REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

BADJI MOKHTAR-ANNABA UNIVERSITY
UNIVERSITE BADJI MOKHTAR-ANNABA

جامعة باجي مختار - عنابة

FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'INFORMATIQUE

Thèse présentée par

Nouaouria-Amri Nabila

En vue de l'obtention d'un Doctorat d'Etat en Informatique

Un Modèle de Mémoire de Cas Pour une Réutilisation Optimale

jury :

Pr Benyettou Abd El Kader	***Université de***	***Oran***	***Président***
Pr Laskri Mohamed Tayeb	***Université de***	***Annaba***	***Rapporteur***
Pr Aissani-Mokhtari Aicha	***Université de***	***USTHB***	***Examinateur***
Pr Sahnoun Zaidi	***Université de***	***Constantine***	***Examinateur***
Pr Sellami Mokhtar	***Université de***	***Annaba***	***Examinateur***

Soutenue publiquement
le 20 Mars 2006

Remerciements

Mes plus sincères remerciements au ***Pr Benyettou Abd El Kader*** pour avoir accepté de présider ma soutenance de thèse.

Mes plus vifs remerciements aux ***Pr Aissani-Mokhtari Aicha,*** Pr ***Sahnoun Zaidi*** ainsi qu'au ***Pr Sellami Mokhtar*** pour avoir bien voulu évaluer mon travail de thèse.

Je ne remercierai jamais assez ***mon Professeur, le Professeur Laskri Mohamed Tayeb*** pour m'avoir dirigée et surtout, pour la confiance qu'il a eu à mon égard, pour sa patience et ses encouragements tout au long de mon travail de recherche.

Je remercie tous ceux qui m'ont aidée et soutenue.

Dédicaces

« Il y a ceux qui font quelque chose :

Ils sont trois qui font quelque chose.

Il y a ceux qui ne font rien :

Ils sont dix qui font des conférences.

Il y a ceux qui croient faire quelque chose

Et ils sont cent qui font des conférences

Sur ce que disent les dix

De ce que font les trois qui font quelque chose.

Et il arrive que l'un des cent dix vienne expliquer

La manière de faire à l'un des trois qui font quelque chose.

Alors l'un des trois intérieurement s'exaspère

Et extérieurement sourit

Mais il se tait car il n'a pas la parole.

D'ailleurs, il a quelque chose à faire ». Citation d'un sage.

Même si je crois faire partie des cents qui croient faire quelque chose,

Que les trois qui font quelque chose

Acceptent ce modeste travail en guise d'excuses :

Aux Maîtres du savoir.

Table des Matières

Modèles et Algorithmes de Recherche

Chapitre 3 :

Un Modèle de Mémoire Supportant la Recherche Guidée par l'Adaptabilité

Chapitre 4 :

Approches d'Extraction de Connaissances d'Adaptabilité

Chapitre 5 :

Evaluation et Résultats

Chapitre 6 :

Autres Travaux de Modélisation

Conclusion Et Perspective

Annexes

Liste des Figures

Liste des Tables

Résumé

Le raisonnement à base de cas (CBR pour Case Based Reasoning) à travers son aspect modélisation, a constitué notre premier champs d'investigation. D'autre part, la mise en avant « la mémoire » comme thème essentiel d'étude pour le raisonnement, a orienté notre problématique vers l'étude de l'efficacité du processus de "rappel".
Cette étude nous a conduit vers la proposition d'un modèle de mémoire prenant en compte un double critère de recherche: la similarité du problème remémoré avec le problème source, et l'adaptabilité de la solution trouvée. Ce qui permettra d'optimiser la réutilisation du cas remémoré.
Afin de garantir la recherche d'un cas qui soit le plus facile à adapter, le mécanisme de recherche doit prendre en considération comment le cas sera adapté sans pour autant procéder à une adaptation complète. Il doit en fait procéder à une prévision du coût d'adaptation durant l'étape de recherche même. Il s'agit d'une anticipation sur l'étape d'adaptation sans l'effectuer dans son intégralité.
Nous nous proposons d'intégrer ce coût d'adaptation dans un modèle de mémoire non classique : Les CRNs.
Cette intégration a généré une seconde problématique, à savoir l'extraction de connaissance d'adaptabilité à partir de la base de cas. Ces connaissances serviront au calcul du coût d'adaptation utilisé comme autre critère de remémoration. Nous avons donc, soulevé cette question et proposé différentes approches pour y répondre.

ملخص

"التفكير بالحالات" عبر جانبه النموذجي يعتبر بداية مجال بحثنا. من جهة أخرى كون "الذاكرة" محور أساسي في دراسة أنماط التفكير، أمر أدى إلى توجهنا نحو الاهتمام بمرحلة "التذكر" و مدى فعاليتها.

هذه الدراسة حملتنا إلى اقتراح نموذج للذاكرة يأخذ بعين الاعتبار معيارين للبحث: تشابه المسالة المتذكرة و المسالة المصدر،و مدى قابلية الحل المتذكر للتكيف. مما يؤدي إلى إعادة استعمال مثلى للحالة المستخرجة من الذاكرة.

من أجل ضمان الحصول على الحالة الأكثر قابلية للتكييف، يجب على آلية البحث أن تأخذ بعين الاعتبار كيفية تكييف الحالة دون الخوض في مرحلة "التكييف". :كي يتسنى لنا ذلك، على آلية البحث أن تقوم بتنبؤات عن كلفة التكييف أثناء مرحلة التذكر. و هذا دون القيام بمرحلة التكييف كاملة.

نقترح في هذه الأطروحة إدماج كلفة التكييف في نموذج غير كلاسيكي للذاكرة آلا و هو . "شبكة إيجاد الحالات"

هذا الإدماج أدى إلى ظهور إشكالية ثانوية في بحثنا و هي: استخراج معرفة التكييف من قاعدة الحالات. هذه المعلومات سوف تستعمل من أجل تقييم كلفة التكيف. قمنا إذا بطرح هذه الإشكالية و اقتراح تشكيلة من الحلول.

Abstract

Case Based Reasoning, through its formal facet was our first field of investigation. Elsewhere, the central role of memory in the inference process, leads us to refine our problem position to study the efficiency of recall step.
So, we have proposed a memory model using two retrieval criteria: the similarity between target problem and source problem and the adaptability of retrieved solution. This will improve the reuse of source case.
In order to warranty the retrieval of a case which is easier to adapt, the search mechanism has to take in count how the case will be adapted without carrying on the whole process of adaptation. It has to forecast the adaptation cost during the retrieval step.
We have proposed the integration of adaptation cost in a non classical memory model: CRN for Case Retrieval Nets.
This integration leads to a second problem, which is the extraction of adaptability knowledge from the case base. This knowledge is used in computing adaptability cost. So, we posed the problem and proposed different approaches to solve it.

Introduction

Dans leur article : « *Quelques points de repères sur l'évolution de l'étude du raisonnement en psychologie cognitive* », T. Ripoll et A. Tricot **[Rip 1995]** terminent par des conclusions jugées révolutionnaires à notre sens. En effet, elles abolissent des siècles de pensée logique en psychologie cognitive et des décennies de raisonnement basé règles en Intelligence Artificielle. Par ailleurs, elles propulsent la mémoire en avant de la scène du Raisonnement.

Dans ce qui suit, nous reprenons l'essentiel de ces conclusions pour introduire notre thèse.

Pour les chercheurs en psychologie cognitive comme pour les pédagogues, il existait une croyance bien établie: l'idéal de la connaissance serait une connaissance abstraite et parfaitement décontextualisée, susceptible de s'appliquer quel que soit le domaine considéré. Les mathématiques et la logique sont considérées comme des disciplines majeures, précisément parce qu'elles impliquent un niveau d'abstraction maximum: elles tendent vers l'universalité.

En fait, cette croyance, nous dirons même cette foi, dans le caractère essentiel de la pensée inférentielle déductive, découle d'une autre croyance, encore plus fondamentale. Il s'agit de la croyance selon laquelle l'essentiel de l'activité cognitive consiste à raisonner et à résoudre des problèmes. Dès lors, l'objectif essentiel de la recherche a consisté à décrire, à formaliser puis à modéliser, les règles, les heuristiques et les algorithmes qui régissent notre faculté de penser et de raisonner. Mais les résultats de cette entreprise, commune à l'Intelligence Artificielle et à la psychologie, n'ont pas été à la hauteur des attentes. D'une part, de tels modèles n'approchent que de très loin les capacités cognitives des humains, d'autre part, de nombreuses fonctions cognitives demeurent hors d'atteinte de tels modèles (l'induction, la reconnaissance de formes, la décision en situation d'incertitude, ...), enfin et surtout de nombreuses données empiriques ont cruellement parlé en défaveur de cette première conception de "raisonner".

Sous la pression de données empiriques problématiques, un renversement théorique profond s'est amorcé et se développe actuellement. Ce renversement théorique a au moins deux facettes :

- La pensée logique est l'exception, l'inférence non démonstrative, plutôt la règle.
- L'homme est un piteux "raisonneur" (ou calculateur) mais un prodigieux "activateur".

Ces deux points ont été clairement mis en évidence à partir d'études sur les joueurs d'échec. Les grands maîtres raisonnent finalement assez peu, ils n'envisagent et ne traitent en profondeur qu'une infime partie des déplacements de pièces qu'autorisent les règles du jeu. En fait, ils parviennent, avec une rapidité incroyable, à retrouver dans leur mémoire les coups pertinents. Leur expertise semble davantage résulter d'une capacité mnésique remarquable que d'une capacité à raisonner, au sens classique qu'on donne à ce terme.

Face à ce constat, l'étude du raisonnement a évolué dans plusieurs directions plutôt complémentaires qu'opposées.

L'étude de la mémoire devient un thème central dont on ne peut plus faire l'économie, quand bien même on ne s'intéresserait qu'au Raisonnement (avec un grand R). Globalement, il semble bien que pour "bien raisonner", l'essentiel est d'activer les connaissances pertinentes. Le transfert d'apprentissage, pierre angulaire et philosophale des didacticiens, peut donc être appréhendé autrement. Ce n'est pas parce qu'on dispose d'une connaissance abstraite et décontextualisée qu'elle pourra nécessairement s'appliquer dans des contextes différents. Le transfert d'apprentissage peut fort bien être réalisé de cas à cas sans médiation par une connaissance abstraite de haut niveau. Comprendre ou favoriser le transfert revient donc à s'interroger sur les conditions d'activation d'un cas par un autre. Le problème général n'est plus "comment raisonne-t-on?", mais "comment active-t-on la bonne connaissance?".

En raisonnement par analogie, par exemple, on se représente le "transfert analogique" comme suit : "je dois résoudre le problème B; je reconnais que le problème B a la même structure que le problème A; donc je résous B comme A". Cette conception n'indique pas pourquoi ni comment le problème A est retrouvé en mémoire. Or, on a montré le rôle important d'indices "de surface" (habillage verbal du problème, situation) dans la récupération en mémoire du problème A. Pourtant ces indices de surface n'ont aucune relation avec la structure du problème, avec sa logique. Ainsi, dans l'étude du raisonnement par analogie, il n'est peut-être pas très intéressant de se demander "comment raisonne-t-on?"; en revanche, la question intéressante est "comment active-t-on la bonne connaissance?".

Les renversements théoriques décrit dans cet article à gros traits semblent suggérer que les connaissances générales et abstraites sont très peu efficaces et donc qu'il n'est pas utile de les enseigner. Evidemment, c'est faux, mais il convient de ne pas mettre la charrue avant les

boeufs. Les connaissances abstraites et générales ne s'enseignent pas, elles se construisent activement. La manière dont on les exploite dépend en grande partie de la manière dont on les a construites, ce qui en retour, détermine les conditions d'applications de telles connaissances. Apprendre à raisonner implique peut être davantage d'apprendre à organiser sa mémoire de telle sorte que les informations pertinentes soient facilement accessibles que d'apprendre à aligner des raisonnements de type logique.

Ces Conclusions introduisent et argumentent de manière intuitive, d'une part, l'intérêt du raisonnement à base de cas (CBR pour Case Based Reasoning) lorsqu'elles abordent les questions « d'inférence non démonstrative » et « de transfert d'apprentissage ». D'autre part, elles mettent en avant « la mémoire » comme thème essentiel d'étude pour le raisonnement.

Ce sont très précisément, les deux points de départ de notre problématique de thèse. A savoir la proposition d'un modèle de mémoire optimisant la réutilisation en raisonnement à base de cas.

Genèse et évolution de la problématique

En affranchissant l'intelligence artificielle du goulot d'étranglement que constituait l'acquisition des connaissances de l'expert, le raisonnement basé cas connaît actuellement un essor considérable aussi bien sur le plan commercial grâce au développement de produits industriels[1] (ReMind, ReCall, Kaidara…), que sur le plan recherche (organisation de congrès :ICCBR, EWCBR,…).

La tendance actuelle sur ce dernier plan porte sur l'aspect modélisation / formalisation et ce pour diverses raisons. La formalisation apportera plus de robustesse aux modèles proposés. Ces derniers supporterons l'expressivité des méthodologies de développement ; qui apporterons un plus à : la productivité, la qualité et la communication au sein d'une équipe de développement, ainsi qu'une bonne gestion des ressources [Ber 1998].

Parce que le CBR prend ses racines en théorie de la mémoire et en raisonnement par analogie, tout effort de modélisation doit tenir compte des aspects sous-jacents, ce qui a motivé un premier travail (voir [Nou 2003]) portant sur la modélisation du raisonnement à partir de ses racines. L'idée a ensuite graduellement évolué vers la modélisation de la mémoire, ressource cruciale pour l'inférence en CBR.

[1] Voir le site www.ai-cbr.org

Nous avions démarré notre étude à partir d'un modèle de mémoire en réseau. Il s'agit des réseaux de recherche de cas (CRN pour Case Retriaval Nets) mis au point par Lenz et son équipe (voir [Len 1996a], [Len 1996b], [Len 1998] et [Len 1999] pour les aspects fondements théoriques et [Hub 2000] pour une illustration pratique).

Dans un premier temps nous avions abordé le modèle d'un point de vue implémentation, afin de mieux maîtriser ces rouages d'une part et de tester son applicabilité d'autre part. Les domaines abordés furent : le tourisme avec l'outil TRAVAGE [Nou 2001a], l'estimation du coût de développement d'un logiciel avec l'outil PROJESTIM [Nou 2001b], puis l'aide au diagnostic en pédiatrie avec PEDIAGNO [Nou 2002]. De ces expériences, il ressort que les CRNs offrent une efficacité intéressante lorsqu'il s'agit de retrouver des cas selon un critère de similarité puisqu'en fait ils sont construits autour d'une sémantique de similarité. Mais qu'en est-il lorsque le critère de recherche doit changer ?

En effet, pour certaines tâches telles que la planification l'explication ou encore la conception, il est impératif de procéder à une adaptation :

"…*Changes have to be made to a memory of the past in order to make it fit to the mold of the present…* " *K.J. Hammond dans [Ham 1992]*

Or, de nombreux travaux ([Kol 1992], [Smy 1993], [Smy 1995])ont démontré que ce ne sont pas les cas les plus similaires qui sont toujours les plus faciles à adapter. Il faut donc envisager une recherche guidée par l'adaptation (AGR pour Adaptation-Guided Retrival). Ce concept fût introduit la première fois par Smyth et Keane dans [Smy 1993], raffiné par les mêmes auteurs dans [Smy 1995], [Smy 1999] puis repris dans son principe par de multiples travaux (voir [Col 1996], [Rou 1996]).

Afin de garantir la recherche d'un cas qui soit le plus facile à adapter, le mécanisme de recherche doit prendre en considération comment le cas sera adapté sans pour autant procéder à une adaptation complète. Il doit en fait procéder à une prévision du coût d'adaptation durant l'étape de recherche même. Il s'agit d'une anticipation sur l'étape d'adaptation sans l'effectuer dans son intégralité.

Nous nous proposons d'intégrer ce principe d'AGR dans un modèle de mémoire non classique : Les CRNs.

Cette intégration a généré une seconde problématique, à savoir l'extraction de connaissance d'adaptabilité à partir de la base de cas. Ces connaissances serviront au calcul du coût d'adaptation utilisé comme autre critère de remémoration. Nous avons donc, soulevé cette question et proposé différentes approches pour y répondre.

Organisation de la thèse

Notre travail est organisé en six chapitres selon les motivations et objectifs cités auparavant. Sommairement, les deux premiers chapitres présentent le paradigme CBR et les organisations de mémoire de cas. Ils offrent un panorama historique sur l'existant présenté comme pré-requis pour notre modélisation.

Les troisième et quatrième chapitres sont les pièces maîtresses de notre travail. Il s'agit de la modélisation d'une mémoire de cas supportant deux critères de recherche : la similarité et l'adaptabilité avec des propositions d'approches permettant l'extraction des connaissances se rapportant à ce second critère. Ils seront directement suivis par un chapitre expérimentations.

Le dernier chapitre expose d'autres directions de modélisation du raisonnement basé cas. Une conclusion générale et des perspectives viendront clôturer le champ d'investigation de notre thèse.

En plus détaillé :

- **Chapitre 1**. Dans ce chapitre, on retrouve l'ensemble des concepts et définitions relatifs au CBR. Ce sera donc, l'ensemble des connaissances de base utilisées lors de la modélisation.

- **Chapitre 2**. Dans ce chapitre, nous focalisons sur le rôle clé de la mémoire et des algorithmes de recherche afin de mieux situer le modèle de mémoire proposé.

- **Chapitre 3.** Dans ce chapitre, nous introduisons une nouvelle hypothèse pour le processus de remémoration. Nous expliquerons graduellement, le pourquoi puis le comment du modèle proposé.

- **Chapitre 4.** Dans ce chapitre, nous introduisons une gamme de techniques d'extraction de connaissances d'adaptabilité et les situons par rapport à la littérature.

- **Chapitre 5.** Ce chapitre sera dédié à l'expérimentation selon un plan d'évaluation motivé par un ensemble d'objectifs fixés à travers les chapitres 3 et 4.

- **Chapitre 6.** Ce chapitre sort du cadre de la problématique de base notre thèse ; tout en restant dans une optique de modélisation CBR. En d'autres termes, il présente d'autres visions possibles. Leurs genèses restent tout de même motivées par la modélisation du CBR.

- Une **conclusion** reprendra les motivations, pointera sur les contributions et présentera une autocritique aussi objective et exhaustive que possible. Des **perspectives** seront obligatoirement à l'horizon.

Chapitre 1 :

Le Raisonnement à Base de Cas Généralités

La majeure partie des applications en intelligence artificielle consiste à reproduire le raisonnement humain. Parallèlement aux travaux menés sur la résolution de problème dans un style déductif, on retrouve le raisonnement à base de cas (CBR[1]) comme mode d'inférence cognitivement proche des mécanismes mentaux de l'homme [Hat 1991]. En effet, dans cette vision des choses, en CBR, on postule que la résolution d'un problème ne se pose pas nécessairement en termes de règles d'inférence, faisant écho en cela à des travaux en psychologie cognitive sur les modèles mentaux [Cap 2002].

