, dans le métabolisme ou dans la régulation), leur structure tridimensionnelle (il y aura des hélices, des feuillets alpha ou bêta…) ou leur localisation (les gènes sont alors organisés suivant leur chromosome, puis leur branche par rapport au centromère et enfin leurs bandes, voir Figure 1). Suivant le centre d'intérêt du biologiste une organisation sera utile et une autre inutile. De plus, il peut éprouver le besoin

d'exprimer qu'un type de gène intervenant dans une maladie génétique particulière (c'est-à-dire avec une fonction particulière) devra être localisé dans un chromosome précis. Ceci pourra être

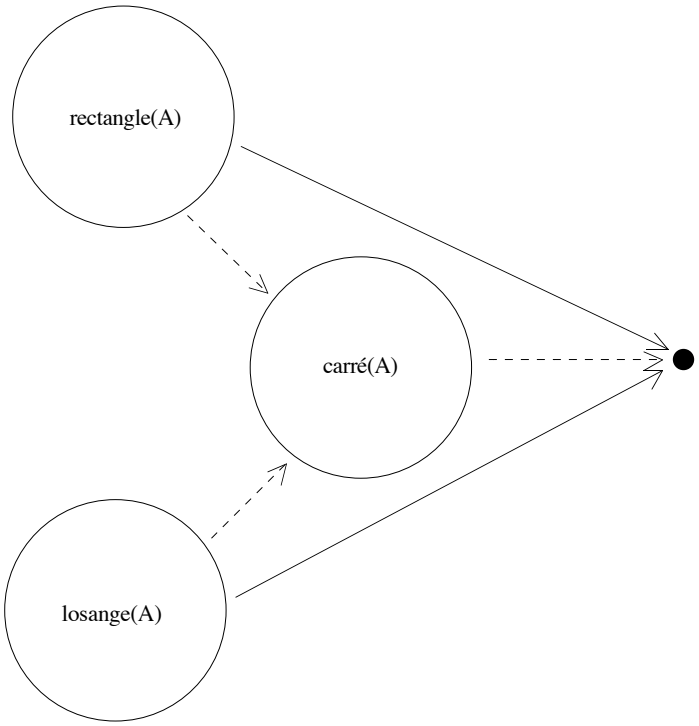


Schéma 11 : Points de vues comme produits. Sous deux points de vue différents, consacrés par exemple à un aspect angulaire et à un aspect métrique, l'objet A est vu comme un rectangle ou un losange. La mise en correspondance de ces deux points de vue permet de partager l'information et d'envisager A comme un carré.

exprimé sous forme de passerelle.

Cependant, l'intégration de la notion de points de vue au sein des représentations de connaissance par objets constitue un problème, car cela demande de reconsidérer plusieurs principes des langages de programmation et de représentation de connaissance par objets :

— un objet n'est pas indissociablement lié à une classe particulière,

— un objet peut évoluer sans que le contrôle soit complètement perdu de la part du système.

Les réponses apportées par l'implémentation de TROEPS à ce problème sont présentées ci-dessous (§II.1).

Le formalisme des systèmes classificatoires — et les opérations présentées au chapitre précédent — doit de plus être enrichi afin de permettre de rendre compte des points de vue et des passerelles (§II.2.2). Ceci conduit à envisager un atelier complet de manipulation de taxonomies intégrant inférences de taxonomies et de passerelles (§II.3).

Perspective historique

La notion de point de vue est, en général, considérée comme une notion annexe mais se trouve très développée dans certains domaines. Par exemple, en bases de données, la notion de vue, implicite dès la création du modèle relationnel [Codd 1970], est implémentée astucieusement peu de temps après [Schmid& 1975]. Elle permet de construire des tables (qui peuvent être considérées en première approximation comme des classes) à partir d'autres tables

existantes. Ces vues peuvent être obtenues par projection (ce qui en fait des vues plus simples) mais aussi par jointure (ce qui en fait des vues plus complexes que chacune des tables initiales mais plus simples que leur ensemble). Cette notion de vue se prolonge avec plus de difficulté dans le domaine des bases de données à objets [Abiteboul& 1991]. La notion de vue est aussi importée en représentation de connaissance dans le langage KRL [Bobrow& 1977] où elle ne concerne que les objets individuellement. Stéphane Le Peutrec a récemment [1998] décrit un ensemble de mécanismes pour abstraire la réponse à une requête dans une base d'objets (par suppression d'attributs, de classes ou rattachement à une classe plus générale). Il définit un langage permettant de spécifier l'abstraction à créer (ou la fonction de création des abstractions). Les opérations réalisent donc l'opération inverse de ce qui a été présenté dans le chapitre précédent. Plutôt que de considérer comment engendrer une vue en fonction de l'information disponible, ce chapitre considère l'agrégation de plusieurs vues décrites indépendamment. Cette notion se trouve aussi dans les méthodologies de spécifications de logiciels multi-points de vue [Nusibeh& 1994].

La notion de point de vue dans les langages de programmation par objets est sans doute due à Bernard Carré [Carré& 1988, Dugerdil 1988]. Les points de vue sont alors utilisés pour orienter la résolution de problèmes d'héritage multiple : suivant le point de vue adopté, l'héritage se fera dans un ordre ou dans un autre. Ils portent sur une entité plus complexe que la classe puisqu'il s'agit d'une taxonomie dont une sous-taxonomie est extraite afin de pouvoir la parcourir plus confortablement.

VIEWS [Davis 1987] est un système d'intelligence artificielle ambitieux qui met en œuvre des représentations diverses nommées vues. Il est possible de partager des objets entre vues et d'attacher des contraintes entre objets de différentes vues. Mais le système n'accorde pas de sémantique particulière aux objets manipulés.

Dans les systèmes d'intelligence artificielle, la notion de point de vue est présente au sein des systèmes de tableau noir où la vision de chaque spécialiste est donnée dans une source de connaissance [Nii 1986]. C'est en partie ce qui a inspiré Olga Mariño pour concevoir le système de points de vue de TROEPS [Mariño& 1990] comme la vision de différents spécialistes sur le même domaine.

Une autre source d'inspiration est la notion de taxonomie en biologie. Alors qu'il est fréquent de parler de la taxonomie des espèces (et il est vrai qu'il existe une taxonomie standardisée au niveau international), il s'avère que le biologiste ne répugne pas à utiliser de nombreuses taxonomies alternatives telles que les clés d'identification [Pankhurst 1991, Lebbe 1998] ou des taxonomies locales destinées à être intégrées plus tard aux taxonomies standards [Beach& 1993]. Pourtant, toutes ces taxonomies concernent bien le même ensemble d'individus. La multiplicité des taxonomies biologiques conduit rapidement à adopter une attitude relativiste vis-à-vis de ces classifications (qui tranche d'autant plus remarquablement avec la volonté de construire des « ontologies » [Guarino 1998] que le statut des taxonomies biologiques a été un sujet de controverse brûlant ces dernières décennies [Mayr 1981]). C'est une des sources d'inspiration de François Rechenmann pour introduire les points de vue tels qu'ils se présentent dans TROEPS.

Les points de vue de TROEPS autorisent la construction de plusieurs taxonomies sur le même ensemble d'objets. C'est donc une notion plus puissante que celle présente dans les bases de

données car au lieu de proposer une classe alternative elle propose toute une hiérarchie de classes alternatives.

Mon travail personnel à ce sujet n'a fait que suivre cette pente. Il a consisté à examiner les problèmes posés par l'implémentation de ces points de vue dans le contexte des langages de représentation par objets et à réexaminer la pertinence des notions utilisées dans TROEPS [Sherpa 1995] afin d'en faire un tout cohérent. À noter que dès 1986, une analyse de la notion de point de vue a été faite dans un cadre logique [Hautamäki 1986] qui partage de nombreux points communs avec les systèmes classificatoires.

Rappel : points de vue en représentation par objets

La notion de points de vue est l'originalité la plus marquante du modèle TROEPS développé au sein du projet SHERPA à l'origine sous l'impulsion d'Olga Mariño et de François Rechenmann

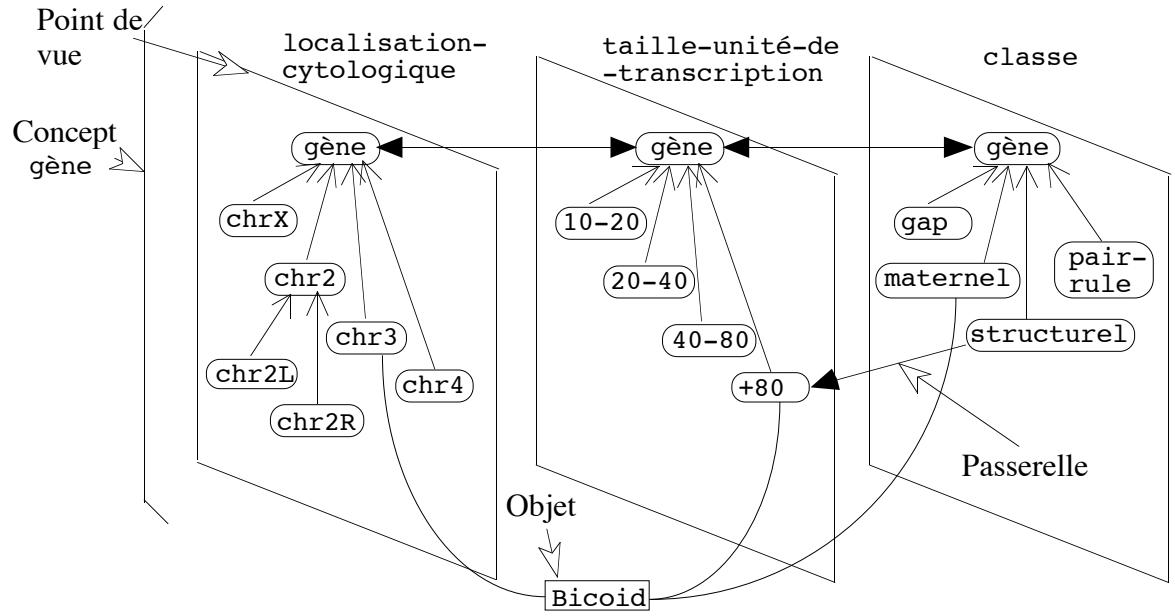


Figure 5 (d'après la base KNIFE) : Les différentes sortes d'entités de TROEPS. Le concept `gène` peut être vu sous les points de vue `localisation-cytologique`, `taille-unité-de-transcription` et `classe` (de gène). Chacun d'entre eux détermine une hiérarchie de classes dont la racine est nommée `gène`. Par exemple, sous le point de vue `localisation-cytologique`, l'ensemble des gènes est décomposé selon leur chromosome. L'objet `Bicoid` est attaché à la classe `chr3` sous le point de vue `localisation-cytologique`, à la classe `+80` sous le point de vue `taille-unité-de-transcription` et à la classe `maternel` sous le point de vue `classe`.

[Mariño& 1990, Mariño 1993]. La notion de *point de vue* peut être rapidement présenté suivant trois aspects détaillés dans la suite :

- la séparation des objets en grandes familles disjointes nommées concepts ;
- l'introduction sur un concept de la multiplicité des taxonomies de classes (correspondant aux classes habituelles des représentations par objets) ;
- l'ajout de la notion de passerelle qui permet d'exprimer des relations d'inclusion entre classes de points de vue différents.

La notion de points de vue est présentée par l'exemple de la Figure 5.

II.1. Les problèmes approchés par TROEPS

La présente section pose trois problèmes de représentation de connaissance par objets qui furent à l'origine de la conception de TROEPS et les solutions qui y sont apportées, en partie en liaison avec les mécanismes de points de vue.

II.1.1. Problème d'évolution

Un problème posé par la structure des représentations de connaissance par objets (et des langages de programmation par objets) est que les objets attachés à une classe ne peuvent pas en changer (ce qui se nomme la *migration d'objet*). Pourquoi changer de classe ? Parce qu'une protéine considérée tout d'abord comme structurelle n'est pas fondamentalement structurelle, c'est d'abord une protéine (en effet, elle peut — une fois sortie de la cellule — cesser d'être structurelle sans cesser d'exister mais elle ne peut cesser d'être une protéine). Ce problème relève de la différence entre « être » et « être vu comme ». Il est intéressant de pouvoir considérer la même protéine comme une protéine structurant la membrane de la cellule lorsque plus d'information est disponible. Il serait intéressant de considérer cette même protéine plus tard comme une protéine régulatrice d'un gène particulier. Il est donc nécessaire de pouvoir la considérer comme telle. Sous l'aspect de la migration d'objets, il y a donc trois problèmes distincts :

- un problème d'*enrichissement* de la connaissance (ou *complétion*), qui apparaît lorsque la connaissance sur la protéine est plus spécifique et qui nécessite un affinement de la représentation,
- un problème d'*évolution* (ou *mise à jour*), qui apparaît lorsque la protéine change — peu utile pour une protéine mais elle peut changer de structure tridimensionnelle et par là même de fonction — et qui nécessite une modification de la représentation, et
- un problème de *point de vue* (ou *perspective*), qui apparaît lorsque la protéine peut être vue sous plusieurs aspects (fonction, régulation, etc.) et qui nécessite un changement de représentation.

Seul le dernier point peut être mécaniquement résolu dans les systèmes de représentation de connaissance classique par l'utilisation de la multi-instanciation ou de la multi-généralisation. Cependant, la classe `protéine` devra être spécialisée en `protéine-structurelle`, `protéine-structurelle-membranaire`... puis `protéine-structurelle-membranaire-régulatrice`,... ce qui n'est pas très satisfaisant [Goodwin 1979]. D'autres mécanismes particulièrement *ad hoc* (destruction puis recréation de l'objet) permettent de traiter les deux autres problèmes [Banerjee& 1987, Nguyen& 1989]. Ils sont particulièrement utilisés lors de la classification.

TROEPS reprend ce problème à son origine. Il contraint à distinguer entre la fonction *ontologique* des classes et leur fonction *taxonomique*. Par ontologique il faut entendre ce qui est lié à l'essence même de la notion représentée et par taxonomique ce qui est lié à une classification des objets dépendant d'un point de vue extérieur[7]. Il sépare alors la notion de concept (ontologique) de la notion de classe (taxonomique). Ainsi, une enzyme est définie

[7] Dans [Hautamäki 1986], la même distinction est faite entre ontologique et « épistémique ».

ontologiquement en tant que protéine et non plus en tant qu'enzyme. Cependant, il peut exister une taxonomie fonctionnelle permettant de classer cette protéine parmi les enzymes, puis les enzymes de restriction. Ainsi, le problème de complétion est résolu. Le problème de perspective connaît aussi un début de solution : dès lors que les objets sont créés hors de leur classe, il est possible de définir plusieurs taxonomies (par exemple, fonction, taille) sur un même concept (celui des protéines). Un même objet peut alors être attaché à plusieurs classes pourvu qu'elles appartiennent à des taxonomies différentes.

II.1.2. Problème d'interprétation des classes et de la spécialisation

Un problème important signalé par William Woods [Woods 1975] est celui de l'interprétation des classes : que signifie une classe, un objet, un lien entre classes ou objets ? Ce problème est en partie tranché par la distinction classe-objet (ou classe-instance), mais pas complètement.

Les classes peuvent être interprétées de manière soit *descriptive*, soit *définitionnelle* — aussi qualifiée de prescriptive. Une classe descriptive introduit les conditions nécessaires pour qu'un objet soit une instance de la classe, alors qu'une classe définitionnelle décrit des conditions nécessaires et suffisantes. Ceci est crucial pour la classification : dans un système définitionnel, en général, la classification détermine (lorsque les objets ont des valeurs pour tous leurs attributs) les classes auxquelles un objet appartient, alors que, dans un système descriptif, elle détermine les classes auxquelles un objet *peut* r. Dans les logiques terminologiques [Napoli 1998], la distinction entre concept primitif et concept défini, tout d'abord fondée sur le fait que certains concepts sont définis en fonction d'autres alors que les concepts primitifs ne peuvent être définis, correspond sémantiquement à la distinction entre classe descriptive et définitionnelle. À l'opposé, le caractère descriptif ou définitionnel des langages de programmation par objets n'a pas été clairement établi (voir, par exemple, la discussion dans [Davis 1987]) : même si les classes sont considérées comme descriptives, dès qu'un mécanisme de classification automatique est introduit, il repose souvent sur une interprétation définitionnelle.

Il y a deux intérêts à disposer de classes descriptives.

— Tout d'abord, il y a beaucoup de classes descriptives à modéliser (soit parce qu'il n'existe pas de conditions suffisantes, soit parce qu'il n'est pas possible de les exprimer) et il faut donc avoir les moyens de le faire.

— Ensuite, tout résultat d'inférence valide dans un monde descriptif est valide dans un monde définitionnel. Les inférences pourront donc y être faites en toute sécurité.

En effet, l'interprétation descriptive est plus faible que l'interprétation définitionnelle : le résultat des inférences est plus restreint alors que plus de notions peuvent être représentées (car elles n'exigent pas de poser une définition complète). Ainsi, si une tasse est caractérisée comme un contenant avec une anse, d'un point de vue définitionnel, il est possible d'inférer qu'un contenant avec une anse est une tasse. D'un point de vue descriptif, il est seulement possible d'inférer que celui-ci peut être une tasse, mais est-ce bien une tasse ?

TROEPS, issu de SHIRKA, s'inscrit dans une tradition héritée des langages de programmation par objets dans laquelle les classes sont descriptives (il est possible de décrire deux classes ayant les mêmes propriétés et une instance d'une de ces classes ne sera pas instance de l'autre).

La popularité des langages de programmation par objets atteste que cette sémantique est bien comprise des utilisateurs actuels des objets. Par ailleurs, il y a de nombreux « domaines » dans lesquels il est possible d'exprimer des conditions nécessaires mais très difficile d'exprimer des conditions suffisantes (voir [Doyle& 1991, Lebbe 1998, Kayser 1998]). Pour ces raisons, TROEPS est, pour l'instant, restreint à une interprétation descriptive des classes.

II.1.3. Problème de nommage et d'identité

L'attachement d'un objet à plusieurs classes incomparables pose un problème connu dans les systèmes à multi-généralisation : le conflit de nom [Ducournau& 1995]. Ce dernier s'énonce comme suit : si deux attributs différents peuvent être nommés de la même manière dans deux classes différentes et que l'objet appartient à ces deux classes, il y a conflit. Il faut noter que le problème des conflits de nom est un problème à double tranchant : soit un attribut ne peut être nommé qu'une fois dans une base et il devient impossible d'utiliser son nom dans des classes incomparables qui ne partageront jamais un objet (« trop de conflits » sont examinés par avance) ; soit deux attributs nommés de la même façon, introduits dans deux classes incomparables, ne sont pas les mêmes attributs et alors le nombre d'attributs est multiplié (des conflits réels ne sont pas explicités) [Dekker 1994].

Dans TROEPS, la solution intermédiaire retenue ramène la définition des attributs et leur nommage au niveau du concept. Ainsi, toutes les classes de ce concept doivent utiliser le nom défini au niveau du concept pour nommer l'attribut et ce nom ne peut nommer un autre attribut. Par exemple, le concept `protéine` et le concept `gène` peuvent avoir un attribut nommé `séquence`, il n'y a aucune ambiguïté sur le fait qu'il ne s'agit pas du même attribut (l'un va contenir une séquence d'acides aminés et l'autre une séquence de nucléotides). Par contre, dans les classes de gènes `récessif` et `structurel`, l'attribut `séquence` est le même et n'a rien à voir avec celui d'une `protéine`.

Au problème des noms s'ajoute le problème d'*identité* des objets. L'identité est aussi du ressort de l'aspect ontologique, c'est-à-dire du concept. Les objets de TROEPS sont nommés au niveau du concept et ce dernier se charge de l'identité des objets, c'est-à-dire qu'il garantit que deux objets différents ne peuvent être identifiés de la même manière et qu'il se charge de retrouver l'objet si l'identificateur est valide. Ceci permet de nouveau de se libérer des conflits de noms au sein d'un même concept et de donner le même nom à deux objets différents, pourvu qu'ils soient dans deux concepts différents. Les bases de données à objets ont promu la notion d'identifiant d'objet par opposition à la *clé* qui permettait de trouver les *n*-uplets dans les bases relationnelles. Dans TROEPS, la décision d'identifier les objets par une clé a été prise afin de disposer de noms plus naturels pour l'utilisateur que les identifiants engendrés par le système ou qu'un nom dont l'utilisateur avait du mal à gérer l'unicité. Les objets sont donc nommés par l'intermédiaire d'un sous-ensemble de leurs attributs (valides pour tout le concept et pouvant varier d'un concept à l'autre).

II.2. Les points de vue dans TROEPS

Les solutions apportées à ces problèmes dans TROEPS sont d'abord présentées avant de considérer l'appréhension des points de vue par le modèle des systèmes classificatoires.

II.2.1. Présentation

L'aspect ontologique de TROEPS est pris en compte par la notion de concept. D'un autre côté, l'aspect taxonomique est pris en compte par les classes. Cela a plusieurs répercussions :

— un objet fait ontologiquement partie du concept, il ne pourra donc, sans rompre son intégrité, migrer vers un autre concept ;

— les objets peuvent être vus sous divers points de vue indépendants de ce qu'ils sont, mais dépendants uniquement de la vision qu'un observateur applique aux objets du concept ;

— un objet est « taxonomiquement » attaché à une classe. C'est-à-dire que si sous un certain point de vue, il a été attaché à une classe, cela ne signifie pas que l'objet ne puisse faire partie d'une autre classe sous un autre point de vue et n'empêche pas qu'il puisse, sous le même point de vue, migrer un objet d'une classe à une autre.

Il faut cependant nuancer ce caractère absolu de l'ontologique : il dépend fortement de l'application dans laquelle la représentation est mise en œuvre. En effet, une application au fonctionnement cellulaire considérera comme ontologique la distinction entre une enzyme et un facteur de croissance (l'un ne pourra pas prendre la place de l'autre) alors que dans une application à l'expression génétique les deux se trouvant la cible des mêmes opérations seront considérés comme des protéines. De même, pour un chimiste, la différence entre ADN et ARN n'est pas aussi fondamentale que pour un généticien moléculariste.

Ces principes se retrouvent naturellement dans l'implémentation du système TROEPS. Ainsi, le concept définit :

— la structure des objets : les attributs qu'il peut avoir sont décrits et typés sous forme d'attributs de concepts (ceux-ci sont définis en fonction d'autres concepts et d'ADT),

— l'identité et l'intégrité des objets : un sous-ensemble de ces attributs forme la clé qui doit être unique et qui permet d'identifier l'objet (le système garantit cette unicité comme il garantit que cette clé ne sera pas modifiée lors de la vie de l'objet),

— l'espace des noms d'attributs, d'objets et de points de vue.

Un `gène` (comme présenté sur la Figure 2) est doté des attributs `nom`, `synonymes`, `taille-de-l-unité-de-transcription`, `séquence`, `protéine` ; il est identifié par l'attribut `nom`. Un `patron d'expression` est identifié par ses attributs `gène`, `type (de mutation)` et `stade (de développement)`.

La définition de l'ensemble des attributs au niveau du concept permet de savoir à quel attribut il est fait référence quelle que soit la classe à laquelle un objet est attaché. Par ailleurs, il indique que n'importe lequel des attributs est applicable aux objets du concept indépendamment de la manière dont ces objets sont vus au travers des classes.

Un point de vue définit une taxonomie de classes en accord avec la vision du domaine qui est appliquée par le modélisateur. L'attachement d'un objet à une classe particulière est contingent.

La classe définit un ensemble de contraintes qui doivent nécessairement être satisfaites par les objets qui y seront attachés. Celles-ci sont décrites par un ensemble de descripteurs

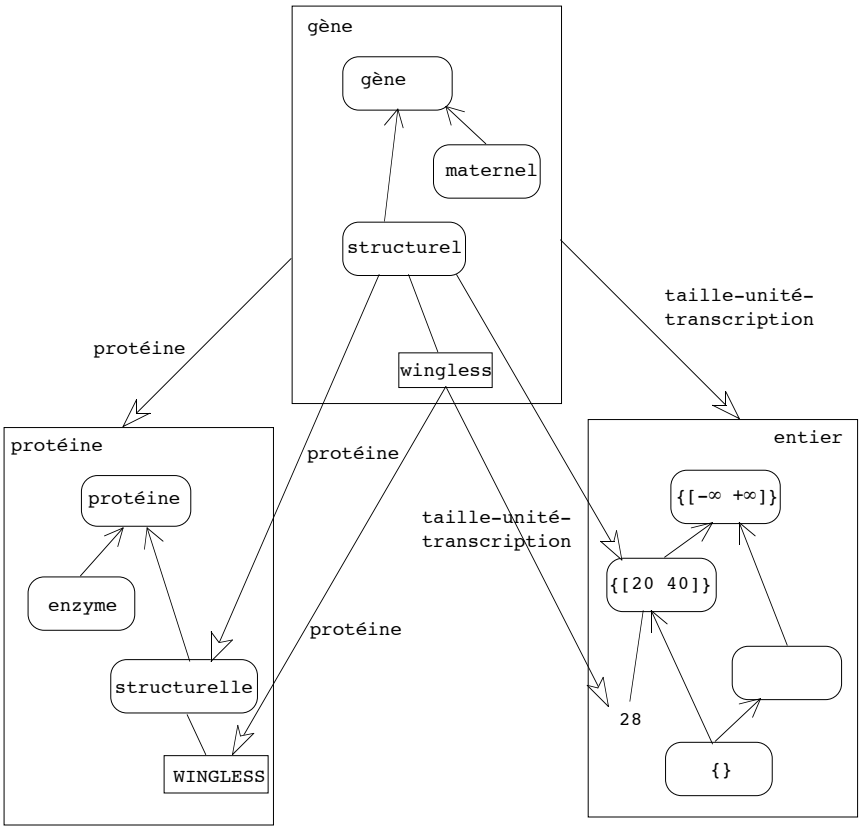


Figure 6 : Arrangement entre les différents niveaux de TROEPS : concept, classe et objet (ici, un seul point de vue est représenté par concept). Il existe des différences suivant que le type de l'attribut de concept est un autre concept (protéine) ou un ADT (entier) : les classes sont contraintes par des expressions de type au lieu d'autres classes, il n'existe qu'un seul point de vue et la taxonomie est un treillis.

(correspondant aux facettes des représentations de connaissance par objets).

Les concepts ne sont pas indépendants entre eux dans le sens où, comme le montre la Figure 6, le type d'un attribut peut être un autre concept (ou un ADT). Cette décomposition se retrouve sur les autres niveaux (classes et objets) sans qu'il y ait de confusion entre les niveaux. Ces liens entre concepts (et donc aussi entre classes ou objets) ne peuvent être circulaires. Ainsi, la protéine correspondant à un gène sera ontologiquement une protéine, de même, la protéine correspondant au gène « wingless » est la protéine « WINGLESS ». Maintenant « wingless » peut être vu comme un gène structurel ou comme un gène récessif et sa protéine comme une protéine structurelle ou comme une protéine membranaire sans que cela n'ait de conséquence sur le gène « wingless » (de même que les réorganisations des classifications des espèces n'ont aucune incidence sur les individus).

Enfin, les *passerelles* permettent d'exprimer des contraintes entre les classes de points de vue différents d'un même concept. Ainsi, si les points de vue sont théoriquement indépendants, il est possible que l'appartenance à une classe sous un point de vue implique l'appartenance à une autre classe sous un autre point de vue. Par exemple, la passerelle entre `structurel` et `+80` dans la Figure 5 indique que les gènes structurels ont forcément une unité de transcription de grande taille.

II.2.2. Polynomies (classifications multiples)

Après avoir introduit l'organisation des points de vues dans TROEPS, il est possible d'en rendre compte dans le contexte des systèmes classificatoires.

Jusqu'ici, la classification dans les systèmes classificatoires a été définie sur une unique taxonomie. TROEPS utilise une classification multiple pour classer un individu dans plusieurs taxonomies simultanément. En bref, la classification multiple détermine, pour plusieurs taxonomies, les ensembles de classes sous lesquelles un individu *i* peut être attaché. Le produit homogène de systèmes classificatoires sera une première approche du problème. Cependant, afin de rendre compte de l'interdépendance des taxonomies, TROEPS introduit la notion de passerelle. Une passerelle relie un ensemble de catégories (la source) à une autre catégorie (la destination ou cible). Ceci signifie que l'intersection des interprétations réelles des classes

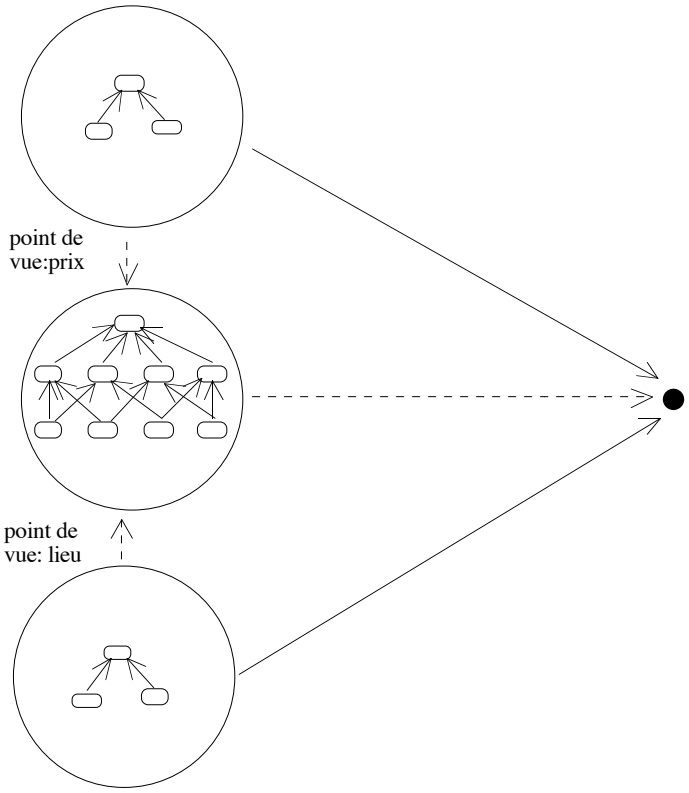


Schéma 12 : Point de vue comme produit (2). En ce qui concerne les points de vue de TROEPS, la mise en correspondance de deux taxonomies — indépendantes — permet de disposer du produit des taxonomies rassemblant dans une unique taxonomie les informations disponibles dans les deux taxonomies initiales. Dans TROEPS, ce produit reste toujours implicite.

sources est incluse dans celle de la classe destination. De manière plus opératoire, cela peut être résumé en : « si un individu est membre de chaque catégorie source alors il est membre de la catégorie destination ». Ajouter des relations entre des catégories de différents points de vue, permet de sortir du cadre du simple produit homogène de taxonomies.