L'idée sous-jacente au CBR est qu'il n'est pas nécessaire de disposer d'un modèle explicatif d'un phénomène ou d'une situation pour déduire une ligne de conduite : il suffit de tirer profit des solutions correspondant à des problèmes déjà résolus. Les applications du raisonnement à base de cas simulent les processus de remémoration et d'exploitation de situations passées [Rou 1994]. Ces systèmes exploitent une base de cas représentant diverses expériences de réalisation d'une tâche pour laquelle le système est conçu.

L'objectif de ce chapitre est d'étudier le paradigme CBR et de mettre en évidence les tâches principales et les besoins de ces systèmes en termes de connaissances et de mécanismes de raisonnement.

Le CBR est aussi situé par rapport à d'autres paradigmes de raisonnement et thèmes de recherche.

1 Historique et Origines

1.1 Historique

C'est d'abord aux Etats Unis qu'émergea le raisonnement à base de cas. La plupart des fondements de l'approche CBR ont été réalisés par Roger Schanck et ses étudiants de l'université de Yale dans les années 80. Les travaux de Roger Schank portaient sur la mémoire dynamique et l'apprentissage pour la compréhension d'histoires énoncées en langage naturel. Un des premiers systèmes CBR CYRUS développé par Janet Kolodner est un

[1] : notons que l'acronyme CBR est issue de «case based reasoning». En français l'acronyme devient RBC pour «Raisonnement Basé Cas», ou encore RàPC «raisonnement à partir de cas», mais comme il est d'utilisation moins fréquente nous avons opté pour le premier.

système qui retrace des évènements impliquant Cyrus Vance dans son rôle de secrétaire d'état. Il implante les idées de Schank c'est à dire une structure de mémoire dynamique pour la recherche d informations.

De nombreux systèmes furent ensuite développés selon le modèle de Schank mais pour des tâches différentes comme CHEF ou JUDGE ou selon d'autre modèles, par exemple un modèle de prototypes comme PROTOS **[Kol 1993]**.

Durant la fin de cette décennie la Defense Advanced Research Projects Agency (DARPA) a démarré l'application des techniques CBR. Ce qui a attiré l'attention des organisations intéressées par les applications commerciales se rapportant aux systèmes à base de connaissance. Le premier *workshop* sur le sujet a eu lieu en 1988. D'autres ont eu lieu en 1989 et en 1991. Cet axe de recherche fait en outre l'objet de *workshop* dans la conférence nationale organisée par l'AAAI (American Association of Artificial Intelligence).

En Europe, les premiers travaux datent de 1989. Le premier *workshop* a eu lieu à Kaiserslautern, en Allemagne en 1993. Actuellement, il existe deux projets importants sur ce thème **[Bar 1999]** : le projet ESPRIT INRECA (Integrated Reasoning from Cases) supporté par la communauté européenne et portant sur l'élaboration de méthodes et boîte à outils pour développer, valider et maintenir des systèmes d'aide à la décision; le second projet FABEL, est à l'initiative du ministère Allemand de la recherche et de la technologie et porte sur la réalisation d'un prototype de recherche basée sur des cas de dessins techniques. Notons qu'en France, une communauté CBR commence petit à petit à ce mettre en place (on peut citer quelques laboratoires ayant inscrit le CBR à l'ordre du jour : CRIN, INRIA, LAFORIA, LIFIA, LRI).

En Algérie, ce paradigme reste encore très mal connu (si ce n'est totalement inconnu). L'intérêt naissant pour cette approche rencontre un obstacle fréquent : la confusion du CBR avec d'autres approches (raisonnement par analogie, réutilisation, apprentissage symbolique...). Nous essayerons de dissiper cette confusion par la suite.

1.2 Origines

Les origines du raisonnement à partir de cas sont la psychologie cognitive pour l'étude de la mémoire et le raisonnement par analogie. Le raisonnement à partir de cas complète le raisonnement par analogie par un mécanisme de mémorisation et d'extraction des expériences. C'est aussi une analogie intra-domaine, conçue pour une tâche bien précise. Par les mécanismes qu'il met en oeuvre, notamment ceux liés à l'exploitation de la base de cas, le raisonnement à partir de cas entretient aussi des relations avec d'autres formes de raisonnement, comme la classification et la catégorisation **[Rou 1994]**.

1.2.1 Théorie de la mémoire

La résolution de problème dans un style CBR fait écho à la position radicale de Minsky **[Cap 2002]** selon laquelle l'approche logique ne peut fonctionner en I.A. En résumant son point de vue, Minsky déclare que le raisonnement logique (déductif) n'est pas assez flexible pour servir de base à la pensée, que la rigueur imposée par la logique formelle n'est pas souhaitable, et qu'en conséquence, les systèmes conçus sont beaucoup trop rigides. Il leur préfère la notion de « frames » qui sont des entités regroupant de façon structurée l'ensemble des informations relatives à un objet ou une situation stéréo typiques. Les « frames » sont reliés entre eux en réseaux sémantiques et c'est ce réseau qui sera exploité/parcouru pour trouver le « frame » le plus adapté à une situation observée. Quand un « frame » est proposé, un processus de mise en correspondance (pattern matching) remplit les informations manquantes. Des facettes particulières (exceptions, valeurs par défaut) complètent le dispositif.

Dans la même mouvance on peut citer les « scripts », introduit par Schank et Abelson **[Kol 1993]**, dont les travaux portaient sur la compréhension de texte énoncé en langage naturel. L'hypothèse fondamentale est que la compréhension d'histoire s'appuie sur des schémas mentaux qui comblent les vides 'présents' naturellement dans les textes. Un « script » décrit un épisode selon un comportement conventionnel découpé en événement qui se succèdent selon un ordonnancement décrit par l'ordre d'apparition attendu de ces événements. Il contient un certain nombre de facettes (but poursuivi, plan relation sociale, rôle tenu… et de manière générale tout ce qui peut jouer sur le comportement dans la mise en

oeuvre du schéma). Tout comme les « frames », ces « scripts » sont unifiés à la situation courante.

Plusieurs théories de la mémoire ont successivement dominé. La dernière étant la théorie de la mémoire dynamique de Schanck. Elle a donné lieu aux premiers systèmes de raisonnement à partir de cas **[Rou 1994]**, **[Kol 1993]**.

Ce qu'il y a lieu de préciser est que la théorie de la mémoire dynamique fournit un guide pour la ***représentation***, la ***segmentation***, l'***accès*** et le ***raisonnement*** sur les «cas».

1.2.2 Le Raisonnement par Analogie

Par définition, l'analogie désigne **«rapport, similitude partielle d'une chose avec une autre»** **[Rou 1994]**. De nombreux exemples d'analogie existent dans notre entourage: l'analogie atome/système solaire, l'analogie courant électrique/cours d'eau. Le raisonnement par analogie consiste à avoir recours à un élément mieux connu pour inférer des informations sur un élément qui l'est moins. Il implique notamment l'évaluation de la ressemblance entre entités et leur mémorisation en vue de leur réutilisation.

Finalités du raisonnement par analogie.

Le raisonnement par analogie est reconnu être très utilisé par l'être humain. Dans la vie quotidienne, face à une situation donnée, l'expérience d'une situation semblable peut s'avérer très utile. En intelligence artificielle, les systèmes mettent en oeuvre ce raisonnement à diverses fins: pour la compréhension du langage naturel, la planification, etc.

Le raisonnement par analogie a pour objectif l'inférence d'informations sur une situation (la cible) à partir de la description d'une situation dans laquelle ces informations sont connues (la source).

Les deux composantes essentielles de l'analogie sont la mise en évidence des caractéristiques ou propriétés communes des situations et la détermination des relations intra- ou inter-domaines. En raisonnement par analogie, la ressemblance entre situations est basée souvent sur des critères syntaxiques (le raisonnement à partir de cas au contraire compare plutôt des ensembles de descripteurs).

Il existe principalement deux contextes d'utilisation du raisonnement par analogie notamment dans le cadre de la résolution de problèmes:

- la résolution du problème s'annonce particulièrement complexe et longue. L'utilisation de la solution d'un problème similaire déjà résolu permet d'accélérer le processus de résolution,
- dans le domaine considéré, il n'existe pas de théorie qui permette de résoudre le problème posé. L'utilisation d'un problème similaire résolu s'avère être la seule issue.

La première analogie est qualifiée d' «heuristique», et la seconde d' «analogie-recours».

Formalisation et terminologie.

Le «paradigme d'analogie» définit l'analogie comme la mise en oeuvre d'un mécanisme de mise en correspondance ou de projection, entre des structures afin de transposer des connaissances d'un univers[2] de base vers un univers cible (**[Rou 1994]**, **[Sma 1994]**), en fonction d'un certain point de vue pouvant correspondre à un but à atteindre ou un problème à résoudre.

Il manipule deux entités appartenant chacune à un univers pouvant être réorganisé selon le point de vue adopté. L'une des deux entités est utilisée pour inférer des connaissances sur la seconde, elle appartient à l'univers de base, l'autre appartient à l'univers cible. Dans la description des mécanismes du raisonnement par analogie, ces deux entités sont appelées simplement «base»et «cible».

Un problème d'analogie s'exprime selon l'expression : **«D est à C ce que B est à A. Connaissant A, B et C, que vaut D?»**. Le principe d'analogie est souvent schématisé par **«le carré d'analogie»** :

[2] : Le terme «univers», quelquefois remplacé par celui de «domaine», désigne l'ensemble des connaissances relatives à un domaine donné. Il contient des éléments de description atomiques ou structurés et un ensemble de connaissances permettant d'inférer de nouvelles connaissances.

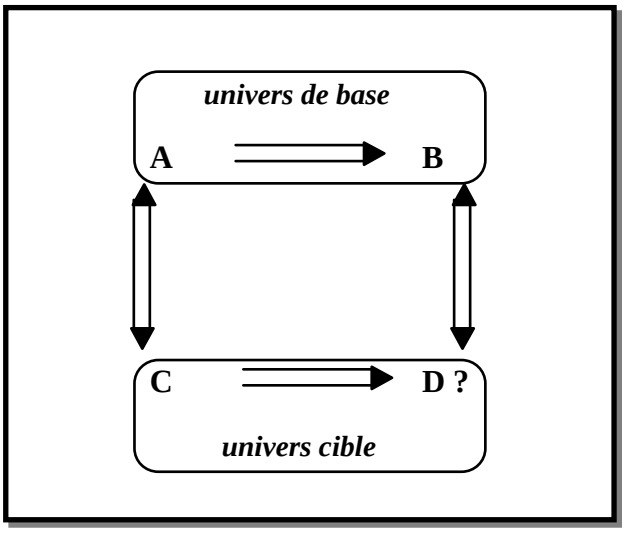


Fig.I.1. Le carré d'analogie

Appliqué à la résolution de problème, ce principe a donné lieu à une définition plus précise [Sma 1994] : **«La résolution de problèmes par analogie consiste à transférer de la connaissance à partir d'épisodes précédents de résolution, aux nouveaux problèmes qui partagent des aspects significatifs de l'expérience précédente et à utiliser les connaissances transférées pour construire des solutions pour les nouveaux problèmes»**.

Par rapport au carré d'analogie, A et B définissent un épisode de résolution, par un problème (A) et sa solution (B). C représente le problème à résoudre et D, la solution recherchée. Carbonell propose deux approches de résolution de problème par analogie (citées dans [Sma 1994]) :

- ***L'analogie par transformation*** tente de réutiliser, avec des modifications mineures, une solution précédemment trouvée pour un problème similaire. Si **d1** représente la différence entre A et C, il s'agit de déterminer **d2**, la différence entre B et D afin d'en déduire D en propageant **d2** dans B. Le carré d'analogie devient alors :

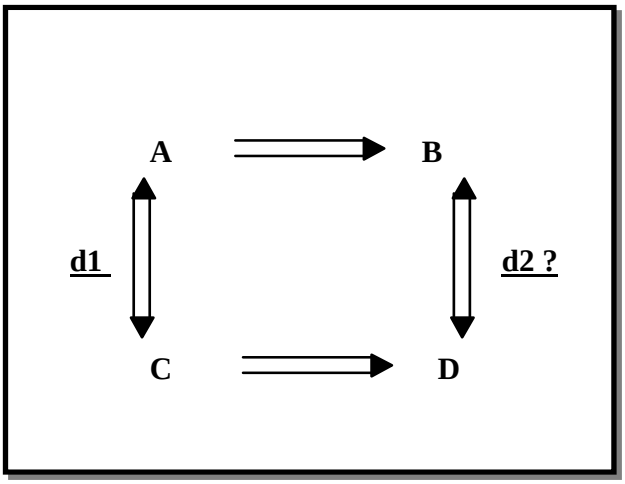


Fig.I.2. L'analogie par transformation.

- ***L'analogie par dérivation :*** il s'agit d'adapter le processus (ou la dérivation) ayant abouti à la solution d'un problème précédent pour construire la solution d'un problème similaire. Si *S* est la similarité entre A et C et *P* la méthode de construction de la solution B à partir du problème A. La résolution du problème C se fait alors par le calcul d'une méthode *P'* à partir de *P* en utilisant *S* (*S* sert à reconnaître les éléments de *P* qui sont encore valables dans la nouvelle situation C) :

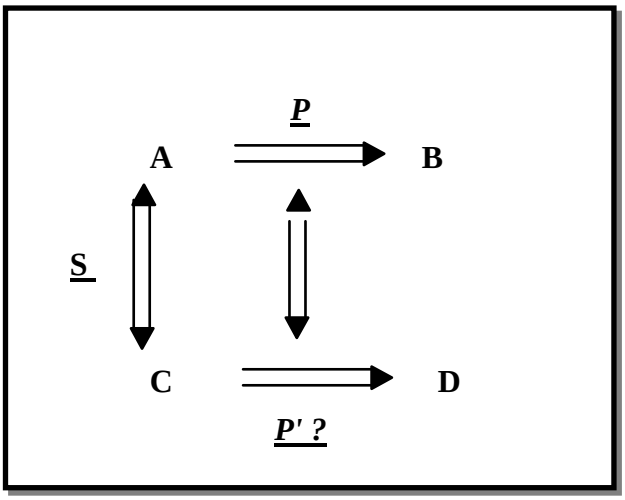


Fig.I.3. L'analogie par dérivation.

Modélisation de l'analogie.

La connaissance actuelle des processus cognitifs de l'analogie permet d'élaborer un modèle conceptuel du raisonnement analogique regroupant l'ensemble des opérations mises en oeuvre, en vue de la réalisation pratique d'un système. Un tel modèle correspond à la succession de quatre étapes **[Hat 1991]**:

- recherche et identification d'une source d'analogie en fonction de la description d'un but à atteindre. Le problème consiste à extraire d'un ensemble de sources mémorisées une ou plusieurs candidates qui semblent prometteuses. Plus généralement, il s'agit de prévoir une organisation de la mémoire des sources en fonction de schémas d'indexation de nature variée.

- élaboration de la mise en correspondance analogique entre la source et le but, et des inférences nécessaires. La difficulté majeure réside dans le fait que l'espace des mises en correspondance est en général de très grande taille et qu'il faut le restreindre à un sous-ensemble de solutions plausibles en utilisant différents types de contraintes.

- calcul et évaluation de la mise en correspondance et des inférences retenues. Il s'agit de calculer l'analogie retenue dans un certain contexte d'utilisation. Cette étape est parfois concomitante à la précédente. Elle intervient à plusieurs niveaux. Elle permet en effet de comparer les prédictions de l'analogie aux résultats typiques du domaine cible, d'adapter et d'améliorer des inférences analogiques inappropriées.

- consolidation et mémorisation de l'analogie élaborée. Les trois étapes précédentes permettent de mettre au point une analogie satisfaisante. Cette quatrième s'intéresse à la mémorisation de la solution obtenue de façon à permettre sa réutilisation dans des contextes similaires. Il s'agit donc du champ de l'apprentissage par analogie, étroitement lié au raisonnement.

La recherche éventuelle de la base qui sera utilisée pour le raisonnement, et la mise en forme préalable des univers en fonction de l'objectif du raisonnement, constituent des sources de variation de ce modèle. De même que l'élaboration des relations inter et intra univers **[Rou 1994]**.

Le raisonnement par cas est une forme de raisonnement par analogie.

Le raisonnement à base de cas est une approche de la résolution de problème basée sur la recherche et l'adaptation de cas, ou les descriptions épisodiques des problèmes et des solutions qui leurs sont associées.

Le raisonnement à base de cas propose une méthodologie pour le traitement de connaissances empiriques **[Wat 1999]**. Par définition, «un cas » est un ensemble de données empiriques **[Bic 1994]**. Un système CBR utilise un ou plusieurs cas déjà rencontrés pour le traitement d'un nouveau cas qui se présente. Le traitement du nouveau cas peut aussi bien faire partie d'une tâche d'analyse, telle que diagnostic ou planification, ou d'une tâche de synthèse telle que apprentissage de concepts.

Un système CBR utilise une base de connaissances contenant un ensemble de cas précédemment rencontrés, cette base est enrichie chaque fois qu'un nouveau cas est traité. La base de cas (base de connaissances de cas) est similaire à une mémoire qui est organisée de sorte à rendre possible la comparaison entre le nouveau cas et les cas mémorisés.

Le CBR peut aussi signifier **[Kol 1991]**: adapter d'anciennes solutions pour satisfaire de nouvelles demandes, utiliser d'anciens cas pour expliquer de nouvelles situations, utiliser d'anciens cas pour critiquer de nouvelles solutions, raisonner sur d'anciens cas pour interpréter de nouvelles situations ou créer une solution équitable pour un nouveau problème.

2 Modélisation des connaissances en CBR

Diverses connaissances sont nécessaires pour l'exécution de chacune de ces phases : les connaissances permettant de représenter de manière adéquate le nouveau problème à résoudre, les connaissances permettant d'évaluer la ressemblance de deux cas, celles

permettant d'organiser la mémoire, ainsi que les connaissances d'adaptation de la solution proposée et les connaissances d'amélioration.

2.1. Le Cas

Un cas est une entité au sein de laquelle sont rassemblées diverses informations sur une situation passée. Le terme «situation» est très général : description de la conception d'un plan en fonction d'un certain nombre de buts, sentence prononcée pour un délit, etc. Un cas est aussi une entité à partir de laquelle un raisonnement est possible par un processus consistant à situer la nouvelle situation par rapport aux circonstances définies dans le cas.

Schank définit trois types de cas **[Rou 1994]**:

- les cas ossifiés assimilables à une règle générale, qui ne sont associés à aucune expérience vécue, les proverbes en sont un exemple,
- les cas paradigmatiques. Ces cas sont moins généraux que les premiers. Ils sont notamment reliés à un ensemble de situations,
- les histoires, qui constituent des cas uniques. Elles sont très détaillées, ce qui les rend exploitables pour diverses utilisations. Elles constituent la base du raisonnement à partir de cas. Ainsi, une «histoire» est analysée une seule fois, avant la constitution de la base de cas, en fonction de l'objectif du système.

Un cas est constitué de descripteurs, appelés aussi dimensions, répartis en trois : la description du problème, la solution et éventuellement les conséquences de son utilisation **[Kol 1993]**.

La description du problème comprend le contexte dans lequel le cas s'est produit. La solution est la solution du problème décrit dans la première partie ou la réaction à cette description (par exemple, la délibération d'un tribunal, la décision prise, etc.). Elle peut aussi décrire le raisonnement utilisé. L'issue du cas est la description du contexte après la mise en oeuvre de la solution.

La description du problème constitue l'état de l'univers lorsque le raisonnement commence. Cela peut être un problème à résoudre ou une situation à interpréter, à classer ou à comprendre. Cette description contient en général :

- Les buts à atteindre dans la résolution du problème.
- Les contraintes sur ces buts.
- Les caractéristiques de la situation et les relations entre ses parties.

La solution dans un cas peut être de différents types, selon la nature de la tâche à réaliser : une interprétation, une classification, un plan, le produit d'une conception, une explication,... Une solution complète contient :

- La solution proprement dite.
- L'ensemble des pas de raisonnement ayant permis la résolution.
- L'ensemble des justifications pour les décisions prises.
- Les solutions acceptables qui n'ont pas été choisies (solutions alternatives), accompagnées de leurs raisonnements et justifications.
- Les solutions rejetées (avec leurs raisonnements et justifications).
- Les prévisions de ce qui résulterait suite à l'application de la solution.

Il est rare que toutes ces informations figurent dans une solution. Elles sont généralement reportées comme connaissances d'adaptation ou d'évaluation/critiques.

L'issue du cas est la réaction de l'univers à l'application de la solution. On y trouve :

- Les résultats de l'application de la solution.
- Si l'issue est conforme ou non aux prévisions.
- S'il s'agit d'un échec ou d'une réussite.
- Explication de la conformité ou de la non conformité aux prévisions.
- Stratégie de réparation.
- Ce qui peut être fait pour éviter le problème.
- Un pointeur vers un autre essai de solution (résultat de la réparation).

Cette partie du cas est généralement omise et les connaissances s'y attachant sont reportées sur les autres étapes du raisonnement.

A priori, plus la représentation des cas est riche, plus les cas peuvent être utilisés pour des buts différents. Mais la construction d'une base de cas ne se conçoit pas indépendamment de la tâche que le système doit réaliser.

Les cas doivent être représentés de manière à pouvoir être discernés les uns des autres et ainsi permettre de retrouver le «bon» cas vis à vis de l'objectif à atteindre. La détermination des informations utiles aux raisonnements futurs constitue le problème de l'indexation.

Dans ce qui suit nous reprenons une modélisation formelle du cas **[Fuc 1997]**, qui tient compte de l'aspect descriptif du problème et de la solution, ainsi que des utilisations du cas au travers des tâches de raisonnement.

Un cas noté C est une connaissance spécifique représentant un épisode de résolution de problème composé de trois parties principales : le problème Pb, sa solution S et le raisonnement R (Fig.I.4).

C = (Pb, R, S),

Le problème est décrit par un ensemble de données descriptives D, et d'une requête Rq indiquant l'objectif du raisonnement, toujours connu. Les données du problème comportent des contraintes à satisfaire par la solution, et constituant une spécification partielle de S.

Pb = (D, Rq).

La solution S construite par R doit permettre de satisfaire les objectifs spécifiés par Pb. La solution à construire par R peut être un objet quelconque (un plan, par exemple).

Le raisonnement R est l'ensemble des étapes de raisonnement, les résultats intermédiaires, les décisions prises pour passer de Pb à S.

Les connaissances représentées dans les cas sont aussi bien descriptives (P,S), qu'opérationnelles (R).

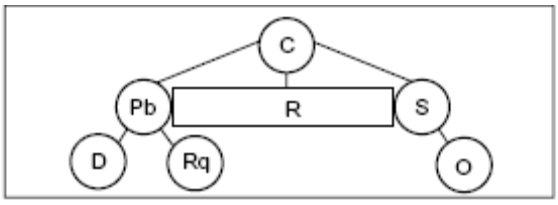


Fig I.4 Un cas C est composé du problème Pb, du raisonnement R et de la solution S.

Le problème comporte les données D et une requête Rq indiquant l'objectif du raisonnement. La solution S est un objet O construit par R.

L'application de cette définition du cas aux différentes catégories de tâches de résolution, permet d'obtenir les définitions suivantes :

- Planification :

Le problème comprend le descriptif d'un état initial et de l'état final à atteindre par la solution. Le raisonnement construit un plan permettant d'atteindre l'état final à partir de l'état initial.

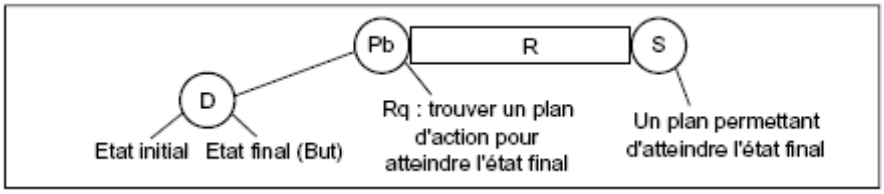


Fig.I.5. Un cas C pour la Planification.

- Conception : la conception est proche de la planification, car il s'agit également d'atteindre un état final exprimé par une spécification partielle de l'objet à concevoir, sous forme de contraintes. Le problème comprend la spécification de l'objet à construire en terme de contraintes. Le raisonnement construit un objet satisfaisant les contraintes à partir des modèles disponibles. La solution peut être un plan ou un autre objet.

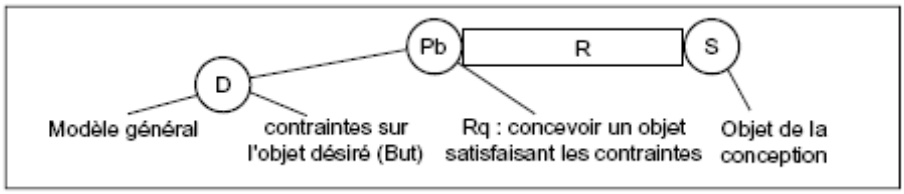


Fig.I.6 Un cas C pour la conception.

- Diagnostic :

Le problème comprend un ensemble de symptômes et un contexte. La solution est une explication des symptômes constatés.

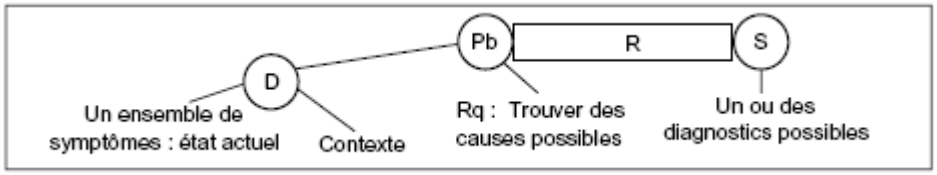


Fig.I.7. Un cas C pour le Diagnostic.

- Aide à la décision :

Le problème décrit une situation associée à un contexte, et l'objectif est de découvrir les éléments pertinents en relation avec la situation dont dépend la décision.

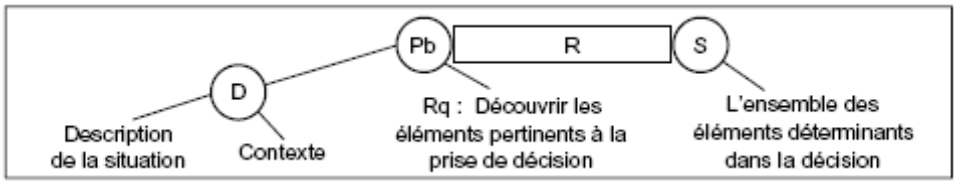


Fig.I.8. Un cas C pou l'aide à la décision.

- Interprétation :

Le problème décrit une histoire et l'objectif est de rapprocher l'histoire avec d'autres afin de prédire des évolutions possibles.

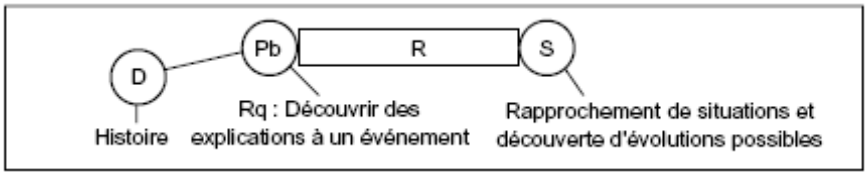


Fig.I.9. Un cas C pour l'Interprétation.

La qualité des cas

La mémoire de cas constitue le coeur du raisonnement. Par conséquent, il est primordial d'avoir une mémoire qui contient des cas de bonne qualité. Autrement dit, le contenu de la mémoire doit être pertinent pour que le fonctionnement soit le plus performant possible. Quelques heuristiques ont définies pour juger le contenu d'une mémoire donnée (**[Fuc 1997]** et **[Smy 1998]**) :

- les cas doivent couvrir la tâche de raisonnement à effectuer,
- ils doivent couvrir les situations de succès les plus communes ainsi que les situations d'échec,
- les cas doivent être collectionnés d'une manière incrémentale pour assurer la couverture du domaine.

Les cas constituant la mémoire doivent être choisis avec attention. Ils doivent être utiles, et il est préférable de ne pas avoir de redondance dans la mémoire.

Le problème de l'utilité, dans les systèmes d'apprentissage en général, apparaît quand la connaissance apprise par le système dans le but d'améliorer sa performance, a l'effet inverse, c'est-à-dire provoque une diminution de performance.

De manière générale, les cas ne sont pas tous égaux dans la mémoire. Il existe des cas qui agissent sur la compétence du système, et d'autres qui influencent sa performance.

Pour modéliser la compétence du système 4 classes de cas ont été distinguées selon les deux critères suivants :

- la couverture d'un cas : l'ensemble des problèmes cibles que ce cas résout,
- l'accessibilité d'un problème, l'ensemble des cas qui peuvent être utilisés pour générer une solution à ce problème.

Les cas sont classés de la manière suivante :

- Les cas essentiels : les cas essentiels sont les cas dont l'effacement réduit directement la compétence du système, ce sont les cas qui ne sont pas accessibles par aucun cas. Ces cas sont généralement isolés et ne peuvent pas être résolus par d'autres.
- Les cas auxiliaires : les cas auxiliaires n'influencent pas la compétence du système, un cas est auxiliaire si la couverture qu'il fournit est subsumée par la couverture de l'un de ses cas accessibles.
- Les cas « ponts » les cas ponts n'influencent pas directement la compétence, ils sont nommés ainsi car leur région de couverture fait la liaison entre des régions qui sont couvertes indépendamment par d'autres cas si des cas de ces régions sont effacés, le cas « pont » devient nécessaire.
- Les cas de support : c'est une classe spéciale des cas « ponts » qui existent en groupe. L'effacement de n'importe quel cas de ce groupe a peu d'influence sur la compétence, par contre l'effacement du groupe entier est semblable à l'effacement d'un cas essentiel.

Cette classification de cas implique un moyen pour ordonner les cas selon leur contribution à la compétence dans le but d'effacer les cas qui diminuent le moins possible cette compétence.

Les cas auxiliaires sont les moins importants, ensuite viennent les cas de support, les cas « ponts » et finalement les cas essentiels **[Fuc 1997]**.

2.2. La Base de Cas

Pour pouvoir fonctionner, le raisonnement utilise des cas stockés dans une base de cas. Celle-ci est supposée représentative de l'ensemble des problèmes susceptibles d'être posés au

système. Plus elle contient de cas, plus le cas sélectionné pour le raisonnement sera semblable au nouveau cas. La solution élaborée sera donc meilleure. Mais plus la base s'accroît, et plus les temps de calcul sont longs. C'est pourquoi les techniques d'organisation de la mémoire et les algorithmes de recherche sont particulièrement importants dans ce raisonnement. On retrouve plusieurs organisations suivant lesquelles la recherche diffère **[Kol 1993]**, **[Rou 1994]** :

L'organisation plate : dans une mémoire plate, les cas sont rangés séquentiellement dans une simple liste, un tableau ou un fichier. Ils sont retrouvés en appliquant une fonction d'appariement séquentiellement à tous les cas. Il n'y a aucune organisation particulière des cas. La recherche est très simple, ce sont en effet les heuristiques d'appariement qui font tout le travail. Comme avantage, on retrouve : La totalité de la base est scrutée; la précision de la recherche est fonction de la qualité de l'appariement et l'ajout d'un nouveau cas n'est pas coûteux. Par contre, cette organisation est coûteuse lorsque la base est trop volumineuse. Pour remédier à cet inconvénient, on utilise des variantes telles que :

- Indexation superficielle à un niveau pour réduire l'ensemble des candidats.
- Partitionnement de la base en tranches que le système peut distinguer.
- Implémentation parallèle telle que le système MBR **[Wal 1992].**

Les réseaux à caractéristiques partagées : Il s'agit de regrouper les cas présentant des similarités dans un même cluster. Les hiérarchies sont formées lorsque les clusters sont subdivisés en sous-clusters. Les méthodes de regroupement utilisées sont celles rencontrées en apprentissage automatique.

Les réseaux de discrimination : Le regroupement effectué dans les réseaux à caractéristiques partagées conduit à une discrimination en second lieu. Dans les réseaux de discrimination, c'est ce qui se passe en 1^{er} lieu : chaque noeud interne est une question qui départage selon la réponse les cas de la base. Les questions les plus importantes sont posées en premier.

Les réseaux redondants de discrimination : Ils fournissent une réponse au problème des informations manquantes. Ils organisent les cas en utilisant différents réseaux de discriminations, chacun avec un ordre différent des questions. Une recherche se fait en parallèle sur les différents réseaux. Si dans l'un des réseaux une question n'a pas de réponse,

on y abandonne la recherche. Au moins, l'un des réseaux retrouvera le cas qui s'apparie s'il existe. L'inconvénient majeur d'une telle organisation est la complexité de sa mise en oeuvre.

3 Modélisation des Tâches

En théorie, le cycle de base du CBR est à trois phases : ***«retrieve, reuse*** and ***store»***. Le système cherche un cas similaire au cas qui se présente, réutilise la solution retrouvée, puis stocke le cas courant pour une utilisation future.
Ce cycle peut être étendu à cinq étapes **[All 1994]**, **[Sma 1994]** :

1) Présentation ou spécification (presentation) : une description du problème est fournie comme entrée du système. Cette description doit se prête aisément à la comparaison entre le cas en entrée et les cas stockée en mémoire (uniformité de la représentation). Un des points clé du CBR est la recherche de cas pertinents, d'où l'importance du procédé qui va étiqueter ou indexer les cas de façon à ce qu'ils puissent être rappelés au moment opportun. Cette indexation s'appuie principalement sur l'extraction des descripteurs les plus caractéristiques du cas **[Mar 1993]**.

2) Recherche ou extraction (retrieval) : le système cherche les cas qui s'unifient le mieux à cette description (closest matching cases). Ces cas sont stockés dans une base de cas. Si la mémoire de cas est organisée selon une structure particulière, un algorithme de recherche dans la mémoire des cas décrit un parcours dans cette structure. Une phase de filtrage ou sélection est souvent effectuée lorsque l'on dispose d'indices parfaitement discriminants permettant d'éliminer un sous-ensemble des cas de la mémoire. Une mesure de similarité peut ensuite être utile pour mesurer précisément la ressemblance entre le cas courant et les cas sélectionnés et donner lieu à un classement de ces cas **[Bar 1992]**.

3) Adaptation : le système utilise le problème courant et le cas qui s'y unifie pour générer une solution à ce problème. L'adaptation constitue le deuxième point difficile (après l'indexation) lors de la conception d'un système CBR. Il faut en effet décider quel type de connaissances il est intéressant de transférer du meilleur cas remémoré. On peut effectuer une analogie transformationnelle, consistant à transformer la solution du cas remémoré pour l'adapter au cas courant ou procéder par dérivation en adaptant la méthode de génération de la solution.

Par ailleurs, la possibilité d'adapter plusieurs cas pour résoudre un problème, de façon simultanée ou en opérant plusieurs remémorations et adaptations simples aux différentes étapes de la résolution, a été jugée plus créative **[Aam94]**.

4) Validation : cette phase sous-entend la possibilité d'une évaluation de la solution proposée en la testant dans un environnement réel ou simulé. Le retour d'information, suite au test, peut alors guider, en cas d'échec de la solution proposée, un processus de réparation.

5) Mise à jour : dans le cas échéant, la solution validée est rajoutée à la base de cas pour une utilisation future. La configuration habituelle des systèmes CBR consiste à ranger systématiquement les cas en mémoire. Une mémorisation plus sélective est toutefois possible et utiliserait des critères spécifiques pour juger si le nouveau cas est utile à apprendre vis-à-vis de la mémoire de cas courante. Un cas est utile à apprendre si à partir de son contenu et en utilisant les possibilités d'adaptation, on peut atteindre un point de l'espace des solutions qui était inaccessible avant l'arrivée de ce nouveau cas.
De façon schématique :

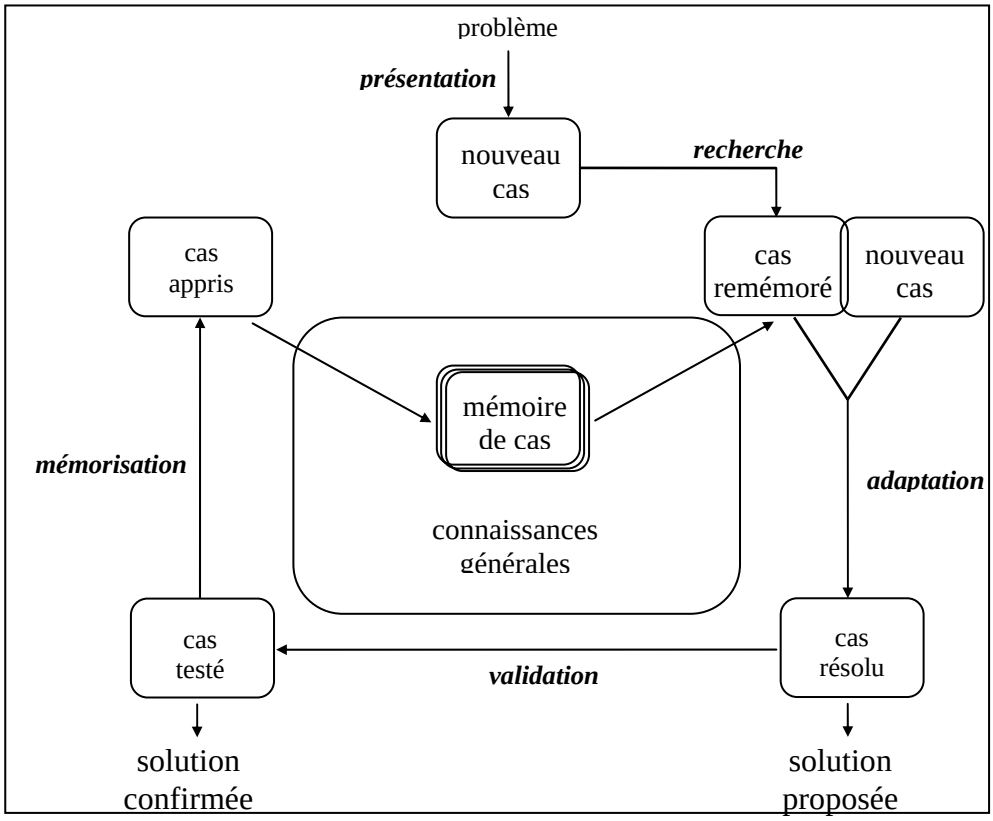


Fig .I.10. Cycle du CBR.