La présente section ne concerne que les taxonomies (c'est-à-dire les systèmes classificatoires indépendamment de $\langle L_C \ll \rangle$). En fait, la catégorisation multi-points de vue, si elle reste possible dans ce cadre, n'a pas été étudiée.

La notion de produit permet de rendre compte de la notion de points de vue. Elle ne permet pas, par contre, d'intégrer les passerelles dans le modèle des systèmes classificatoires.

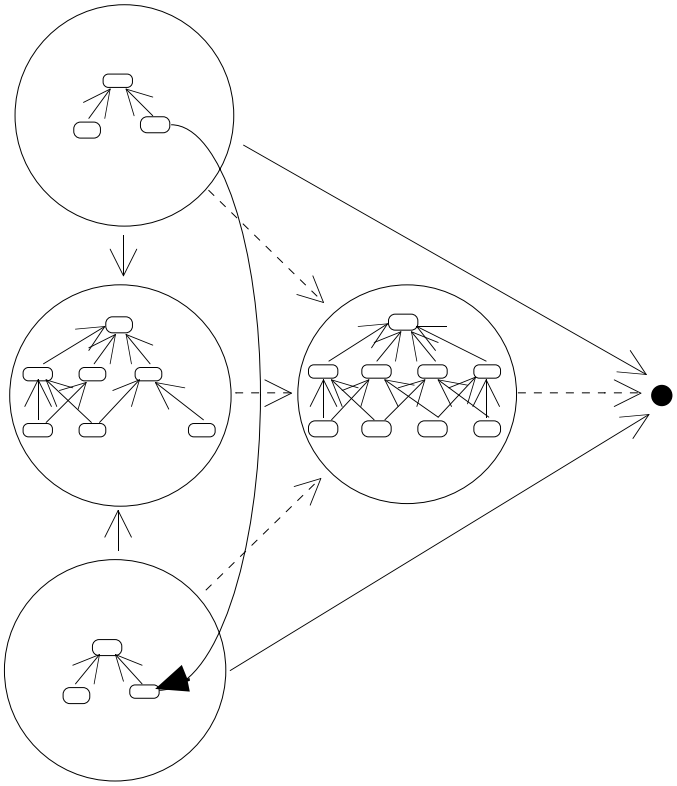


Schéma 13 : Polynomie. La présence d'une passerelle entre deux représentations du produit permet d'avoir une taxonomie produite plus restreinte que le simple produit des taxonomies initialement disponible (car la passerelle indique l'équivalence entre certaines classes des deux taxonomies). La polynomie permet de rompre l'indépendance des taxonomies initiales.

DÉFINITION (passerelle, fondation) : Une *passerelle* dans un produit disjoint de systèmes classificatoires homogènes $\langle C \leq \ll \rangle$ est un élément (s, d) de $\pi C \times \sigma C$ dans lequel σ est une projection non vide orthogonale à π . Le couple (π,σ) est nommé la *fondation* de la passerelle.◊

DÉFINITION (acceptabilité) : Une passerelle (s,d) de fondation (π,σ) est *acceptable* dans $\langle C \leq \ll \rangle$ si et seulement si

$$\forall c \in C, \pi c \leq_\pi s \Rightarrow \sigma c \ll_\sigma d \,. \qquad \Diamond$$

CONSÉQUENCE : Soit (s,d) une passerelle acceptable de fondation (π,σ), $\forall c_1, c_2 \in C, \left(\pi c_1 \leq_\pi s\right) \wedge \left(d \leq_\sigma \sigma c_2\right) \Rightarrow \sigma c_1 \ll_\sigma \sigma c_2$ ◊

Ainsi, les passerelles d'une catégorie (c_1) vers une autre (c_2) telles que l'interprétation abstraite de la première n'est pas incluse dans la seconde est inacceptable. Soit cette passerelle est incorrecte, soit elle est risquée (dans le sens où si tous les individus attachés à c_1 doivent l'être à c_2, alors il serait souhaitable que la description de c_1 soit plus spécifique que celle de c_2).

DÉFINITION (validité) : Une passerelle (s,d) de fondation (π,σ) est *valide* dans $\langle C \leq \ll \rangle$ si elle est acceptable et si $I_R(s) \subseteq I_R(d)$. ◊

L'acceptabilité permet de considérer seulement les passerelles qui satisfont le critère de sous-catégorisation pour le produit. La validité constitue une justification sémantique de la passerelle. Mais si l'acceptabilité peut être vérifiée ce n'est pas le cas de la validité.

DÉFINITION (polynomie ou produit de systèmes classificatoires avec passerelles) : Une *polynomie* ($\langle\langle S\ P\rangle$) est constituée d'un produit disjoint homogène de systèmes classificatoires (S) et d'un ensemble de passerelles (P) valides pour S. ◊

Comme une polynomie est un produit homogène, les individus considérés pas TROEPS sont le produit du même individu ($i,i,...i$) qui est noté i par abus de langage. Mais la polynomie n'est pas un réel système classificatoire. Afin de la considérer comme telle il faut réussir à intégrer les passerelles dans les structures du système classificatoire. Les passerelles modifient l'ordre partiel du produit de systèmes classificatoires en conséquence de la définition de la validité des passerelles.

PROPOSITION : À une polynomie ($\langle\langle C \leq \ll\rangle\ P\rangle$) correspond un système classificatoire $\langle C' \leq'|_{C'} \ll\rangle$ tel que $\leq'$ soit défini par :

si $c_1 \leq c_2$ alors $c_1 \leq' c_2$;
si $\exists (s,d) \in P$ de fondation (π,σ);$(\pi c_1 \leq_\pi s) \wedge (d \leq_\sigma \sigma c_2)$ alors $c_1 \leq' c_2$;
$\leq'$ est fermé par transitivité.

$\langle C \leq'\rangle$ est un pré-ordre auquel peut-être associé un ordre partiel $\langle C' \leq'|_{C'}\rangle$. dans lequel les éléments de C' sont les éléments maximaux des classes d'équivalence. ◊

Par conséquent, la classification sous plusieurs points de vue se réduit à la classification dans une polynomie. Celle-ci peut être considéré comme un simple système classificatoire auquel les opérations définies au chapitre I peuvent être appliquées et retourneront des résultats en accord avec la définition de polynomie.

La notion de projection peut être étendue aux polynomies. Ceci est utile afin de ne considérer que certaines parties de la polynomie.

DÉFINITION (projection dans une polynomie) : Une projection π d'une polynomie vers une autre est telle que l'image de la polynomie ($\langle\langle S\ P\rangle$) est une autre polynomie ($\langle\langle S'\ P'\rangle$) et

— $S' = \pi S$;
— $P' = \{(s,d) \in P, (\pi s = s) \wedge (\pi d = d)\}$. ◊

La projection ne préserve que les passerelles dont les sources et la destination sont préservées par la projection. Dans le cas contraire les passerelles projetées seraient trop restreintes — s'appliquant à plus d'individus qu'à l'origine. Par exemple, ce n'est pas parce que tous les gènes structurels et récessifs sont de taille moyenne que, si l'aspect structurel n'est pas considéré, tous les gènes récessifs sont de taille moyenne.

Il faut remarquer que cette projection n'inverse du produit (appliquer le produit aux résultats des projections unitaires, retourne le produit classique de systèmes classificatoires). On peut ajouter, à titre d'objectif plus que de conjecture, qu'il est souhaitable que cette projection commute avec la transformation d'une polynomie en système classificatoire.

II.2.3. Inférence de taxonomie multi-points de vue

De la définition de la catégorisation dans les systèmes classificatoires (voir p19) et de l'interprétation des points de vue en systèmes classificatoires découle naturellement une re-définition de la classification et de la catégorisation.

Puisque la polynomie est un produit de systèmes classificatoires, il est clair que la définition de dissimilarité construite s'appliquant à un produit de systèmes classificatoires pourrait s'appliquer telle quelle et que cette dissimilarité est utilisable. À la différence de la précédente cependant, cette dissimilarité n'est pas fonction directe des composants (ceci est dû à la définition de polynomie). Une autre formulation, tirant parti de la taxonomie présente sous chaque point de vue, est utilisée [Valtchev 1998].

DÉFINITION (dissimilarité multi-points de vue) : Soit une polynomie de n systèmes classificatoires, soient o et $o' \in L_l$ dans ce produit. Soit $aggr'$ une fonction d'agrégation et λ'_i le poids associé au facteur i (isolé par la projection π_i), une dissimilarité entre o et o' est définie par :

$$\delta(o,o') = \underset{i=1}{\overset{n}{aggr'}}\ \lambda'_i.\overline{\delta_i}(o,o') \qquad \Diamond$$

Une polynomie de n points de vue permet de définir 2^n produits (par projection par exemple). Si la mesure de dissimilarité doit être indépendante de la manière d'obtenir le produit, il est conseillé d'utiliser une combinaison linéaire normalisée comme fonction d'agrégation (ainsi la mesure obtenue en produisant trois systèmes classificatoires sera la même que celle obtenue en en produisant d'abord deux puis le troisième).

II.3. Impact des points de vue

La notion de point de vue a de nombreux avantages pour les applications. Elle permet de faire de la classification et de la migration d'objets très facilement. Ceci n'est pas dû aux points de vue en tant que tels mais plutôt à l'architecture qui détache l'aspect taxonomique de l'aspect ontologique.

Les points de vue permettent de ne présenter à un utilisateur qu'une hiérarchie restreinte (le produit des taxonomies est implicitement présent dans TROEPS mais les classes du produit ne sont jamais construites). Ainsi, il est possible de contraindre la valeur d'un attribut d'objet à être dans la classe `structurel` dans le point de vue `classe` et `récessif` dans le point de vue `expression` sans s'adresser à une classe spécifique (`structurel-récessif` ou `récessif-structurel` ?). À l'instar des vues des bases de données, les points de vue permettent aussi de ne prendre en compte qu'un sous-ensemble des attributs.

Il est possible de créer tout d'abord les concepts et les relations qu'ils entretiennent avant de créer la moindre classe, ce qui d'un point de vue méthodologique semble satisfaisant (cependant il est aussi possible d'ajouter un attribut au concept — et ultérieurement d'en faire mention dans les classes — même lorsque des points de vue existent).

Il est donc aussi possible de créer un nouveau point de vue, de sélectionner les attributs qui y seront visibles et de demander à l'algorithme d'inférence de taxonomie de créer une taxonomie (l'interface de TROEPS demande pour cela les poids à attribuer à chacun des attributs à considérer). Cet algorithme créera des classes, y attachera les objets correspondant à chacune et inférera une description de la classe [Valtchev 1999b]. La taxonomie ainsi obtenue peut être utilisée de diverses manières :

— comme un outil d'analyse exploratoire, elle permettra de distinguer des classes intéressantes dans l'ensemble d'objets ;
— comme une taxonomie de représentation de connaissance classique ;
— comme un mécanisme permettant de détecter des manques soit dans l'échantillon courant, soit dans le paramétrage de l'algorithme d'inférence de taxonomie.

Enfin, il est possible d'utiliser un algorithme d'inférence de passerelles [Euzenat 1993c] qui permet de rechercher les passerelles pouvant exister entre différents points de vue. Ces passerelles, correspondant à des règles de type clause de Horn dans un calcul des prédicats monadique sans fonction, sont très intéressantes car elles permettent de présenter certaines régularités (toutes les protéines influant sur la transcription sont situées dans le noyau ?). En fonction d'un point de vue destination et d'un ensemble de points de vue sources, l'algorithme est capable d'obtenir toutes les passerelles acceptables ou valides dans la base courante (il peut en effet fonctionner soit en comparant les descriptions des classes — intension —, soit les ensembles d'objets attachés aux classes — extension). L'inférence de passerelles est utile pour :

— concevoir des passerelles qui vont subsister dans la base de connaissance ;
— rendre compte des différences entre plusieurs algorithmes d'inférence de taxonomie (ou plusieurs paramétrages du même algorithme).

Conclusion

La notion de point de vue permet la représentation multiple d'un même domaine. L'introduction des points de vue a nécessité une réflexion sur la place des classes et des concepts se partageant les aspects taxonomiques et ontologiques de la représentation. Par ailleurs, les points de vue peuvent être interprétés en termes de systèmes classificatoires et être mis en relation de manière à définir un système classificatoire qu'ils approximent.

Les passerelles introduisent un moyen supplémentaire d'exprimer les liens entre les représentations. Ces liens s'intègrent une fois de plus dans la structure des systèmes classificatoires.

Ceci facilite l'appréhension par un utilisateur d'unités plus petites et plus précisément adaptées à sa tâche tout en permettant aux algorithmes (de classification, d'inférence de taxonomies ou de passerelles) de fonctionner indifféremment sur ces petites unités ou sur des assemblages de ces unités.

Comme dans le chapitre précédent, des travaux afin de rendre compte de liens circulaires entre les concepts seraient une extension nécessaire de ce modèle. TROEPS, tel que présenté dans les deux derniers chapitres fonctionne, en particulier en tant que serveur de connaissance à

travers HTTP [Euzenat 1996c, Alemany 1998] et est utilisé dans plusieurs applications [Euzenat& 1997b].

Le chapitre suivant considérera encore des relations entre représentations différentes, cette fois-ci organisées hiérarchiquement. Le langage utilisé sera très différent et la comparaison de ce qui est exprimé sera envisagé plus explicitement.

Crédits

La notion de points de vue et l'organisation générale de TROEPS proviennent du travail d'Olga Mariño [Mariño 1993]. La présentation originale de l'approche, ainsi que certaines re-formulations de ce chapitre, sont ma contribution personnelle, aiguillonnée par de nombreux débats menés au sein des groupes « classification et objets » du PRC-GDR « intelligence artificielle » et « évolution des langages à objets » du GDR « programmation ».

L'introduction des points de vue au sein des systèmes classificatoires est ma contribution ; sa prise en compte dans les algorithmes de catégorisation, celle de Petko Valtchev. D'une manière générale, les travaux présentés ci-dessus ont influencé la mise en œuvre de TROEPS.

L'implémentation du logiciel TROEPS est un travail collectif du projet SHERPA réalisé sous ma direction. Elle a plus particulièrement bénéficié des travaux de Christophe Alemany (stage d'ingénieur CNAM) et de Loïc Tricand De La Goutte (ingénieur expert) que j'ai dirigés également.

III. GRANULARITÉ DANS LES SYSTÈMES RELATIONNELS

Les algèbres de relations binaires sont très utilisées en représentation de connaissance pour représenter les relations entre objets, en particulier dans le temps ou l'espace (§III.1). La même situation peut être représentée avec différents formalismes, mais il est aussi possible de la représenter avec le même formalisme à différents niveaux de détails ou granularité. Cette possibilité est particulièrement importante dans le contexte de l'utilisation d'agents logiciels autonomes disposant de capteurs de résolutions différentes. Les représentations, exprimées dans le même langage, sont alors dans un rapport d'approximation.

Cependant, la simple superposition de plusieurs représentations de la même situation sous des granularités différentes n'est pas forcement consistante. Le rapport d'approximation s'exprime alors par des règles régissant le comportement des opérateurs de changement de granularité (§III.2). Définir des opérations de changement de granularité est utile dans ce contexte car cela permet (entre autres) d'évaluer si deux situations sont intrinsèquement contradictoires ou si elles peuvent être la représentation de la même situation sous deux granularités différentes.

Six contraintes s'appliquant à tout système relationnel (c'est-à-dire une algèbre de relations binaires munie d'une structure de voisinage) et permettant de définir de tels opérateurs sont présentées (§III.2). Ces contraintes ont été appliquées aux algèbres de relations usuelles pour représenter le temps et l'espace (algèbres de points, d'intervalles, "Region-Connection Calculi", §III.3).

Ces travaux dénotent donc une notion d'approximation qui se révèle plus complexe que celles développées jusqu'à présent (en particulier pour les représentations de connaissance par objets). Ce paramétrage d'une notion d'approximation au départ intuitive doit être prise en compte dans une image plus large des rapports entre représentations.

Problème : représenter une situation sous différentes granularités

Le langage de la représentation de connaissance par objets est momentanément abandonné pour considérer des problèmes de représentations multiples intervenant dans une représentation relationnelle.

L'espace physique est un exemple typique de domaine représenté — par une carte — sous différentes granularités. Une carte est chargée de représenter un territoire. Par exemple, la carte Didier Richard consacrée au Vercors n'est pas une simple carte au 1/50 000^{e} elle contient aussi sur sa couverture une carte au 1/1 000 000^{e} et un schéma de la région Rhône-Alpes au 1/4 000 000^{e} sur laquelle se trouvent les contours des autres cartes de la collection. Ce type de disposition est tout d'abord pratique, mais il pose aussi beaucoup de problèmes aux cartographes. Ce sont les problèmes rencontrés lors de la construction numérique de la granularité : Où se trouve le centre des villes lorsqu'elles doivent être représentées par un point à une échelle plus petite ? Quels éléments doivent disparaître (la seconde carte retient les villes principales, les autoroutes, les nationales et les fleuves alors que la dernière se cantonne aux grandes villes et fleuves) ? Bien que la carte porte sur sa couverture la mention d'une échelle, elle n'est en aucun cas une projection géométrique du territoire à cette échelle. En effet, alors qu'une projection conserve toute la structure, la carte oublie un grand nombre d'éléments. Pire,

si le territoire couvert est bien à l'échelle indiquée, ce n'est pas le cas de beaucoup d'éléments figurant sur la carte : ainsi les routes y figurent à une échelle bien plus grande que les forêts.

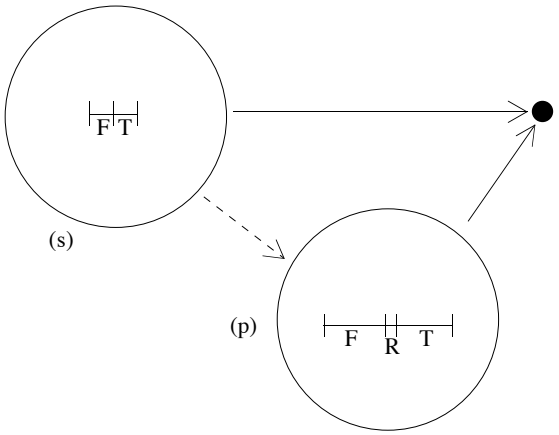


Schéma 14 : Changement de granularité comme approximation. Le but recherché dans ce travail est de rendre compte du rapport d'approximation entre deux situations exprimées sous des granularités différentes (et où plus ou moins de détails apparaissent).

La problématique de la généralisation cartographique [Muller& 1995, Barkowsky& 1997] consiste, à partir d'une représentation fine, à établir une représentation moins fine qui constituera la carte. Elle se doit d'établir les règles permettant d'obtenir la carte ou les propriétés qui doivent être conservées entre les deux représentations (par exemple, faut-il conserver le nombre de villes traversées par une route ? Faut-il conserver le nombre de virages en épingle dans une route de montagne ?). Comme il n'y a pas un facteur d'échelle mécanique entre deux représentations, il est nécessaire de déterminer dans quelle mesure deux représentations correspondent bien à la description du même domaine.

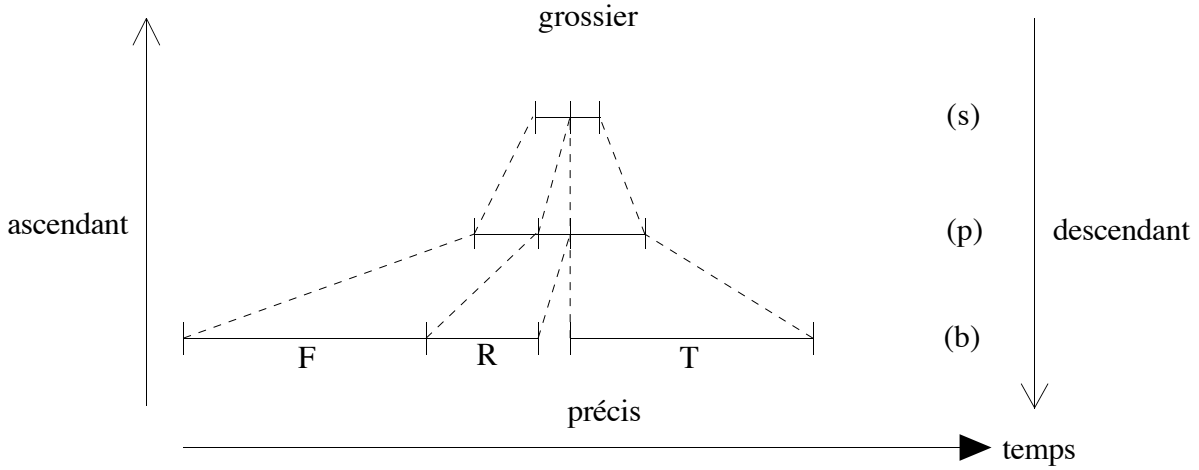


Figure 7 : L'exemple de l'accident d'avion. La période de fonctionnement de l'avion (F), celle où il a des ratés (R) et celle où il perd de l'altitude — représentées par des intervalles — sont perçues diversement par l'observateur au sol (s), le pilote (p) ou la boîte noire (b).

Le travail présenté ici, à cet égard moins ambitieux, se place dans le cadre qualitatif des algèbres de relations. Il cherche à établir les règles permettant de rendre compte de cet effet de changement de granularité entre deux représentations. Comme il est exposé ci-dessous sous le jour de la représentation temporelle, l'exemple suivant décrit les problèmes posés.

Plusieurs témoignages sont apportés sur un accident d'avion. Ils impliquent un témoin au sol (s) disant que « le moteur a arrêté de fonctionner (F) et l'avion s'est immédiatement mis à tomber (T) », le pilote (p) disant que « l'avion fonctionnait correctement (F) jusqu'à ce qu'il ait des ratés (R) après quoi il a perdu de l'altitude (T) » et la boîte noire (introuvable, b) révélant que « l'avion a eu une courte période de ratés (R) suivie d'une courte période de fonctionnement correct avant que l'avion ne commence à perdre de l'altitude (T) ». Cette situation, visualisée sur la Figure 7 constitue un ensemble de représentations apparemment contradictoires. Le problème principal consiste à déterminer si elles le sont intrinsèquement (par exemple si l'un des intervenants ment ou est défaillant) ou si ce sont des représentations compatibles d'une même situation sous différentes granularités.

Ce travail sera utile lorsqu'il faudra intégrer les informations de plusieurs bases de données ou lorsque divers agents (logiciels ou humains) n'ayant pas été conçus pour décrire les situations avec le même niveau de détail devront communiquer.

Une fois encore, l'approche développée ici est indicative non impérative. Elle ne permet pas de dire comment une représentation sera représentée sous une autre granularité mais comment elle le peut (en éliminant comment elle ne le peut pas).

Il est possible de revenir une fois de plus à la biologie puisque les généticiens développent des cartes génomiques représentant un même objet (un génome) à différentes échelles. Plusieurs types de cartes peuvent être distinguées :

— des cartes cytogénétiques (mesurées en pourcentages de la longueur du chromosome) dans lesquelles les gènes sont localisés au sein des bandes de couleurs observées ou obtenues sur les chromosomes ;
— des cartes génétiques (mesurées en centiMorgans) construites statistiquement qui expriment la proximité entre deux gènes en fonction du taux recombinaison observé dans les populations (plus deux gènes sont exprimés ensemble plus ils seront proches) ;
— des cartes physiques (mesurées en kilobases) construites en fonction de la séquence d'ADN sur laquelle les gènes sont précisément localisés dans la suite de nucléotides.

Le problème est que ces cartes ne sont pas décrites dans le même langage. Indépendamment des unités, les méthodes employées font que leurs résultats, tous intéressants, ne sont pas forcément compatibles (voir Figure 8). Les travaux qui sont présentés ci-dessous ne s'appliquent donc pas forcément directement dans ce domaine, bien que certains auteurs aient utilisé le même langage de représentation [Schmeltzer 1995], car il n'existe pas (ou pas nécessairement) d'approximation entre les niveaux.

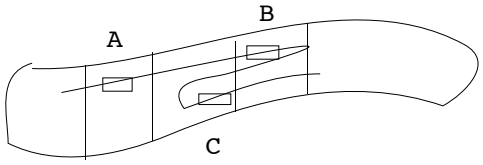


Figure 8 : Si le dessin de la figure représente un brin d'ADN au sein d'un chromosome dont les bandes sont visibles, la carte génétique ordonnera les gènes dans le sens A-B-C alors que la carte cytogénétique donnera un ordre A-C-B. Il est par ailleurs possible, si A et C sont plus souvent exprimés ensemble que d'autres couples mais que A est plus souvent exprimé avec B que C, d'obtenir un ordre B-A-C (ou C-A-B) dans la carte génétique.

Perspective historique

Le problème de la granularité est certainement posé dans de nombreux domaines. La généralisation cartographique est étudiée par les cartographes depuis très longtemps. Cependant, le développement de la cartographie numérique pousse à en formaliser l'approche.

Ramesh Patil [1981] décrit les faits médicaux (la description du patient) sous plusieurs niveaux de détails et autorise la décomposition d'un nœud sous le niveau en dessous. Comme la représentation utilisée est un réseau sémantique, il représente de manière uniforme les attributs et la causalité (essentielle afin de faire de la simulation qualitative). Mais il distingue une granularité focale qui relie les objets (et leurs attributs) d'une granularité causale qui relie les liens de causalité (et peut-être des objets intermédiaires). La représentation multi-niveaux est utilisée pour chercher de manière focalisée les différentes causes d'une maladie. Ainsi, si aucune liaison directe entre symptômes et hypothèses n'est disponible à un niveau, il ira rechercher une liaison indirecte, c'est-à-dire à un niveau inférieur. De manière duale, le système est capable d'expliquer les causalités des niveaux les plus élevés en les instanciant dans les niveaux les plus bas. Il décrit ainsi les effets de la salmonellose jusqu'au niveau chimique.

Dans un sens plus systématique, les premiers travaux sont sans doute ceux de Jerry Hobbs [1985] dont le but est de représenter le sens d'un texte sous plusieurs granularités.

En ce qui concerne le temps en général et les algèbres de relations temporelles en particulier, aucun travail n'avait été publié, lorsqu'en 1990, Philippe Kirsch, Jean-Marc Pugin et moi-même nous sommes penchés sur ce problème. Notre but était la réponse pour le compte de Bull à un appel d'offre de Sollac (filiale d'Usinor) requérant explicitement de savoir maîtriser les changements d'échelle dans la représentation de l'évolution au cours du temps du fonctionnement d'un haut-fourneau. Ces réflexions ont été abandonnées au sein du CEDIAG, mais j'ai continué à y songer et à développer tout d'abord un certain nombre de contraintes nécessaires aux opérateurs de changement de granularités [Euzenat 1993e] et les opérateurs correspondants pour l'algèbre d'intervalles temporels.

Indépendamment, Angelo Montanari et ses collègues [Montanari& 1992] travaillaient sur l'aspect quantitatif (dans le cadre de la spécification de programmes temps-réel). Une série de travaux sur ce sujet ont été menés pour culminer avec la thèse d'Angelo Montanari [1996]. Par ailleurs, dans le domaine des bases de données, d'autres auteurs s'intéressaient au problème toujours quantitatif, tout d'abord dans la perspective de convertir des dates [Wiederhold& 1993] puis en se rapprochant de la problématique de la granularité [Bettini& 1998].

En 1992, Christian Freksa [1992] publiait un article introduisant la notion de « voisinages conceptuels » au sein des algèbres de relations. J'ai alors pu compléter l'ensemble des contraintes par celle de « compatibilité avec le voisinage » et montrer que les opérateurs que j'avais proposés étaient les uniques opérateurs possibles [Euzenat 1995e]. Par ailleurs, Robin Hirsch [1996] a montré depuis comment construire de manière automatique une algèbre d'intervalles, ce qui permet de généraliser encore l'approche utilisée.

À part quelques observations très parcellaires, il n'existe pas à ce jour de théorie liant à la fois l'aspect qualitatif et l'aspect quantitatif [Euzenat& à paraître].

III.1. Rappel : systèmes relationnels

Un travail considérable a été accompli sur la représentation qualitative du temps. Plusieurs notions concernant les algèbres de relations binaires temporelles, utilisées dans la suite, sont brièvement rappelées.

Une structure légèrement étendue par rapport aux algèbres de relations binaires classiques sera considérée en y adjoignant un voisinage : une structure topologique sur la base de l'algèbre. Ainsi, après la définition d'une algèbre de relation étendue (§III.1.1), sont présentés ci-dessous les algèbres d'instants (§III.1.2) et d'intervalles (§III.1.3) ainsi que le passage de l'une à l'autre (§III.1.4).

III.1.1. Algèbre de relations étendue

Une algèbre de relations binaires (algèbre de relations dans la suite) est une structure $\langle A,\wedge,\vee,*,1,0,1',\neg\rangle$ où $\langle A,\wedge,\vee,1,0\rangle$ est une algèbre booléenne ; * est une loi interne binaire associative d'élément neutre 1' (à gauche et à droite) et distributive par rapport à $\vee$; $\neg$ est un opérateur unaire interne involutif et distributif par rapport à $\wedge$, $\vee$ et *.

Un type particulier d'algèbres de relations est considéré ici dans lequel A est l'ensemble des parties d'un ensemble de base Γ clos par l'opérateur $\neg$ (ici $^{-1}$), $\wedge$ et $\vee$ étant les opérations ensemblistes usuelles ($\cap$ et $\cup$). De telles algèbres de relations seront notées par $\langle 2^{\Gamma},\cap,\cup,o,\Gamma,\{\},e,^{-1}\rangle$.

Lorsque l'ensemble Γ est considéré comme un ensemble de relations binaires possibles entre des objets d'un domaine particulier, o est la composition de relations et $^{-1}$ permet d'obtenir la relation réciproque.

L'utilisation d'ensembles de relations permet de pallier le manque de connaissance sur l'exacte relation entre deux objets en exprimant une disjonction de relations possibles (les relations de l'ensemble). Ainsi, si les trois relations de la base sont <, > et =, il est possible d'exprimer que *x* est avant ou égal à *y* par $x\{<=\}y$. Deux exemples bien connus d'algèbres de relations sont donnés ci-après.

Les algèbres de relations sont étendues ici en considérant une notion de voisinage, dit voisinage conceptuel [Freksa 1992, Nökel 1988]. Un voisinage *N* sera simplement une relation binaire réflexive et symétrique entre les éléments de Γ. Cette relation est représentée par un graphe non orienté.

Dans le cadre de la représentation de relations entre objets dans un espace topologique, deux relations sont voisines lorsque la relation entre deux objets peut passer directement de l'une à l'autre par une « déformation » continue des objets. Les voisinages conceptuels sont très utiles pour engendrer les tables de composition [Randell& 1992] ou les compresser [Freksa 1992].

III.1.2. Algèbre d'instants

Un instant est une entité temporelle sans durée (qui peut aussi être qualifiée de point par analogie avec un point sur une droite). Il peut être représenté métriquement par une date. La représentation qualitative des instants consiste à les identifier et à identifier les relations qu'ils

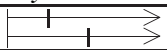
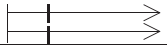
relation (r) : $x\ r\ y$	x/y	réciproque : $y\ r^{-1}\ x$
antérieur (<)		postérieur (>)
simultané (=)		=

Tableau 1 : Les 3 relations de A_3.

o3	>	=	<
>	>	>	<=>
=	>	=	<
<	<=>	<	<

Tableau 2 : Table de composition de A_3.

entretiennent. L'algèbre d'instants présente trois relations possibles qui sont mutuellement exclusives et exhaustives. Elles sont nommées antérieur (<), postérieur (>) et simultané (=). L'ensemble {<,=,>} est nommé A_3, il est la base de l'algèbre d'instants (nommée aussi A_3).

Le Tableau 2 donne la table de composition entre ces relations. Il est parfois possible de déduire de l'information plus précise à partir de celle donnée. Ainsi, si x est antérieur ou simultané à y (x{<=}y) qui est antérieur à z (y{<}z), il est possible de déduire que x{<}z si la composition de < et = est <, et celle de < et < est <, aussi ({<=}o{<}={<o<}∪{=o<}={<}∪{<}={<}).

Le graphe de voisinage pour A_3 est donné sur la Figure 9.

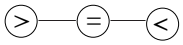


Figure 9 : Graphe de voisinage pour A_3. Le graphe est composé des relations comme nœuds et des voisinages conceptuels comme arcs.

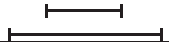
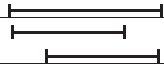
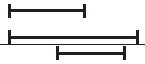
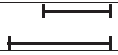
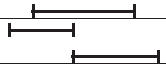
relation (r) : x r y	x/y	réciproque : y r^{-1}x
avant (b)		après
pendant (d)		contient
recouvre (o)		recouvert par
commence (s) (et termine avant)		commencé par (et termine après)
termine (f) (et commence après)		terminé par (et commence avant)
rencontre (m)		rencontré par
égale (e)		e

Tableau 3 : (d'après [Allen 1983]). Les 13 relations de A_{13}.

III.1.3. Algèbre d'intervalles

Un intervalle est une entité temporelle connexe qui dure. Il peut être vu comme un segment sur une droite. Une représentation métrique de l'intervalle est un couple de dates (date de début et date de fin) ou une date et un cardinal (date de début et durée). Les intervalles sont manipulés

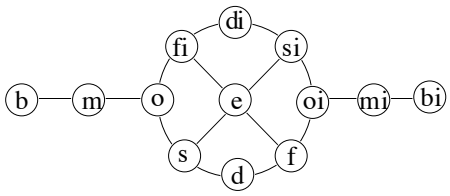


Figure 10 (d'après [Nökel 1988]) : Graphe de voisinage pour A_{13}. La déformation considérée (nommée A) est la translation continue d'une extrémité d'un intervalle, les trois autres étant fixes. Ainsi, si un intervalle en chevauche un autre (o) en bougeant la fin de celui qui est recouvert vers le passé ou la fin de celui qui recouvre vers le futur, ils vont se rencontrer (fi) ; en bougeant le début de celui qui est recouvert vers le futur ou la fin de celui qui recouvre vers le passé, ils vont se rencontrer (m) ; en bougeant le début de celui qui est recouvert vers le passé ou le début de celui qui recouvre vers le futur, ils vont se rencontrer (s).

à l'aide de 13 relations mutuellement exclusives et exhaustives (voir Tableau 3; l'algèbre sera nommé A_{13}).

L'opérateur de composition est représenté par une table [Allen 1983] similaire à celle du Tableau 2. Elle prendrait trop de place et n'est donc pas montrée ici.

Le voisinage dépend du type de modification que les éléments subissent. Freksa [1992] en propose trois : (A) déplacement continu d'une extrémité sans déplacer l'autre, (B) translation de l'intervalle et (C) déplacement simultané des deux extrémités sans modifier le centre de gravité. Un graphe de voisinage pour A_{13} est donné dans la Figure 10. Il correspond au A-voisinage, le seul voisinage qui sera considéré ici.

III.1.4. Des instants aux intervalles

Il existe un rapport étroit entre l'algèbre d'instants et l'algèbre d'intervalles. Ce rapport a été plus systématiquement exploré récemment [Hirsch 1996]. Ainsi, à n'importe quelle algèbre de relations correspond une algèbre d'intervalles en fonction de n'importe quelle relation r de l'algèbre initiale. Dans une telle algèbre les objets x en relation sont des couples d'objets $(x^- x^+)$ de l'algèbre initiale contraints par la relation r (c'est-à-dire que $x^- r x^+$). Ici, la relation r sera toujours une relation d'ordre. L'algèbre d'intervalles est définie comme suit :

- les relations entre intervalles x et y sont obtenues en considérant les quadruplets de relations (r_1, r_2, r_3, r_4) tels que $x^- r_1 y^-$, $x^- r_2 y^+$, $x^+ r_3 y^-$ et $x^+ r_4 y^+$.
- la relation réciproque est obtenue par distribution de la réciprocité et inversion de r_2 et r_3,
- la composition est obtenue par le produit matriciel où l'addition est remplacée par l'union et la multiplication par la composition.

Ainsi, l'algèbre d'intervalles A_{13} peut se définir à partir de l'algèbre d'instants A_3. La seule contrainte dans la définition d'un intervalle est que le début précède la fin au sens de la relation d'ordre (<). Lorsque le temps est régi par une relation d'ordre total (comme dans A_3), il y a potentiellement 3^4=81 combinaisons possibles. Cependant, les intervalles doivent respecter les contraintes suivantes $x^- < x^+$ et $y^- < y^+$. Ces deux contraintes sont appelées γ. À partir de ces deux contraintes il est possible de déduire 5 combinaisons de relations points à points qui sont impossibles dans un ordre :

1. $\gamma \cup \{x^- \geq y^-, x^+ \leq y^-\}$
2. $\gamma \cup \{x^- \geq y^+, x^+ \leq y^+\}$
3. $\gamma \cup \{x^- \leq y^-, x^- \geq y^+\}$
4. $\gamma \cup \{x^+ \leq y^-, x^+ \geq y^+\}$
5. $\gamma \cup \{x^- \geq y^+, x^+ \leq y^-\}$

L'élimination de toutes les combinaisons satisfaisant l'une de ces contraintes laisse seulement les 13 relations de A_{13} telles qu'exprimées au Tableau 4. La notation $\Rightarrow$ (resp. $\Leftarrow$) sera utilisée pour la conversion de A_{13} vers A_3 (resp. de A_3 vers A_{13}).

Deux techniques ont été proposées dans [Ligozat 1994] pour obtenir les voisinages conceptuels d'une manière générale et, en particulier, lors du passage à l'intervalle. La première

$x$ry	$x^-r_1y^-$	$x^-r_2y^+$	$x^+r_3y^-$	$x^+r_4y^+$	$x$ry	$x^-r_1y^-$	$x^-r_2y^+$	$x^+r_3y^-$	$x^+r_4y^+$
b	<	<	<	<	b^{-1}	>	>	>	>
d	>	<	>	<	d^{-1}	<	<	>	>
o	<	<	>	<	o^{-1}	>	<	>	>
s	=	<	>	<	s^{-1}	=	<	>	>
f	>	<	>	=	f^{-1}	<	<	>	=
m	<	<	=	<	m^{-1}	>	=	>	>
e	=	<	>	=					

Tableau 4 : Les 13 relations entre intervalles exprimées en fonction des relations entre leurs extrémités.

utilise les relations de connexité dans une représentation géométrique des relations comme des surfaces du demi-plan réel ; elle ne se généralise pas aisément. La seconde, considérant que tout intervalle décompose l'espace en trois zones, étant fortement fondée sur la structure d'ordre total des entiers naturels, n'est pas simplement adaptable à tout type d'intervalle.

III.2. Opérateurs de changement de granularité et leurs contraintes

À l'instar des autres chapitres, le but ici est de concevoir que deux représentations d'une même situation peuvent différer suivant la granularité. Les situations sont donc représentées dans un système relationnel. Dans l'exemple de la Figure 7, il y a trois représentations : (s) contenant F{m}T, (p) contenant F{m}R et R{m}T et (b) contenant F{m}R et R{b}T. Il devient alors clair que l'union de deux quelconques de ces descriptions donnera lieu à une description inconsistante (car F{m}T≠F{b}T=F{m}o{m}T). Comment accommoder le fait qu'il s'agisse de descriptions d'une même situation ?

Le principe développé ici considère deux opérateurs ↑ et ↓ (ascendant et descendant) capables de transformer une relation en un ensemble de relations susceptibles de la représenter sous une granularité plus grossière ou plus fine (l'opérateur → sera utilisé, lorsque n'importe lequel peut être utilisé). Lorsqu'il sera nécessaire de préciser les granularités initiales (g) et finales (g'), la notation ${}_g\uparrow^{g'}$ et ${}^g\downarrow_{g'}$ sera utilisée. Ces opérateurs seront généralisés aux disjonctions de relations et ainsi à l'ensemble des représentations.

Il sera alors possible de répondre aux questions essentielles :

— deux représentations sont-elles compatibles (l'une peut-elle être une représentation plus grossière que l'autre) ?
— quelles sont les positions relatives de deux représentations (laquelle est la plus grossière) ?
— quelles sont les représentations possibles de la situation représentée par une situation particulière sous une granularité plus grossière (resp. plus fine) ?

Les effets du changement d'échelle permettent d'imaginer des opérateurs de conversion ; un ensemble de propriétés qui doivent être satisfaites par tout système d'opérateurs de conversion est produit ci-dessous. Les contraintes sont purement algébriques et ne se réfèrent pas à une algèbre de relations particulière.

En fait, cet ensemble de propriétés est relativement restreint. La section suivante (§III.3) montrera cependant qu'il est suffisant pour restreindre le nombre d'opérateurs correspondant aux algèbres de relations temporelles présentées plus haut (§III.1) à un seul (en plus des opérateurs attendus que sont l'identité et la conversion vers tout).

III.2.1. Auto-préservation

L'*auto-préservation* signifie que, quelle que soit la conversion, une relation doit toujours avoir la chance d'être dans sa propre conversion. C'est une propriété qui permet que sous deux granularités proches la relation reste la même (ou encore que la vision d'une relation sous une granularité particulière ait une chance d'être aussi précise que sous une granularité un peu plus précise).

[1] $r \in \rightarrow r$ (auto-préservation)

Techniquement cette propriété ancre la conversion dans le voisinage de la relation perçue. Elle évite la proposition d'opérateurs ne tenant pas compte de la relation initiale (par exemple en convertissant une relation en sa réciproque).

III.2.2. Compatibilité avec le voisinage

Une propriété considérée initialement était la *préservation de l'ordre* — énoncée dans [Hobbs 1985] comme une équivalence : $\neg(\exists x,y)\ x>y \wedge \rightarrow x < \rightarrow y$. Cette propriété considère qu'une relation d'ordre (<) oriente la représentation. Elle dit que :

si $x > y$ alors $\neg(\rightarrow x < \rightarrow y)$ (préservation de l'ordre)

Cependant, la préservation de l'ordre a le défaut de requérir une relation d'ordre. Sa généralisation algébrique serait l'*évitement des réciproques* :

si $x\ r\ y$ alors $\neg(\rightarrow x\ r^{-1} \rightarrow y)$ (évitement des réciproques)

Cependant l'évitement des réciproques est une sur-généralisation de la préservation de l'ordre et entre en conflit avec l'auto-préservation dans le cas de relations auto-réciproques (c'est-à-dire telles que $r=r^{-1}$). La *compatibilité avec le voisinage*, bien que non énoncée explicitement dans [Euzenat 1993e] avait été prise en compte informellement : elle contraint la conversion d'une relation à former un voisinage conceptuel (et ainsi la conversion d'un voisinage conceptuel à être un voisinage conceptuel).

[2] $\forall r.\ \forall r',r'' \in \rightarrow r\ .\ \exists r_1,\ldots r_n \in \rightarrow r.\ r_1=r',\ r_n=r''$ et $\forall i \in [1,n\text{-}1]\ N(r_i,r_{i+1})$

(compatibilité avec le voisinage)

Une telle propriété a déjà été rapportée par Christian Freksa [1992] qui considère qu'un ensemble de relations doit être un voisinage conceptuel s'il doit être vu comme une représentation grossière d'une relation particulière. [2] est plus faible que les deux propositions précédentes car il n'interdit pas à la réciproque d'une relation d'être dans la conversion de celle-ci. Mais dans ce cas, elle contraint à ce qu'un ensemble de relations liant les deux opposés par voisinages soient aussi dans la conversion. La compatibilité avec le voisinage semble être la bonne propriété en partie parce qu'à l'opposé des deux précédentes elle n'interdit pas un opérateur qui convertirait toute relation à être l'ensemble support de l'algèbre (considérant ainsi une représentation non informative).

III.2.3. Distributivité conversion-réciprocité

Une propriété évidente pour la conversion est la symétrie qui établit que la conversion d'une relation entre un objet et un second doit être la réciproque de la conversion de la relation entre le second et le premier. Il est clair que la relation entre deux objets temporels étant symétrique, la conversion doit respecter cette symétrie.

[3] $(\rightarrow r^{-1}) = (\rightarrow r)^{-1}$ (distributivité de $\rightarrow$ sur $^{-1}$)

III.2.4. Compatibilité inverse

La *compatibilité inverse* dit que les opérateurs de conversion doivent être compatibles, c'est-à-dire que si la relation entre deux objets peut être vue comme une autre relation sous une autre granularité (après une certaine conversion), alors cette seconde relation à laquelle est appliquée l'opération inverse doit permettre de retrouver la relation initiale.

[4] $r \in \bigcap_{r' \in \uparrow r} \downarrow r'$ et $r \in \bigcap_{r' \in \downarrow r} \uparrow r'$ (compatibilité inverse)

Cette propriété correspond à l'expression $r \in \downarrow r'$ ssi $r' \in \uparrow r$.

Par exemple, si quelqu'un dans la situation (p) de la Figure 7 est capable d'imaginer que sous une granularité plus fine (par exemple la situation b) il y a un laps de temps entre la période de ratés et la perte d'altitude alors cette personne doit être prête à accepter que si elle est dans la situation (b) elle peut imaginer qu'il n'y a pas de délai entre eux sous une granularité plus grossière (comme p).

III.2.5. Idempotence

Une propriété considérée en général au premier abord est la *transitivité générale* :

$$g \rightarrow g' \cdot g' \rightarrow g''\ r = g \rightarrow g''\ r$$

Cette propriété est trop forte ; elle impliquerait par exemple que :

$$g \rightarrow g' \cdot g' \rightarrow g\ r = r$$

Bien entendu ce n'est pas possible car il n'y aurait alors pas de perte d'information lors du changement de granularité. Et si cela était possible, il n'y aurait alors plus besoin d'opérateurs de changement de granularité : chaque situation serait représentée de la même manière sous chaque granularité. Néanmoins, la *transitivité cumulée* peut être considérée :

$$ {}_{g}\uparrow^{g'}\cdot_{g'}\uparrow^{g''} r = {}_{g}\uparrow^{g''} r \text{ et } {}^{g''}\downarrow_{g'}\cdot{}^{g'}\downarrow_{g} r = {}^{g''}\downarrow_{g} r $$

Cependant, dans un calcul purement qualitatif, les différences de granularités (g) ne sont pas pertinentes et cette propriété devient une propriété d'*idempotence* des opérateurs :

[5] $\uparrow\cdot\uparrow=\uparrow$ et $\downarrow\cdot\downarrow=\downarrow$ (idempotence)

À première vue, il peut sembler plus naturel de disposer d'opérateurs non idempotents qui soient de moins en moins précis avec le nombre de conversions. Cependant, si la non idempotence est naturelle pour l'aspect quantitatif, elle ne l'est pas dans le cas qualitatif. En effet, l'application d'un opérateur doit donner le même résultat que la conversion soit entre deux granularités rapprochées ou éloignées. Si ce n'était pas le cas, et sachant qu'en accumulant des granularités proches la granularité éloignée peut être atteinte, le résultat serait différent en passant par dix petites conversions ou par une grande. Ainsi, si l'on veut avoir ${}_{g}\uparrow^{g'}\cdot_{g'}\uparrow^{g''} r = {}_{g}\uparrow^{g''}$, il faut avoir $\uparrow\cdot\uparrow=\uparrow$.

III.2.6. Indépendance de la représentation

L'indépendance de la représentation signifie que la conversion doit donner le même résultat qu'elle soit établie sur une algèbre d'intervalles ou sur l'algèbre dont celle-ci dépend.

[6] $\rightarrow r = \Leftarrow\rightarrow\Rightarrow r$

et (indépendance de la représentation)

$\rightarrow r = \Rightarrow\rightarrow\Leftarrow r$

Bien entendu, comme $\Leftarrow$ requiert que la relation entre les bornes autorise le résultat à être un intervalle, il y a des restrictions des résultats obtenus (ces restrictions correspondent exactement à la disparition d'un intervalle, ce qui n'est pas considéré ici).

En conclusion, rien ne dit que ces contraintes conduisent à un unique couple d'opérateurs ascendant/descendant pour une algèbre de relations donnée, mais ceux qui les satisfont seront de bons candidats.

DÉFINITION (système d'opérateurs) : Soit une algèbre de relations binaires, un couple d'opérateurs de conversion satisfaisant [1-5] est un système d'opérateurs de conversion cohérent pour cette algèbre. ◊

Il faut noter que le cadre ainsi défini concerne deux opérateurs reliés par des contraintes mais qu'il n'y a aucune spécificité de l'opérateur ascendant ou descendant. Par convention, si le système contient une unique relation d'équivalence (définie comme une relation e telle que $e=eoe=e^{-1}$ [Hirsch 1996]), l'opérateur qui transforme cette relation en un ensemble de relations strictement plus grand sera nommé l'opérateur descendant. Ceci correspond à l'intuition qui veut que plus la vue est grossière plus les éléments sont indiscernables (et sont alors sujets à

tomber sous la relation d'équivalence). Implémenter cette exigence comme une nouvelle contrainte aurait au moins deux conséquences :

— simplifier la recherche des solutions ;

— permettre d'introduire des contraintes spécifiques au sens de conversion.

Cependant, comme cela n'a pas été nécessaire pour le travail présenté ici, la solution la plus générale (c'est-à-dire n'introduisant pas l'orientation) a été retenue.

III.3. Résultats

Un certain nombre de résultats spécifiques aux représentations temporelles et spatiales ont pu être obtenus sur les opérateurs de conversion de granularité. Les résultats concernant les opérateurs ont été présentés dans [Euzenat 1993e] puis dans [Euzenat en révision] une preuve qu'ils étaient uniques a été fournie. Les résultats concernant les rapports entre composition et conversion ont été publiés dans [Euzenat 1993e]. Enfin ceux concernant l'existence et la non

relation : r	↑r	↓r
<	< =	<
=	=	< = >
>	> =	>

Tableau 5 : Opérateurs de conversion ascendant et descendants pour l'algèbre d'instants.

existence d'opérateurs ont été obtenus récemment [Euzenat& à paraître]… en réponse à une question posée lors de la présentation de [Euzenat 1995e].

III.3.1. Granularité pour l'algèbre d'instants

PROPOSITION : Le Tableau 5 définit les seuls opérateurs non auto-inverses de conversion ascendante/descendante pour A_3. ◊

Les opérateurs du Tableau 5 correspondent très bien à l'intuition. Par exemple, si l'exemple de la Figure 7 est modélisé par les bornes des intervalles (x^- pour la borne antérieure et x^+ pour la borne postérieure de x) il s'écrit en (b) par $F^+=R^-$ (le moteur s'est arrêté quand il a commencé à avoir des ratés), $R^-<R^+$ (le début des ratés est antérieur à leur fin), $R^+<T^-$ (la fin des ratés est antérieure à la perte d'altitude), en (p) par $R^+=T^-$ (la fin des ratés coïncide avec la perte d'altitude) et en (g) par $R^-=R^+$ (la période de ratés n'est pas perçue). Ceci est compatible avec la conversion de $R^+<T^-$ en $R^+=T^-$ (car $= \in \downarrow <$) et $R^-=R^+$ en $R^-<R^+$ ($< \in \uparrow =$)

III.3.2. Granularité pour l'algèbre d'intervalles

Puisque l'algèbre de Allen est une algèbre d'intervalles de plein droit, la contrainte d'indépendance de la représentation peut être appliquée pour déduire ses opérateurs de conversion de granularité. Ceci produit les seuls opérateurs possibles pour l'algèbre d'intervalles. Le Tableau 6 montre la traduction ainsi opérée des points vers les intervalles.

r	$\uparrow\Rightarrow r$				$\uparrow r$	$\downarrow\Rightarrow r$				$\downarrow r$
b	<=	<=	<=	<=	b m	<	<	<	<	b
d	>=	<=	>=	<=	d s f e	>	<	>	<	d
o	<=	<=	>=	<=	o s m e f^{-1}	<	<	>	<	o
s	=	<=	>=	=	s e	<=>	<	>	<	o s d
f	>=	<=	>=	=	f e	>	<	>	<=>	o^{-1} f d
m	<=	<=	=	<=	m	<	<	<=>	<	b m o
e	=	<=	>=	=	e	<=>	<	>	<=>	o f^{-1} d^{-1} s e s^{-1} d f o^{-1}

Tableau 6 : Transformation des opérateurs ascendants et descendants entre instants vers les quadruplets d'intervalles.

Ainsi la table des opérateurs de conversion pour l'algèbre d'intervalles est donnée ci-dessous. Les opérateurs satisfont les mêmes propriétés que ceux de l'algèbre d'instants.

PROPOSITION : Les opérateurs ascendant/descendant pour A_{13} du Tableau 7 sont les seuls qui satisfassent la propriété [6] par rapport aux opérateurs pour A_3 du Tableau 5. ◊

PROPOSITION : Les opérateurs ascendant/descendant pour A_{13} du Tableau 7 satisfont les propriétés [1-5]. ◊

relation : r	$\uparrow r$	$\downarrow r$	réciproque : r^{-1}	$\uparrow r^{-1}$	$\downarrow r^{-1}$
b	b m	b	b^{-1}	b^{-1} m^{-1}	b^{-1}
d	d f s e	d	d^{-1}	d^{-1} s^{-1} f^{-1} e	d^{-1}
o	o f^{-1} s m e	o	o^{-1}	o^{-1} s^{-1} f e m^{-1}	o^{-1}
s	s e	o s d	s^{-1}	s^{-1} e	d^{-1} s^{-1} o^{-1}
f	f e	d f o^{-1}	f^{-1}	f^{-1} e	d^{-1} f^{-1} o
m	m	b m o	m^{-1}	m^{-1}	o^{-1} m^{-1} b^{-1}
e	e	o f^{-1} d^{-1} s e s^{-1} d f o^{-1}			

Tableau 7 : Opérateurs de conversion ascendante et descendante des relations entre intervalles.

Le lecteur est invité à vérifier sur l'exemple de la Figure 7 que ce qui a été dit de l'algèbre d'instants est valide pour l'algèbre d'intervalles. La situation (b) est décrite par F{m}R (la période de fonctionnement normal rencontre celle des ratés) et R{b}T (la période de ratés précède strictement celle de la perte d'altitude) ; la situation (p) est décrite par R{m}T (la fin des ratés coïncide avec la perte d'altitude) et la situation (g) où la période de ratés n'apparaît plus par R{m}T (la période de fonctionnement normal précède immédiatement la perte d'altitude). Ceci est encore une fois compatible avec l'idée que sous une granularité plus grossière b peut devenir m ($m \in \uparrow b$) et sous une granularité plus fine m peut devenir b ($b \in \downarrow m$).

L'opérateur ascendant ne satisfait pas la condition de compatibilité des voisinages pour les B-voisinages (où les objets sont continûment translatés) car elle est violée par d, s et f. Il ne la satisfait pas non plus pour les C-voisinages (où ils sont continûment expansés ou contractés en préservant le même centre) car elle est violée par o, s et f. L'explication de cela est que ces deux

derniers voisinages ne sont pas fondés sur une translation indépendante d'une borne de l'intervalle mais sur une translation contrainte des deux bornes simultanément alors que cette indépendance est utilisée lors de la transformation de l'algèbre d'instants vers l'algèbre d'intervalles.

Les opérateurs obtenus correspondent exactement à la fermeture des relations que Gérard Ligozat [1990] a introduite dans son propre formalisme. Ceci est naturel car la fermeture, comme les opérateurs de conversion, procurent toutes les relations adjacentes (voir [Euzenat en révision]).

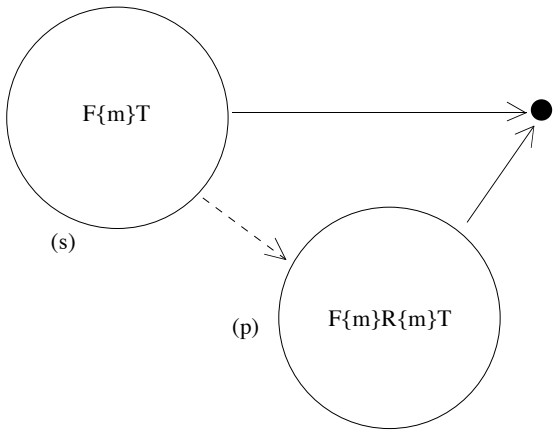


Schéma 15 : Changement de granularité comme approximation (2). Les changements possibles de granularité sont formellement exprimés mais ne correspondent pas à un homomorphisme. En effet, F{m}T n'est pas conservé (ni déductible) dans (p).

Par ailleurs, si seuls les liens de causalités sont considérés dans le système de [Patil 1981], ils peuvent être assimilés à la relation "meets" de l'algèbre d'intervalle (plus exactement, on peut se considérer dans A_5 où l'on dispose de b, m, e, mi, bi). On observe alors des transformations différentes d'ici : $\downarrow$m={b, m}, $\uparrow$b={b, m}, le reste étant l'identité. L'aspect discret des objets observés empêche de considérer des superpositions nécessaires dans l'appauvrissement de A_{13} vers A_5.

III.3.3. Résultats d'existence pour les algèbres de relations binaires

La question de l'existence, en général, d'opérateurs de changement de granularité correspondant aux contraintes [1-5] peut être posée. Comme les solutions pour l'algèbre d'instants (et donc d'intervalles) ont été obtenues par énumération, le travail doit être refait pour chaque algèbre. Cependant, les deux résultats suivants sont obtenus [Euzenat en révision]. Le premier signifie qu'il existe des petites algèbres sans opérateur non triviaux.

PROPOSITION : L'algèbre de relations binaires fondée sur deux éléments a et a^{-1} tels que $N(a,a^{-1})$ n'a pas d'autres opérateurs de conversion de granularité satisfaisant [1-5] que l'identité et l'application non informative. $\Diamond$

Un résultat plus intéressant est celui de l'existence de tels opérateurs pour un ensemble important d'algèbres. Il ne fournit cependant que des conditions suffisantes pour de tels

opérateurs. Bien que constructive, cette proposition ne permet pas d'engendrer toutes les solutions.

PROPOSITION : Soit une algèbre de relations binaires contenant (au moins) deux relations a et b telles que $N(a,b)$ (il est supposé que $N(a^{-1},b^{-1})$), il existe un couple d'opérateurs ascendant/descendant de conversion défini par :

1. si a et b sont auto-réciproques :
   - $\downarrow a = \{a, b\}$, $\uparrow b = \{a, b\}$
   - le reste étant l'identité ;
2. si a est seul auto-réciproque :
   - $\downarrow a = \{a, b, b^{-1}\}$,
   - $\uparrow b = \{a, b\}$, $\uparrow b^{-1} = \{a, b^{-1}\}$,
   - le reste étant l'identité ;
3. si a et b ne sont pas auto-réciproques :
   - $\downarrow a = \{a, b\}$, $\uparrow b = \{a, b\}$,
   - $\downarrow a^{-1} = \{a^{-1}, b^{-1}\}$, $\uparrow b^{-1} = \{a^{-1}, b^{-1}\}$,
   - le reste étant l'identité. ◊

Il faut noter qu'il y a en général de nombreux opérateurs pour une algèbre donnée. Cependant, des contraintes du type de celle liant l'algèbre d'intervalles à l'algèbre d'instants restreignent beaucoup le nombre de solutions.