En pratique, le nombre d'étapes[3] peut varier largement. L'organigramme du CBR est décrit dans la figure –I.11-.

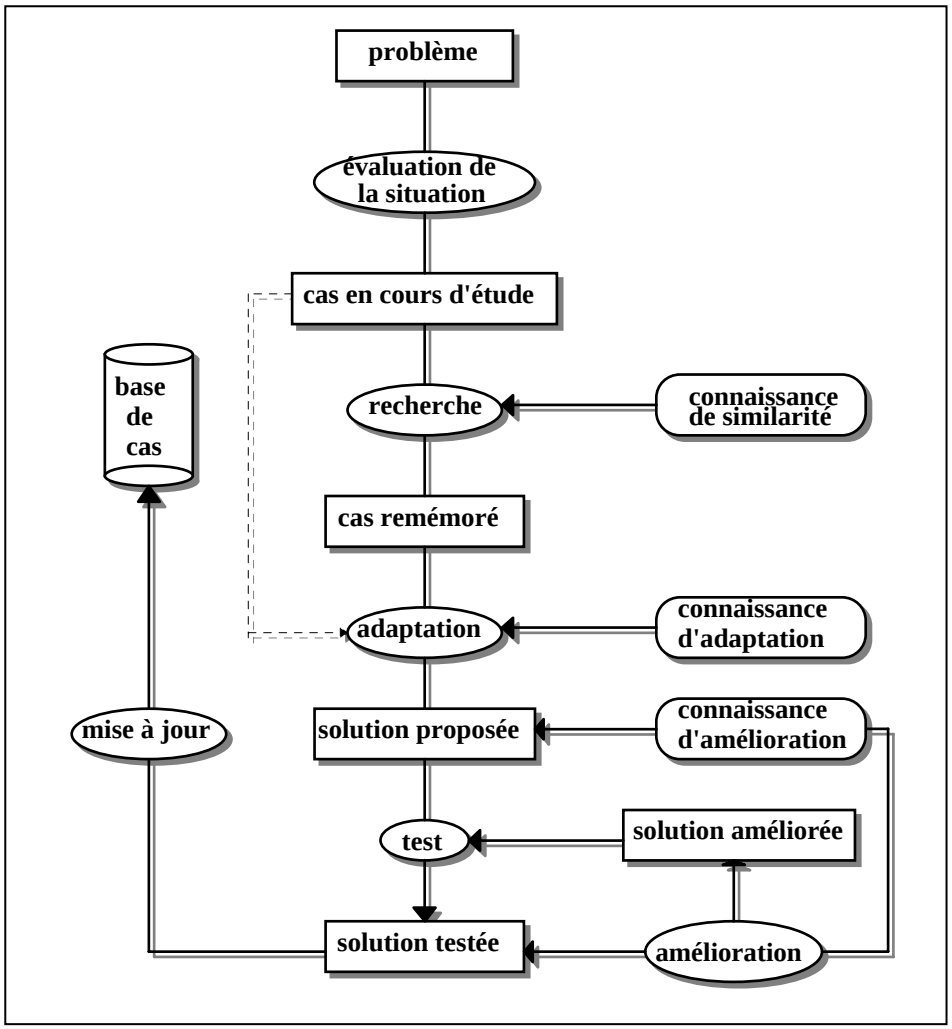


Fig.I.11. Organigramme du raisonnement à partir de cas

Ce schéma représente le raisonnement à partir de cas le plus complet. En effet, le système teste lui-même la solution et l'améliore si nécessaire. Il décrit un exemple de raisonnement utilisé dans le cadre de la résolution de problèmes. Un système dédié à l'interprétation de situations ne comporte pas les phases de proposition de la solution, et donc d'évaluation et d'adaptation de la solution proposée.

[3] Ce schéma peut varier en fonction de la tâche et également du domaine d'application pour lesquels le système est conçu.

La première étape consiste à fournir au système le problème à résoudre. Celui-ci peut être décrit *a priori* sous n'importe quelle forme. Une phase d'évaluation de la situation, ou problème à résoudre, modifie la représentation du problème posé de manière à ce qu'il puisse être comparé aux cas de la base de cas. Ensuite commence la recherche du meilleur cas à utiliser pour résoudre le problème posé. Celle-ci est réalisée en deux phases. La première consiste en l'exploration de la base de cas et l'évaluation de la similarité des cas «remémorés» avec le nouveau problème, que nous appellerons également «cas en cours d'étude». La deuxième étape, la phase de classement, permet de choisir le meilleur.

La solution de ce cas est alors utilisée pour construire la solution recherchée. Elle est adaptée au nouveau problème, puis évaluée. Si elle ne convient pas, elle peut être modifiée.

Le nouveau problème résolu peut à son tour être utilisé pour un raisonnement ultérieur du système. C'est pourquoi il est également mémorisé dans la base de cas. Ceci constitue l'apprentissage de base d'un système de raisonnement à partir de cas.

Le modèle de spécification des tâches :

Le CBR peut être vu sous un autre angle lorsque l'on s'intéresse au flux d'informations inter tâches, les informations utilisées pour contrôler les traitements et les mécanismes qui supportent les tâches (figure I.12). Une tâche est décrite par les informations qu'elle traite en entrée, une étiquette précisant les fonctions réalisées par la tâche, les informations qu'elle produit en sortie, les modèles de connaissances utilisés, et des mécanismes de raisonnement ou d'inférence mis en œuvre.

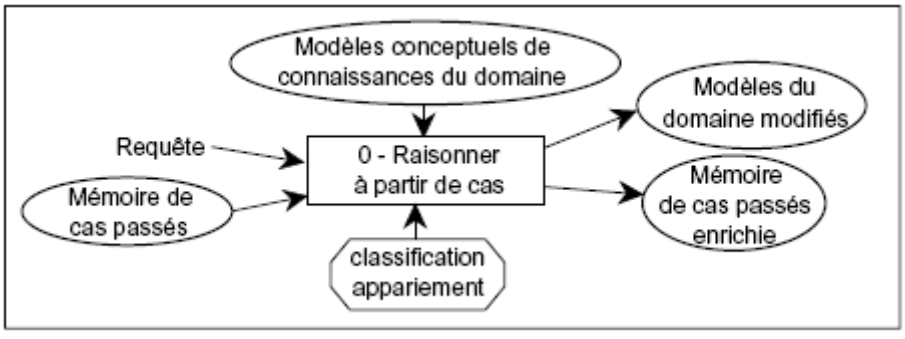


Fig.I.12.La spécification (incomplète) de la tâche principale Raisonner à partir de cas.

La tâche est activée par une requête formulant un problème à résoudre et fournit en sortie un cas contenant une solution au problème. Elle utilise des modèles conceptuels de

connaissances du domaine d'application et une mémoire de cas passés qu'elle enrichit par apprentissage. Elle utilise des mécanismes de classification et d'appariement.

Le modèle de décomposition de tâches

Si ce premier modèle traduit bien l'ensemble des éléments agissant sur la réalisation d'une tâche, il ne permet pas facilement de mettre en évidence les décompositions de tâches et leur ordonnancement. Le second formalisme hiérarchique décrit une décomposition de tâches en sous tâches (figure .I.13.). La tâche principale la plus abstraite est située à la racine de la hiérarchie, et reliée à ses sous tâches situées plus bas. Les arcs matérialisant les relations de composition de tâches sont annotés à l'aide de trois symboles permettant de préciser la nature de la relation entre la tâche principale et ses sous tâches **[Aam 1994], [Fuc 1997]**.

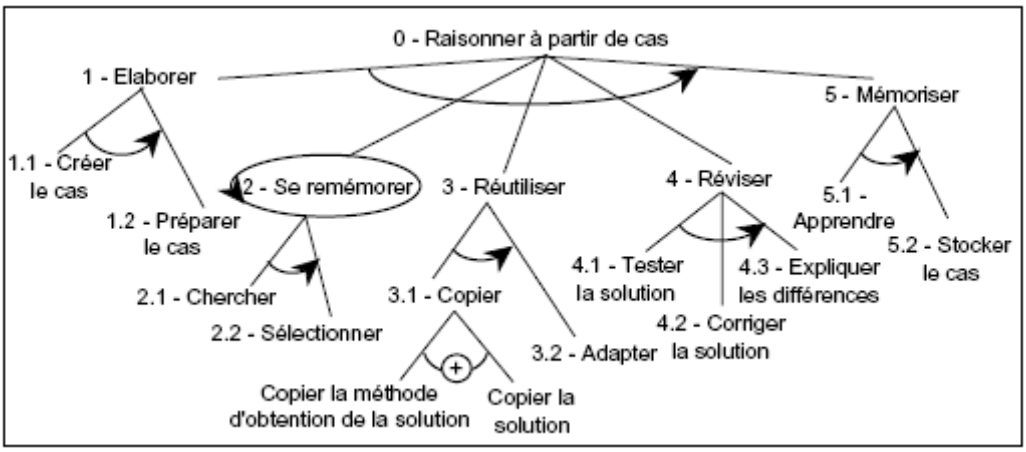


Fig.I.13. La décomposition (incomplète) de la tâche Raisonner à partir de cas.

La tâche principale Raisonner à partir de cas est réalisée par les sous tâches Elaborer, se remémorer, réutiliser, Réviser puis Mémoriser en séquence. La tâche se remémorer est répétitive, et la tâche Copier peut être réalisée par l'une ou l'autre des sous tâches Copier la méthode d'obtention de la solution ou Copie la solution. L'absence d'annotation indique un ordre non défini (en séquence ou en parallèle).

La formalisation utilisée permet de définir plus précisément les rôles joués par les connaissances : entrées, sorties, contrôle, mécanique du raisonnement, et a pour objectif de mettre en évidence la transformation d'un cas dans les différentes étapes de raisonnement.

3.1. Elaboration du problème

La plupart des modèles de CBR ne comprennent pas de tâche d'élaboration du cas qui est néanmoins toujours implicite. En effet, avant d'effectuer la remémoration d'un cas, il est

nécessaire de créer un nouveau cas, de collecter des informations afin de décrire le problème, de déterminer les indices qui permettront de rechercher un cas dans la mémoire des cas passés.

La tâche d'élaboration est déclenchée par un agent extérieur (un utilisateur, un logiciel), et a pour point de départ une requête qui donne les premières informations connues et contextuelles sur le problème à résoudre. Un nouveau cas est crée et la description du problème est ensuite complétée en collectant d'autres informations. Puis sont déterminés les indices à utiliser pour remémorer un cas passé (figure I.14)

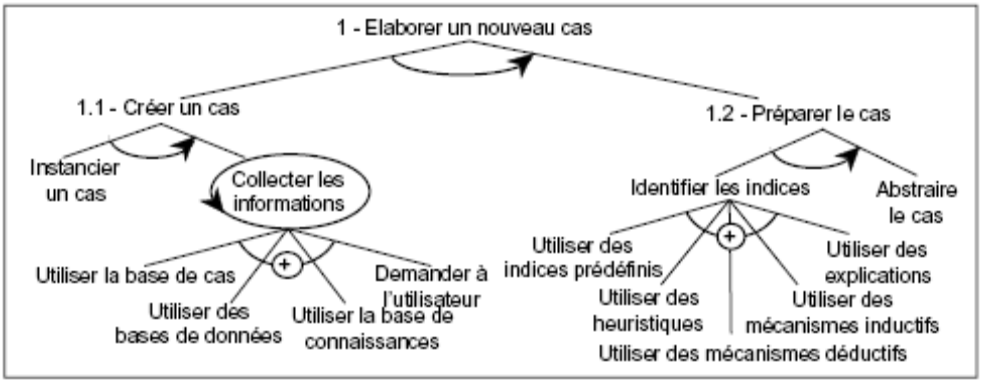


Fig.I.14. Décomposition de la tâche <Elaborer un nouveau cas>.

La tâche <Elaborer un nouveau cas> est composée de deux sous tâches principales. La première sous tâche crée un nouveau cas, et la seconde prépare le cas en évaluant quels sont les meilleurs indices à utiliser pour la tâche remémoration.

La tâche Créer un nouveau cas est composée de deux sous tâches. La première instancie le modèle de cas, et la deuxième collecte les informations pour décrire le problème.
La deuxième sous tâche de la tâche d'élaboration consiste à choisir les indices permettant de diriger la recherche de cas. Ceci permet d'éliminer des attributs peu pertinents qui risquent de bruiter la recherche.

3.2. Remémoration

La remémoration choisit un cas ayant le plus de chances de faciliter la résolution du problème. La tâche de remémoration choisit dans la mémoire des cas passés un cas similaire au problème courant. Dans de nombreux systèmes, cette tâche exploite un modèle

d'indexation pour identifier les points d'entrée dans la mémoire des cas passés, des modèles de similarité et des modèles du domaine afin d'évaluer la ressemblance (figure I.15).

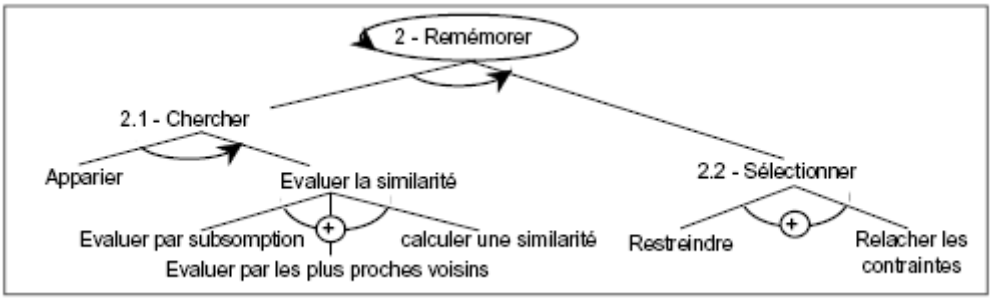


Fig.I.15. Décomposition de la tâche <se remémorer>

La tâche de remémoration procède à la sélection progressive d'un sous ensemble de cas, en commençant avec la mémoire de cas intégrale jusqu'à obtenir un seul cas dont la solution sera réutilisée pour le problème courant. La tâche de remémoration est répétitive et à chaque étape le sous ensemble de cas sélectionné est raffiné à l'aide de critères plus restrictifs. Chaque étape de raffinement commence par une recherche qui consiste en un appariement suivi d'une évaluation permettant de chiffrer la similarité. Les cas sont alors ordonnés en fonction de l'évaluation de similarité. La sélection choisit un ou plusieurs cas en fonction de cet ordonnancement.

La tâche de recherche peut consister à parcourir une hiérarchie d'index et à comparer chaque noeud avec les indices. S'il y a appariement, le sous arbre est alors sélectionné et ses descendants sont comparés de la même manière. L'ensemble des cas sélectionnés est raffiné au fur et à mesure de l'évolution dans la hiérarchie. Pour le calcul de similarité, le processus est généralement réalisé en une seule phase. Une autre forme de recherche appelée propagation d'activation consiste, à partir des caractéristiques du nouveau cas, à parcourir des relations vers des connaissances (connaissances du domaine ou connaissances d'adaptation par exemple) et en activant des cas atteints par différentes relations à la fois. Nous verrons un exemple de cela en chapitre III.

Dans la plupart des systèmes CBR, la remémoration est réalisée en deux phases : dans un premier temps, les cas sont filtrés (on parle aussi de discrimination), puis dans un deuxième temps la similarité des cas sélectionnés est évaluée.

3.3. Réutilisation

La solution du cas remémoré est réutilisée pour le nouveau problème moyennant généralement des adaptations.

L'adaptation est indispensable car le cas remémoré n'est jamais strictement identique au nouveau cas. Elle consiste à modifier la solution du cas remémoré pour prendre en compte les différences entre spécifications de problèmes. Généralement, la solution du cas remémoré est copiée et certaines parties sont substituées en fonction des spécificités du nouveau problème. Beaucoup de systèmes se contentent d'une recopie simple de la solution du cas remémoré ou d'une composition des solutions de plusieurs cas remémorés dans différents cycles de raisonnement distincts. L'adaptation ne change pas radicalement la solution mais en modifie certaines parties et peut la réorganiser.

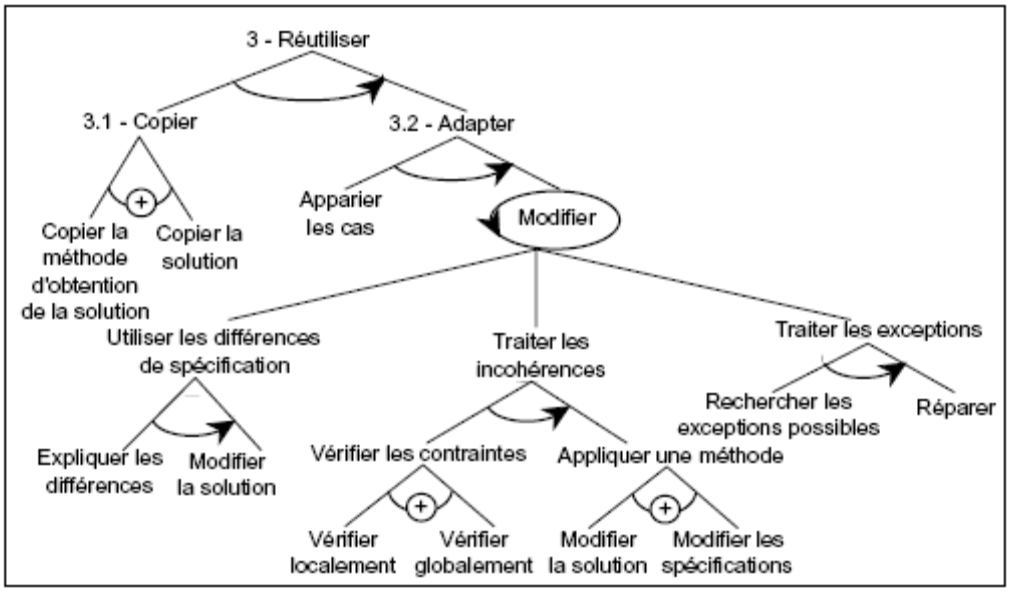


Fig.I.16. Décomposition de la tâche <Réutiliser>

La tâche <Réutiliser> (figure.I.16.) commence par copier soit la solution, soit la méthode d'obtention de la solution du cas remémoré, puis procède à l'adaptation. Les systèmes distinguent généralement l'adaptation par transformation et l'adaptation par dérivation. Dans le premier cas, c'est la solution du cas elle même qui est réutilisée, alors que dans le deuxième cas c'est la trace du raisonnement ayant produit la solution du cas qui est réutilisée. L'adaptation par dérivation recrée complètement la solution pour le nouveau problème en ré appliquant, sur les données du nouveau problème, le raisonnement qui avait produit la solution du cas passé.