III.3.4. Granularité et inférence

La composition de relations symboliques est le mécanisme d'inférence privilégié dans les systèmes de représentation symbolique du temps. Une des propriétés intéressantes à cet égard est l'indépendance des résultats de cette opération par rapport à la granularité (équation [7]). La distributivité de $\rightarrow$ sur o dénote l'indépendance des inférences par rapport à la granularité sous laquelle elles sont effectuées.

[7] $$\rightarrow(\rho_1 \circ \rho_2) = (\rightarrow\rho_1) \circ (\rightarrow\rho_2)$$

(distributivité de $\rightarrow$ sur o)

Cette propriété est uniquement satisfaite pour la conversion ascendante dans A_3.

PROPOSITION : La conversion ascendante pour A_3 satisfait la propriété [7]. ◊

La propriété [7] n'est pas valide en ce qui concerne l'algèbre d'intervalles : soient trois intervalles *x*, *y* et *z* tels que *x*b*y* et *y*d*z*, l'application de la composition donne *x*{b o m d s}*z*, ce qui une fois converti sous une granularité plus grossière donne *x*{b m e d f s o f^{-1}}*z*. Par contre, si la conversion est tout d'abord appliquée, elle retourne *x*{b m}*y* et *y*{d f s e}*z* ce qui, une fois la composition appliquée donne *x*{b o m d s}*z*. Ce résultat s'explique parce qu'en convertissant d'abord, l'information qu'il existe un intervalle *y* empêchant *x* de terminer *z* est perdue. Par contre, si la relation liant *y* à *x* et *z* est conservée, alors la propagation retrouvera la précision perdue : {b m e d f s o^{-1}}o{b o m d s}={b o m d s}. Cependant, ceci ne peut être garanti car il

est possible que y soit si petit qu'il ne soit pas considéré sous la granularité plus grossière. Alors l'information correcte par rapport à cette granularité est celle retournée en appliquant tout d'abord la composition : x peut rencontrer la fin de z sous une telle granularité.

Si l'équation [7] ne peut être satisfaite pour la conversion ascendante dans l'algèbre d'intervalles, la conversion ascendante est super-distributive par rapport à la composition.

PROPOSITION : La conversion ascendante pour A_{13} satisfait la propriété suivante :

[8] $$(\uparrow \rho_1) \circ (\uparrow \rho_2) \subseteq \uparrow(\rho_1 \circ \rho_2)$$

(super-distributivité de $\uparrow$ sur $\circ$)

Un phénomène similaire apparaît avec les opérateurs de conversion descendants (à la fois pour l'algèbre d'instants et celle d'intervalles). Soient trois instants x, y et z tels que $x>y$ et $y=z$, d'une part la composition des relations produit $x>z$ ce qui une fois converti donne $x>z$. D'autre part, la conversion des deux relations initiales donne $x>y$ et $y\{<=>\}z$ ce qui retourne $x\{<=>\}z$ par application de la composition. C'est la situation inverse de la situation précédente : elle tient compte du fait que l'indiscernabilité de deux instants ne peut être assurée sous une granularité plus précise. Bien entendu, si tout est d'abord converti, le résultat est aussi précis que possible. La conversion descendante est sous-distributive par rapport à la composition.

PROPOSITION : La conversion descendante pour A_{13} et A_3 satisfait la propriété suivante :

[9] $$\downarrow(\rho_1 \circ \rho_2) \subseteq (\downarrow \rho_1) \circ (\downarrow \rho_2)$$

(sous-distributivité de $\downarrow$ sur $\circ$)

Ces deux dernières propositions peuvent être utiles afin de propager les contraintes en vue d'obtenir un résultat en accord avec une granularité précise : par exemple, dans le cas de la conversion ascendante, si aucun intervalle ne disparaît, il vaut mieux d'abord appliquer la conversion puis la composition.

Ces dernières propriétés ont été découvertes indépendamment dans le cas qualitatif [Euzenat 1993e] et dans le cas quantitatif [Bettini& 1998] par le biais d'un algorithme d'approximation des contraintes quantitatives.

Conclusion

La granularité temporelle, et plus généralement la granularité dans les algèbres de relations, est exclusivement une affaire d'approximation entre représentations. Les travaux présentés ci-dessus donnent un cadre général pour le changement de granularité qui fixe les limites de celui-ci. L'utilisation de ce type d'opérateurs est, comme cela a déjà été dit, de discerner lors de la confrontation de deux représentations les représentations intrinsèquement contradictoires de celles qui peuvent n'être qu'un effet de la granularité. Le chapitre suivant examinera la confrontation de représentations indépendantes liée plus fortement à l'intervention des utilisateurs.

Ce travail peut être inscrit dans un cadre un peu plus vaste concernant les relations entre représentations temporelles d'une même situation. Ainsi (voir Figure 11), il est possible de considérer diverses représentations d'une même situation dépendant de différents langages (Quantitatifs, qualitatifs ou d'affaiblissement de représentations qualitatives — utilisant moins de relations) ou de différentes résolutions (par une discrétisation ou une granularité plus grossière). Cette simplification de la représentation de l'espace en fonction de la tâche à accomplir a aussi été mis en évidence dans le travail de Laurent Buisson [1990]. Ce travail se prolonge dans le cadre de la thèse de Vincent Cligniez où la représentation de l'espace est le support de l'intégration de modèles de simulation numérique. Ainsi, à l'instar de ce qui a été présenté pour les représentations par objets, le système ARSEN [Cligniez 1998] est capable de produire certains types de représentations, adaptés à un modèle de simulation, d'intégrer le résultat d'une simulation dans la représentation (qui va être utilisée par un autre modèle de simulation) et de maintenir les dépendances entre ces différentes représentations.

Un ensemble de programmes MAPLE [Redfern 1994] permettant de passer de certaines représentations à d'autres de manière automatique a été développé [Euzenat 1998a]. Cela permet de disposer d'un jeu de construction de représentations intéressant. Le but d'une exploration de ces représentations serait d'établir les conditions de commutation de la Figure 11 et, par exemple, de fonder quantitativement les opérations qualitatives présentées ici.

Le changement de granularité qualitatif pourrait, par ailleurs, être fondé sur un modèle qualitatif du changement d'échelle en introduisant la notion d'erreur dans cette modélisation comme l'a étudié Hidde De Jong [1998]. En effet, le changement de granularité peut être vu comme une erreur (ou un jeu) dans la perception de la situation.

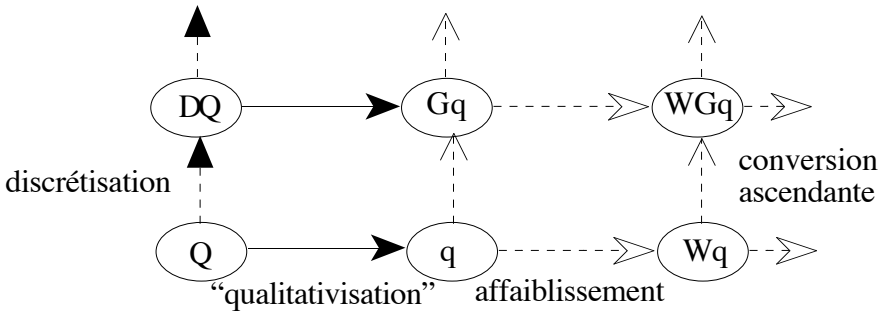


Figure 11 : Les relations existant entre différentes représentations de la même situation. Elle dépendent de la nature de la représentation (Quantitative/qualitative), de son degré de simplification (Continu/Discret — Exact/Granularisé) et de l'expressivité du langage utilisé (qui peut être affaibli). Les flèches sans extrémités signifient que le type de transformation peut continuer à s'appliquer sur les représentations du même type.

Crédits

Les travaux présentés dans cette partie sont le fruit d'une recherche personnelle que j'ai continué à développer au cours des ans.

J'ai eu l'occasion d'encadrer certains étudiants en liaison avec cette problématique. Johannes Stein a développé sous ma direction pour son „Diplom-Arbeit" [Stein 1991] une application test mêlant (la première version de) TROEPS et changement de granularité dans les termes décrits ici. Damien Raczy a effectué un DEA de sciences cognitives sous ma direction dans lequel il a étudié

la notion de voisinages conceptuels chez les sujets humains et son application à la théorie des modèles mentaux ([Johnson-Laird 1983]) [Raczy 1996]. Enfin, Vincent Cligniez a développé dans le cadre de sa thèse (co-encadrée par René-Michel Faure, Laurent Buisson et moi-même) un travail sur la représentation de l'espace destinée à l'intégration de différents modèles [Cligniez 1998].

IV. CONSTRUCTION COLLABORATIVE DE BASES DE CONNAISSANCE

Les rapports entre différents points de vue sur un domaine à modéliser peuvent être pris en compte lors de la collaboration de plusieurs spécialistes concourant à la conception d'une base de connaissance. Un des aspects intéressants de cette problématique consiste à éviter que les bases obtenues ne soient la simple juxtaposition de bases individuelles (comme ce peut être le cas avec les points de vue). Pour cela il faut doter les collaborateurs d'outils leur permettant de confronter leur connaissance afin d'en permettre la fusion (§IV.1). Cette approche est justifiée par des besoins applicatifs dans le cadre de la construction de bases de connaissance scientifique [Euzenat 1995a] ou des mémoires d'entreprises [Euzenat 1996b]. Deux voies sont explorées ici pour établir la synthèse des diverses représentations.

Une première voie est celle de la révision (au sens logique du terme) de bases de connaissance par objets (§IV.2). En partant d'une définition sémantique de TROEPS, les notions d'inconsistance et d'incohérence, qui dénotent des conflits entre les différentes modélisations, sont caractérisées syntaxiquement. Une notion de révision utilisant la notion de modèles maximaux consistants et sélectionnant ceux-ci suivant des critères globaux (minimisation du changement) est ensuite présentée. Ceci permet d'obtenir les « meilleures » représentations du domaine fusionnant deux représentations qui pourront être présentées aux utilisateurs. Ces travaux constituent la première contribution à la révision au sein d'une représentation de connaissance par objets.

La seconde voie (§IV.3) doit surtout être vue comme une aide à la confrontation des bases de connaissance. Elle instaure un protocole permettant de construire un ensemble de bases de connaissance structurées en un arbre et contenant les représentations adoptées par un nombre croissant de personnes. Au delà des détails du protocole d'acceptation d'une contribution à une base de connaissance (fondée sur la métaphore de soumission à un journal) [Euzenat 1995a], le protocole permet d'examiner les rapports entre les différentes bases mises en jeu.

Problème : concilier les représentations de différentes provenances

Jusqu'ici, la coexistence de diverses représentations, soit en rapport d'approximation soit incomparables, a été organisée. Le chapitre précédent a montré, dans un cadre restreint, comment détecter des incompatibilités entre représentations et l'inférence de passerelles (voir §II.3) permet d'établir des rapports entre représentations. Mais aucun moyen n'a été donné pour confronter les représentations. C'est l'objet du présent chapitre.

Le problème posé par ce dernier chapitre est celui de l'agrégation de différentes représentations provenant de sources diverses. La préoccupation principale est de permettre à divers agents (des chercheurs ou des agents logiciels), disposant de représentations d'un domaine, de les partager. Une préoccupation semblable a été présentée au chapitre précédent : des agents décrivant une situation à des granularités différentes.

Contrairement aux problèmes posés précédemment, celui-ci est très ouvert et n'est pas posé dans un cadre formellement fermé. Par ailleurs, en fonction du résultat visé, plusieurs solutions peuvent être retenues qui sont présentées ci-dessous (voir Schéma 16). La première est une approche par union qui retient tout ce qui est exprimé par chacun des agents — ceci rejoint l'approche par points de vue de TROEPS — alors que la seconde est une approche par

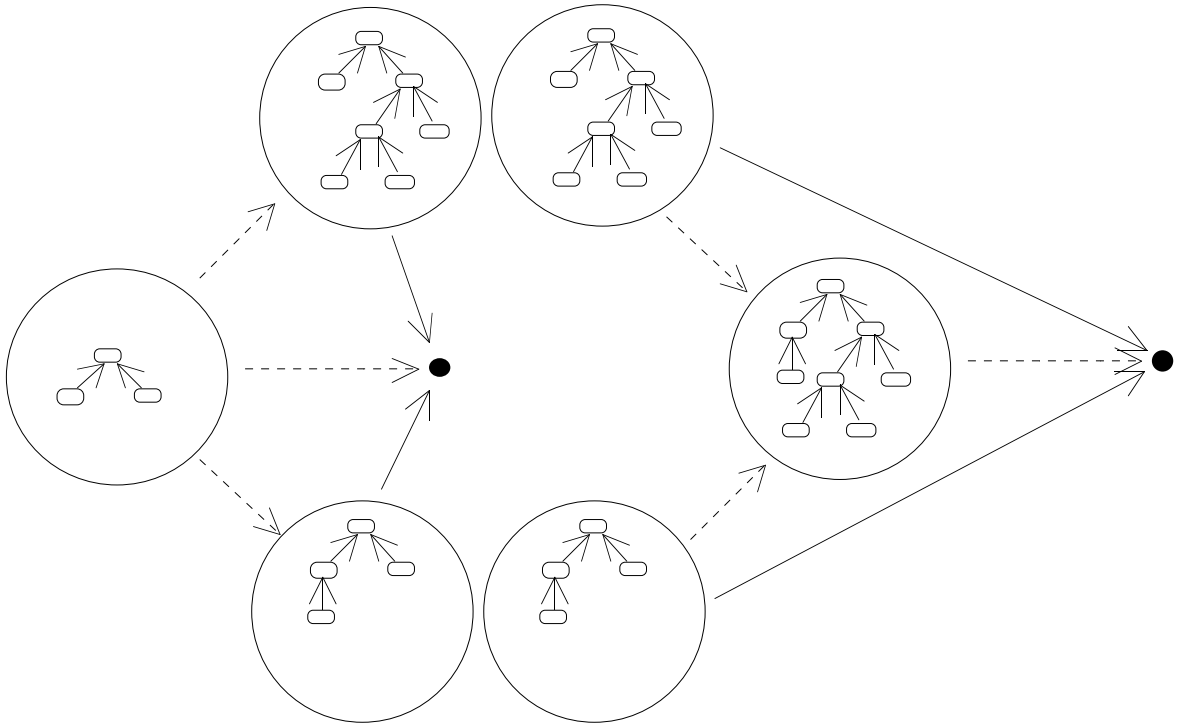


Schéma 16 : L'agrégation de représentations comme produit ou comme intersection ? Si deux représentations, à l'instar des points de vue du chapitre II, considèrent deux taxonomies différentes, il est possible d'agréger ces représentations en en prenant le plus petit dénominateur commun (à gauche) ou le plus grand dénominateur commun (à droite). Le résultat sera différent en ce que dans le premier cas, la représentation résultante est approximée par les représentations initiales alors que dans le second, c'est l'inverse.

intersection qui ne retient que ce que tous les agents expriment. Cette problématique a déjà été considérée dans de nombreux travaux [Baral& 1992, Cholvy 1994].

Un problème particulier posé par ces solutions est leur caractère à la fois universel et non discutable — qui est, bien entendu, justifié dans le cadre où elle sont utilisées. L'approche développée ici se place dans le contexte de l'aide à la construction de bases de connaissance (plus particulièrement scientifique) et cherche à fournir une aide aux différents agents. Cette aide leur est fournie en leur signalant les antagonismes entre les différentes descriptions et en proposant des solutions pour les surmonter. À la suite de quoi, un protocole permet aux agents eux-mêmes de décider ce qui doit être intégré à une base de connaissance représentant le consensus entre ces agents.

Illustrer le problème décrit, par la biologie, est relativement simple. Comme cela a déjà été présenté au chapitre II, certains biologistes s'intéressent à la fonction des gènes, d'autres à l'interaction entre les gènes, d'autre encore à la structure des gènes. Chacun dans son domaine fait progresser la connaissance en génétique et la connaissance globale sur le gène. Il peut sembler naturel que cette connaissance puisse être mise en commun car, par exemple, il est important de connaître la structure d'un gène lorsqu'il est inhibé par un autre. Cependant, l'idée de mettre en commun le contenu de bases de connaissance décrivant le génome est délicate. Il risque en effet d'y avoir conflit entre le contenu des différentes bases (par exemple, une appréhension purement théorique des interactions géniques a permis de supputer une interaction qu'une connaissance sur la structure permet d'expliquer autrement — deux gènes régulés par un même troisième). Il est vital de présenter aux différents biologistes les conflits entre leurs

modélisations et leurs données de manière à ce qu'ils puissent en discuter. C'est le rôle des travaux présentés ici.

Perspective historique

Les travaux présentés dans ce chapitre sont relativement composites et sont donc reliés à de nombreux champs d'activité : bases de données distribuées, révision, systèmes multi-agents, travail collaboratif… C'est donc un champ d'activité relativement ancien qu'il ne serait pas raisonnable de songer résumer ici. Il est cependant intéressant d'en décrire l'histoire récente.

Notre préoccupation pour la construction collaborative de bases de connaissance est liée à l'avènement d'Internet et à la disponibilité d'un réseau chez les utilisateurs et principalement des biologistes. C'est en 1992 que François Rechenmann [1993] a proposé l'idée de la construction de bases de connaissance par plusieurs chercheurs. Ce travail s'inscrivait plutôt en opposition avec la vague montante du partage de connaissance [Neches& 1991] qui consistait à ce qu'une base de connaissance construite en un point puisse être utilisée par de nombreux utilisateurs. Des travaux semblables étaient sans doute sous-jacents à la métaphore de la communauté scientifique [Kornfeld& 1981] dans le domaine des systèmes multi-agents. Par contre, le modèle de tableau noir, développé en intelligence artificielle distribuée [Nii 1986], n'est pas focalisé sur la construction de bases de connaissance (mais sur la résolution de problèmes, une fois la base construite). Il existe aussi tout un courant en intelligence artificielle, perpendiculaire à toutes les sous-branches, qui cherche à reproduire les principes de la découverte scientifique [Falkenhainer& 1988, Karp 1993, Martin 1996, De Jong& 1997, Corruble& 1997].

Les aspects de fusion de bases de connaissance étaient présents dans de nombreux domaines. C'est tout d'abord le cas en bases de données où d'important travaux avaient été faits dans le cadre de PROLOG sur ce qui allait s'appeler bases de données fédérées [Baral& 1992, Subramanian 1994]. C'était aussi le cas sur la fusion de données [Cholvy 1994]. C'était enfin le cas en représentation de connaissance sous la forme de la révision dans les bases de connaissance [Alchourrón& 1985]. Ce dernier cas est sans doute le plus important dans la suite car il a servi de guide aux développements présentés et il correspond bien au cadre de la correction de bases. Le domaine de la révision a fait couler beaucoup d'encre ces 15 dernières années (voir un panorama dans [Sombé 1994]) mais semble dans une impasse dans le sens où, à côté de nombreux travaux théoriques, ne figure aucune application pratique en raison de sa complexité algorithmique.

Par ailleurs, très peu d'études ont été dévolues à la révision dans les représentations par objets. Le travail le plus important concerne la révision dans les logiques terminologiques [Nebel 1990, 1992]. Mais la complexité du test de consistance dans les logiques terminologiques n'a pas permis de développer des systèmes opérationnels. Récemment Norman Foo [1995] a proposé des méthodes pour réviser des bases de graphes conceptuels, Yolanda Gil [&1997] des scripts procéduraux pour corriger des méthodes de résolution de problèmes et de nombreux autres travaux ont étudié la réorganisation de bases de données à objets (avec des propriétés invariantes et des règles pour les restaurer) [Banerjee& 1987]. Ces travaux donnent une idée des méthodes à mettre en œuvre pour réviser effectivement une base d'objets mais ne

considèrent pas explicitement la sémantique des formalismes de représentation de connaissance. Seul, Theodoratos [1995] a étudié le problème considéré ici dans le cadre d'un langage légèrement différent.

L'originalité de nos travaux dans ce domaine [Crampé 1997, Crampé& 1998] consistent principalement d'une part à leur adaptation très étroite à une représentation de connaissance par objets qui permettra de les implémenter et d'autre part à l'inclusion de l'utilisateur dans la boucle de révision.

Le second point important, plus relié au travail collaboratif, concerne l'édition simultanée d'une base de connaissance et le protocole qui permet de construire une base consensuelle. L'édition de base de connaissance sur le "worldwide web" date de quelques années seulement et peu de systèmes l'implémentent. Elle oblige à traiter sérieusement le problème de la concurrence d'accès simultané aux bases de connaissance. Très peu de travaux s'intéressent à ce domaine : APECKS [Tennison& 1998] permet l'édition concurrente sans contrôle ; le site interactif de systématique (SIS) permet l'annotation concurrente plus que l'édition ; ONTOLINGUA [Farquhar& 1995] permet de protéger l'accès en édition et de faire collaborer un groupe d'utilisateurs mais n'exerce aucun contrôle sur les modifications effectuées [Alemany 1998] ; GFP-EDITOR [Karp& 1997] implémente un mécanisme de contrôle optimiste qui permet à chaque utilisateur d'éditer une copie et qui tente de régler les problèmes lors de l'intégration ("commit") de la connaissance ; Co4 [Euzenat 1997a] permet à chacun d'éditer une base locale qui est une base de plein droit et exerce un contrôle a priori lors de l'intégration dans la base qui doit être réglée par l'ensemble des utilisateurs (et indépendamment de la consistance de la connaissance ajoutée).

Nos travaux sur le protocole de Co4 sont assez liés aux travaux sur les systèmes multi-agents et en particulier à l'approche qui consiste à concevoir des agents capables de choisir dynamiquement le protocole avec lequel ils vont tenir une conversation [Demazeau 1995] (par opposition à des agents obéissant toujours au même protocole fut-il très général tel que celui du réseau de contractants [Smith 1980]). Ils sont aussi en rapport avec la problématique de la distribution de la connaissance parmi un ensemble d'agents (on peut voir dans le Schéma 16, une incarnation de la distinction entre connaissance distribuée et connaissance commune [Fagin& 1995]). Par ailleurs, le protocole de Co4 décrit ci-dessous est très complet par rapport aux protocoles développés dans la littérature sur les systèmes multi-agents.

Enfin, la conception du protocole de Co4 a profité des travaux effectués depuis des années dans l'établissement de consensus dans les systèmes distribués [Fisher& 1985, Chaudhri& 1992, Kumar& 1996].

IV.1. Architecture et principe

Le problème de la construction de bases de connaissance consensuelles s'est posé dans le contexte de la construction de bases de connaissance scientifique et plus précisément dans le cadre de la génétique moléculaire [Rechenmann 1995]. Le projet SHERPA, après l'utilisation des outils qu'il développe dans ce cadre, s'est engagé en 1992 dans un projet destiné à industrialiser l'approche.

À l'époque le contexte de la génétique moléculaire était très particulier :

— les grands projets de séquençages ne pouvaient se faire qu'en faisant collaborer des dizaines de laboratoires ;
— les biologistes étaient par ailleurs parfaitement habitués à l'utilisation des réseaux informatiques, ne serait-ce que pour consulter les grandes bases de données de séquences.

À titre exploratoire, la possibilité d'exploiter ces caractéristiques afin de permettre aux chercheurs de construire des bases de connaissance de manière collaborative a été considéré comme sujet de travail. Pour leur permettre d'appréhender l'outil d'une manière naturelle, la métaphore de la soumission à une revue scientifique et plus généralement la métaphore de la collaboration entre scientifiques a été adoptée.

IV.1.1. Principes

Au départ, la métaphore de la recherche scientifique était assez vague. Elle consistait à faire dialoguer les chercheurs entre eux dans le dessein de construire une base de connaissance consensuelle. Il s'agissait de leur fournir une aide afin de comparer des éléments de connaissance et d'établir s'ils étaient contradictoires, similaires, plus ou moins généraux l'un que l'autre [Rechenmann 1995]. Cette approche ne peut être abordée telle quelle, mais un certain nombre d'éléments trouvent un support dans le logiciel TROEPS (détection d'inconsistance, mécanisme de classification, mesure de similarité).

Dans les développements qui ont suivi, seule une partie de cette métaphore a été préservée : celle de la soumission à une revue scientifique où, lorsqu'un utilisateur juge que certains éléments de connaissance doivent être intégrés à une base consensuelle, il va les soumettre à cette base qui organise le processus de relecture. Les autres souscripteurs auront alors connaissance de la soumission et pourront donner leur avis en acceptant, rejetant ou amendant la proposition initiale. Le protocole mis en œuvre pour implémenter ceci diffère de celui des revues actuelles dans le sens où :

— la base de connaissance doit rester consistante (mais il est possible de soumettre un retrait de connaissance) ;
— toute modification doit être acceptée par l'ensemble des parties (*consensus*).

L'architecture logicielle (voir Figure 12) proposée permet de mettre les bases de connaissance des différents intervenants en relation. Cette architecture ne préjuge pas de l'utilisateur des bases feuilles (ce peut être un programme ou un être humain) alors que les bases consensuelles ou bases de groupe fonctionnent sans intervention humaine. Chaque base de groupe est réputée représenter le consensus établi entre ses bases souscriptrices. L'architecture est récursive, si bien que le consensus peut être établi par exemple d'abord entre les membres d'un même laboratoire avant d'être discuté entre les laboratoires. L'intérêt de multiplier les bases de connaissance est d'une part d'éviter la modification entrant en conflit avec celle d'autres intervenants (comme dans ONTOLINGUA [Farquhar& 1995]) et d'autre part de permettre aux intervenants de poursuivre leur travail personnel sans avoir à le divulguer prématurément. Ce dernier point permet de pallier certains inconvénients dus aux délais de prise de décision (voir un exemple au §IV.3.1).

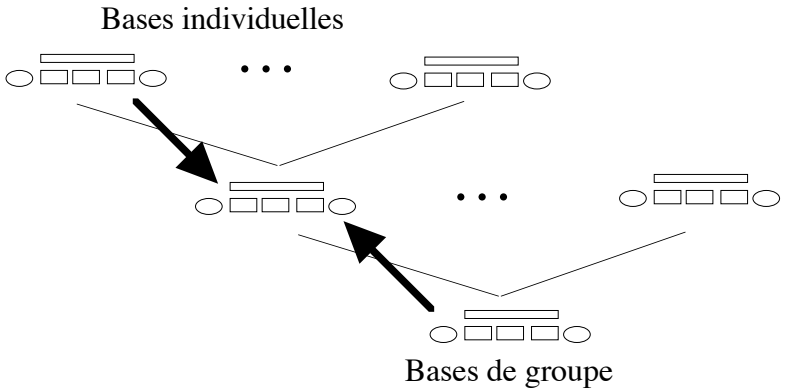


Figure 12 : Architecture hiérarchique et flot de messages (flèches sombres) de CO_4. Les bases sont organisées en un arbre dont les feuilles sont les bases individuelles et les nœuds représentent le consensus entre leurs fils. Les messages descendants comprennent la soumission d'une proposition, les réponses d'acceptation, de refus où de contre-proposition. Les messages ascendants comprennent la diffusion des propositions acceptées et des appels à commentaires sur les propositions soumises.

Le dispositif rapidement tracé obéit à quelques principes simples qui sont résumés ci-dessous :

Uniformité : Chacun (base individuelle ou base de groupe) dispose du même outil pour représenter la connaissance. Ceci est bien entendu nécessaire pour assurer l'intégration des contributions (à noter que le logiciel autorise l'intégration de connaissance sous forme d'objets mais aussi sous forme textuelle ou terminologique).

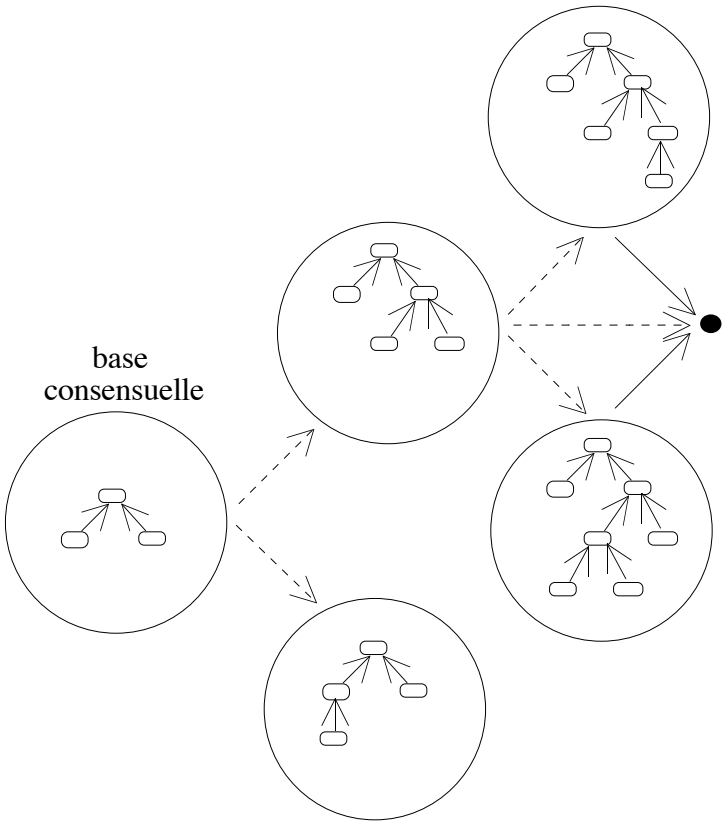


Schéma 17 : Statiquement, l'architecture des bases dans CO4 semble correspondre au modèle de l'intersection : les bases consensuelles renferment ce qui est commun aux bases souscriptrices. Ainsi, il semble que la construction éloigne les intervenants de l'objectif.