La tâche adapter procède à un appariement (lorsque ça n'est pas déjà fait durant la remémoration) puis, modifie la solution en se focalisant sur les différences entre spécifications de problèmes pour déterminer quels éléments de solution doivent être modifiés

et quelles méthodes d'adaptation appliquer sur la solution. La tâche modifier contrôle la cohérence de la solution à la suite des modifications. Des connaissances d'adaptation spécifiques permettent de réaliser des ajustements supplémentaires afin d'obtenir une solution cohérente vis à vis de la spécification du problème ou des connaissances disponibles. Lorsqu'il y a trop de contraintes, l'espace des solutions est vide et il est nécessaire de relâcher des contraintes afin de trouver des solutions.

La tâche <Traiter les exceptions> anticipe des échecs potentiels lors de la mise en œuvre. Les exceptions passées rencontrées sont consultées dans le cas remémoré en parcourant des index spécifiques. Si cette opération n'a pas déjà été réalisée durant la phase de remémoration.

3.4. Révision

La solution obtenue après adaptation est proposée. Différentes sources permettent d'obtenir des retours sur le succès ou l'échec de cette solution. D'une part la mise en oeuvre de la solution permet de détecter des échecs, et d'autre part sa soumission à un utilisateur lui permet d'apporter des corrections.

La révision est souvent considérée comme la continuation de l'adaptation. Nous avons considéré la tâche d'adaptation comme l'ensemble des tâches modifiant la solution avant sa mise en œuvre. La tâche de révision prend en compte les retours après la mise en oeuvre de la solution permettant de détecter des cas d'échecs non connus auparavant, et prépare à l'apprentissage.

La révision permet d'effectuer des corrections de la solution et d'exploiter les différences entre la solution avant et après correction (figure I.17).

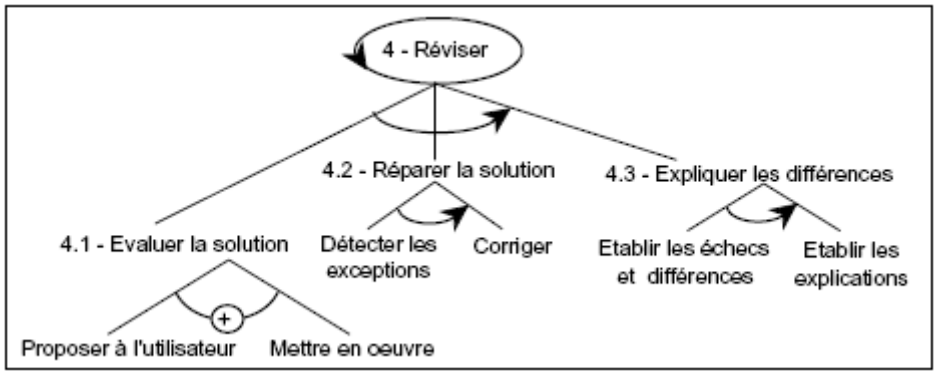


Fig.I.17. Décomposition de la tâche <Réviser>.

Les différences observées entre la solution après adaptation et la solution après révision peuvent être considérées comme des échecs de raisonnement. Ces échecs sont généralement dus au fait que le processus d'adaptation ne possède pas toutes les connaissances lui permettant de vérifier la validité de la solution. Ces échecs peuvent servir de prétexte pour lancer un processus d'apprentissage afin d'acquérir de nouvelles connaissances, à condition d'identifier la cause de l'échec. L'identification d'une cause d'échec peut être typiquement réalisée grâce à un processus de diagnostic qui recherche des explications possibles de l'échec constaté en exploitant un modèle causal (s'il existe). L'explication peut également être simplement demandée à un utilisateur.

Les explications sont ensuite exploitées par la tâche d'apprentissage pour acquérir de nouvelles connaissances afin d'éviter des échecs ultérieurs et améliorer les compétences de raisonnement.

3.5. Mémorisation

Un cas résolu est stocké s'il apporte de nouvelles compétences au système, afin de pouvoir être remémoré ultérieurement. Au fur et à mesure que de nouveaux cas sont rencontrés et résolus, ils sont ajoutés dans la mémoire des cas, et les index sont mis à jour.

La tâche mémoriser comporte la composante d'apprentissage proprement dite du CBR. La figure .I18. modélise deux tâches d'apprentissage en distinguant d'une part l'exploitation des résultats de la tâche de révision afin de modifier les modèles de connaissances pour tenir compte des échecs, et d'autre part le stockage du cas dans la mémoire

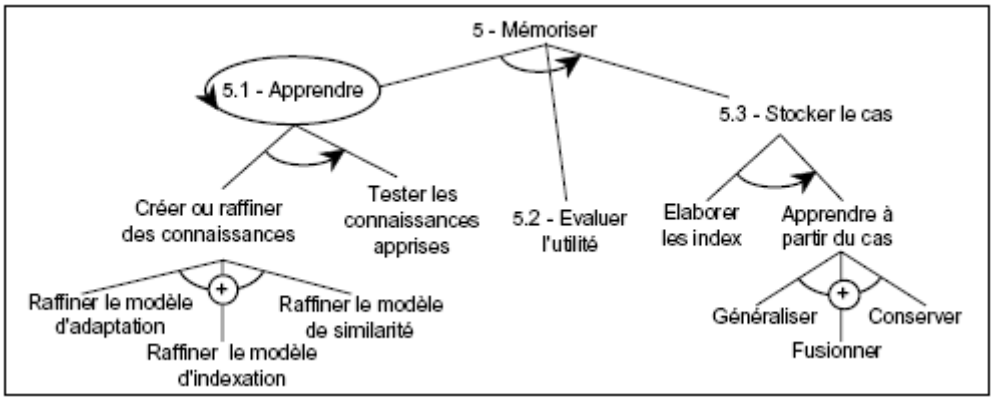


Fig.I.18. Décomposition de la tâche <Mémoriser>.

La composante d'apprentissage est fondamentale en CBR car elle permet au système d'évoluer au fur à mesure que de nouveaux cas sont acquis. L'apprentissage dans le CBR couvre différents aspects. Il consiste d'abord à acquérir de nouveaux cas résolus et à mettre en place les index appropriés pour permettre leur remémoration ultérieure. La mise en place d'index permet de prendre en compte les échecs rencontrés par la solution et les stratégies utilisées pour les corriger. La formation de nouveaux concepts, ou l'affinement de concepts existants par des méthodes de généralisation permet d'enrichir les connaissances du domaine. L'introspection permet de diagnostiquer des erreurs d'indexation ou des difficultés d'adaptation, et de proposer des modifications de modèles pour améliorer le raisonnement.

L'apprentissage est souvent déclenché lorsque des échecs sur les résultats prévus sont apparus lors de la mise en oeuvre de la solution. Le principe de base est qu'une erreur commise ne doit pas être répétée, et des index spécifiques sont mis en place afin de remémorer, en même temps que le cas, des informations sur les échecs potentiels.

4 Styles du CBR

Il existe deux styles du CBR (**[Kol 1991]**, **[Kol 1992]**, **[Ash 1988]** et **[Ris 1989]**): le style résolution de problème et le style interprétation.

Dans la résolution de problèmes, on adapte d'anciennes solutions à de nouveaux problèmes : les anciennes solutions peuvent fournir des solutions quasi-correctes ou prévenir contre des erreurs potentielles et des échecs. Dans ce style le CBR peut supporter une variété de tâches de résolution de problèmes telles que : planification, diagnostic, conception....

Dans le style résolution de problème, une solution au nouveau problème est proposée à la base de la solution correspondant au cas extrait. Cette étape est suivie d'une adaptation, ajustement de l'ancienne solution à la nouvelle situation; puis une étape de critique: processus d'évaluation de la nouvelle solution.

Dans le style interprétatif, les cas sont utilisés pour évaluer ou justifier de nouvelles situations, tel un juriste utilisant des cas précédents comme argument pour classer de nouvelles situations. Il y a évaluation même lorsque aucune méthode claire et tranchante n'est disponible. On peut aussi interpréter des situations dont les frontières sont floues ou ouvertes-

finies (open_ended). En style interprétatif, une interprétation de base (ou un résultat désiré) est proposée, parfois basée sur les extraits, parfois imposées par l'extérieur (ex: lorsqu'un client exige de son avocat un certain résultat). Cette étape est suivie par la justification, processus de création d'un argument pour la solution proposée; ce qui est réalisé par comparaison et opposition de la nouvelle situation aux cas antérieurs. Et enfin, la critique: processus de mise au point (déboggage), réalisé par la génération de situations hypothétiques et le test de l'argument sur ces situations.

Ces étapes, sont dans un sens, récursives. La critique et l'adaptation par exemple, requièrent l'extraction de nouveaux cas. Par ailleurs, il y a plusieurs boucles dans le processus. La critique peut mener à une adaptation additionnelle et peut être même à une réévaluation. De plus, lorsque le raisonnement ne progresse pas en utilisant un cas, il peut être reinitialisé entièrement avec un nouveau cas.

5 Méthodes de construction des systèmes CBR

Des méthodologies guidant le développement des systèmes CBR sont apparues. Elles partagent les trois phases suivantes **[All 1994]**, **[Ber 1998]** et **[Bar 1999]**:

5.1 Conception de la Base de cas

On développe une représentation générale des cas, en utilisant les sources matérielles d'information (documents, enregistrement de bases de données, notes de l'expert...). Ceci étant accompli en coordonnant les efforts entre utilisateur, gestionnaire et concepteur du système. La conception de la base de cas peut mener à la constitution d'un lexique de termes utilisé pour décrire les caractéristiques du problème, la sélection des caractéristiques appropriées pour l'indexation des cas, la spécification de schémas de bases de données utilisées pour le stockage des cas (ainsi que la définition de standards d'écriture de la base de cas).

5.2 Développement initial de la Base de cas

Une base initiale est développée pour fournir les grandes lignes de l'application. Elle sera par la suite, revue en extension par le concepteur et l'utilisateur et itérativement raffinée jusqu'à ce qu'elle couvre une large portion du domaine d'application.

5.3 Développement continu et maintenance

La base initiale est utilisée et raffinée à travers les étapes de validation et stockage. Une partie de l'organisation gère la base de cas ainsi que la représentation de cas de la même façon qu'une base de données traditionnelle. Des techniques de contrôle de qualité statistiques peuvent être utilisés pour connaître la précision et l'utilité des cas.

Un environnement typique de développement de CBR fournit: des schémas par défauts de représentation de cas, un flot prédéfini de résolution de problèmes support de décisions. Des formats sont utilisés pour l'édition de cas, de caractéristiques et de solutions. Des utilitaires sont fournis pour l'indexation manuelle et automatique, l'importation automatique de cas à partir des enregistrements d'une table de BD relationnelle ainsi que le groupage conceptuel (conceptuel clustering) de cas pour l'analyse et la recherche efficace.

6 Situation du CBR

Malgré l'émergence du raisonnement à base de cas en tant que paradigme à part entière, il est souvent confondu avec d'autres paradigmes.

Au cours de notre recherche, nous avons souvent rencontré des réactions du style : «***Ceci est de la réutilisation pure et simple!***» ou «***c'est de l'apprentissage automatique!***» ou encore «***c'est du raisonement par analogie***».

Le CBR entretient certes, d'étroites relations avec ces concepts, mais nous ne pouvons et ne devons le confondre avec l'un deux sans connaitre les limites. L'objectif de cette section est justement de situer le raisonnement à base de cas par rapport à ces concepts.

6.1 CBR et Analogie

Bien que le raisonnement par analogie fût à l'origine du CBR, ce dernier s'en est distingué de par le développement des technique de remémoration inspirées des théories de la mémoire.

En réalité, le CBR peut être considéré comme une forme particulière du raisonnement par analogie. Ce dernier infère des connaissances sur une entité appartenant à un univers cible, en important des connaissances sur une entité appartenant à un univers source. En CBR, les univers source et cible sont confondus.

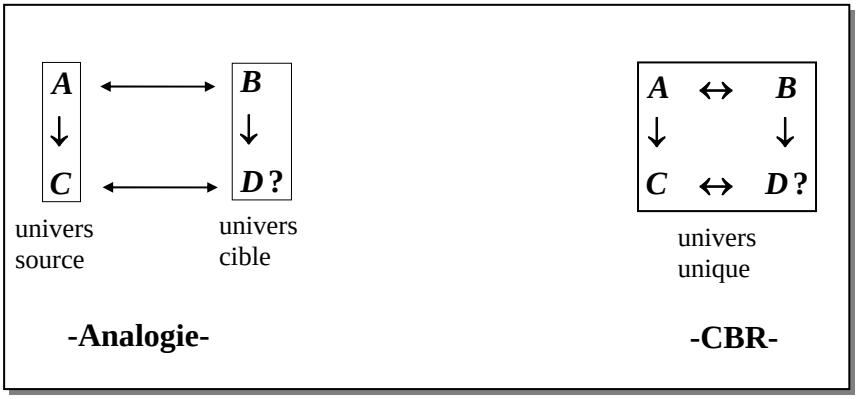


Fig.I.19. CBR et Analogie.

Un exemple illustrant parfaitement cette situation a été cité par Kodratoff **[Kod 1993]** :

Roméo disant à Juliette qu'elle est son «soleil», détecte une similarité entre le soleil et Juliette, en l'occurrence, que le soleil est un objet astronomique qui conditionne la vie sur les planètes qu'il attire, et que Juliette est un objet humain qui conditionne la vie de Roméo qu'elle attire. Les propriétés physiques du soleil correspondent également à des propriétés psychologiques de Juliette. Il s'agit là d'une analogie faite entre le soleil et Juliette.

En raisonnement par cas, le concept de Juliette est très éloigné du concept soleil (Ils ne sont pas de même nature et n'appartiennent pas au même univers), on ne peut donc comparer leur cas. Mais Juliette est très proche de Marguerite dans Faust (deux jeunes filles innocentes etc...) et on peut essayer d'appliquer le cas Juliette à celui de Marguerite.

On constate là l'importance d'un univers unique pour le raisonnement par cas. C'est ce qui en fait une variante simple du raisonnement par analogie. Ce dernier est très naturel mais difficile à mettre en oeuvre dans la pratique **[Hat 1993]**.

6.2 CBR et Apprentissage Automatique

La flexibilité et la possibilité d'acquisition incrémentale de connaissances épisodiques font des systèmes CBR des systèmes capables d'évolution. Cette évolution peut être qualifiée

d'apprentissage «paresseux» par opposition à l'apprentissage «agressif»occasionné par certaines méthodes d'apprentissage inductif **[Sma 1994]**.

En effet, l'apprentissage à base de cas se caractérise par une abstraction «paresseuse» de l'expérience antérieure. Contrairement à une abstraction «agressive» ou une compilation sévère de l'expérience en prévision d'un usage futur, cette méthode préserve le maximum d'informations sur les cas traités jusqu' à ce que une utilisation réelle se présente.

Kodratoff a établi une taxonomie des techniques de raisonnement utilisées pour effectuer l'apprentissage **[Kod 1993]** :

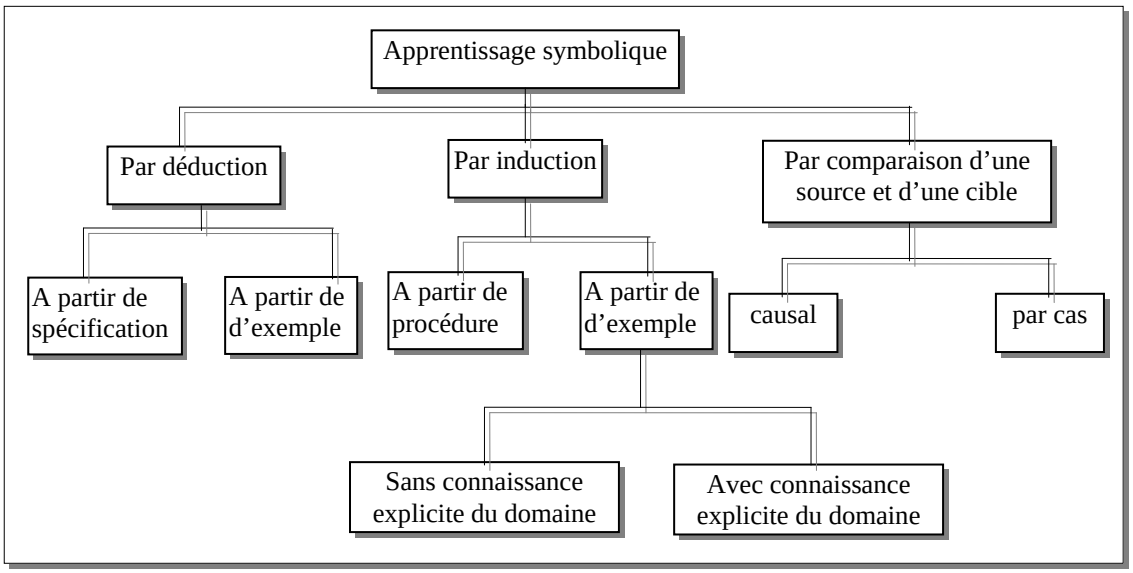


Fig.I.20. techniques de raisonnement pour l'apprentissage.

Il situe l'apprentissage par cas au même niveau que l'apprentissage causal, puisque tous les deux sont basés sur la comparaison d'une source et d'une cible. A la différence que pour l'apprentissage causal, on recherche des relations de causalité alors que le raisonnement par cas recherche des relations de proximité.

6.3 CBR et Réutilisation

La réutilisation est souvent systématiquement associée au génie logiciel. On la définit comme étant l'application des connaissances d'un système, déjà opérationnel, à un autre système similaire **[Erz 1999]**. D'un point de vue CBR, dans le cycle de base : <Retrieve, Reuse, Store>.

On note la place centrale qu'occupe ce concept. Mais il s'agit là d'une réutilisation à sens plus large que celui du génie logiciel. On réutilise un épisode passé pour résoudre un problème actuel. Cet épisode constitue une leçon apprise lors du processus de résolution avec tout ce que cela inclut comme connaissance opératoire ou factuelle. En fait, cette forme de réutilisation est inhérente au processus d'importation de connaissances d'une entité source vers une entité cible que l'on retrouve aussi bien en raisonnement par analogie que par cas.