Consistance : L'accent est mis sur la consistance entre les apports de chacun. Si dans un premier temps ceci peut être vu comme une contrainte trop forte, il faut considérer que c'est surtout une aide afin que les conflits soient explicités et discutés entre les différents intervenants et non ignorés (comme cela peut se passer dans les revues scientifiques textuelles). Même s'il semble naturel que des spécialistes aient une vision contradictoire (entre eux) d'un même domaine, il est inutile de la stocker telle quelle [Euzenat 1996b] : il vaut mieux qu'ils rendent compte des différences et si possible de leurs raisons. La conception d'une révolution reposant sur des principes complètement nouveaux [Kuhn 1959] peut toujours être développée dans une nouvelle base qui sera ensuite confrontée à l'ancienne base. Enfin, le but est d'obtenir une vision cohérente du domaine considéré.

Bonne volonté : L'ensemble de cet édifice est basé sur la bonne volonté des intervenants (ils soumettent leur connaissance et leur opposition à certaines contributions) et leur bon usage (ils ne diffèrent pas des publications en ne répondant pas aux appels à commentaires, il ne consomment pas les ressources inutilement, ils ne soumettent pas de propositions qu'ils savent ne pas approximer la situation cible). Indépendamment de comment cette hypothèse de bonne volonté se traduit — et peut être relâchée — il est clair qu'un tel système ne peut fonctionner qu'entre des individus ayant un intérêt commun bien compris (celui de construire ensemble une base de connaissance)… comme c'est normalement le cas lorsque l'on participe conjointement à la progression de la connaissance.

IV.1.2. Défis

Afin de développer un tel système, deux problèmes techniques spécifiques ont été identifiés et examinés dès 1995.

Le premier problème concerne l'élaboration d'un opérateur de révision permettant de ne pas heurter les utilisateurs lorsque le système ne peut accepter une connaissance inconsistante. En effet, il n'est pas commun, même chez les scientifiques, d'appréhender toute l'étendue des assertions proférées et ainsi de pouvoir anticiper les conflits possibles. Par ailleurs, même lorsque le conflit est détecté, il n'est pas aisé de déterminer d'où il provient. Ceci est d'autant plus vrai si la connaissance considérée met en jeu des domaines divers que l'auteur ne connaît pas forcément. Un système de révision permettant de proposer aux utilisateurs les moyens de modifier la base afin d'éliminer ce type de conflit a donc été développé. Les résultats de ces travaux sont exposés au §IV.2.

Le second problème consiste à trouver un moyen de gérer la communication entre les différents utilisateurs de telle sorte que ceux-ci ne mettent pas en péril le contenu de la base de connaissance consensuelle. À cet effet, un protocole de communication a été conçu pour permettre aux utilisateurs de soumettre la connaissance à une base consensuelle et de contrôler le contenu de cette dernière. Les résultats de ces travaux sont présentés au §IV.3.

Le langage d'expression de la connaissance concernant le premier problème sera celui de la représentation de connaissance par objets déjà présenté au chapitre I. La seconde solution ne dépend d'aucun langage de représentation particulier pour peu qu'il existe des primitives (test de consistance, ajout d'éléments, etc.) pour intégrer le protocole et le langage.

IV.2. Révision dans une base de connaissance par objets

Le travail présenté ici a pour but de transférer les résultats établis en logique dans le domaine des représentations de connaissance par objets et, surtout, d'obtenir un système utilisable dans le domaine de la correction d'erreurs durant l'édition d'une base de connaissance. Deux points importants sont donc à prendre en compte :

— la révision doit tirer parti de la structure des formalismes objets (elle sera conçue sur un langage qui autorise une détection rapide des inconsistances et proposera une interprétation des révisions minimales en termes d'objets) ;
— la révision peut tirer parti de la présence de l'utilisateur pour choisir interactivement parmi les révisions minimales.

Une formalisation d'un langage de représentation fondé sur les objets est tout d'abord présentée (§IV.2.1) afin de caractériser formellement la révision à effectuer. Puis une manipulation syntaxique préservant la sémantique est présentée (§IV.2.2). Elle permet à la fois de stocker les bases de connaissance sous une forme facilement manipulable et d'exprimer les notions de révision plus simplement. La révision est alors introduite et les critères de minimalité présentés (§IV.2.3) avant de montrer en quoi ceux-ci se transcrivent facilement en termes d'objets. Enfin, une implémentation interactive de cette notion de révision au sein de TROEPS est décrite (§IV.2.4). Les preuves des propositions se trouvent dans [Crampé 1997].

IV.2.1. Langage de représentation de connaissance par objets

La définition d'un langage de représentation par objets sur lequel fonder la révision a deux objectifs : définir formellement la sémantique des objets et permettre une transposition plus aisée de la révision logique vers les objets. Bien que le langage présenté dans cette section puisse sembler simple, il couvre une partie importante des besoins des bases de connaissance dont le but n'est pas de résoudre tous les problèmes imaginables mais plutôt d'organiser la connaissance d'un domaine et de répondre à quelques requêtes simples (par exemple [Euzenat& 1997b]). Cela dit, il serait bon d'élargir la révision à l'ensemble de TROEPS (par exemple en ajoutant les constructeurs liste et ensemble et surtout en considérant les références circulaires).

assertion	signification	exemple	prédicat
$v{=}v'$	v est la même valeur que v'	$3 = 3$, $\neg(3 = 9)$	$=_\tau$
$v{:}d$	v appartient au domaine d	$3 : [0\ 9]$, $\neg(3 : [6\ 9])$	$\in_\tau$
$d{<:}d'$	d est un sous-domaine de d'	$[6\ 9] <: [0\ 9]$	$\leq_\tau$
$d \wedge d'$	le plus grand domaine commun à d et d'	$[0\ 9] \wedge [6\ 12] = [6\ 9]$	
$\{v\} = \bigwedge_i d_i$	v est le seul élément du plus grand domaine commun aux d_i	$\{6\} = [0\ 6] \wedge [6\ 9]$	

Tableau 8 : Différentes assertions fournies par le système de type. Soit elles correspondent directement à un prédicat communiqué par le système de type, soit elles en sont dérivées.

La du langage est donnée ci-dessous. Comme attendu, le langage contient des objets et des classes. Il existe une notion de spécialisation entre classes et d'attachement d'un

objet à une classe. Les objets ont des attributs et ces attributs sont typés dans les classes et prennent une valeur dans les objets. Les valeurs d'attributs peuvent être soit des objets, soit des valeurs d'un domaine extérieur (voir §I) comme des entiers. Ces valeurs et leurs domaines sont manipulés à l'aide de prédicats fournis par le système de type. Ils permettent de tester les assertions du Tableau 8. Ces assertions ne font pas partie des formules du langage (elles seront donc utilisées uniquement en tant que test dans les règles d'inférence).

DÉFINITION (Grammaire).

```
<KB>          ::= {<assertion>*}
<assertion>   ::= <spec>
                | <attachment>
                | <class_slot>
                | <inst_slot>
<spec>        ::= <class_name> ≤ <class_name>
<attachment>  ::= <inst_name> ∈ <class_name>
<class_slot>  ::= <class_name>.<slot_name>⊆<domain>
                | <class_name>.<slot_name>⊆<class_name>
<inst_slot>   ::= <inst_name>.<slot_name> = <value>
                | <inst_name>.<slot_name> = <inst_name>
<class_name>  ::= <identifier>
<inst_name>   ::= <identifier>
<slot_name>   ::= <identifier>
```

<identifier> sont des chaînes de caractères. <value> et <domain> sont respectivement des entiers et des intervalles d'entiers ci-dessous. ◊

Il existe de plus des contraintes méta-syntaxiques :

— une base de connaissance est un ensemble fini d'assertions,

— le graphe de la relation de spécialisation (≤) est sans circuit, et

— le graphe de la relation de restriction de domaine (⊆) composé avec ≤ est sans circuit.

Il faut noter que les classes et les objets peuvent être introduits plus d'une fois : la multi-généralisation (plusieurs super-classes pour une classe) et la multi-instanciation (des objets attachés à plusieurs classes) sont autorisées. La Figure 13 présente graphiquement un fragment de base de connaissance exprimé dans ce langage.

La sémantique du langage est donnée en référence à un domaine d'interprétation. Les objets sont interprétés comme des éléments de ce domaine alors que les classes en sont des sous-ensembles. La relation de spécialisation est interprétée comme l'inclusion ensembliste et l'attachement comme l'appartenance ensembliste.

Cette sémantique n'est pas celle des logiques terminologiques [Nebel 1990] parce que le langage présenté est purement descriptif (voir §II.1.2) et que les formules ne sont pas uniformément réductibles à la subsomption (c'est-à-dire que les formules — exprimables ou déductibles — peuvent aussi être des assertions d'appartenance ou de restriction de domaine d'attribut). Elle n'est pas non plus celle des "frame logics" [Kifer& 1995] parce que la sémantique de la spécialisation est plus stricte et plus précise (ce n'est pas une relation binaire

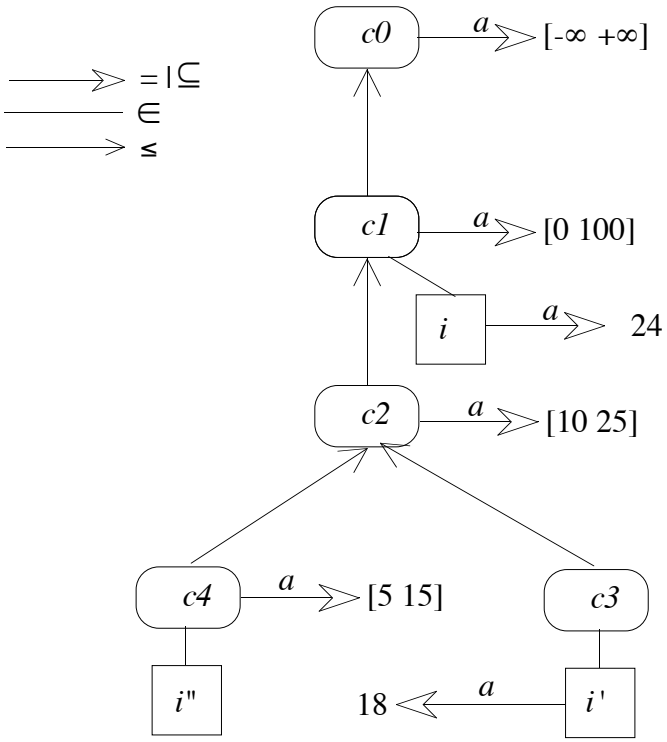


Figure 13 : $B=\{c_1 \leq c_0 ; c_2 \leq c_1 ; c_3 \leq c_2 ; c_4 \leq c_2 ; i \in c_1 ; i' \in c_3 ; i'' \in c_4 ; c_0.a \subseteq [-\infty +\infty] ; c_1.a \subseteq [0\ 100] ; c_2.a \subseteq [10\ 25] ; c_4.a \subseteq [5\ 15] ; i.a=24 ; i'.a=18\}$.

arbitraire mais l'inclusion ensembliste). Le système présenté est plus proche de la sémantique des algèbres de traits ou des Φ-termes [Aït-Kaci& 1993].

DÉFINITION (Interprétation). Soit B une base de connaissance (avec O, C, A des ensembles de noms d'objets, de classes et d'attributs respectivement et τ l'ensemble de toutes les valeurs possibles) et D un domaine. Une interprétation est un couple $\langle D,I \rangle$, dans lequel I est appelé fonction d'interprétation :

I:	$O \rightarrow D$	injective
	$C \rightarrow 2^D$	
	$A \rightarrow (D \rightarrow \tau \cup D)$	$I(a)$ est totale ◊

L'injectivité de I sur O garantit l'hypothèse de nom unique pour les objets.

DÉFINITION (Satisfaction). Une interprétation $\langle D,I \rangle$ satisfait une assertion δ ($\models_{\langle D,I \rangle} \delta$) si

$\models_{\langle D,I \rangle} c \leq c'$	ssi	$I(c) \subseteq I(c')$
$\models_{\langle D,I \rangle} c.a \subseteq d$	ssi	$I(a\|c) \subseteq I(d')$
$\models_{\langle D,I \rangle} o \in c$	ssi	$I(o) \in I(c)$
$\models_{\langle D,I \rangle} o.a = v$	ssi	$I(a)(I(o)) = I(v)$
$\models_{\langle D,I \rangle} c.a \subseteq c'$	ssi	$I(a\|c) \subseteq I(c')$
$\models_{\langle D,I \rangle} o.a = o'$	ssi	$I(a)(I(o)) = I(o')$

$I(a|c)$ est le domaine d'interprétation de l'attribut a lorsque le co-domaine est restreint à c. ◊

Comme d'habitude, un *modèle* d'une base de connaissance est une interprétation qui en satisfait toutes les assertions, une assertion est une *conséquence* d'une base de connaissance si elle est satisfaite dans tous les modèles de cette base et une base de connaissance est *consistante* si elle possède au moins un modèle (et *inconsistante* si elle n'en possède pas).

Il est utile de décrire un système déductif pour ce langage car les inconsistances sont parfois dues aux inférences faites dans la base et parce que les règles d'inférence sont un guide naturel pour remonter aux sources des inconsistances par un mécanisme d'abduction.

Une procédure de déduction est donnée ci-dessous sous la forme d'un ensemble de règles de déductions spécifiques des représentations par objets. Certaines règles (≤-réflexivité) sont techniques (ne servant qu'à obtenir la complétude), d'autres sont bien connues (≤-transitivité, héritage). La ∈-inférence est une règle originale (suggérée dans [Ducournau 1996]) présente

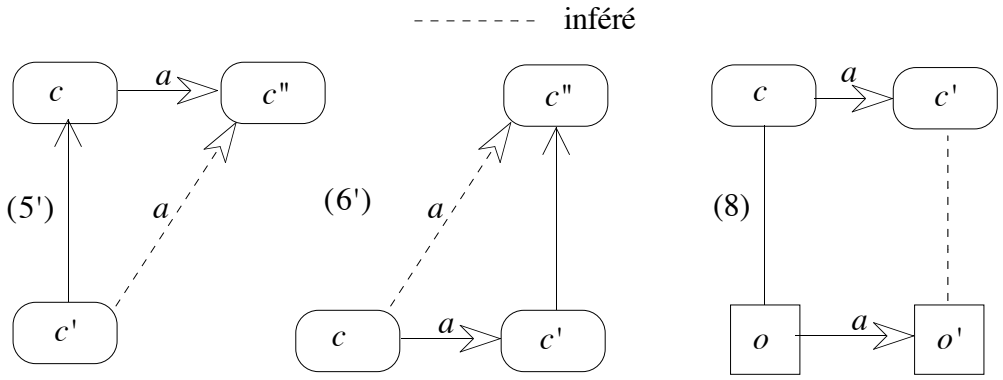


Figure 14 : Exemples des règles (5'), (6') et (8).

dans très peu de systèmes de représentation par objets mais pourtant requise par la sémantique.

DÉFINITION (Règles d'inférence). $\forall\, c, c', c'', c_i \in C, o \in O, a \in A, v \in V$ et $d, d', d_i \in T$:

(1) (≤-réflexivité) $\dfrac{}{c \leq c}$

(2) (≤-transitivité) $\dfrac{c \leq c' \quad c' \leq c''}{c \leq c''}$

(3) (∈-fermeture) $\dfrac{c \leq c' \quad o \in c}{o \in c'}$

(4) (d-intersection) $\dfrac{c.a \subseteq d \quad c.a \subseteq d'}{c.a \subseteq d \wedge d'}$

(5) (d-héritage) $\dfrac{c.a \subseteq d \quad c' \leq c}{c'.a \subseteq d}$

(5') (c-héritage) $\dfrac{c.a \subseteq c'' \quad c' \leq c}{c'.a \subseteq c''}$

(6) (d-fermeture) $\dfrac{c.a \subseteq d}{c.a \subseteq d'}\ d <: d'$

(6') (c-fermeture) $\dfrac{c.a \subseteq c' \quad c' \leq c''}{c.a \subseteq c''}$

(7) (o-valuation) $\dfrac{\{c_i.a \subseteq d_i \quad o \in c_i\}}{o.a = v}\ \{v\} = \bigwedge_i d_i$

(8) (∈-inférence) $\dfrac{c.a \subseteq c' \quad o \in c \quad o.a = o'}{o' \in c'}$

La *clôture déductive* (Cn(*B*)) d'une base de connaissance *B* par les règles est définie comme l'ensemble des assertions obtenues par l'application systématique des règles sur les assertions de *B* et la déductibilité par l'appartenance d'une assertion à la clôture.

DÉFINITION (Déduction). Une assertion δ est *déductible* d'une base de connaissance *B* ($B \vdash \delta$), ssi $\delta \in \mathrm{Cn}(B)$. ◊

Comme escompté, la déduction est correcte et complète par rapport à la sémantique.

PROPOSITION (Correction). Soit B une base de connaissance et δ une assertion, si $B \vdash \delta$ alors $B \models \delta$. ◊

PROPOSITION (Complétude partielle). Soit B une base de connaissance *consistante* et δ une assertion,

si $B \models \delta$ alors $B \vdash \delta$. ◊

Il est notable que la complétude est seulement partielle parce qu'elle requiert la consistance de la base de connaissance. Comme toujours, lorsqu'une base de connaissance est inconsistante (c'est-à-dire qu'elle n'a pas de modèle), toute assertion est satisfaite dans tous les modèles (puisqu'il n'y en a pas). Dans le langage présenté ici, la complétude pleine et entière pourrait être obtenue (avec l'aide des résultats de la section suivante) en ajoutant des règles d'inférences spécifiant que lorsqu'une inconsistance est détectée, toute assertion peut être déduite. L'inconsistance serait alors absolue, c'est-à-dire qu'en cas d'inconsistance, toutes les formules du langage seront des théorèmes.

Cependant, en rester à la consistance partielle permet de conserver une inconsistance locale (qui ne se propage pas à l'ensemble des assertions mais reste liée à celles qui l'ont provoquée). Ceci permet à la révision de considérer rapidement les causes de l'inconsistance.

IV.2.2. Forme normale et manipulation syntaxique

Bien qu'une présentation théorique des représentations de connaissance par objets ait été fournie, les implémentations sont généralement très différentes. Elles pratiquent en général un processus de normalisation permettant de compresser la base et de récupérer efficacement l'information.

Une forme normale pour les bases de connaissance par objets est présentée ici parce qu'elle permet de définir des traitements syntaxiques utilisés par la révision. Une caractérisation constructive de cette forme normale est donc donnée avant de caractériser syntaxiquement l'inconsistance.

La forme normale est d'abord caractérisée axiomatiquement par trois propriétés. C'est une base de connaissance équivalente à la base de connaissance initiale (1 et 2) dans laquelle les suppressions sont effectives (3) c'est-à-dire que si une assertion est supprimée de la forme normale, elle ne sera plus déductible. Cette propriété est précieuse lors de la révision car elle évite les suppressions non minimales.

DÉFINITION (Forme normale). La *forme normale* d'une base de connaissance B, notée NF(B), est telle que :

(1) $B \vdash \text{NF}(B)$,
(2) $\text{NF}(B) \vdash B$,
(3) pour tout δ de NF(B), $\text{NF}(B)-\{\delta\} \nvdash \delta$. ◊

La proposition suivante affirme que la forme normale existe, est unique et peut être obtenue par une simple procédure. Elle consiste simplement à partir de la clôture de la base de connaissance et à appliquer les règles de déduction dans le sens inverse. Bien entendu, l'implémentation ne part pas de la clôture déductive mais mélange inférence et calcul de la forme normale.

PROPOSITION (Caractérisation de la forme normale). Soit B une base de connaissance, la forme normale NF(B) de B est définie par :

NF(B) = Cn'(B)
- $\{(c \leq c'') ; (c \leq c'), (c' \leq c'') \in Cn'(B)\}$
- $\{(o \in c) ; (o \in c'), (c' \leq c) \in Cn'(B)\}$
- $\{(c.a \subseteq d) ; (c.a \subseteq d') \in Cn'(B) \text{ et } d' <: d\}$
- $\{(c.a \subseteq c') ; (c.a \subseteq c''), (c'' \leq c') \in Cn'(B)\}$
- $\{(o.a = v) ; \{c_i.a \subseteq d_i), o \in c_i\} \in Cn'(B) \text{ et } \{v\} = \wedge_i a_i\}$
- $\{(c.a \subseteq d) ; \{c_i.a \subseteq d_i, c \leq c_i\} \in Cn'(B), \wedge_i a_i = a\}$
- $\{(c.a \subseteq c') ; (c''.a \subseteq c'), (c \leq c'') \in Cn'(B)\}$
- $\{(o \in c) ; (o' \in c'), (c'.a \subseteq c), (o'.a = o) \in Cn'(B)\}$

où $Cn'(B) = Cn(B) - \{(c \leq c) \in Cn(B)\}$ ◊

Parmi les avantages de la forme normale figure l'opportunité d'isoler syntaxiquement la source d'une inconsistance. Ceci, combiné avec la propriété de complétude partielle ci-dessus, permet de localiser rapidement l'inconsistance.

PROPOSITION (Inconsistance syntaxique). Une base de connaissance B est inconsistante si et seulement si l'une des assertions suivantes est vérifiée :

(1) $(o.a = x) \in B, (o.a = x') \in B$ avec $x \neq x'$.

(2) $B \vdash o \in c, B \vdash c.a \subseteq d, (o.a = v) \in B$ avec $\neg(v : d)$

(3) Il existe I tel que pour chaque $i \in I, B \vdash o \in c_i, (c_i.a_1.a_2 \ldots a_n \subseteq * d_i) \in B$ et $\wedge_i d_i = \emptyset$

où $c.a_1.a_2 \ldots a_n \subseteq * d$ signifie « il existe $c_1, \ldots c_n, c'_1, \ldots c'_n$ tel que pour tout $1 \leq j \leq n$ $(c_j \leq c'_j) \in B$ et $(c'_j.a_j \subseteq c'_{j+1}) \in B$ (avec $c = c_1$ et $(c'_n.a_n \subseteq d) \in B$) ». ◊

PROPOSITION (Complexité). La complexité du calcul de la forme normale et de la détection de la consistance est polynomiale. ◊

IV.2.3. Révision

La définition de la révision dans le langage de représentation par objets est donnée ci-dessous avant de caractériser syntaxiquement et sémantiquement la minimalité des révisions. Qui plus est, cette révision minimale a une interprétation naturelle dans le contexte des représentations par objets. Ceci permettra de les présenter aisément aux utilisateurs.

Les notions classiques de base contractée (un sous-ensemble consistant d'une base de connaissance dans laquelle une assertion est inconsistante) et de base révisée (une base contractée dans laquelle l'assertion problématique est ajoutée) sont introduites.

DÉFINITION (Contraction d'une base de connaissance). Soit B une base de connaissance consistante et δ une assertion telle que $B \cup \{\delta\}$ soit inconsistante. B' est une *contraction* de (B,δ) ssi $Cn(B') \subset Cn(B)$, (c'est-à-dire $B' \not\models B$ et $B \models B'$) et $B' \cup \{\delta\}$ est consistante. ◊

DÉFINITION (Révision d'une base de connaissance). Soit B une base de connaissance consistante et δ une assertion telle que $B \cup \{\delta\}$ soit inconsistante. B' est une *révision* de (B,δ) ssi $B' = B'' \cup \{\delta\}$ avec B'' une contraction de (B,δ) ◊

Parmi les diverses révisions possibles d'une base de connaissance, un ordre partiel peut être défini qui sélectionne les révisions les plus proches de la base de connaissance initiale. Cet ordre est fondé sur la conséquence (et donc sur la déductibilité).

DÉFINITION (Ordre entre les contractions). Soient B' et B'' des contractions de (B,δ), $B' \propto B''$ ssi n'importe laquelle de ces deux propositions équivalentes est satisfaite :

(1) $B' \models B''$
(2) $Cn(B'') \subseteq Cn(B')$ ◊

La notion de minimalité est ainsi strictement fondée sur la conservation du maximum de déductions ou la limitation des modifications dans les anciens modèles.

La révision est bien un mécanisme qui permet de comparer diverses représentations.

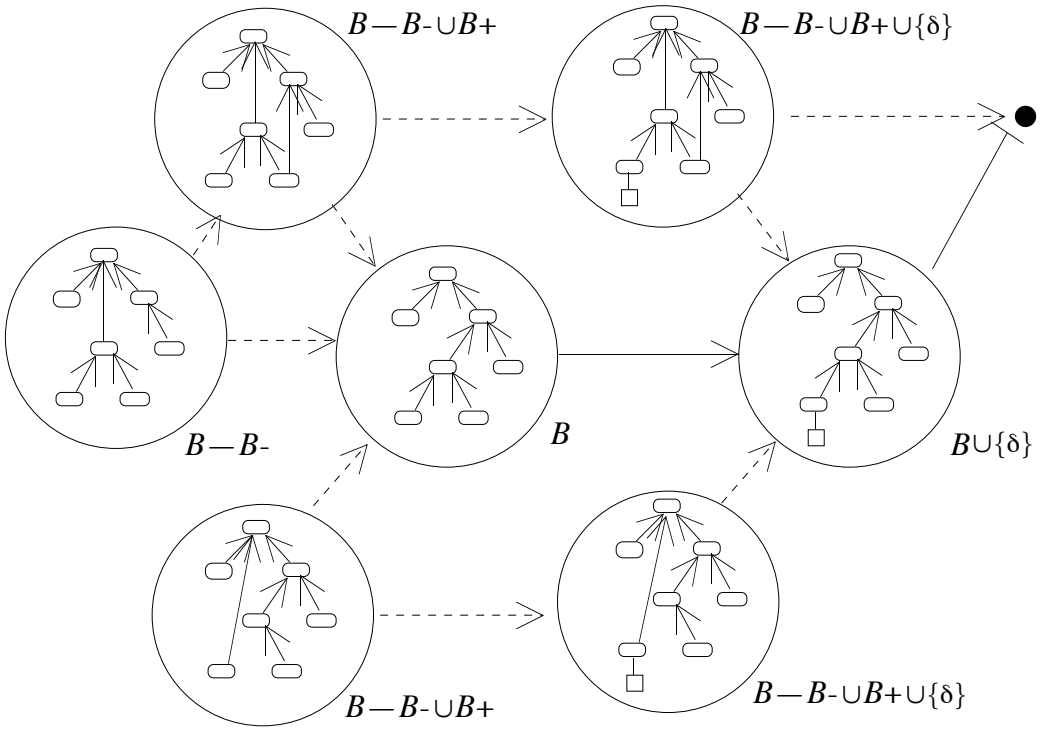


Schéma 18 : Une partie du schéma mis en œuvre par la révision. Celui-ci présuppose que δ est un élément de l'approximation du domaine et que, par conséquent, B (qui est inconsistant avec δ) ne peut être une approximation de celui-ci (car ce dernier est forcément consistant). Le but de la révision est donc de trouver une sous-base de B $(NF(B) - B'_{-}) \cup B'_{+}$ à laquelle δ peut être ajouté. Pour cela, les contractions peuvent être vues comme les approximations de B compatibles avec δ et les contractions minimales comme celles qui n'approximent d'autres contractions qu'elles-mêmes.

Cependant elle prend en compte une représentation initiale qui n'est pas une approximation du domaine.

DÉFINITION (Contraction minimale). B' est une *contraction minimale* de (B,δ) ssi B' est une contraction de (B,δ) et il n'y a aucune $B''\neq B'$ telle que B'' est une contraction de (B,δ) et $B''\propto B'$. ◊

Afin de manipuler les contractions, il est préférable qu'elles soient des ensembles finis d'assertions. La forme normale sera donc utilisée. La forme normale d'une contraction B' de (B,δ) peut toujours être écrite en fonction de celle de B comme $B'=\mathrm{NF}(B')=(\mathrm{NF}(B)-B'_-)\cup B'_+$ avec les contraintes suivantes :

$$B'_- \cap B'_+ = \varnothing,$$
$$B'_- \subseteq \mathrm{NF}(B) \text{ et}$$
$$B'_+ \subseteq \mathrm{Cn}(B)$$

Cette formulation est unique et correspond à la forme normale de B à laquelle un ensemble d'assertions B'_- est ôté et un autre B'_+ déductible de B est ajouté. Sous la vision atomique de la base de connaissance, cette minimalité peut être caractérisée sémantiquement et syntaxiquement. Les deux caractérisations sont équivalentes.

PROPOSITION (Caractérisation sémantique). B' est une contraction minimale de (B,δ) ssi $\forall\gamma$ telle que $B\models\gamma$ et $B'\cup\{\delta\}\not\models\gamma$ alors $B'\cup\{\delta\}\cup\{\gamma\}$ est inconsistant. ◊

PROPOSITION (Caractérisation syntaxique). Soient B' et B'' deux contractions de (B,δ), B' est plus petite que B'' ($B'\propto B''$) ssi $B'_- \subseteq B''_-$ et $B' \vdash B''_+$. ◊

L'opérateur ainsi défini satisfait les postulats AGM [Alchourrón& 1985] de base (les six premiers postulats classiques des opérateurs de révision) et un ensemble maximal des contraintes de [Hansson 1996]. Il peut être caractérisé comme un opérateur maxi-choix (qui choisit une unique révision parmi celles possibles) [Alchourrón& 1985] où le choix est ici fait par l'utilisateur.

Une force de l'opération de révision résultante est de pouvoir être interprétée en termes d'objets à la place de manipulation de formules. Dans le cas du langage décrit, la terminologie objet peut être substituée aux formules logiques. En fait, une révision d'une base de connaissance par objets est un ensemble de modifications de type suivant :

- attacher un objet à une classe moins spécialisée dans la hiérarchie de classes ;
- supprimer la valeur d'un attribut d'objet ;
- élargir une restriction d'un attribut de classe (en élargissant le domaine ou en utilisant une classe moins spécialisée comme domaine) ;
- monter une classe plus haut dans la hiérarchie de classes.