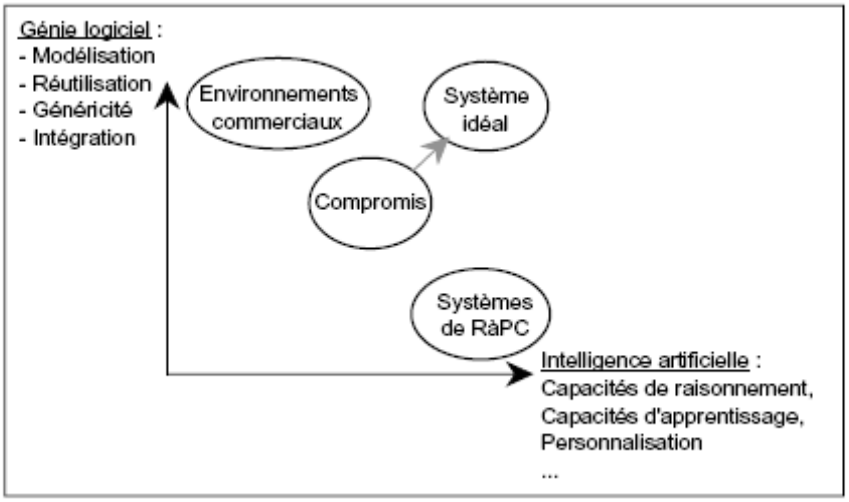


Fig I.21 Les axes génie logiciel/intelligence artificielle

Par ailleurs, dans la figure I.22, les deux paradigmes sont situés selon deux axes : intelligence artificielle et génie logiciel et caractérisant des objectifs différents, souvent contradictoires.

Pour l'aspect génie logiciel, il s'agit de fournir un environnement méthodologique pour guider la conception et des outils associés facilitant la réutilisation de fonctionnalités fournies. Les outils commerciaux sont dédiés à la conception et la réalisation d'applications utilisant le CBR et doivent répondre plus à des objectifs de génie logiciel dans le but de fournir un outil directement réutilisable dans différents domaines d'application.

En ce qui concerne l'aspect intelligence artificielle, il s'agit d'obtenir de bonnes capacités de raisonnement, de faciliter l'acquisition de connaissances et l'apprentissage. Ceci nécessite une définition explicite des connaissances, ce qui rend les systèmes plus souples quant à l'ajout de fonctionnalités. Les systèmes CBR reposant sur un environnement de représentation de connaissances permettent plus facilement l'implantation des modèles des

connaissances de manière explicite. Ils sont par conséquent potentiellement plus ouverts pour une personnalisation à des domaines d'application particuliers. Néanmoins, la personnalisation reste difficile si un cadre méthodologique adéquat ne guide pas la conception.

Le système idéal répondant à la fois aux deux objectifs génie logiciel et puissance de raisonnement et d'apprentissage n'est évidemment pas réaliste. Néanmoins, essayer de s'en approcher en trouvant un bon compromis entre ces deux composantes est un objectif ambitieux.

Chapitre 2 :

La Mémoire de Cas
Modèles et Algorithmes de Recherche

Le coeur du raisonnement à base de cas : la mémoire

1 Introduction

La mémoire expérimentale constitue le centre d'un système de raisonnement à partir de cas. La qualité du raisonnement fourni par le système reflète la richesse de son expérience (en terme de qualité des cas stockés dans la mémoire) et la façon avec laquelle sont organisés ces cas.

La mémoire est utilisée pendant deux étapes importantes du cycle du CBR : l'étape de remémoration de l'expérience passée pendant la résolution d'un problème, et l'étape d'apprentissage ou de mémorisation de la nouvelle expérience acquise en forme de cas.

Par conséquent, le contenu de la mémoire, sa structure, les mécanismes d'indexation utilisés pour assurer l'accès au cas approprié, jouent un rôle crucial dans le rappel et l'apprentissage.

Dans la suite de ce chapitre, nous présentons des modèles cognitifs de la mémoire, l'organisation de la mémoire, ainsi que les algorithmes de recherche basés sur l'hypothèse de similarité.

2 Théorie de la Mémoire

Plusieurs théories de la mémoire ont successivement dominé. La dernière étant la théorie de la mémoire dynamique de Schanck. Elle a donné lieu aux premiers systèmes de raisonnement à partir de cas **[Rou 1994]**, **[Kol 1993]**.

Théorie de Lindsay, Norman et al (1972).

Des chercheurs en psychologie cognitive et en intelligence artificielle ont proposé une organisation de la mémoire sous forme de réseau sémantique. Selon cette théorie, le monde est décrit par un ensemble de faits statiques, immuables. Les limites de cette théorie pour la modélisation de la mémoire sont : toutes les données ne sont pas représentables sous formes de faits statiques. La taille des souvenirs peut varier et les souvenirs eux-mêmes peuvent

évoluer. La deuxième critique qui lui est faite est de ne pas expliciter l'origine des connaissances qu'elle contient.

La théorie de la mémoire épisodique (Tulving 72 et 83).

Pour tenir compte de l'aspect éphémère de la mémoire, Tulving propose d'ajouter à la mémoire sémantique une mémoire épisodique. La mémoire reçoit et stocke des informations sur des épisodes, des histoires ou des événements datés, et des informations sur des relations spatiales et temporelles entre événements. La mémoire n'est plus figée.

La mémoire conceptuelle (Schanck 72).

Parallèlement à ces recherches, en intelligence artificielle, Schanck s'intéresse à partir de 1972 à la compréhension du langage naturel, à des phrases puis à des textes. Pour cela, il propose la mémoire conceptuelle. La mémoire conceptuelle reprend le modèle précédent mais la mémoire sémantique est décomposée en une partie statique (lexicale) et une partie dynamique (associations et relations entre concepts) modifiée avec l'expérience. Pour Schanck, le savoir est acquis par l'expérience et il montre l'importance du processus d'acquisition des connaissances dans une théorie de la mémoire.

Théorie de la mémoire dynamique.

Nos mémoires sont dynamiques. Elles changent suite à nos expériences : les nouvelles situations que l'on rencontre, les questions que l'on se pose face à ces situations et la façon d'y répondre. L'effet de bord de ce processus **[Kol 1993]**, est qu'une mémoire dynamique ne se comporte jamais de façon exactement identique, deux fois de suite car elle change après chaque expérience.

Retrouver la bonne expérience en mémoire, est la clé de succès dans le raisonnement. L'expérience adéquate peut être une structure de connaissance générale (schéma), représentant une description normative d'une expérience, ou une instance (un cas) de cette structure générale. La bonne expérience est celle qui peut donner lieu aux meilleures inférences.

Identifier les expériences adéquates en mémoire, constitue le processus de rappel. L'analyse des rappels suggère deux façons différentes de structurer la mémoire, l'une organise les situations selon les activités similaires, l'autre organise les situations selon leur similarité du point de vue interaction buts-plan. Dans la théorie de Schank, la première est dite MOP's (Memory Organization Packets), la seconde TOP's (pour Thematic Organization Packets)

a) Les MOPs **[Rou 1994]** : les scripts avaient été proposés pour la compréhension du langage naturel en 1973. Schank les réactualise en 82 et les généralise en MOPs. Un script décrit un ensemble d'informations détaillées sur un événement stéréotypé, comme «aller au restaurant». En raison de leur description très précise, des scripts comme la visite chez un dentiste et la visite chez un généraliste sont très différents alors qu'ils se déroulent de manière semblable.

Un MOP est une généralisation de plusieurs scripts. Les scripts «aller chez le dentiste» et «aller chez le généraliste » peuvent être généralisés en le script «aller chez le docteur». Un MOP comprend une succession de scènes. Chaque scène correspond à une étape et un ou plusieurs scripts décrivent comment la réaliser. Par exemple la scène «prendre un rendez-vous» comprend l'appel téléphonique, la demande de rendez-vous et le choix de la date. La structuration de la mémoire en MOPs permet sa mise à jour par la création éventuelle de nouveaux MOPs. Cette théorie a été utilisée dans plusieurs systèmes CBR dont le système CYRUS.

b) Les TOPs: constituent une généralisation des MOPs et permettent la remémoration trans-domaine. Ils ont notamment été utilisés dans CHEF (système CBR).

Ce qu'il y a lieu de préciser est que la théorie de la mémoire dynamique fournit un guide pour la ***représentation***, la ***segmentation***, l'***accès*** et le ***raisonnement*** sur les «cas».

3 Organisation des Cas en Mémoire

Les processus de mémorisation et de remémoration sont fortement liés à la façon d'organiser les cas dans la mémoire. Dans cette section, nous allons reprendre avec plus de

détails, différentes méthodes d'organisation des cas en mémoire, déjà citées en chapitres I. Pour chacune des méthodes, nous détaillons les processus de mémorisation et la remémoration. Il est évident que le choix de la structure de mémoire dépend entièrement de la tâche à accomplir, du domaine d'application, de la complexité des étiquettes et du nombre des cas disponibles.

Deux catégories pour l'organisation de la mémoire sont distinguées :

La mémoire plate : il s'agit de mémoriser tous les cas dans une liste séquentielle. Nous détaillons ce type de mémoire dans la section suivante,

La mémoire hiérarchique : quand la mémoire des cas est large, il y a une nécessité d'organiser les cas hiérarchiquement. Ceci permet de simplifier la remémoration. Dans la suite, nous présentons deux approches pour l'organisation hiérarchique des cas en mémoire : les réseaux à traits (ou caractéristiques) partagés et les arbres de discriminations.

3.1 La Mémoire Plate

Les cas sont stockés séquentiellement dans une simple liste ou fichier, c'est la plus simple structure à imaginer pour une mémoire. Les avantages d'une telle structure est que pendant la remémoration, tous les cas existant dans la mémoire sont testés, ceci garantie une remémoration précise qui dépend de la qualité de la fonction d'appariement. De plus, la mémorisation n'est pas coûteuse, il suffit d'ajouter le nouveau cas à la fin du fichier.

L'inconvénient majeur est le temps de remémoration qui augmente linéairement avec la taille de la mémoire.

Il existe plusieurs variantes pour l'organisation en mémoire plate :

Indexation superficielle : L'indexation se fait dans un seul niveau, chaque descripteur (attribut, valeur) choisi comme étiquette pointe vers les cas correspondants (qui contiennent ce descripteur dans leur représentation).

Pendant la phase de remémoration, les cas qui sont pointés par ce descripteur sont sélectionnés, et ensuite la fonction d'appariement est appliquée juste à ce groupe des cas, et non pas à tous les cas. Ceci marche très bien quand ces descripteurs permettent d'extraire un petit nombre de cas et quand les étiquettes sont suffisamment descriptives.

Partitionnement de la mémoire : Le système doit reconnaître à quelles partitions une nouvelle situation appartient. La fonction d'appariement est appliquée seulement aux cas appartenant aux partitions sélectionnées.

Extraction parallèle : La fonction d'appariement est appliquée d'une manière parallèle à tous les cas qui existent dans la mémoire.

3.2 Organisation en Réseaux à Traits Partagés

Il s'agit d'organiser les cas en réseaux, dans le sens où les cas qui partagent des caractéristiques sont groupés ensemble. Chaque noeud d'un tel graphe contient les caractéristiques partagées par les cas qui sont au niveau inférieur. Les feuilles contiennent les cas eux-mêmes. La méthode générale de la remémoration à partir d'un tel réseau (structure) consiste à retrouver le noeud qui correspond le mieux à l'entrée ou au problème traité, jusqu'à ce qu'on arrive à une feuille.

Pour construire une telle architecture, qui a la plupart du temps la forme d'un arbre, les étapes suivantes sont réalisées **[Kol 1993]**, **[Mal 1996]** :

1. choisir une méthode de regroupement pour partitionner les cas,
2. créer la racine de l'arbre N, C étant l'ensemble des cas à organiser,
3. mettre tous les attributs en commun entre les cas dans N,
4. partitionner C en utilisant la méthode de regroupement choisie dans la première étape,
5. pour chaque partition :
    a) créer un noeud Ni dont le père est N,
    b) si Ni contient plus qu'un seul cas, répéter à partir de l'étape 3 avec :N = Ni et C = les cas dans la partition,
    c) sinon, mettre les attributs des cas dans le noeud Ni.

D'un autre côté, la mémorisation d'un nouveau cas nécessite une mise à jour de l'arbre de la façon suivante (on commence par la racine) :

1. comparer le nouveau cas avec les cas dans ce niveau, trouver le noeud N qui correspond le mieux ;
2. si le cas partage tous les attributs avec le noeud N continuer, sinon retourner à 1 ;
3. si N est une feuille (un cas) regrouper les traits en commun et les mettre dans le noeud N, construire deux nouveaux nœuds, N1 et N2 enfants de N, placer dans N1 l'ancien cas et dans N2 le nouveau cas ;

4. sinon, créer un nouveau noeud M au même niveau de N, placer le nouveau cas dans ce nœud, associer à M le même parent que N.

Les réseaux à traits partagés garantissent une remémoration efficace en terme de temps de calcul par rapport à la mémoire plate. Ceci est dû à l'organisation hiérarchique des cas qui permet de gagner du temps pendant la remémoration. Néanmoins, ces méthodes présentent un certain nombre d'inconvénients :

- L'ajout des nouveaux cas dans le réseau est une opération compliquée en complexité de calcul,
- la nécessité d'un grand espace,
- il n'est pas du tout évident de garder l'arbre optimal,
- la remémoration parfaite n'est pas garantie car les cas ne sont pas tous visités,

3.3 Arbres de discrimination

La différence essentielle entre un arbre à traits communs et un arbre de discrimination est que dans le premier, l'ordre des noeuds dans l'arbre se fait suivant une méthode de regroupement des cas selon les traits communs, tandis que dans le deuxième l'ordre est choisi selon un critère qui sert à classer les attributs suivant leur capacité de discrimination.

Après avoir fixé une liste de priorités qui reflète le niveau de l'attribut dans l'arbre, celui-ci est construit. Chaque noeud correspond à une question concernant la valeur de l'un de ces attributs, chaque branche correspond à une réponse, les feuilles contiennent les cas.

Les arbres de discrimination partagent les mêmes avantages et inconvénients des réseaux à traits partagés. On peut ajouter aussi les deux inconvénients suivants :

- la nécessité de la construction d'une liste de priorités pour les attributs ;
- l'arbre est incapable de donner une classification en cas de présence des valeurs manquantes.

4 La Recherche Guidée par la Similarité

Motivation :

En Base de Données, la recherche d'une donnée se fait en présentant une certaine clé. De même en recherche d'information, un mot-clé est présenté. Le problème avec cela est que l'on peut tomber dans deux situations différentes : le système ne fournit aucune réponse (Silence) ou qu'il donne différentes réponses (Bruit).

L'efficacité et la précision sont deux objectifs important de l'étape de recherche.

L'efficacité des différentes méthodes de recherche dépend grandement des points suivants :

- la représentation des cas,
- la structure de la base,
- la mesure de similarité,
- la précision de la réponse attendue.

Par ailleurs, ces caractéristiques elles même dépendent du domaine et de l'application.

La tâche :

La tâche centrale de la recherche étant :

- Soient :
  - Une base de cas BC=$\{C_1, C_2, \ldots, C_n\}$,
  - Une mesure de similarité Sim,
  - Une requête R
- On veut :
  - Le cas le plus similaire C_i ou bien
  - Les m cas les plus similaires $\{C_1, C_2, \ldots, C_m\}$ (ordonnés ou pas) ou bien
  - Tous les cas $\{C_1, C_2, \ldots, C_m\}$ qui possède au moins une similarité Sim_{min} avec la requête R.

Le problème posé : Comment organiser la base de cas pour une recherche efficace ?

Le tableau suivant résume les différentes approches de recherche guidées par la similarité [Ric 2004]:

Type	Méthodes	Restriction /Similarité	Convient à …
Force brute	Recherche Séquentielle	Non	Petites Bases de Cas Similarité Simple
Basé index	Kd-Tree	Réflexivité, Monotonie Pas de Similarité de classe	Petit nombre d'attribut Grandes bases de cas
	Fish & Shrink	Réflexivité, Monotonie Inégalité triangulaire	Similarité complexe Petites bases de cas
	CRN	Monotonie Pas de Similarité de classe	Peu d'attribut numériques Grande bases de cas

Tab.II.1. Principales méthodes de recherche guidées similarité

4.1 La Recherche Séquentielle

Il s'agit du schéma de recherche le plus simple dans lequel il est possible d'utiliser les structures suivantes :

STRUCTURE DE DONNEES :

```
Type :
        SimCas =        Record
                Cas : CAS ;
                Similarité :[0..1] ;
                        end
        SimCasQueue = Array[1..m] of  SimCas ;

Variables :
        Scq : SimCasQueue ;
        BC : Array[1..n] of Cas ;
```

ALGORITHME DE RECHERCHE :

```
        Scq[1..m].similarité :=0 ;
        FOR  i:= 1 TO n DO
                IF Sim(R, BC[i]) > Scq[m].similarité
                        THEN insert BC[i] in Scq;
        Return Scq;
```

Propriété de la recherche séquentielle :

- Complexité est de l'ordre : O(n).

Inconvénients :

- Pose problème si la base est très grande
- L'effort de recherche est indépendant de la requête.
- L'effort de recherche est indépendant de m.

Avantage :

- Implémentation simple.
- Aucune structure additionnelle d'indexation n'est requise.
- Des mesures arbitraires de similarité peuvent être utilisées (voir **[Bar 1992]**).

4.2 La Recherche Basée Kd-Tree

Il s'agit d'une recherche dans un arbre binaire de dimension K. L'idée étant de décomposer la base de cas itérativement en petites partitions en utilisant une structure

arborescente. La recherche se fera de manière descendante avec d'éventuels retours en arrière.

Définition :

Etant donnés :

- k domaines ordonnés $T_1, \ldots, T_k$ pour les attributs $A_1, \ldots, A_k$,
- Une base de cas $BC \subseteq T_1 \times \ldots \times T_k$ et
- Un paramètre b (représentant la taille du 'bucket' ou seau)

Un kd-Tree T(BC)pour une base de cas BC est un arbre binaire récursivement défini par :

- Si $|BC| \leq b$: T(BC) est un nœud feuille (appelé 'bucket') étiqueté par BC.
- Si $|BC| > b$: T(BC) est alors un arbre possédant les propriétés suivantes :
  - La racine est étiquetée par un attribut A_i et une valeur $V_i \in T_i$
  - La racine possède deux kd-tree successeurs $T_{\leq}$ ($BC_{\leq}$) et $T_{>}$ ($BC_{>}$)
  - Où $BC_{\leq} := \{(X_1, \ldots, X_k) \in BC \mid Xi \leq V_i\}$ et
    $BC_{>} := \{(X_1, \ldots, X_k) \in BC \mid Xi > V_i\}$

Propriétés d'un Kd-Tree :

Un Kd-tree partitionne la base de sorte que la racine représente la totalité des cas. Un nœud feuille représente un sous-ensemble de la base non partionnable. A chaque nœud interne la base est subdivisée selon la valeur d'un attribut.

La sélection d'un attribut de partitionnement peut se faire selon différentes techniques, telle que l'utilisation de l'entropie.