La minimalité elle-même peut être traduite en termes d'objets.

PROPOSITION (Interprétation objet de la minimalité). B' est une contraction minimale de (B,δ) si et seulement si :

(1)B' est une contraction de (B,δ).
(2)$\forall\ B''$ contraction de (B,δ), si $B''_- \subseteq B'_-$ et $B'_+ \subseteq B''_+$, alors $B'=B''$.
(3)$\forall (c \leq c') \in B'_+$, $\forall k \neq c'$, si $B' \vdash k \leq c'$ et $(c \leq k) \in \mathrm{Cn}(B)$, alors $(c \leq k)$ ne peut être ajouté dans $B' \cup \{\delta\}$.
(4)$\forall (o \in c) \in B'_+$, $\forall c' \neq c$, si $B' \vdash c' \leq c$ et $(o \in c') \in \mathrm{Cn}(B)$, alors $(o \in c')$ ne peut être ajouté dans $B' \cup \{\delta\}$.
(5)$\forall\ (c.a \subseteq d) \in B'_+$, $\forall (c',d') \neq (c,d)$, si $d' \leq : d$, $B' \vdash c \leq c'$ et $(c'.a \subseteq d') \in \mathrm{Cn}(B)$, alors $(c'.a \subseteq d')$ ne peut être ajouté dans $B' \cup \{\delta\}$.
(6)$\forall\ (c.a \subseteq k) \in B'_+$, $\forall (c',k') \neq (c,d)$, $k,k' \in C$, si $B' \vdash k' \leq k$, $B' \vdash c \leq c'$ et $(c'.a \subseteq k') \in \mathrm{Cn}(B)$, alors $(c'.a \subseteq k')$ ne peut être ajouté dans $B' \cup \{\delta\}$.
(7)$\forall (o \in c) \in B'_+$, $\forall c' \neq c$, o', a si $(c'.a \subseteq c) \in \mathrm{Cn}(B)$, $(o' \in c') \in \mathrm{Cn}(B)$ et $(o'.a=o) \in \mathrm{Cn}(B)$, alors une des trois assertions ne peut être ajoutée dans $B' \cup \{\delta\}$.
(8)$\forall (o.a=v) \in B'_+$, pour tout ensemble minimal d'assertion $\{c_i.a \subseteq d_{ij},\ o \in c_i\} \in \mathrm{Cn}(B)$ tel que $\{v\} = \wedge d_{ij}$, alors une des assertions ne peut être ajoutée dans $B' \cup \{\delta\}$. ◊

L'« interprétation objet de la minimalité » dit que la notion de minimalité peut être interprétée sur les hiérarchies de spécialisation. Son interprétation est que si une assertion (3 : relation de spécialisation, appartenance à une cl : restriction de domaine d'attribut, 8 : valuation d'attribut) est ajoutée dans une base de connaissance minimale, c'est parce qu'une autre assertion permettant de la déduire (3 : relation de spécialisation, 4 : appartenance à une classe, 5, 6 : restriction de domaine d'attribut, 7 : valuation d'attribut,

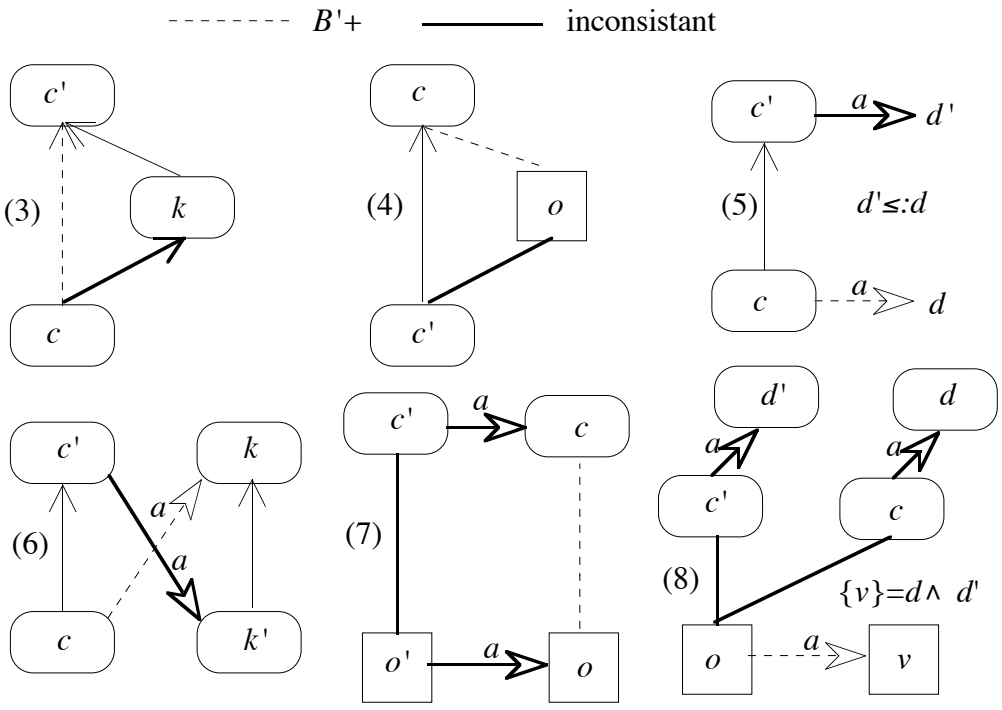


Figure 15 : Illustration de différents cas de la proposition (interprétation objet de la minimalité).

appartenance à une classe ou restriction de domaine d'attribut, 8 : appartenance à une classe ou restriction de domaine d'attribut) ne peut rester dans la base sans inconsistance. Cela signifie, par exemple, que si une classe est remontée, elle l'est dans la classe la plus spécialisée possible.

IV.2.4. Implémentation et fonctionnement

Le but de cette opération de révision est d'être implémentée au dessus de TROEPS. Les résultats obtenus jusqu'à présent (parce qu'ils sont toujours caractérisés syntaxiquement sur la forme normale) permettent son implémentation. Le contexte de fonctionnement de cette opération est décrit ci-dessous avec quelques détails de l'implémentation non déductibles de la théorie. Son fonctionnement est illustré par un exemple.

L'opération de révision est utilisée lorsqu'il y a interaction avec le système. Comme il peut exister de nombreuses révisions minimales, la tâche de choisir celle à appliquer est laissée à l'utilisateur (bien qu'il puisse y avoir certaines stratégies générales telles que de ne pas modifier le schéma conceptuel ou — au contraire — de ne pas modifier les données). Afin d'être synthétique, les choix sont présentés à l'utilisateur de manière ordonnée (suivant le processus d'abduction suivi par l'opérateur). Ainsi la procédure de révision est la suivante :

1) l'utilisateur tente de modifier la base ;
2) le système détecte une inconsistance, la présente à l'utilisateur et propose certaines assertions à supprimer (y compris la modification initiale) ;
3) l'utilisateur sélectionne l'assertion à supprimer, ceci peut être suffisant ou requérir d'autres itérations pour choisir parmi les différents moyens de retirer cette assertion.

En fonction de l'assertion à supprimer de la b ut exister plusieurs assertions qui permettent de la déduire qu'il faut supprimer de la base comme cela est montré par la Figure 16. Le système peut parcourir les différentes possibilités de correction en fonction de ce graphe.

Détecter une inconsistance introduite par une nouvelle assertion est très efficace (polynomial). Cependant, la recherche de toutes les révisions (même minimales) reste impraticable ("intractable") dans ce langage. Tirer avantage de la présence de l'utilisateur pour choisir parmi les possibilités à explorer permet d'obtenir n'importe quelle révision en temps polynomial (parce qu'une seule branche est explorée et que le test de consistance est efficace). Cependant, si l'utilisateur décide d'explorer toutes les branches, ceci peut être très long.

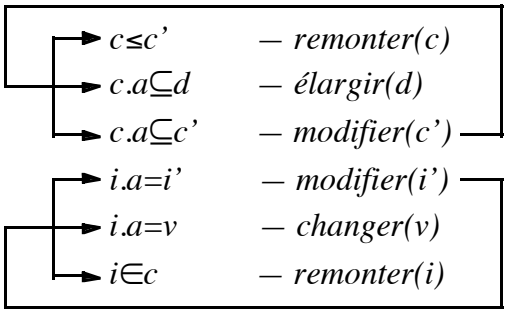


Figure 16 : Schéma général du parcours de l'espace de révision en fonction des assertions entrant en conflit. En fonction de l'assertion à remettre en cause (et de son explication en langage objet), l'action à accomplir peut être simple (changer) ou requérir la remise en cause d'autres assertions. L'acyclicité de TROEPS permet à ce processus de terminer.

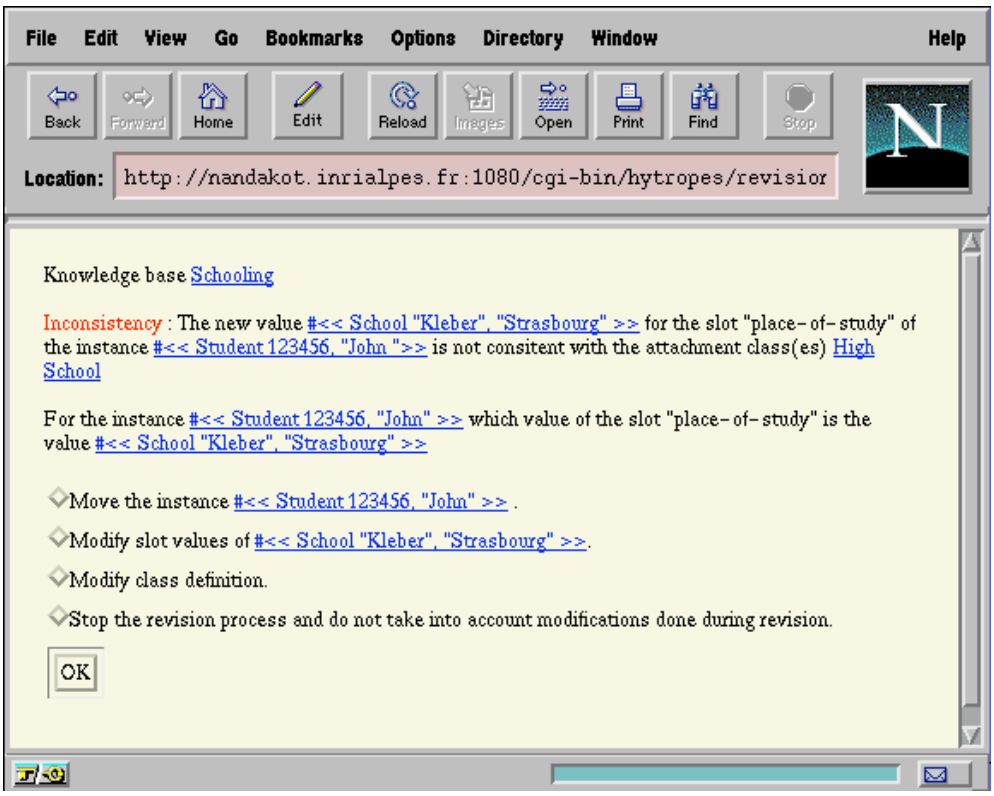


Figure 17 : Copie d'un écran d'exécution de la révision. À partir d'un diagnostic d'inconsistance, le système propose un ensemble de voies possibles pour remédier au problème correspondant aux actions de la Figure 16.

Le système décrit est implémenté au dessus de TROEPS. Il rattrape les erreurs déclenchées par TROEPS et, si l'erreur est révisable (il ne s'agit pas, par exemple, d'une erreur typographique), le système de révision gère le problème. Dans ce cas, les assertions impliquées dans l'inconsistance sont connues précisément. Pour chaque type d'inconsistance, les stratégies possibles de révision sont compilées (par exemple bouger une classe, bouger un objet). Lorsqu'une inconsistance particulière est détectée, la stratégie adaptée est appliquée aux objets concernés. Les choix faits par l'utilisateur sont ordonnés de telle sorte que les premières modifications faites ne puissent être remises en cause par les suivantes (par exemple, les assertions sur les hiérarchies de classes sont mises en cause avant celles sur les objets). Ainsi, la minimalité de la solution globale est généralement assurée par la minimalité de chaque étape. Il y a cependant quelques exceptions où un choix a un impact sur une modification précédente et une vérification a posteriori est alors requise (spécialement pour vérifier qu'une assertion sur un objet n'a pas été supprimée inutilement). L'interaction avec l'utilisateur est gérée au travers d'HTTP comme pour le reste du système. À chaque moment, l'utilisateur peut décider d'abandonner la piste courante et de revenir sur un choix précédent. Ceci est implémenté en empilant les alternatives (et peut prendre un espace polynomial).

Par exemple, si la nouvelle valeur donnée pour un attribut d'un objet est inconsistante, le programme présente à l'utilisateur les différentes solutions présentes dans la Figure 17. La figure affiche l'inconsistance courante et les révisions minimales possibles. Ceci est décrit sous forme objet.

Le premier choix fera apparaître la hiérarchie de classes et la classe dans laquelle l'objet peut être déplacé pour résoudre l'inconsistance. Le second choix proposera de modifier l'objet qui est la valeur de l'attribut modifié. Le troisième choix va guider l'utilisateur dans la modification des restriction d'attributs entrant en conflit avec la nouvelle valeur : cela conduira à élargir le domaine restreignant l'attribut ou à remplacer la classe restreignante par une classe plus générale. Le quatrième choix consiste à rejeter la valeur, c'est-à-dire à abandonner la modification. Ainsi, l'utilisateur choisit à chaque étape ses modifications préférées — parmi celles possibles — et le programme garantit que la révision finale est une modification minimale de la base initiale.

IV.2.5. Argumentation

La complexité de la révision persiste mais elle est transférée à l'utilisateur. La raison de cela est que la base de connaissance ne contient pas la connaissance nécessaire pour choisir parmi les révisions possibles [Friedman& 1996]. Les quelques tests réalisés ont montré que ce compromis semble satisfaisant. Cependant les utilisateurs peuvent toujours être dépassés par les choix à faire. C'est pourquoi l'aspect argumentatif est considéré.

Si la révision permet théoriquement de trouver toutes les modifications possibles permettant à un utilisateur d'ajouter de la connaissance à une base, elle ne résout pas l'un des problèmes posés initialement : la désorientation de l'utilisateur qui ne connaît pas profondément toute l'étendue de la base. À cet égard, un dispositif permettant à cet utilisateur de faire collaborer les spécialistes des différents domaines (et, plus pratiquement, les auteurs de la connaissance mise en cause) au choix de la révision à réaliser serait très utile.

Un tel dispositif s'inscrit dans le domaine de recherche sur les collecticiels (CSCW) et plus particulièrement dans le domaine de l'argumentation assistée par ordinateur (CSCA) [Buckingham Shum 1997]. La problématique consiste à étudier dans quelle mesure il est possible de construire un système d'argumentation dont les différents chemins soient balisés par le contenu de l'application (ici la base de connaissance).

À cet effet, cette année, un système d'argumentation classique (IBIS [Conklin& 1988]) a été développé et intégré au système de révision [Lecourtier 1998]. Les types de nœuds sont ceux autorisés par le système IBIS (problème, solution, argument) à ceci près que les problèmes et les solutions sont proposés par la révision. Ce système permet à l'utilisateur gêné face à un choix qu'il ne sait faire de demander aux différents auteurs des éléments de connaissance à remettre en cause leurs arguments en faveur et en défaveur de la solution proposée par le système de révision. Ces auteurs peuvent, si cela leur est nécessaire, développer les alternatives proposées par la révision afin d'affiner leur jugement (par exemple, ils peuvent dire : d'accord pour élargir le domaine de cette classe mais pas celui de sa sous-classe). L'idée principale est donc :

— que les utilisateurs puissent choisir la révision à opérer en connaissance des arguments pour et contre exprimés par les spécialistes du domaine concernés ;
— que le processus d'argumentation soit uniquement borné par les choix possibles lors de la révision et non par n'importe quelle discussion impliquant la connaissance problématique.

Le système de révision, en dehors de sa valeur intrinsèque, a une grande importance au sein de Co4. En effet, il permet aux utilisateurs, avant de soumettre une partie de leur base de connaissance à la base consensuelle, de vérifier de la compatibilité entre les deux bases. Si elles sont incompatibles, il peut alors faire appel au système de révision pour décider quelles modifications de la base de groupe soumettre afin d'accommoder sa contribution. S'il n'est pas sûr de son choix de révision, il peut faire appel à l'argumentation.

Une fois ce problème de consistance réglé, il est temps de soumettre sa contribution à la base de groupe et de discuter avec les autres souscripteurs de leur perception de la contribution. Pour cela, le protocole de Co4 va organiser la procédure de vote destinée à s'assurer de l'aspect consensuel de la base de groupe. Il est maintenant présenté.

IV.3. Protocole de collaboration

Le protocole de collaboration obéit presque exclusivement aux principes donnés dans le paragraphe de présentation. Seuls seront donc présentés ci-dessous un exemple de l'utilisation de ce protocole (§IV.3.1) puis une présentation et discussion des propriétés qu'il satisfait (§IV.3.2). Plus de détails — y compris la description complète du protocole et les preuves de propriétés — sont disponibles dans [Euzenat 1997a].

IV.3.1. Exemple

Lorsqu'un souscripteur a suffisamment confiance dans une partie de la connaissance exprimée dans sa base individuelle, il peut la soumettre à la base de groupe à laquelle il souscrit. Ceci est réalisé en circonscrivant la partie à soumettre (en TROEPS une entité particulière et tout ce qu'elle contient) et en appelant la procédure de soumission (sur la partie gauche de la Figure 18, en cliquant sur le bouton « Soumettre »). Afin de compléter la requête, le système fait la différence entre ce qui est à soumettre et ce qui est déjà dans la base de groupe et prend en compte les modifications de la base de groupe que l'utilisateur n'a pas voulu intégrer dans sa base. Puis il transmet la requête à la base de groupe. Si la connaissance soumise est consistante avec le contenu de la base de groupe, la requête est rendue anonyme puis transmise aux autres souscripteurs sous la forme d'un appel à commentaires afin qu'ils évaluent la contribution.

Cet appel à commentaires parvient à l'ensemble des souscripteurs (dans la partie droite de la Figure 18, il apparaît sous l'en-tête "Proposals"). Les utilisateurs peuvent lire ces soumissions (en cliquant sur "More") ou les inclure dans leur propre base individuelle. Ceci peut conduire à une détection d'inconsistance ou de redondance que l'utilisateur pourra utiliser dans sa réponse ou dans sa confection d'une contre-proposition. En réponse à cet appel à commentaire, l'utilisateur peut répondre par l'acceptation lorsqu'il estime que la proposition doit être intégrée à la base de groupe, le rejet s'il estime qu'elle ne doit pas l'être ou la contre-proposition lorsqu'il propose des modifications (boutons correspondants sur la partie droite de la Figure 18).

Lorsque la base de groupe dispose de suffisamment de commentaires, elle intègre ou non la modification à son contenu. Elle modifie l'appel à commentaires (en l'annulant si la proposition est rejetée ou en le remplaçant par un plus spécifique s'il s'agit d'une contre-proposition). Si la proposition a été acceptée, elle en informe tous ses souscripteurs.

Il peut cependant s'avérer que les recherches menées par l'un des souscripteurs entrent en conflit avec ce qu'il vient d'accepter. Aussi peut-il refuser d'intégrer le nouveau contenu dans sa base individuelle. Ce contenu est alors stocké dans le journal de la base individuelle (bouton

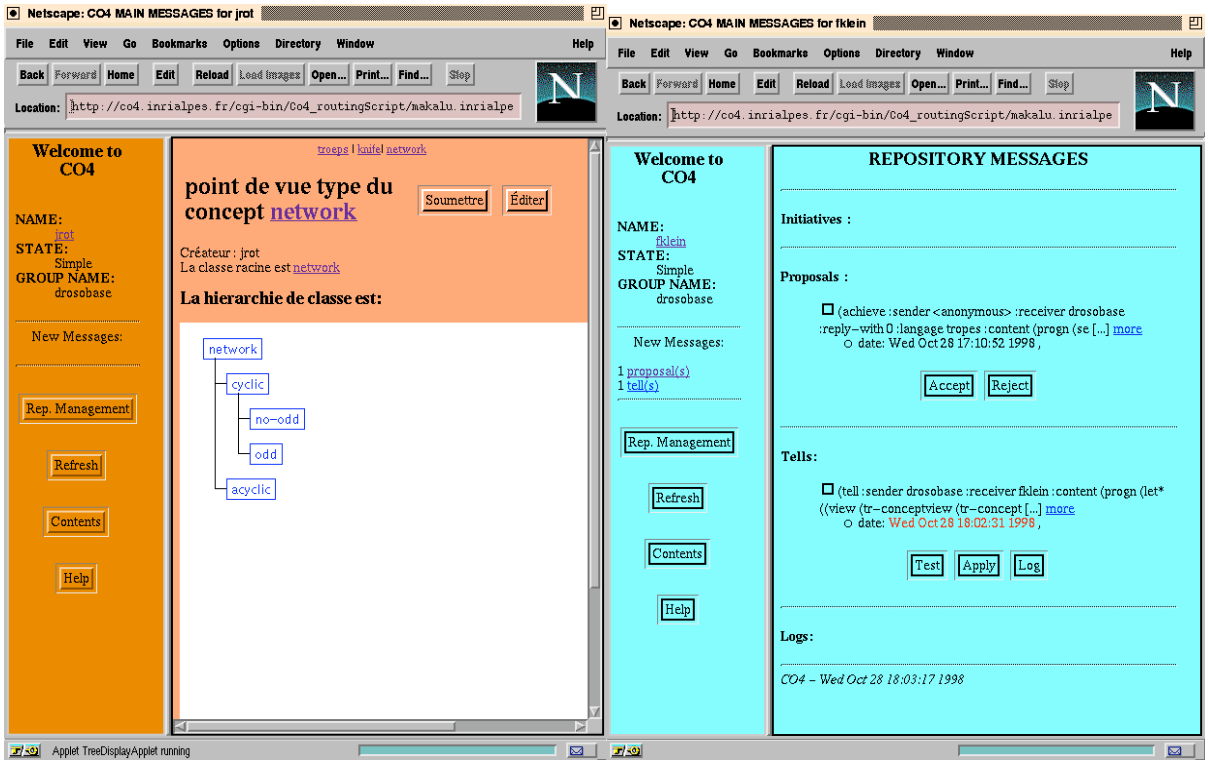


Figure 18 : Interface de la bibliothèque de CO4 connectée à TROEPS. La partie gauche présente le point de vue qu'une utilisatrice soumet à la base de groupe et la partie droite présente l'état des soumissions reçues par un autre souscripteur.

"Log" sur la Figure 18). Le fait que chacun puisse maintenir une base différente de la base de groupe permet l'exploration de voies divergentes. Cela permet aussi à la communication, la négociation et l'acceptation d'être asynchrone. En fait, cela reproduit le fonctionnement des comités de lecture : les évaluateurs peuvent prendre leur temps pour examiner une contribution avec attention parce que cela n'empêche pas les auteurs de continuer à travailler.

IV.3.2. Propriétés

Les propriétés intéressantes pour un protocole tel que celui qui est ébauché ici sont les suivantes :

(0) que le protocole soit complètement spécifié ;

(1) que les bases ne puissent être dans un état inconsistant ;

(2) que n'importe quel utilisateur puisse soumettre sa connaissance n'importe quand ;

(3) que l'acceptation (resp. le rejet) d'une proposition soit subordonnée à l'accord (resp. au non-accord) de tous les (resp. au moins un) souscripteur(s) ;

(4) qu'une décision (accepté/refusé/retiré) pour toute proposition soit prise dans un laps de temps fini ;
(5) que le nombre de messages soit minimum.

La terminaison n'est pas forcément souhaitable puisque le système peut-être utilisé indéfiniment. À la place, (4) propose une propriété de terminaison locale. La propriété la plus simple à obtenir (0) est le fait qu'à chaque message corresponde une réponse.

PROPOSITION (Intelligibilité). Pour toute performative et tout type de récepteur (groupe ou base individuelle) il y a une règle se déclenchant sur ce message (dans toutes les conditions). ◊

D'autres résultats concernant ce protocole dépendent de quatre hypothèses de travail. Elles sont toutes présentées en une seule fois.

HYPOTHÈSES. Le protocole est considéré ici sous les hypothèses suivantes :

(a) il y a toujours, au bout d'un laps de temps fini, une réponse à une requête émise vers une base individuelle,
(b) il n'y a pas un nombre infini de contre-propositions à une proposition,
(c) il n'y a pas, dans un laps de temps fini, un nombre infini de nouvelles souscriptions, et,
(d) différentes propositions examinées simultanément sont indépendantes. ◊

Chaque hypothèse sera évoquée dans le cadre de la propriété où elle est utile. Les hypothèses (b) et (c) sont des hypothèses raisonnables et ne seront pas discutées. L'hypothèse (a) est difficile à garantir dans la réalité mais elle est absolument nécessaire pour que la terminaison soit vérifiée. Ce n'est pas une propriété spécifique au protocole de Co_4 ; elle est partagée par toutes les tentatives de résoudre le "transaction commit problem" dans les systèmes de bases de données distribuées (même pour les votes à la majorité) [Fischer& 1985]. Les temporisations ("time-out") sont les procédés habituellement utilisés pour résoudre ce problème.

L'hypothèse (d) est aussi très difficile à garantir mais nécessaire pour obtenir la consistance. Elle peut être garantie en détectant les dépendances et en retardant une requête tant qu'une décision n'a pas été prise sur toutes les autres requêtes dont elle dépend (affaiblissant ainsi la propriété d'équité). Le même problème se pose dans le cadre législatif où des amendements contradictoires peuvent être déposés sur un même texte. La procédure de vote permet de se prémunir de ce problème [Assemblée nationale 1959]. Ainsi, chaque texte de loi est examiné soit en bloc (et ne peut être amendé), soit article par article auquel cas les articles sont examinés amendements par amendements dans l'ordre d'importance décroissante (d'abord les amendements de suppression de l'article, puis les amendements qui changent de moins en moins le texte initial et dans l'ordre de ce texte). Une disposition spéciale est par ailleurs utilisée pour des amendements contradictoires qui peuvent être examinés de concert mais toujours votés de manière ordonnée. Dés qu'un amendement est accepté sur un article, tous les amendements concurrents (ceux avec lesquels il y a inconsistance) sont considérés comme rejetés.

Les propositions suivantes ont pu ainsi être établies :

PROPOSITION (Consistance). Sous l'hypothèse (d), les bases de groupe ne sont jamais dans un état inconsistant. ◊

PROPOSITION (Vivacité et équité). Les souscripteurs peuvent soumettre des propositions à leur base de groupe à tout moment. ◊

PROPOSITION (Consensus). Toute soumission est acceptée si et seulement si tous les souscripteurs (au moment de son introduction dans la base) l'ont acceptée. ◊

PROPOSITION (Terminaison). Sous les hypothèses (a-d) toute soumission atteint un état d'acceptée ou de rejetée au bout d'un laps de temps fini. ◊

En sus des preuves analytiques des propositions citées plus haut, le protocole a été partiellement (sans les contre-propositions) décrit en LOTOS et vérifié par une approche combinatoire ("model checking") [Pecheur 1997].

Il est notable que certaines propriétés ne se transfèrent pas facilement vers un protocole majoritaire ou intersectif. En effet, si l'intersection et le consensus sont des fonctions monotones décroissantes de l'ensemble des souscripteurs, ce n'est pas le cas de la majorité (ajouter des souscripteurs peut élargir ou réduire l'ensemble des propositions acceptées).

Par rapport au schéma donné en introduction, le protocole de Co4 assure donc un schéma d'intégration du contenu des bases de type intersectif dans lequel chaque contribution fait progresser vers un modèle plus complet. En conclusion, la base consensuelle finira par être plus

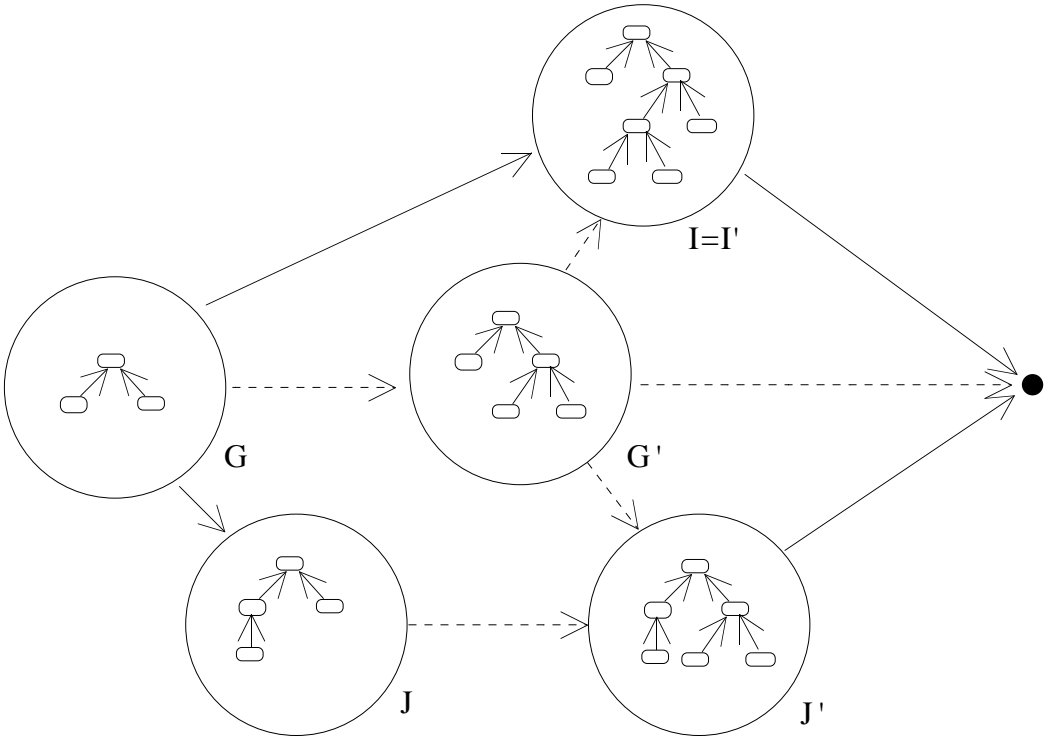


Schéma 19 : L'agrégation de représentations dans Co4 bien que correspondant statiquement au modèle intersectif (voir Schéma 17) permet la progression de la représentation vers des représentations plus complètes. Les trois triplets I-J-G et I'-J'-G' correspondent à un modèle intersectif. Cependant, entre G et G' s'opère un enrichissement de toutes les bases simultanément. Ce fonctionnement est semblable à celui du jeu de rugby où l'on ne peut passer la balle qu'en arrière mais on peut malgré tout parvenir à l'embut.

complète que toutes les bases individuelles initiales (voir Schéma 19). Il permet donc, comme c'est le but de ce travail, un partage de la connaissance qui enrichit l'ensemble des intervenants.

Conclusion

L'activité du scientifique peut être vue comme la construction et la manipulation de modèles de plus en plus proches de l'objet étudié (ou plutôt de sa représentation à un niveau d'abstraction et dans un langage particulier). Mais s'il est réjouissant de voir dans les progrès de ce travail de modélisation une progression monotone enrichissant progressivement le modèle, il n'en est pas forcément ainsi.

Il arrive que la représentation considérée ne soit pas une approximation et qu'il faille la modifier pour la remettre dans ce qui semble être une approximation de ce domaine. La révision, fondée sur l'idée que la représentation initiale n'est tout de même pas si loin d'une bonne approximation, va chercher une représentation possible qui soit le plus proche possible de celle-ci.

Par ailleurs, la multiplicité des représentations possibles fait qu'il n'est pas forcement aisé d'assembler plusieurs représentations pour en obtenir une représentation plus complète mais aussi plus difficile à appréhender. Le protocole de CO4 tente de résoudre ce problème en aidant les intervenants à construire une telle représentation de manière consistante, consensuelle et incrémentale. La théorie telle que présentée dans la dernière partie est que la base consensuelle croît et se rapproche donc du domaine à modéliser. Ce n'est cependant pas le cas car les soumissions peuvent, à l'instar de la révision, contribuer à supprimer des éléments de la base consensuelle.

Ainsi, les travaux présentés dans ce chapitre, et dont le but est de permettre aux modélisateurs de construire des bases de connaissance sur lesquelles ils s'accordent, constituent une fois de plus un travail sur la manipulation de représentation et la manière de passer d'une représentation à une autre.

Encore une fois, les résultats ne sont qu'indicatifs et non impératifs. Cependant, ces derniers travaux sont différents de ceux présentés dans les chapitres précédents principalement parce qu'ils font intervenir de manière centrale l'utilisateur du système et non uniquement un système formel.

Crédits

La construction collaborative de bases de connaissance consensuelles est mon axe de travail principal au sein du projet SHERPA. Ce travail a été initié par François Rechenmann qui a proposé la métaphore de la soumission à une revue scientifique. La concrétisation de cette métaphore, l'élaboration de l'architecture du système et du protocole de collaboration sont ma contribution personnelle.

Le protocole a été décrit en LOTOS et validé par les soins de Charles Pecheur en séjour post-doctoral au sein de l'action VASY de l'INRIA Rhône-Alpes ; il a été développé sous ma direction par Loïc Tricand De La Goutte, ingénieur expert au sein du projet SHERPA.

Les travaux sur la révision sont pour moi le prolongement naturel de ceux que j'ai pu mener dans le cadre de ma thèse sur les systèmes de maintien du raisonnement [Euzenat 1990, 1991b, 1993d]. Si les travaux dans cette dernière voie se sont taris, c'est, en ce qui me concerne, parce que la problématique de la révision permet d'envisager la réaction à une inconsistance sur un

terrain sémantique et non plus algorithmique. Le travail mené sur la révision constitue la thèse d'Isabelle Crampé que j'ai dirigée et celui sur l'argumentation le travail de DEA de Claire Lecourtier que j'ai également dirigée.

CONCLUSION ET PERSPECTIVES

Ce mémoire a présenté certains travaux que j'ai conduits sous le point de vue très particulier des rapports qu'entretiennent de multiples représentations d'un domaine. D'autres travaux relèvent toujours de cette problématique mais n'ont pas été évoqués car plus anciens :

— Le raisonnement hypothétique peut aussi être considéré sous le jour de la représentation multiple d'un même domaine. Ce mémoire n'en fait pas état car mes travaux sur le sujet se sont arrêtés rapidement après ma thèse [Euzenat 1990, Euzenat 1991b]. Ma dernière contribution fut une description précise des algorithmes de systèmes de maintenance du raisonnement à propagation [Euzenat 1993d] n'entrant pas dans ce cadre.

— J'ai développé quelques travaux sur la notion d'oubli dans une représentation de connaissance fondée sur les systèmes de maintien du raisonnement [Euzenat& 1990, Strecker 1991]. L'oubli, en général, permet de passer d'une représentation à une représentation plus « abstraite » qui l'approxime.

Il est clair que chacun des chapitres présentés plus haut traite d'un sujet différent avec des approches différentes. Cela dit, à l'exception du chapitre III, l'ensemble des résultats obtenus sont maintenant intégrés dans le logiciel Co4 qui comprend TROEPS, ses algorithmes de classification, d'inférence de passerelles, d'inférence de taxonomies, un éditeur distant utilisant le protocole HTTP, les mécanismes de révision et bientôt d'argumentation, ainsi que le protocole de collaboration.

Car les travaux présentés, s'ils cherchent à développer une vision raisonnée et à long terme de représentations de connaissance, sont aussi le support d'applications plus immédiates permettant de nourrir la réflexion sur les outils proposés (voir [Euzenat& 1995] à propos de nos précédents outils). Ainsi, les deux terrains d'application de ces outils ont été, de manière privilégiée :

— les bases de connaissance scientifique plus particulièrement adaptées au domaine de la biologie moléculaire. Une réalisation notable utilisant TROEPS est celle de KNIFE [Euzenat& 1997b], une base de connaissance dédiée aux interactions géniques dans l'embryon de drosophile, développée en collaboration avec le LGPD de Marseille.

— les mémoires techniques. Indépendamment des multiples notions que peut recouvrir ce terme dans le cadre plus large de la gestion de connaissance dans les entreprises, il est intéressant d'étudier comment les bases de connaissance peuvent être utilisées afin d'organiser la représentation d'un domaine particulier — en tant que structure de stockage et de consultation mais surtout en tant qu'outil d'analyse exploratoire. Dans ce cadre le projet SHERPA poursuit des travaux avec la société STMicroelectronics autour de la représentation de la connaissance au sein du département informatique.

La présentation donnée ici de mes travaux vise à faire ressortir l'aspect modulaire des systèmes développés, que cette modularité soit interne (les utilisations variées de la structure de système classificatoire) ou externe (l'utilisation de schémas pour présenter les points communs entre les approches développées).

En effet, les formalismes de représentation de connaissance sont de plus en plus spécialisés (les uns dédiés à l'expression des plans d'action, les autres à celle des relations temporelles…) et il est important, s'ils sont développés de manière modulaire, de préserver la possibilité de les intégrer. Il faut donc aller vers un moyen d'assembler les différents formalismes à utiliser. J'espère que la présence des schémas dans ces pages a permis au lecteur de se rendre compte de la façon dont cet assemblage peut se faire (certains assemblages de schémas se déduisent très facilement des schémas élémentaires proposés et n'ont pas été donnés).

Nous disposons aujourd'hui d'un substrat de logiciels, de savoir-faire, de contrats et de contacts. Il est maintenant temps de passer au niveau supérieur afin d'organiser cela dans une vision globale, indépendante de nos formalismes et puissante (c'est-à-dire génératrice de nouveaux résultats).

À cette fin il me semble que le choix de la relation d'approximation est la meilleure base et que la théorie des catégories, évitée tout au long de ce document, est l'instrument idéal pour envisager les relations entre représentations. Il me faut remarquer que cette conviction me vient certainement de la lecture de [Aït-Kaci& 1993]. J'ai amorcé un tel travail dans le cadre de la représentation temporelle [Euzenat 1995d]. Développer proprement dans cette théorie les observations faites dans ce mémoire est bien entendu un travail à très long terme, sans doute peu productif en termes d'applications potentielles et de publications parce que si abstrait qu'il paraîtra simpliste ou inutile (mais de récents travaux se penchent sérieusement sur ce sujet [Goguen 1998]). En tout cas, un mémoire d'habilitation à diriger les recherches est sans doute l'endroit approprié pour en faire état.

De plus, j'ai insisté dans l'exposé sur le caractère indicatif et non impératif des formalismes développés. Ce caractère provient principalement de la prise en compte d'utilisateurs auxquels on laisse choisir la solution cible une fois que les solutions impossibles ont été écartées. Bien entendu, cela est un peu facile car les occasions de faire appel à l'utilisateur sont justement celles où le programme est traditionnellement en difficulté. On peut donc considérer que les problèmes sont évacués. Dans certains cas, ce peut être vrai (l'utilisateur ne sait pas mieux résoudre le problème que la machine), dans d'autres c'est faux pour diverses raisons (l'utilisateur est le seul à posséder la connaissance, l'intuition ou le pouvoir).

L'aspect indicatif ne cadre pas forcément avec le développement d'applications exhibant un « comportement intelligent ». Il se borne à circonscrire le problème à l'ensemble gigantesque des représentations possibles. Mais si cette structure peut être utilisée de manière à se placer sur le niveau le plus approprié, l'organisation des représentations suivant leur approximation devrait être un atout pour appliquer des techniques de type méta-raisonnement [Pitrat 1990]. De plus, la conservation des propriétés de l'approximant vers l'approximé est un point fort pour construire des algorithmes progressifs passant d'une représentation à une représentation plus précise [Giunchiglia& 1997]. Enfin, savoir plaquer à cette structure une mesure permettant d'évaluer chaque représentation en fonction de critères propres à l'utilisation et à son contexte [Russell& 1991, 1997] est à mon avis la voie la plus prometteuse développée actuellement pour la production d'un raisonnement adapté. Il restera donc toujours très important de pouvoir

développer l'espace des possibles à la demande car la situation, et donc le critère d'évaluation, change d'un individu à l'autre et d'une application à l'autre[8].

L'attention portée à l'insertion du modélisateur dans la boucle de représentation de connaissance est d'ailleurs présente dans d'autres aspects de mon travail comme une modeste contribution à l'étude psychologique de l'adéquation de certaines constructions de langages de programmation par objets [Euzenat 1997b].

Par ailleurs, la sémantique des langages est pensée pour les implémenteurs de systèmes (compilateur). En fait, elle l'est tout autant pour les utilisateurs (programmeurs). D'aucuns déplorent que ce que la théorie a fait pour populariser une syntaxe formelle au travers de la "Backus-Naur form" [ISO/IEC 14997:1996] elle n'a pu le faire pour la sémantique ("popular semantics" [Schmidt 1997]). J'ai défendu ailleurs l'intérêt d'une sémantique dénotationelle simple [Euzenat 1994c] pour les mêmes raisons, certainement corrélées avec les applications dans lesquelles je me suis impliqué (surtout en tant que tuteur). Bien entendu, dans l'absolu, il faudrait disposer d'utilisateurs rompus aux concepts les plus évolués ou de machines capables de se passer de l'entrée des utilisateurs. Mais les contraintes actuelles doivent être prises en compte.

L'attention portée aux utilisateurs a d'autres implications qui n'ont pas été évoquées jusqu'à présent. La problématique présentée ici a été inspirée de l'idée de raisonnement à profondeur variable [Bonté& 1988, Kayser 1997]. Cela dit, à l'exception du travail sur la granularité (voir la légende du Schéma 15) et celui sur la révision, l'ensemble des travaux présentés ici se placent strictement dans un cadre où le passage d'un niveau à l'autre se fait dans le respect du niveau le plus grossier alors que le raisonnement à profondeur variable, au contraire, développe une théorie non monotone permettant de ne plus déduire au niveau précis ce qui était déductible au niveau grossier.

Ma conviction d'informaticien, créant des outils pour des utilisateurs peu rompus aux joies des structures complexes, est différente. En effet, devant un problème non abordé, il est tentant de le considérer tel quel : systèmes non-linéaires, comportements discontinus, programmes non-déterministes, logique non-monotone… Cela mène à des formulations difficiles à expliquer (en particulier à des utilisateurs) et à implémenter. L'approche qui consiste à supprimer tous ces « non- », outre son intelligibilité, apporte des propriétés très importantes en informatique de compositionnalité et d'extensibilité. Ce genre de propriété, consistant à toujours conserver les propriétés établies de manière à disposer d'une base solide, me semble être à la base de nouveaux cadres tels que le modèle de programmation d'Oz [Smolka 1995].

[8] L'après-midi même du jour où j'écris cela, j'assiste à un séminaire de Jean Sallantin dont le premier transparent dit en substance: il faut une ontologie faisant l'inventaire des êtres, un principe de rationalité permettant de construire ces ontologies et des techniques d'apprentissage le mettant en œuvre dans une démarche. C'est bien ceci qui est proposé dans ce dernier paragraphe.

LÉGENDES DES GRAPHISMES

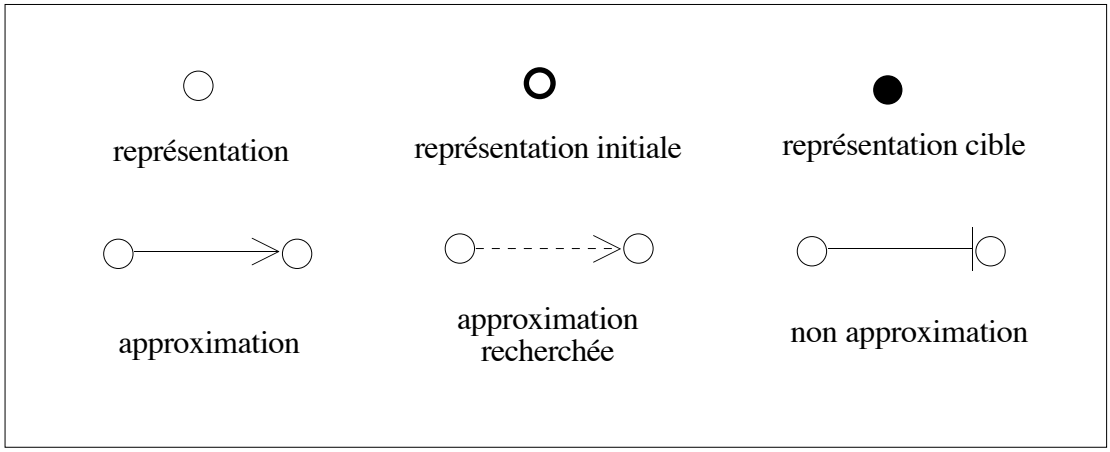


Figure 19 : Légende des schémas.

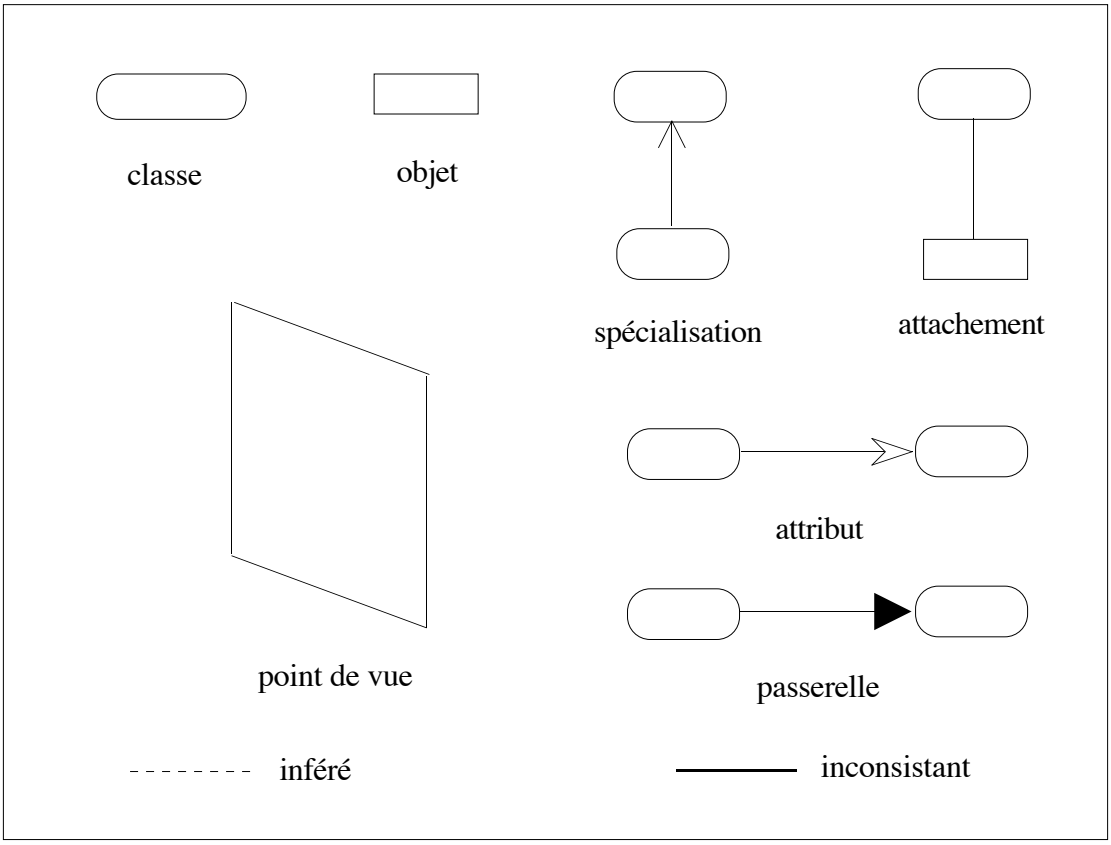


Figure 20 : Légende des représentations par objets.

RÉFÉRENCES

[Abiteboul& 1991] Serge Abiteboul, Anthony Bonner, Objects and views, *SIGMOD Record* 20(2): 239-247 (Actes SIGMOD, Denver (CO US), 1991), 1991

[Aït-Kaci& 1993] Hassan Aït-Kaci, Andreas Podelski, Towards a meaning of LIFE, *Journal of logic programming*16(3-4):195-234, 1993

[Alchourrón& 1985] Carlos Alchourrón, Peter Gärdenfors, David Makinson, On the logic of theory change: partial meet contraction and revision functions, *Journal of symbolic logic* 50(2):510-530, 1985

[Alemany 1998] Christophe Alemany, Étude et réalisation d'une interface d'édition de bases de connaissances au travers du World Wide Web, Mémoire d'ingénieur, CNAM, Grenoble (FR), 1998 `ftp://ftp.inrialpes.fr/pub/sherpa/rapports/cnam-alemany.ps.gz`

[Allen 1983] James Allen, Maintaining knowledge about temporal intervals, *Communications of the ACM,* 26(1):832-843, 1983

[Assemblée nationale 1959] Assemblée nationale, Règlement de l'assemblée nationale, Petite loi 29, Assemblée nationale, Paris (FR), 21 juillet 1959 (tel qu'en vigueur depuis le 25 mars 1998) `http://www.assemblee-nationale.fr/5/5ab.htm`

[Baader& 1991] Franz Baader, Philipp Hanschke, A scheme for integrating concrete domains into concept languages, Actes 12th IJCAI, Sydney (AU), pp452-457, 1991

[Banerjee& 1987] Jay Banerjee, Won Kim, Hyoung-Joo Kim, Henry Korth, Semantics and implementation of schema evolution in object-oriented databases, *SIGMOD records* 16(3):311-322 (actes SIGMOD, San Francisco (CA US), 1987), 1987

[Baral& 1992] Chitta Baral, Sarit Kraus, Jack Minker, Venkataram Subrahmanian, Combining knowledge bases consisting in first order theories, *Computational intelligence* 8(1):45-71 1992

[Barkowsky& 1997] Thomas Barkowsky, Christian Freksa, Cognitive requirements on making and interpreting maps, *Lecture notes in computer science* 1329:347-361, 1997

[Barr& 1990] Michael Barr, Charles Wells, Category theory for computing science, Prentice Hall, Hemel Hempstead (GB), 1990 (rev. 1995)

[Barthélémy& 1988] Jean-Pierre Barthélémy, Alain Guénoche, Les arbres et les représentations des proximités, Masson, Paris (FR), 1988

[Beach& 1993] James Beach, Sakti Pramanik, John Beaman, Hierarchic taxonomic databases, dans Renaud Fortuner (éd.), Advanced computer methods for systematic biology, The Johns Hopkins university press, Baltimore (ML US), pp241-256, 1993

[Bettini& 1998] Claudio Bettini, X. Wang, Sushil Jajodia, A general framework for time granularity and its application to temporal reasoning, *Annals of mathematics and artificial intelligence* 22(1):29-58, 1998

[Berthier 1994] Denis Berthier, L'agent rationnel abstrait : objet de l'IA ?, *Revue d'intelligence artificielle* 8(4):327-359, 1994

[Bisson 1995] Gilles Bisson, Why and how to define a similarity measure for object-based representation systems, dans Nicolaas Mars (éd.), Towards very large knowledge bases (actes 2nd international conference on building and sharing very large-scale knowledge

bases (KBKS), Enschede (NL) 1995) IOS press, Amsterdam (NL), pp236-246, 1995 ftp://ftp.inrialpes.fr/pub/sherpa/publications/bisson95a.ps.gz

[Bobrow& 1977] Daniel Bobrow, Terry Winograd, An overview of KRL: knowledge representation language, *Cognitive science* 1(1):3-45 (rep. dans [Brachman& 85], pp263-295, 1985), 1977

[Bonté& 1988] Eric Bonté, Jacqueline Castaing, Philippe Grandemange, Stéphane Grumbach, Daniel Kayser, François Lévy, Description succincte d'un raisonneur à profondeur variable, Actes 8ièmes journées sur les systèmes experts et leurs applications, Avignon (FR), pp117-132, 1988

[Borgida& 1992] Alex Borgida, Ronald Brachman, PROTODL: a customizable knowledge base management system, Actes 1st Conference on Information and Knowledge Management, Baltimore (MA US), pp482-490, 1992

[Brachman 1985] Ronald Brachman, "I lied about the trees" or, defaults and definitions in knowledge representation, *AI magazine* 6(3):80-93, 1985

[Brachman& 1985] Ronald Brachman, Hector Levesque, (éds.), Readings in knowledge representation, Morgan Kaufmann, Los Altos (CA US), 1985

[Buckingham Shum 1997] Simon Buckingham Shum, Computer-supported collaborative argumentation resource site, 1998 http://kmi.open.ac.uk/sbs/csca

[Buisson 1990] Laurent Buisson, Le raisonnement spatial dans les systèmes à base de connaissances — application à l'analyse de sites avalancheux, Thèse d'informatique, université Joseph Fourier, Grenoble (FR), 1990

[Capponi 1995] Cécile Capponi, Identification et exploitation des types dans un modèle de représentation des connaissances par objets, Thèse d'informatique, université Joseph-Fourier, Grenoble (FR), 1995

[Capponi& 1995] Cécile Capponi, Jérôme Euzenat, Jérôme Gensel, Objects, types and constraints as classification schemes (abstract), Actes international symposium on "Knowledge Retrieval, Use, and Storage for Efficiency", Santa-Cruz (CA US), pp69-73, 1995 ftp://ftp.inrialpes.fr/pub/sherpa/publications/capponi95a.ps.gz

[Cardelli& 1991] Luca Cardelli, John Mitchell, Operations on records, *Mathematical Structures in Computer Science* 1(1):3-48, 1991

[Carré& 1988] Bernard Carré, Gérard Comyn, On multiple classification, points of view and object evolution, dans Jacques Demongeot, Thierry Hervé, Vincent Rialle, Christophe Roche (eds.), Artificial intelligence and cognitive sciences, Manchester University Press, Manchester (GB), pp49–62, 1988

[Chaudhri& 1992] Vinay Chaudhri, Vassos Hadzilacos, John Mylopoulos, Concurrency control for knowledge bases, Actes 3rd KR, Cambridge (MA US), pp762-773, 1992

[Cholvy 1994] Laurence Cholvy, Fusion de sources d'informations contradictoires ordonnées en fonction des thèmes, *Revue d'intelligence artificielle* 8(2):187-204, 1994

[Cligniez 1998] Vincent Cligniez, Un outil de représentation générique de l'espace pour l'étude des risques naturels, Thèse d'informatique, Université Jean Monnet, Saint-Étienne (FR), 1998

[Codd 1970] Edgar Codd, A relational model of data for large shared data banks, *Communications of the ACM* 13(6):377-387, 1970

[Conklin& 1988] Jeffrey Conklin, Michael Begemann, gIBIS: A hypertext tool for explanatory policy discussion, *ACM transactions on office information systems* 6(4):303-331, 1988

[Corruble& 1997] Vincent Corruble, Jean-Gabriel Ganascia, Induction and the discovery of the causes of scurvy: a computational reconstruction, *Artificial intelligence* 91:205-223, 1997

[Cousot 1996] Patrick Cousot, Abstract interpretation, *ACM Computing surveys* 28(2):324-328, 1996 `ftp://lix.polytechnique.fr/pub/ESPRIT/LOMAPS/LOMAPS-ENS-X-28.ps.gz`

[Crampé 1997] Isabelle Crampé, Révision interactive dans une base de connaissance à objets, Thèse d'informatique, université Joseph-Fourier, Grenoble (FR), 1997 `ftp://ftp.inrialpes.fr/pub/sherpa/theses/these-crampe.ps.gz`

[Crampé& 1998] Isabelle Crampé, Jérôme Euzenat, Object knowledge base revision, Proc 13th ECAI, Brighton (UK), pp3-7, 1998 `ftp://ftp.inrialpes.fr/pub/sherpa/publications/crampe98a.ps.gz`

[Davis 1987] Harley Davis, VIEWS: muliple perspectives and structured objects in a knowledge representation language, Master thesis, MIT, Cambridge (MA US), 1987

[Davis 1991] Randall Davis, A tale of two knowledge servers, *AI magazine* 12(3):118-120, 1991 `http://www.aaai.org/Magazine/Issues/Vol12/12-03/Davis.pdf`

[Day& 1984] William Day, Herbert Edelsbrunner, Efficient algorithms for agglomerative hierarchical clustering methods, *Journal of classification* 1(1):7-24, 1984

[De Jong& 1997] Hidde De Jong, Arie Rip, The computer revolution in science: steps towards the realization of computer supported discovery environments, *Artificial intelligence* 91(2):225-256, 1997

[De Jong 1998] Hidde De Jong, Computer-supported analysis of scientific measurements, Thèse, Université de Twente, Enschede (NL), 1998

[Dekker 1994] Lenneke Dekker, FROME : représentation multiple et classification d'objets avec points de vue, Thèse d'informatique, université des sciences et technolgie de Lille, Lille (FR), 1994

[Demazeau 1995] Yves Demazeau, From interactions to collective behaviour in agent based systems, Actes 1st European conference on cognitive science, Saint-Malo (FR), 1995

[Doyle& 1991] Jon Doyle, Ramesh Patil, Two theses of knowledge representations: language restrictions, taxonomic classification and the utility of representation services, *Artificial intelligence* 48(3):261-297, 1991

[Ducournau 1996] Roland Ducournau, Des langages à objets aux logiques terminologiques : les systèmes classificatoires, Rapport de recherche 30, LIRMM, Montpellier (FR), 1996 `ftp://ftp.lirmm.fr/pub/LIRMM/papers/1996/RRI-Ducournau-96.ps`

[Ducournau 1998] Roland Ducournau, La logique des objets : application à la classification incertaine, dans [Ducournau& 1998], pp351-379, 1998

[Ducournau& 1995] Roland Ducournau, Michel Habib, Marianne Huchard, Marie-Laure Mugnier, Amedeo Napoli, Le point sur l'héritage multiple, *Techniques et science informatiques* 14(3):309-345, 1995

[Ducournau& 1998] Roland Ducournau, Jérôme Euzenat, Gérald Masini, Amedeo Napoli (éds.), Langages et modèles à objets : état des recherches et perspectives, INRIA, Rocquencourt (FR), 1998 http://co4.inrialpes.fr/lmobook/

[Dugerdil 1988] Philippe Dugerdil, Contribution à l'étude de la représentation de connaissances fondée sur les objets : le langage ObjLog, Thèse, université d'Aix-Marseille, Luminy (FR), 1988

[Euzenat 1990] Jérôme Euzenat, Un système de maintenance de la vérité à propagation de contextes, Thèse d'informatique, Université Joseph Fourier, Grenoble (FR), 1990, 131p. ftp://ftp.inrialpes.fr/pub/sherpa/theses/euzenat.ps.gz

[Euzenat 1991b] Jérôme Euzenat, Contexts for nonmonotonic RMSes, Actes 12th IJCAI, Sydney (AU), pp300-305, (24-30 août) 1991

[Euzenat 1993a] Jérôme Euzenat, Une définition abstraite de la classification et son application aux taxonomies d'objets, Actes 2ndes journées «représentations par objets» (RPO), La Grande Motte (FR), pp235-246, 1993 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat93a.ps.gz

[Euzenat 1993c] Jérôme Euzenat, Brief overview of T-TREE: the TROPES Taxonomy building Tool, dans Philip Smith, Clare Beghtol, Raya Fidel, Barbara Kwasnik (éds.), Avances in classification research 4 (Actes 4th ASIS SIG/CR classification research workshop, Columbus (OH US), pp69-87, 1993), Learning information, Medford (NJ US), 1994 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat93c.ps.gz