La recherche se fait en parcourant l'arborescence de manière descendante.

4.3 La Recherche Basée Fish & Shrink

Supposons deux cas considérés comme similaires dans un domaine donné et selon un aspect précis. Si l'on change l'aspect considéré ces deux cas deviennent dissimilaires. Malheureusement, lors de la construction des bases de cas, il n'est pas aisé de changer de point de vue car l'on utilise des similarités statiques pour organiser la base de cas [Sch 1996].

L'idée de l'algorithme de Fish & Shrink est de considérer le cas comme étant une structure supportant de multiples facettes ou aspects. La similarité de l'ensemble des cas de la base sera calculée selon chacun des aspects du cas et stockée préalablement. Lors de la

recherche, on peut sélectionner les aspects à considérer et éventuellement les combiner selon leurs poids si l'on doit prendre en compte plus d'un aspect.

Ainsi, la base de cas peut être vue comme un réseau de cas (voir fig.II.1). Où un polyèdre représente un cas.

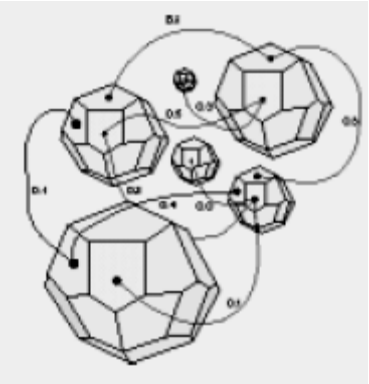


Fig.II.1. La base de cas vue comme réseau de cas.

Une facette du polyèdre représente un aspect et l'étiquette d'un arc reliant deux cas représente la distance calculée entre ces cas selon l'aspect considéré.

Propriétés de Fish & Shrink :

Inconvénients :

- Les distances entre aspects de cas doivent être calculées au préalable.
- La fonction distance doit satisfaire l'inégalité triangulaire.

Avantages :

- Calcul flexible de distance pour la requête.
- Différentes tâches de recherche peuvent être effectuées.
- Efficace, puisque différents calculs de distance peuvent être sauvegardés.
- Peut être utilisé comme solution « anytime ».

Convenable lorsque l'application nécessite des calcul de similarité coûteux (ex : représentation de graphes).

4.4 La Recherche Basée CRN

Là aussi, il s'agit d'une base de cas organisée en réseau mais selon une optique totalement différente. En effet, les CRNs (pour Case Retrieval Nets) incarnent le principe de mémoire associative donc accessible par le contenu.

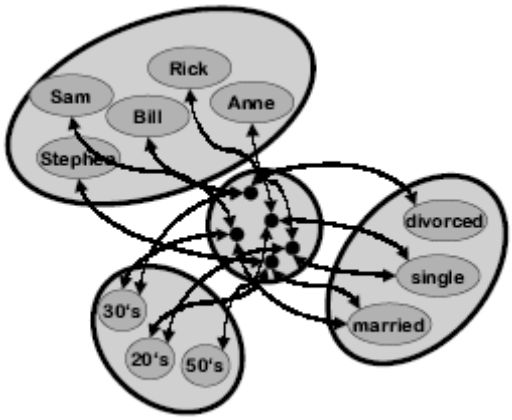


Fig.II.2. Principe d'une mémoire associative extrait de [Len 1999]

Ce principe (Fig.II.2) s'écarte de la représentation classique des connaissances en mettant en commun, toute donnée susceptible d'être partagée par plusieurs objets. Exemple : Bill et Sam ont chacun 30 ans. La valeur 30 ans n'est représentée qu'une fois dans le réseau tout en étant utilisable aussi bien pour Sam que pour Bill.

Une présentation plus technique des CRNs est donnée en Annexe 1.

5 Conclusion

Il est important de rappeler que l'objectif de ce chapitre est de donner une vue sommaire, pas nécessairement exhaustive, sur la mémoire en CBR. Aussi, avons nous allégé sa présentation en se limitant aux principes et caractéristiques des modèles de mémoire orientés similarité.

Quant aux CRNs, nous nous sommes limités à donner la ligne directrice puisqu'une description détaillée est donnée en Annexe 1 et qu'une extension en est donnée en chapitreIII.

Chapitre 3 :

Un Modèle de Mémoire Supportant la Recherche Guidée par l'Adaptabilité

« *Every time you run into a problem which in standard AI technology would be solved by searching a space of possibilities,* ***in CBR what you do is search the space of actuality; you search the space of what does happen, what has happened, rather than the space of what could happen;*** *because the space of what could happen is outrageously huge, and the space of what has happened is nice and tiny and tractable in comparison* » K.J. Hammond cité dans **[Bro 1994]**

1 Introduction

Le *"space of what does happen, what has happened"* est materialisé par la mémoire de cas. Retrouver la bonne expérience en mémoire constitue la clé de succès de l'inférence en CBR. C'est le rôle de l'étape de rappel ou remémoration.

Pour fonctionner correctement, le CBR utilise des cas stockés dans une structure de mémoire. Ces cas sont représentatifs de l'ensemble des problèmes rencontrés dans le domaine. Plus riche est la mémoire de cas, plus efficace est le système en terme de capacité de résolution. Or, cette richesse peut induire un coût de recherche prohibitif pour retrouver la bonne expérience à réutiliser. C'est pourquoi l'organisation de la mémoire ainsi que les algorithmes de recherche sont particulièrement important dans ce mode de raisonnement.

Dans la suite de ce chapitre, nous nous intéressons au processus de rappel en étudiant deux points de vue différents : la similarité et l'adaptabilité. Le système « Déjà-Vu » est présenté comme illustration du critère d'adaptabilité. Nous présentons ensuite, un troisième point de vue qui intègre les deux critères, suivi d'un modèle de mémoire incorporant ce point de vue. Nous concluons enfin, en situant notre approche (le troisième point de vue) relativement aux deux critères.

2 L'Hypothèse de Similarité

L'une des hypothèse majeures faites en Intelligence Artificielle est que les expériences similaires peuvent guider le raisonnement future, la résolution de problème ou l'apprentissage ; c'est ce que Smith (dans **[Smy 1999]**) a qualifié d'*Hypothèse de Similarité*.

Il est intéressant de considérer que le CBR fonctionne sur deux espaces de symboles distincts, un espace de spécification ou de description de problème et un espace de description de solution (figure.III.1.).

La recherche opère dans l'espace problème en tentant de trouver des appariements entre la description du problème cible et la partie problème des cas sources.

L'adaptation quant à elle, opère dans l'espace solution en procédant à des transformations sur les parties de la solution du cas source pour satisfaire le cas cible.

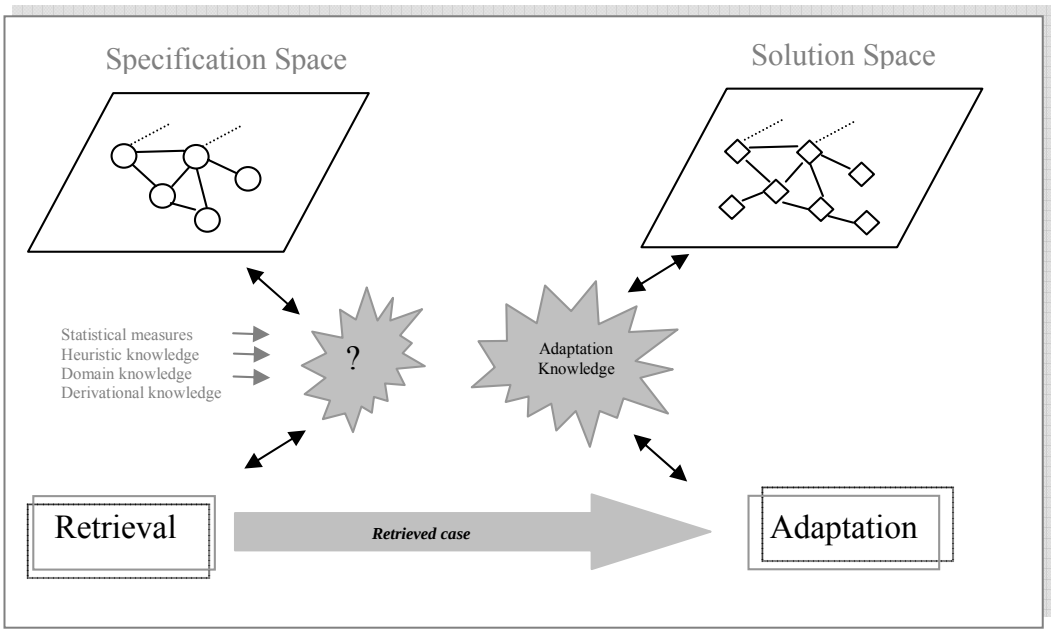


Fig.III.1. Du problème à la Solution, quel est le lien ?

Dans cette perspective, la remémoration est l'exploration de l'espace problème en vue de trouver le bon ensemble d'appariements entre les spécifications du cas cible et celles des cas sources. L'adaptation est l'exploration de l'espace solution en vue de trouver le bon ensemble de transformation de la solution du cas source en une solution du cas cible.

Les approches conventionnelles de recherche font une nette séparation entre les étapes de recherche et d'adaptation, en supposant que la similarité dans l'espace problème peut être utilisé pour prédire l'utilité future du cas. En ignorant ainsi le lien entre l'espace problème et l'espace solution, nous ignorons aussi toute communication réelle entre l'étape de recherche et l'étape d'adaptation.

Par ailleurs, l'hypothèse de similarité est basée sur des 'connaissances de similarités' qui se traduisent en mesure de similarité. Généralement ces mesures sont interprétées et implémentées par de simples distances géométriques **[Sta 2002]**. Cependant, dans de multiples domaines d'application, ces distances ne suffisent pas pour obtenir des résultats

raisonnables. En effet, une simple distance géométrique entre descriptions de problèmes source et cible peut constituer une mauvaise approximation de la qualité de la solution trouvée d'un point de vue résolution.

Une illustration abstraite de ce problème est montrée en figure.III.2. La partie gauche représente l'espace problème *P*, la partie droite représente l'espace solution correspondant *S*. Une connexion entre les deux espaces est donnée par deux cas $Case_1$ et $Case_2$, constitué respectivement des descriptions problèmes P_1, P_2 et des description solutions S_1, S_2. De plus, nous avons P_a la description d'un problème non résolu.

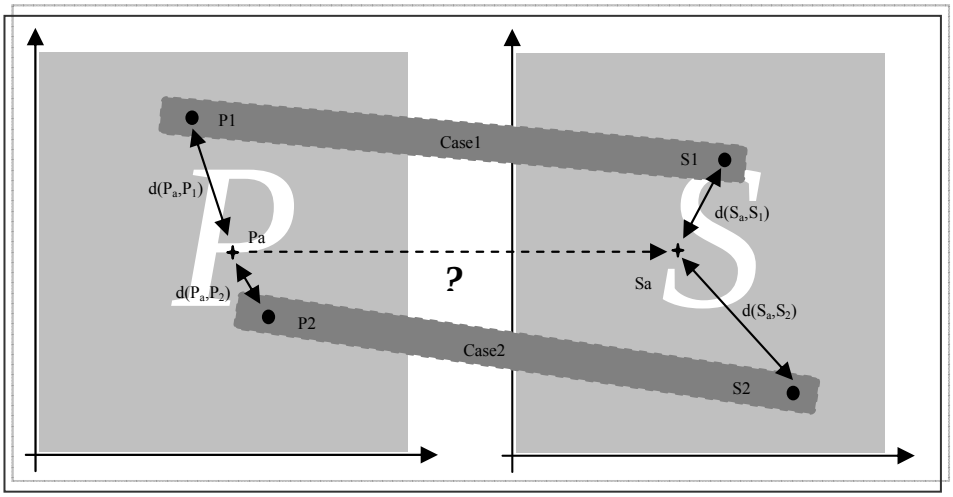


Fig.III.2. Du problème à la solution, comment doit être le lien ?

Etant donnée cette situation, tâchons de trouver une solution pour P_a. Supposons qu'il existe une solution S_a optimale mais inconnue. En appliquant le CBR par l'utilisation d'une simple distance géométrique sur l'espace *P* comme mesure de similarité, le processus de recherche va assigner à $case_2$ une qualité de solution plus grande que elle de $case_1$ puisque $d(P_a,P_1)>d(P_a,P_2)$. Cependant en analysant, les relations dans l'espace *S* des solutions, il s'avère que la solution S_1 contenue dans $case_1$ est plus proche de la solution optimale S_a que celle retrouvée S_2 ($d(S_a,S_2)>d(S_a,S_1)$).

Si l'on interprète la distance géométrique entre solution comme mesure de qualité de la solution, il est évident qu'une telle approche de recherche n'est pas optimale.

3 Recherche Guidée par l'Adaptabilité : le système «Déjà-Vu »

De nombreux travaux (**[Kol 1992]**, **[Smy 1993]**, **[Smy 1995]**) ont démontré que ce ne sont pas les cas les plus similaires qui sont toujours les plus faciles à adapter. Il faut donc

envisager une recherche guidée par l'adaptation (AGR pour Adaptation-Guided Retrival). Ce concept fût introduit la première fois par Smyth et Keane dans **[Smy 1993]**, raffiné par les mêmes auteurs dans **[Smy 1995]** puis repris dans son principe par de multiples travaux (**[Col 1996]**, **[Rou 1996]**).

Dans ce qui suit nous présentons « Déjà-Vu » le premier sysème CBR mettant en œuvre le principe AGR. « Déjà-Vu » est un système CBR pour la conception de logiciels de pilotage d'installations industrielles. Le cycle CBR classique est adapté pour permettre la réutilisation combinée de plusieurs cas de conception.

Le domaine d'application principal est le contrôle de véhicules robotisés dans une aciérie. Ces véhicules se déplacent sur des voies interconnectées pour transporter des objets d'un endroit à un autre de l'usine, avec des vitesses et directions différentes. Une des tâches principales étudiée est le transport, le chargement / déchargement de rouleaux et de bobines d'acier par des véhicules.

Modèle de connaissances et cas

Les connaissances du domaine décrivent la configuration de l'usine et sont structurées en classes d'objets **[Smy 1995]** : véhicules, réservoirs, produits, pistes. Ces classes possèdent des spécialisations, par exemple il existe deux sortes de véhicules, les véhicules à bobines ou les ponts roulants à bobines. Les objets sont décrits dans une hiérarchie de composition. La spécification des tâches précise leur nature, les machines requises, les conditions initiales, les buts de la tâche et les contraintes de fonctionnement.

Les tâches réalisées par les programmes de pilotage varient en complexité, par exemple des tâches de déplacement, de chargement / déchargement, de transfert. Le domaine des tâches de pilotage d'installations industrielles peut être facilement structuré et décomposé d'une part en tâches plus simples et à différents niveaux d'abstraction. Le langage de programmation utilisé présente l'avantage d'être de haut niveau et les opérations peuvent être représentées par des noeuds dans un graphe de solutions. Ce graphe peut ensuite être compilé en langage exécutable.

Les solutions de conception sont mémorisées sous forme de hiérarchies de cas. Ces hiérarchies comportent d'une part des cas abstraits, correspondant à des solutions abstraites, qui décomposent un problème donné en sous problèmes, et d'autre part des cas de conception concrets fournissant des morceaux de programmes résolvant un sous problème. Cette organisation permet de réutiliser des portions de conceptions complexes plus facilement et de décomposer des problèmes cibles en sous problèmes plus simples. Le cas racine de la

hiérarchie est le problème de conception le plus abstrait, et le code nécessaire pour générer la solution est situé dans les cas feuilles.

Les cas comportent une partie description et une partie solution. La partie description décrit les caractéristiques de la tâche. Les solutions sont des organigrammes du programme. Pour les cas abstraits, les solutions sont écrites avec des extensions du langage de programmation sous forme de commandes abstraites.

Les cas sont représentés par des frames. Un frame principal décrit la structure de la tâche et ses caractéristiques principales. Ce frame est relié à la base de connaissance et aux modèles de l'installation en désignant les frames correspondant aux entités utilisées par la tâche. Ces frames entités contiennent des détails supplémentaires.

Résolution de problème

Le principe de base de « Déjà-Vu » est la conception d'une solution en réutilisant plusieurs cas de manière combinée : le CBR hiérarchique -CBRH- étend le cycle classique du CBR pour la combinaison de multiples cas. Le CBRH réutilise plusieurs cas à des niveaux d'abstraction différents.

L'approche de décomposition de « Déjà-Vu » pour la conception facilite l'adaptation combinant des parties de cas (les cas spécifiques de conception) pour résoudre de nouveaux problèmes de conception. L'approche hiérarchique pour la représentation des cas est utilisée afin de réduire la complexité de l'adaptation.

La décomposition d'un problème complexe en sous problèmes ne nécessite pas de procédure ou connaissances particulières, car elle résulte de la remémoration et de l'adaptation de cas abstraits.

Le processus de résolution de problèmes CBRH nécessite plusieurs cycles de raisonnement CBR. A chaque cycle, le système remémore un cas, l'adapte et l'intègre dans la solution. Si le cas remémoré est abstrait, il correspond alors à une décomposition de problème et son traitement implique l'exécution récursive de plusieurs cycles de CBR correspondant à la résolution des sous problèmes. Les relations de décomposition sont exploitées pour identifier les spécifications de sous problèmes. L'intégration de plusieurs éléments dans la solution cible réétablit les relations de décomposition. Les cas sources abstraits de « Déjà-Vu » contiennent explicitement la connaissance de décomposition, ce qui est utile dans des domaines où aucune connaissance statique de décomposition n'est disponible. Par conséquent, plus la traversée de la hiérarchie est rapide et plus la remémoration est efficace.

« Déjà-Vu » utilise une méthode de remémoration guidée par l'adaptabilité. Les auteurs argumentent leur approche par le fait que ce ne sont pas toujours les cas sources les plus similaires au cas cible qui sont les plus facilement adaptables, particulièrement lorsque la similarité est basée sur des caractéristiques de surface. La remémoration doit alors rechercher non seulement des cas similaires, mais surtout des cas facilement adaptables. Dans « Déjà-Vu », les connaissances concernant l'adaptabilité sont stockées dans les parties conditions des règles d'adaptation. Le processus de remémoration retourne non seulement le cas source le plus facilement adaptable mais également les règles d'adaptation à appliquer.