[Euzenat 1993d] Jérôme Euzenat, Multiple labelling generators in non monotonic RMS graphs, Rapport de recherche 2076, INRIA, Grenoble (FR), 1993 ftp://ftp.inrialpes.fr/pub/sherpa/rapports/rr-inria-2076.ps.gz

[Euzenat 1993e] Jérôme Euzenat, Représentation granulaire du temps, *Revue d'intelligence artificielle* 7(3):329-361, 1993 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat93d.ps.gz

[Euzenat 1994a] Jérôme Euzenat, Classification dans les représentations par objets : produits de systèmes classificatoires, Actes 9ième RFIA, Paris (FR), pp185-196, 1994

[Euzenat 1994c] Jérôme Euzenat, KR and OOL co-operation based on semantics non reducibility, Actes 11th ECAI workshop on integrating object-orientation and knowledge representation, Amsterdam (NL), sans pagination, 1994 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat94b.ps.gz

[Euzenat 1995a] Jérôme Euzenat, Building consensual knowledge bases: context and architecture, dans Nicolaas Mars (éd.), Towards very large knowledge bases (actes 2nd international conference on building and sharing very large-scale knowledge bases (KBKS), Enschede (NL), 1995) IOS press, Amsterdam (NL), pp143-155, 1995 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat95a.ps.gz

[Euzenat 1995d] Jérôme Euzenat, A categorical approach to time representation: first study on qualitative aspects, Actes séminaire IJCAI «spatial and temporal reasoning», Montreal (CA), pp145-152, 1995 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat95d.ps.gz

[Euzenat 1995e] Jérôme Euzenat, An algebraic approach for granularity in qualitative time representation, Actes 14th International Joint Conference on Artificial Intelligence,

Montreal (CA), pp894-900, 1995
ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat95e.ps.gz

[Euzenat 1996b] Jérôme Euzenat, Corporate memory through cooperative creation of knowledge bases and hyper-documents, Actes 10th knowledge acquisition workshop, Banff (CA), pp(36)1-18, 1996
http://www.inrialpes.fr/sherpa/papers/euzenat96b/euzenat96b.html
ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat96b.ps.gz

[Euzenat 1996c] Jérôme Euzenat, HyTropes: a WWW front-end to an object knowledge management system, Actes 10th knowledge acquisition workshop demonstration track, Banff (CA), pp(62)1-12, 9-14 novembre 1996
http://www.inrialpes.fr/sherpa/papers/euzenat96c/euzenat96c.html
ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat96c.ps.gz

[Euzenat 1997a] Jérôme Euzenat, A protocol for building consensual and consistent repositories, Rapport de recherche 3260, INRIA Rhône-Alpes, Grenoble (FR), septembre 1997, 46p. ftp://ftp.inrialpes.fr/pub/sherpa/rapports/rr-inria-3260.ps.gz

[Euzenat 1997b] Jérôme Euzenat, Influence des classes intermédiaires dans les tests de classification, Session posters 4es journées « langages et modèles à objet », Roscoff (FR), 5p. 1997
ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat97c.ps.gz

[Euzenat 1998a] Jérôme Euzenat, Algèbres d'intervalles sur des domaines temporels arborescents, Actes 11ième RFIA, Clermont-Ferrand (FR), ppIII-385-394, 1998

[Euzenat 1998b] Jérôme Euzenat, Représentation de connaissance par objets, dans [Ducournau& 1998], pp293-319, 1998

[Euzenat 1998d] Jérôme Euzenat, Sémantique des représentations de connaissance, Support de cours, Université Joseph-Fourier, Grenoble (FR), 1998

[Euzenat en révision] Jérôme Euzenat, Granularity in relational spaces: application to time and space representation, en révision

[Euzenat& 1990] Jérôme Euzenat, Libero Maesano, An architecture for selective forgetting, Actes 8th AISB conference, Leeds (GB), pp117-128, 1991

[Euzenat& 1995] Jérôme Euzenat, François Rechenmann, Shirka, 10 ans, c'est Tropes ?, Actes 2ndes journées «langages et modèles à objets», Nancy (FR), pp13-34, 1995
ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat95f.ps.gz
http://www.inrialpes.fr/sherpa/papers/shirka10/shirka10.html

[Euzenat& 1997b] Jérôme Euzenat, Christophe Chemla, Bernard Jacq, A knowledge base for *D. melanogaster* gene interactions involved in pattern formation, Actes 5th international conference on intelligent systems for molecular biology, Halkidiki (GR), pp108-119, 1997 ftp://ftp.inrialpes.fr/pub/sherpa/publications/euzenat97b.ps.gz

[Euzenat& à paraître] Jérôme Euzenat, Angelo Montanari, Time granularity, à paraître

[Fagin& 1995] Ronald Fagin, Joseph Halpern, Yoram Moses, Moshe Vardi, Reasoning about knowledge, The MIT press, Cambridge (MA US), 1995

[Falkenhainer& 1988] Brian Falkenhainer, Shankar Rajamoney, The interdependencies of theory formation, revision and experimentation, Rapport de recherche DCS-R-88-1439, University of Illinois, Urbana-Champain (IL US), 1988

[Farquhar& 1995] Adam Farquhar, Richard Fikes, Wanda Pratt, James Rice, Collaborative ontology construction for information integration, Rapport de recherche 63, Knowledge system laboratory, Stanford university, Stanford (CA US), 1995 ftp://ksl.stanford.edu/pub/KSL_Reports/KSL-95-63.ps

[Filman 1988] Robert Filman, Reasoning with worlds and truth maintenance in a knowledge-based programming environment, *Communications of the ACM* 31(4):382-401, 1988

[Fischer& 1985] Michael Fischer, Nancy Lynch, Michael Paterson, Impossibility of distributed consensus with one faulty process, *Journal of the ACM* 32(2):374-382, 1985

[Foo 1995] Norman Foo, Ontology revision, *Lecture notes in computer science* 954:16-31, 1995 ftp://ftp.cs.su.oz.au/ksg/papers/ontology.revision.ps.gz

[Freksa 1992] Christian Freksa, Temporal reasoning based on semi-intervals, *Artificial intelligence,* 54(1):199-227, 1992

[Friedman& 1996] Nir Friedman, Joseph Halpern, Belief revision: a critique, Actes 5th KR conference, Cambridge (MA US), pp421-431, 1996

[Gaines 1993] Brian Gaines, A class library implementation of a principled open architecture knowledge representation server with plug-in data types, Actes 13th IJCAI, Chambéry (FR), pp504-509, 1993

[Gensel 1995a] Jérôme Gensel, Integrating constraints in an object-based knowledge representation system, *Lecture notes in computer science* 923:67-77 (Manfred Meyer (éd.), Constraint processing), 1995 ftp://ftp.inrialpes.fr/pub/sherpa/publications/gensel95a.ps.gz

[Gensel 1995b] Jérôme Gensel, Contraintes et représentation de connaissances par objets : application au modèle TROPES, Thèse d'informatique, université Joseph Fourier, Grenoble (FR), 1995 ftp://ftp.inrialpes.fr/pub/sherpa/theses/gensel.ps.gz

[Gil& 1997] Yolanda Gil, Marcelo Tallis, A script-based approach to modifying knowledge bases, Actes 14th AAAI, Providence (RI US), pp377-384, 1997

[Giunchiglia& 1997] Fausto Giunchiglia, Adolfo Villafiorita, Toby Walsh, Theories of abstraction, *AI communications* 10(2):167-176, 1997

[Goguen 1998] Joseph Goguen, An introduction to algebraic semiotics, with application to user interface design, actes Computation for metaphor, analogie and agents, Aizu (JP), 1998 http://www-cse.ucsd.edu/users/goguen/ps/as.ps.gz

[Goodwin 1979] James Goodwin, Taxonomic programming with KLONE, Rapport de recherche, Linköping university, Linköping (SE), 1979

[Guarino 1998] Nicola Guarino (éd.), Formal ontology in information systems, IOS press, Amsterdam (NL), 1998 http://www.ladseb.pd.cnr.it/infor/ontology/papers/FOIS98.ps

[Hansson 1996] Sven Hansson, A test battery for rational database updating, *Artificial intelligence* 82(1):341-352, 1996

[Hautamäki 1986] Antti Hautamäki, Points of view and their logical analysis, *Acta philosophica fennica* 41, 1986

[Hirsch 1996] Robin Hirsch, Relation algebras of intervals, *Artificial intelligence* 83(2):267-295, 1996

[Hobbs 1985] Jerry Hobbs, Granularity, Actes 9th IJCAI, Los Angeles (CA US), pp432-435, 1985

[Ilog 1991] Ilog S.A., SMECI : manuel de référence, Ilog, Gentilly (FR), 1991

[ISO/IEC 14997:1996] JTC1/WG19, Syntactic metalanguage – Extended BNF, International standard ISO/IEC 14977:1996, International Organization for Standardization/International Electrotechnical Commission, 1996

[Johnson-Laird 1983] Philip Johnson-Laird, Mental models, Cambridge university press, Cambridge (UK), 1983

[Karp 1993] Peter Karp, Design method for scientific hypothesis formation and their application to molecular biology, *Machine learning* 12(1-3):89-117, 1993

[Karp& 1995] Peter Karp, Karen Myers, Thomas Gruber, The Generic Frame Protocol, Actes 14th IJCAI, Montréal (CA), pp768-774, 1995

[Karp& 1997] Peter Karp, Vinay Chaudhri, Suzanne Paley, A collaborative environment for authoring large knowledge bases, 1997, submitted for publication `http://www.ai.sri.com/pubs/papers/Karp9704:Collaborative/document.ps`

[Kayser 1997] Daniel Kayser, La représentation de connaissances, Hermès, Paris (FR), 1997

[Kayser 1998] Daniel Kayser, Ontologically yours, *Lecture notes in computer science* 1453:35-48, 1998

[Kifer& 1995] Michael Kifer, Georg Lausen, James Wu, Logical foundations of object-oriented and frame-based languages, *Journal of the ACM* 42(4):741-843, 1995

[Kornfeld& 1981] William Kornfeld, Carl Hewitt, The scientific community metaphor, *IEEE transactions on man, systems and cybernetics* 11(1):24-33 (rep. rapport de recherche AI-memo 641, MIT, Cambridge (MA US), 1981), 1981

[Kuhn 1959] Thomas Kuhn, The essential tension: tradition and innovation in scientific research, dans C. Taylor (éd.), Actes 3rd University of Utah research conference on the identification of creative scientific talent, Salt Lake City (UT US), pp162-174 (trad. fr. Pierre Jacob (éd.), La tension essentielle : tradition et innovation dans la recherche scientifique, dans Pierre Jacob, De Vienne à Cambridge : l'héritage du positivisme de 1950 à nos jours, Gallimard, Paris (FR), 1980), 1959

[Kumar& 1996] Akhil Kumar, Kavindra Malik, Optimizing the costs of hierarchical quorum consensus, *Acta informatica* 33(3):255-276, 1996

[Lebbe 1998] Jacques Lebbe, Représentations par objets et classifications biologiques, dans [Ducournau& 1998], pp421-447, 1998

[Lecourtier 1998] Claire Lecourtier, Révision collaborative d'un serveur de connaissances, DEA d'informatique, INPG-Université Joseph Fourier, Grenoble (FR), 1998 `ftp://ftp.inrialpes.fr/pub/sherpa/rapports/dea-lecourtier.ps.gz`

[Le Peutrec 1998] Stéphane Le Peutrec, Mécanismes d'abstraction dans les représentations de connaissances par objets, Thèse d'informatique, Université de Rennes 1, Rennes (FR), 1998 `ftp://ftp.irisa.fr/techreports/theses/1998/lepeutrec.ps.gz`

[Leuschner 1991] D. Leuschner, A mathematical model for classification and identification, *Journal of classification* 8(1):99-113, 1991

[Levesque& 1987] Hector Levesque, Ronald Brachman, Expressiveness and tractability in knowledge representation and reasoning, *Computational intelligence/intelligence informatique* 3(2):78-93, 1987

[Ligozat 1990] Gérard Ligozat, Weak representations of interval algebras, Actes 8th AAAI, Boston (MA US), pp715-720, 1990

[Gérard Ligozat, Towards a general characterization of conceptual neighborhoods in temporal and spatial reasoning, dans Frank Anger, Rasiah Loganantharaj (éds.), actes 12th AAAI workshop on spatial and temporal reasoning, Seattle (WA US), pp55-59, 1994

[Mac Gregor 1991] Robert Mac Gregor, The evolving technology of classification-based knowledge representation systems, dans John Sowa (éd.), Principles of semantic networks: exploration in the representation of knowledge, Morgan-Kauffman, San-Mateo (CA US), pp385-400, 1991

[Mackworth 1977] Alan Mackworth, Consistency in networks of relations, *Artificial intelligence* 8(1):99-118, 1977

[Mariño 1991] Olga Mariño, Classification d'objets composites dans un système de représentation de connaissances multi-points de vue, Actes 8ième RFIA, Villeurbanne (FR), pp233-242, 1991

[Mariño 1993] Olga Mariño, Raisonnement classificatoire dans une représentation à objets multi-points de vue, Thèse d'informatique, université Joseph Fourier, Grenoble (FR), 1993 `ftp://ftp.inrialpes.fr/pub/sherpa/theses/marino.ps.gz`

[Mariño& 1990] Olga Mariño, François Rechenmann, Patrice Uvietta, Multiple perspectives and classification mechanism in object-oriented representation, Actes 9th ECAI, Stockholm (SE), pp425-430, 1990

[Martin 1996] Éric Martin, Découverte scientifique et rationalité, Actes 9th Journées françaises d'apprentissage, Sête (FR), pp164-175, 1996

[Mayr 1981] Ernst Mayr, Biological classification: toward a synthesis of opposing methodologies, *Science* 214(10):510-516, 1981

[Michalski& 1983] Ryszard Michalski, Robert Stepp, Automatic construction of classifications: conceptual clustering versus numeric taxonomy, *IEEE transaction on pattern analysis and machine intelligence* 5(4):396-409, 1983

[Minsky 1974] Marvin Minsky, A framework for representing knowledge, Rapport technique AI-memo 306, MIT, Cambridge (MA US), (rep. dans Patrick Winston (éd.), The psychology of computer vision, Mac Graw-Hill, New York (NY US), pp211-277, 1975 ; rep. dans [Brachman& 85], pp245-262, 1985), 1974

[Montanari 1996] Angelo Montanari, Metric and layered temporal logic for time granularity, ILLC DIssertation Series 1996-02, University of Amsterdam, Amsterdam (NL), 1996

[Montanari& 1992] Angelo Montanari, Enrico Maim, Emanuele Ciapessoni, Elena Ratto, Dealing with time and granularity in the event calculus, Actes 4th FGCS, Tokyo (JP), pp702-712, 1992

[Muller& 1995] J.-C. Muller, J.-P. Lagrange, Robert Weibel, F. Salgé, Generalization: state of the art and issues, dans J.-C. Muller, J.-P. Lagrange, Robert Weibel (éds.), GIS and generalization, Taylor and Francis, London (UK), pp3-17, 1995

[Napoli 1992] Amedeo Napoli, Représentations à objets et raisonnement par classification en intelligence artificielle, Thèse d'état, Université de Nancy 1, Nancy (FR), 1992

[Napoli 1998] Amedeo Napoli, Une introduction aux logiques de descriptions, dans [Ducournau& 1998], pp321-350, 1998

[Nayak& 1995] P. Pandurang Nayak, Alon Levy, A semantic theory of abstraction, Actes 14th IJCAI, Montreal (CA), pp196-202, 1995

[Nebel 1990] Bernhard Nebel, Reasoning and revision in hybrid representation systems, *Lecture notes in computer science (lecture notes in artificial intelligence)* 422, 1990

[Nebel 1992] Bernhard Nebel, Syntax-based approaches to belief revision, dans Peter Gärdenfors (éd.), Belief revision, Cambridge tracts in theoretical computer science 29, Cambridge university press, Cambridge (GB), pp52-88, 1992

[Neches& 1991] Robert Neches, Richard Fikes, Tim Finin, Thomas Gruber, Ramesh Patil, Ted Senator, William Swartout, Enabling technology for knowledge sharing, *AI Magazine* 12(3):36-56, 1991

[Nguyen& 1989] Gia Toan Nguyen, Dominique Rieu, Schema evolution in object-oriented database systems, *Data & Knowledge Engineering* 4(1):43-67, 1989

[Nii 1986] Penny Nii, Blackboard systems, *AI magazine* 7(2):38-53 & 7(3):82-106, 1986

[Nökel 1988] Klaus Nökel, Convex relations between time intervals, Rapport de recherche 17, SEKI, Kaiserslautern (DE), 1988

[Nusibeh& 1994] Bashar Nusibeh, Jeffrey Kramer, Anthony Finkelstein, A framework for expressing the relationships between multiple views in requirement specifications, *IEEE transactions on software engineering* 20(10):760-773, 1994

[Pankhurst 1991] Richard Pankhurst, Practical taxonomic computing, Cambridge university press, Cambridge (GB), 1991

[Patil 1981] Ramesh Patil, Causal representation of patient illness for electrolyte and acid-base diagnosis, PhD thesis, MIT, Cambridge (MA US), 1981

[Pecheur 1997] Charles Pecheur, Specification and verification of the Co_4 distributed knowledge system using LOTOS, Rapport de recherche 3259, INRIA Rhône-Alpes, Montbonnot (FR), 1997 (version courte dans les actes de 12th IEEE international conference on automated software engineering, Incline Village (NE US), 1997)

[Pierce 1991] Benjamin Pierce, Basic category theory for computer scientists, The MIT press, Cambridge (MA US), 1991

[Pitrat 1990] Jacques Pitrat, Métaconnaissance : futur de l'intelligence artificielle, Hermès, Paris (FR), 1990

[Raczy 1996] Damien Raczy, Esquisse de formalisation de la théorie des modèles mentaux, mémoire de DEA de sciences cognitives, INPG, Grenoble (FR), 1996

[Randell& 1992] David Randell, Anthony Cohn, Zhan Cui, Computing transitivity tables: a challenge for automated theorem provers, *Lecture notes in computer science* 607:786-790, 1992

[Rechenmann 1993] François Rechenmann, Building and sharing large knowledge bases in molecular genetics, Actes 1st International Conference on Building and Sharing of Very Large-Scale Knowledge Bases (KBKS), Tokyo (JP), pp291-301, 1993 `ftp://ftp.inrialpes.fr/pub/sherpa/publications/rechenmann93.ps.gz`

[Rechenmann 1995] François Rechenmann, Knowledge bases and computational molecular biology, dans Nicolaas Mars (éd.), Towards very large knowledge bases (actes 2nd international conference on building and sharing very large-scale knowledge bases (KBKS), Enschede (NL), 1995) IOS press, Amsterdam (NL), pp7-12, 1995

[Rechenmann& 1988] François Rechenmann, Patrice Uvietta, Pierre Fontanille, SHIRKA, manuel d'utilisation, Rapport interne, IMAG, Grenoble (FR), 1988 (rev. 1989) `ftp://ftp.inrialpes.fr/pub/sherpa/rapports/manuel-shirka.ps.gz`

[Redfern 1994] D. Redfern, The Maple handbook: Maple V release 3, Springer-Verlag, New-York (NY US), 1994

[Russell& 1991] Stuart Russell, Eric Wefald, Principles of meta-reasoning, *Artificial intelligence* 49:361-395, 1991

[Russell 1997] Stuart Russell, Rationality and intelligence, *Artificial intelligence* 94(1):57-77, 1997
`http://www.elsevier.nl/cas/tree/store/artint/sub/noncas/acc/1460.ps.gz`

[Schmeltzer 1995] Olivier Schmeltzer, Modélisation de cartes génomiques : une formalisation et un algorithme de construction fondé sur le raisonnement temporel, Thèse d'informatique, université Joseph-Fourier, Grenoble (FR), 1995

[Schmid& 1975] Hans Albrecht Schmid, Phillip Bernstein, A multi-level architecture for relational data base systems, Actes 1st VLDB, Framingham (MA US), pp202-226, 1975

[Schmidt 1997] David Schmidt, On the need for a popular formal semantics, *ACM Sigplan notices* 32(1):115-116, 1997

[Sherpa 1995] Projet Sherpa, Tropes 1.0 reference manual, Rapport interne, INRIA Rhône-Alpes, Grenoble (FR), juin 1995 (rev. Tropes 1.1 reference manual, 1997, 120p. ; Troeps 1.2 reference manual, 1998, 145p.), 85p.
`http://hytropes.inrialpes.fr/docs/tropes-manual.html`
`ftp://ftp.inrialpes.fr/pub/sherpa/rapports/troeps-manual.ps.gz`

[Smith 1980] Reid Smith, The contract net protocol: high level communication and control in a distributed problem solver, *IEEE transactions on computers* 29(12):1104-1113 (rep. dans Alan Bond, Les Gasser (éds.), Readings in distributed artificial intelligence, pp357-366, Morgan Kauffman, San Mateo (CA US), 1988), 1980

[Smolka 1995] Gert Smolka, The Oz programming model, *Lecture notes in computer science* 1000:324-343, 1995

[Sombé 1994] Léa Sombé (éds), Special issue on revision and updating in knowledge base, *International journal of intelligent systems* 9(1), 1994

[Stefik 1995] Mark Stefik, Introduction to knowledge systems, Morgan Kauffman, San Francisco (CA US), 1995

[Stein 1991] Johannes Stein, Entwicklung eines Zeitmodells mit der Fähigkeit zur Variation der Granularität und dessen Implementierung im Repräsentationswerkzeug Tropes, Diplomarbeit, Kaiserslautern Universtät, Kaiserslautern (DE), 1991

[Strecker 1991] Martin Strecker, Oubli dans des systèmes de connaissances, Mémoire de DEA informatique, UJF-INP, Grenoble (FR), 1991

[Subrahmanian 1994] Venkataram Subrahmanian, Amalgamating knowledge bases, *ACM transactions on database systems* 19(2):291-331, 1994

[Tan 1993] Shong-Ye Tan, Types et classes dans les bases de connaissances à objet, Mémoire de DEA informatique, UJF-INP, Grenoble (FR), 1993

[Tennison& 1998] Jenifer Tennison, Nigel Shadbolt, APECKS: a tool to support living ontologies, Actes 11th KAW, Banff (CA), 1998 http://ksi.cpsc.ucalgary.ca/KAW/KAW98/tennison/

[Theodoratos 1995] Dimitri Theodoratos, Updating object-oriented schema structures viewed as logical theories, Rapport de recherche 12, ERCIM, Rocquencourt (FR), 1995 ftp://ftp.inria.fr/associations/ERCIM/research_reports/ps/1295R043.ps

[Tversky& 1984] Barbara Tversky, Kathleen Hemenway, Objects, parts and categories, *Journal of experimental psychology* 113(2):169-193, 1984

[Valtchev 1998] Petko Valtchev, Infering class taxonomies by automatic clustering of software objects, Actes 6th IFCS, Roma (IT), 1998 à paraître

[Valtchev 1999a] Petko Valtchev, An algorithm for minimal insertion in a type lattice, *Computational intelligence* 15(1):63-78, 1999

[Valtchev 1999b] Petko Valtchev, Construction automatique de taxonomies dans une représentation de connaissances par objets, Thèse d'informatique, université Joseph-Fourier, Grenoble (FR), 1998

[Valtchev& 1996] Petko Valtchev, Jérôme Euzenat, Classification of concepts through products of concepts and abstract data types, dans Edwin Diday, Yves Lechevalier, Otto Opitz (éds.), Ordinal and symbolic data analysis, Springer Verlag, Heildelberg (DE), pp3-12, 1996 ftp://ftp.inrialpes.fr/pub/sherpa/publications/valtchev96a.ps.gz

[Valtchev& 1997] Petko Valtchev, Jérôme Euzenat, Dissimilarity measure for collections of objects and values, *Lecture notes in computer science* 1280:259-272, 1997 ftp://ftp.inrialpes.fr/pub/sherpa/publications/valtchev97c.ps.gz

[Wiederhold& 1993] Gio Wiederhold, Sushil Jajodia, Witold Litwin, Integrating temporal data in a heterogeneous environment, dans Abdullah Uz Tansel, James Clifford, Shashi Gadia, Sushil Jajodia, Arie Segev, Richard Snodgrass (éds.), Temporal databases, Benjamin/Cummings, Redwood city (CA US), pp564-578, 1993

[Williams 1984] Chuck Williams, ART: The advanced reasoning tool, conceptual overview, Inference Corporation, Los Angeles (CA US), 1984

[Woods 1975] William Woods, What's in a link: foundations for semantic networks, dans Daniel Bobrow, Alan Collins (éds.), Representation and understanding: studies in cognitive science, New-York Academic Press, New-York (NY US), pp35-82 (rep. dans [Brachman& 85], pp217-241, 1985), 1975

INDEX DES TERMES ET DES NOMS PROPRES

Les noms et termes utilisés dans ce mémoire sont indexés ci-dessous. La page de définition du terme correpond au numéro de page en italique.

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z

Knowledge representations: from approximation to confrontation

Jérôme Euzenat

A knowledge representation formalism aims at modelling a precise domain. There are several such formalisms and, within one of them, there can be numerous models of the same domain. The present report is devoted to the study of the relationships between multiple representations of the same situation. It describes the work of the author between 1992 and 1998 progressing from the approximation relation to the confrontation of the representations.

First, the notion of an approximation between object-based representations is developed mainly with regard to the taxonomy-based mechanisms (classification, categorisation, clustering). A classification scheme allows to consider these mechanisms in a homogeneous way and puts forward the approximation relation between the initial representation and the final one. It is shown how to build a knowledge representation system according to the principles of classification schemes.

The second chapter introduces the advantages and problems of providing alternative taxonomies (organising the same set of objects) in a knowledge representation system. It considers some choices that can be made in implementing multiple taxonomies. These taxonomies are rephrased in terms of classification schemes.

Granularity is the topic of the third chapter. Contrary to the other chapters, the representation formalism is an algebra of binary relations enabling to express the relations (in time or space for instance) between objects. The relationship between representations is constrained since the representations are ordered by the level of detail they consider. The rules governing such representations are presented.

Last, the fourth chapter aims at confronting the various representations in order to take advantage of them (i.e. building a consistent and consensus representation). The goal of the presented works is the implementation of a system for co-operatively building consensus knowledge bases in which the users build a common knowledge base from the content of their individual ones. Two particular problems are considered: the design of a revision mechanism for object-based knowledge representation formalisms allowing the users to deal with inconsistency and the design of a knowledge submission protocol ensuring the gathering of consensus knowledge.

Although limited, this insight into possible studies of the relationships between representations reveals the non-imperative character of the developed solutions, which are well suited to the interaction between the user and the representation system.

KEY WORDS: knowledge representation — approximation — knowledge base — modelling — object-based representation — viewpoint — bridge — classification — categorisation — clustering— granularity — time representation— algebra of binary relations — revision — consensus — TROEPS — CO4.

Représentations de connaissance : de l'approximation à la confrontation

Jérôme Euzenat

Un formalisme de représentation de connaissance a pour but de permettre la modélisation d'un domaine particulier. Bien entendu, il existe divers langages de ce type et, au sein d'un même langage, divers modèles peuvent représenter un même domaine. Ce mémoire est consacré à l'étude des rapports entre de multiples représentations de la même situation. Il présente les travaux de l'auteur entre 1992 et 1998 en progressant de la notion d'approximation, qui fonde la représentation, vers la confrontation entre les diverses représentations.

Tout d'abord la notion d'approximation au sein des représentations de connaissance par objets est mise en avant, en particulier en ce qui concerne l'ensemble des mécanismes tirant parti de la structure taxonomique (classification, catégorisation, inférence de taxonomie). À partir de la notion de système classificatoire qui permet de rendre compte de ces mécanismes de manière unique on montre comment un système de représentation de connaissance peut être construit.

Le second chapitre introduit la possibilité de tirer parti de multiples taxonomies (sur le même ensemble d'objets) dans un système de représentation de connaissance. La multiplicité des représentations taxonomiques est alors introduite en tant que telle et justifiée. Ces multiples taxonomies sont replacées dans le cadre des systèmes classificatoires présentés auparavant.

La notion de granularité, qui fait l'objet du troisième chapitre, concerne la comparaison de représentations diverses de la même situation sachant qu'elles ont un rapport très particulier entre elles puisqu'elles représentent la même situation sous différentes granularités. À la différence des autres chapitres, celui-ci n'est pas situé dans le cadre des représentations de connaissance par objets mais dans celui des algèbres de relations binaires utilisées pour représenter le temps et l'espace.

Le quatrième chapitre, enfin, va vers la confrontation des différentes représentations de manière à en tirer le meilleur parti (obtenir une représentation consensuelle ou tout simplement une représentation consistante). Le but des travaux qui y sont présentés est de développer un système d'aide à la construction collaborative de bases de connaissance consensuelles. À cette fin, les utilisateurs veulent mettre dans une base commune (qui doit être consistante et consensuelle) le contenu de leurs bases de connaissance individuelles. Pour cela, deux problèmes particuliers sont traités : la conception d'un mécanisme de révision, pour les représentations de connaissance par objets, permettant aux utilisateurs de traiter les problèmes d'inconsistance et la conception d'un protocole de soumission de connaissance garantissant l'obtention d'une base commune consensuelle.

Cet aperçu partiel des travaux possibles dans l'étude des relations entre représentations est limité, mais il met en évidence le caractère non impératif des solutions proposées qui s'appliquent bien au cadre où le modélisateur interagit avec le système de représentation.

MOTS CLÉ : représentation de connaissance — approximation — bases de connaissance — modélisation — représentation par objets — point de vue — passerelle — classification — catégorisation — inférence de taxonomie— granularité — représentation temporelle — algèbre de relations binaires — révision — consensus — TROEPS — CO4.