Les connaissances d'adaptation sont de deux catégories. Celles de la première catégorie, appelées spécialistes d'adaptation, offrent des procédures d'adaptation spécialisées pour des objets ou tâches particuliers. Celles de la deuxième catégorie, appelées stratégies d'adaptation sont plus générales. Chaque connaissance d'adaptation comporte d'une part une partie action spécifiant les étapes d'adaptation de la solution, et d'autre part une partie condition décrivant de manière détaillée les types de modifications réalisées par cette connaissance. Les parties action des connaissances d'adaptation sont décrites au moyen d'opérateurs de transformation et de manipulation des graphes représentant la solution. Afin d'exploiter les connaissances d'adaptation durant la phase de remémoration, les parties conditions définissent les conditions nécessaires à l'application des connaissances d'adaptation. Elles comportent deux parties, les caractéristiques et les tests. Les caractéristiques indiquent le contexte d'applicabilité de la connaissance d'adaptation avec les caractéristiques dérivées et des informations sur la tâche. Les tests réalisent des vérifications pour évaluer les besoins en adaptation. Les spécialistes d'adaptation sont utilisés pour réaliser des modifications locales conformément aux différences de spécification des problèmes source et cible, et opèrent soit sur des cas abstraits ou concrets. Les stratégies d'adaptation réalisent des modifications plus globales en détectant des incohérences, ou notamment en cas de problèmes d'interaction.

L'adaptation de cas de décomposition suppose la modification des relations. Ces modifications vont de simples substitutions à la suppression ou l'insertion de relations. Une fois le cas de décomposition modifié, un nouvel ensemble de sous problèmes est spécifié. Il peut être résolu en se remémorant des cas de n'importe quelle hiérarchie de cas.

Le processus de remémoration est fondé sur la prédiction de l'adaptabilité de cas afin d'assurer la facilité d'adaptation d'un cas source pour le problème cible. Afin d'assurer l'adaptabilité des cas remémorés « Déjà-Vu » implante une méthode de remémoration guidée par l'adaptabilité. De cette manière, la connaissance d'adaptation applicable est identifiée au moment de la remémoration. Le processus de remémoration est réalisé en 4 étapes. Dans la

première étape, le problème cible est analysé et les caractéristiques importantes sont mises en évidence. Elles servent à activer des connaissances d'adaptation pertinentes. Dans la deuxième étape, les spécialistes d'adaptation inactivés servent à éliminer les cas inutilisables pour résoudre le problème cible. Dans une troisième étape, les spécialistes d'adaptation actifs permettent de sélectionner les cas adaptables pour les besoins de la cible. Ces cas sont dits adaptables localement. Dans la dernière étape, les stratégies d'adaptation tentent de reconnaître des conflits dans les cas localement adaptables. Un coût d'adaptation global est alors calculé pour chaque cas en combinant les coûts des spécialistes d'adaptation et des stratégies d'adaptation. Les cas sont ordonnés par coût décroissant.

L'approche développée pour la remémoration des cas repose non pas sur leur ressemblance avec un problème cible, mais sur une évaluation de leur facilité d'adaptation. Le processus d'adaptation a été étudié en détail et utilise deux sortes de connaissances : les spécialistes qui agissent localement et les stratégies qui contrôlent le processus de manière plus globale.

Les processus de remémoration et réutilisation sont étroitement liés et les cas sont mis en relation avec les connaissances d'adaptation qui servent à la recherche des cas, ce qui permet d'évaluer un coût prévisionnel d'adaptation. L'adaptation procède aux modifications nécessaires mises en évidence lors de la remémoration, et les stratégies sont mises en oeuvre après contrôle de la cohérence de la solution suite aux modifications réalisées par les spécialistes d'adaptation.

4 La Réutilisation mise en équation

Rappelons que l'objectif de la recherche est de fournir un cas candidat à la réutilisation. Dans les sections précédentes nous avons vu deux philosophies de choix du candidat. La première se limite à l'exploration de l'espace problème avec le risque de biaiser les distances sur l'espace solution, il s'agit de l'hypothèse de similarité. La seconde, fait abstraction de l'espace problème et procède à une exploration de l'espace solution, il s'agit de l'approche AGR.

En fait, le meilleur critère de choix du candidat se résume à : *« retrouver le cas le plus similaire dans sa partie problème et le plus adaptable dans sa partie solution »* ce qui revient à l'intégration des deux critères : similarité et adaptabilité à la fois.

Pour mieux envisager cette intégration, revenons sur l'une des racines du CBR à savoir le paradigme d'analogie.

En fait, la réutilisation en CBR revient à un transfert analogique de connaissances entre cas source et cas cible selon le schéma présenté en figure.III.3.

Rappelons par ailleurs, que la réutilisation consiste en deux tâches qui sont : copier et adapter.

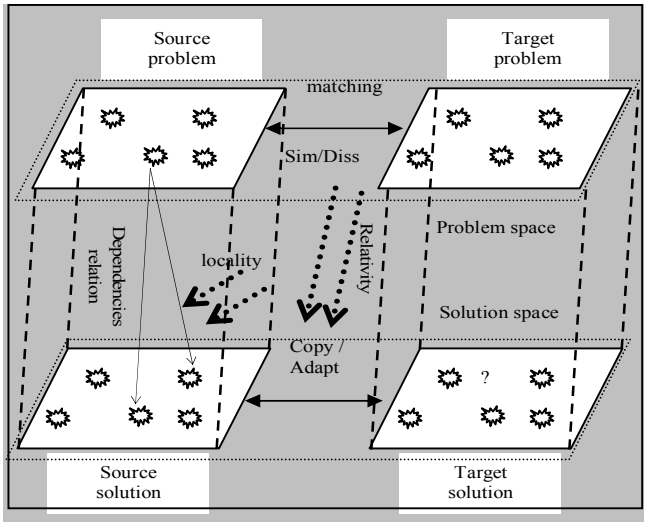


Fig.III.3. Le carré d'analogie revisité (extrait de [Nou 2004b])

L'appréciation de l'adaptabilité, qui est la potentialité d'un cas à être adapté, va être estimée à travers son dual appelé 'coût d'adaptabilité' et noté μ(cas). Il est définie comme étant l'approximation de l'effort à fournir pour adapter une solution. Son calcul est basé sur les relations de dépendances présentes entre partie problème et partie solution du cas source. C'est un score établit à partir des liens problème /solution uniquement pour les parties problèmes concernés par la dissimilarité.. Ainsi:

μ(partie Solution % partie problème dissimilaire) = Score des opérateurs d'adaptation applicable.

$\mu(case) = \sum \mu($ partie Solution % partie problème dissimilaire$)$

L'appréciation de ce score peut se faire dans une étape de prétraitement comme nous le verrons en chapitre IV.

Le cas source est défini par des descripteurs du problème dans l'espace problème et des descripteurs de la solution dans l'espace solution. Le cas cible dispose uniquement des descripteurs problème. L'inférence consiste à apparier le problème source et le problème cible. En fonction des similarités et dissimilarités issues de cet appariement, le moteur d'inférence décide de ce qu'il faut *copier* de la solution source et ce qu'il faut *adapter* pour satisfaire les contraintes imposées par le cas cible. Le coût de transfert de connaissances entre les univers source et cible peut être formalisé par :

Tcost(source, target) = similarity * copy cost + dissimilarity * adapt cost

Lorsque les problèmes source et cible sont identiques (similarité maximale = dissimilarié nulle), la réutilisation se limite à la tâche de copie, et il n' y aura pas d'adaptation. Seul le terme en similarité agira, le terme en dissimilarité étant annulé.

D'un autre côté, lorsque le problème source présente des dissimilarités avec le problème cible sur tous les descripteurs, la dissimilarité est maximale et le coût d'adaptation sera calculé sur l'ensemble des descripteurs. L'effort d'adaptation sera donc plus important (cela reviendrait à une génération de la solution 'from the scratch').

Un example illustratif.

L'exemple est inspiré de **[Ber 1998]** et **[Fuc 2000]**. Il s'agit du choix de configuration PC (Ce qui a par ailleurs, insipré l'application I du chapitre V.).

Un problème est composé d'un ensemble de descripteurs relatifs aux préférences ou besoins de l'utilisateur tels que : Games, Music, Word processing, Programming, Image processing, Easiness to handle and Price. La solution sera décrite en termes de : Master board, Processor, Additional pointer, CD-Rom, Colour, Screen, Sound card and Printer.

Notons que le descripteur "Price" peut être vu comme un descripteur problème (en terme de contrainte imposée par l'utilisateur) ou comme descripteur solution lorsqu'il n'est pas contraint (doit être déterminé).

A partir de l'analyse du domaine, l'expert peut établir l'ensemble des dépendances suivant (figure.III.4.) :

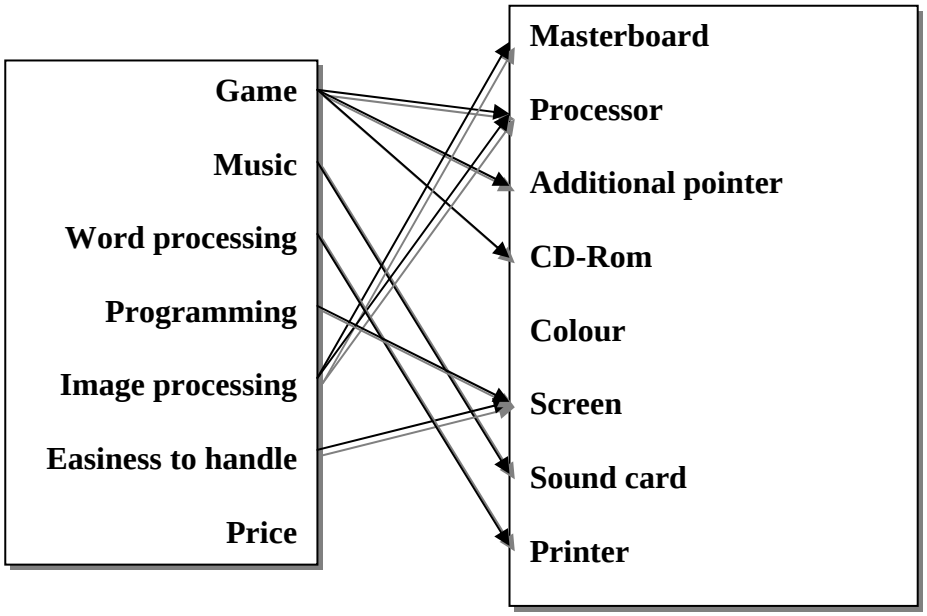


Fig.III.4. Dépendances des descripteurs

- Le besoin d'une configuration 'game' influence le choix des descripteurs solution 'processor', 'additional pointer' et 'CD-Rom'.
- Le descripteur 'music' agit sur 'sound card'.
- Le descripteur 'word processing' influence 'printer'.
- Le processeur 'Programming' contrôle le choix de 'screen'.
- 'Image processing' nécessite un 'processor' puissant ainsi qu'une 'master card' adéquate.
- 'Easiness to handle' guide le choix de 'screen'.

Pour effectuer une adaptation, il est nécessaire de quantifier ces dépendances : jusqu'à quel degré la contrainte 'game' va influencer la puissance de 'processor', par exemple.

Pour nos besoins (estimation de l'adaptabilité), il n'est pas nécessaire de connaître plus de détails. Il nous suffit d'établir l'existence de ces relations de dépendance.

En général, deux situations pourraient occurer:

Situation 1:

Lorsque le problème source présente des dissimilarités avec le problème cible sur les descripteurs suivants: 'Game' et 'Programming'. Nous aurons à adapter le choix des quatre descripteurs qui leurs sont associés. Ainsi:

μ(Game) = 3.

μ(Programming) = 1.

μ(source case) = 4

Situation 2:

Lorsque le problème source présente des dissimilarités avec le problème cible sur les descripteurs suivants: 'Music' et 'Programming'. Nous aurons à adapter le choix des deux descripteurs qui leurs sont associés. Ainsi:

μ(Music) = 1.

μ(programming) = 1.

μ(source case) = 2

Dans les deux situations la difference portrait sur deux descripteur. Cependant le cas sousrce de la situation 2 est plus facile (moins coûteux) à adapter que celui de la situation 1.

5 Le Modèle de Mémoire

Dans cette section, nous proposons l'intégration du critère d'adaptabilité dans un modèle de mémoire construit selon le critère de simmilarité. Il s'agit du modèle de mémoire CRN.

Deux arguments majeurs ont motivé notre choix :

- Mémoire performante : jusqu'à 10 fois plus rapide qu'une recherche séquentielle (voir **[Len 1999]**).
- Incarne parfaitement l'hypothèse de similarité, restait donc à lui injecter le critère d'adaptabilité.

Le principe des CRNs est inspiré des réseaux de neurones et des modèles de mémoires associatives. L'idée est que le processus de rappel d'un cas ne se fait pas en parcourant un chemin dans une arborescence mais plutôt de façon reconstructive en récupérant graduellement les entités d'information constituants le cas.

Les connaissances de base dans les CRNs sont les entités d'information (IEs). Un cas est un ensemble de ces entités. Une mémoire de cas est alors : un réseau de nœuds correspondant aux IEs du domaine ainsi que de nœuds additionnels dénotant le cas (voir figure.III.5.). Les nœuds IEs sont connectés par des arcs de ***similarité*** et les nœuds cas sont accessibles à partir

des IEs les constituants à travers des arcs de ***pertinence***. Différents degrés de similarité et de pertinence peuvent être exprimés en faisant varier le poids des arcs.

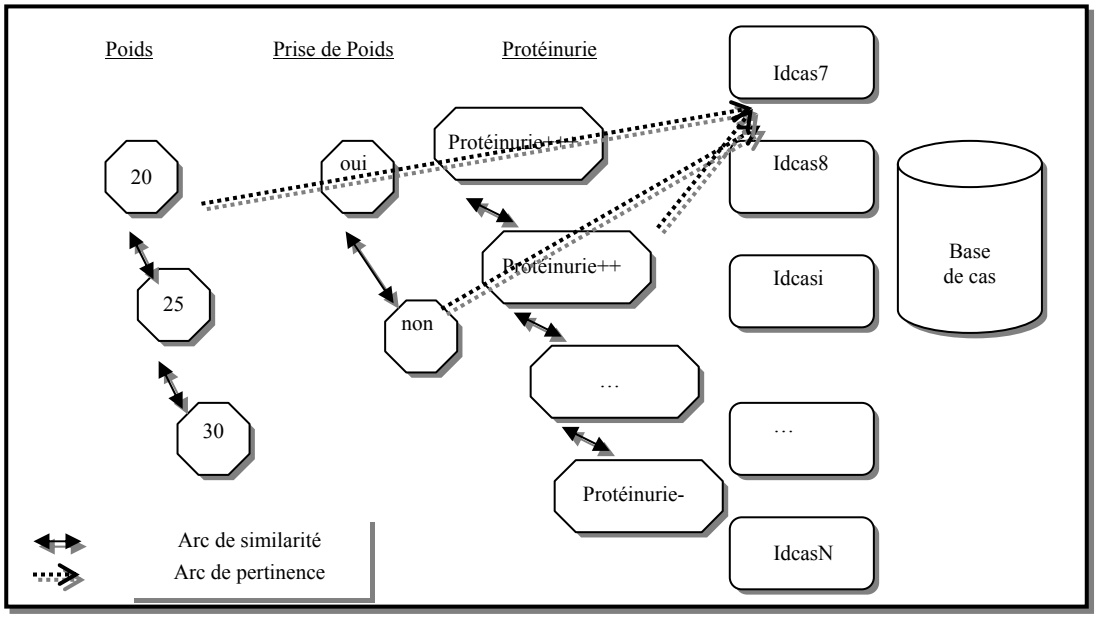


Fig.III.5. Extrait d'un CRN de l'application PEDIAGNO [Nou2002]

A partir de cette structure la recherche est faite en trois étapes :

1. Activation des IEs donnés par la requête (cas à résoudre).
2. propagation des activations correspondant à la similarité à travers le réseau des IEs,
3. collecte des activations finales dans les nœuds cas correspondants.

Le modèle CRN présente des avantages certains, néanmoins la recherche qu'il offre est guidée par la similarité ce qui ne répond pas toujours aux besoins en adaptation puisque le cas le plus similaire n'est pas forcément le plus adaptable **[Smy 1993]**.

Pour cette raison, nous prévoyons une extension des CRNs pour la prise en compte du critère d'adaptabilité.

Précisons d'abord certains points critiques pour cette étude :

- la relativité de l'adaptation : la mesure de l'adaptabilité d'un cas dépend du problème posé (dépendance du contexte). Cette connaissance n'est donc pas associée au cas stocké dans la base mais appréciée relativement au contexte d'utilisation du cas.
- Localité de l'appréciation de l'adaptabilité : En effet, elle se fait par rapport à une partie du cas (au niveau de la solution) et non à la totalité du cas (la partie problème

est sensée être proche), puisque l'objectif ici est d'adapter la partie du cas source afin de satisfaire les contraintes du cas cible.

- Il s'agit bien d'une appréciation rapide (une sorte de quick-look) dont l'objectif est d'orienter la pise de décision au stade de la recherche et non d'une estimation exacte telle que celle effectuée lors de l'étape d'adaptation.

L'appréciation de l'adaptabilité qui se fait à travers le coût d'adaptabilité $\mu(e)$ ne doit donc pas apparaître au niveau IEs d'un cas mais doit être apprécié localement puis propagé lorsqu'une requête est soumise en entrée.

La prise en compte de ce coût lors du processus de propagation se fera par l'introduction d'une fonction $\beta : \Re \times \Re \rightarrow \Re$ exprimant la combinatoire entre similarité et adaptabilité. Elle peut être définie à travers une heuristique inspirée de l'équation de réutilisation présentée auparavant.

Pourquoi cette combinatoire ?

En fait, on aurait pu réfléchir à une collecte simultanée (durant la même activation) mais séparée (dans des containers différents) des coefficients de similarité et d'adaptabilité. Pour se faire, il suffirait de considérer l'ensemble C des nœuds de cas non pas comme un ensemble de nœuds atomiques mais un ensemble de nœuds d'agrégats, permettant ainsi la collecte de deux valeurs à la fois. Cette solution à elle seule ne suffirait pas puisqu'on ne cherche pas un cas similaire uniquement ou adaptable uniquement mais les deux à la fois : ***similaire dans sa partie problème et adaptable dans sa partie solution***. Ce qui revient à une sorte de recherche multicritères.

De manière formelle, on utilisera les définitions suivantes :

Définition 1 : ***(Entité d'information)***

Une entité d'information (IE) est une connaissance atomique dans le domaine. Elle présente la plus fine granularité dans la représentation des connaissances du cas et de la requête.

Définition 2 : ***(Le cas)***

Le cas consiste en un unique descripteur (ou identificateur) de cas et d'un ensemble d'IEs. La requête est constituée uniquement d'un ensemble d'IEs.

Definition 3: (Adaptation Guided Retrieval Net: AGRN)

Un Adaptation Guided Retrieval Net (AGRN) est défini par une structure

$N = [E,C,\sigma,\rho,\beta,\Pi]$ avec:

E: est un ensemble fini de noeuds IEs.

C: est un ensemble fini de nœuds Cas.

σ: est une fonction de similarité $\sigma : E \times E \rightarrow \Re$

elle décrit la similarité $\sigma(e_i,e_j)$ entre les IEs e_i et e_j.

ρ : est une fonction de relevance ou pertinence $\rho : E \times C \rightarrow \Re$

elle décrit la relevance ρ(e,c) de l'IE e au noeud cas c.

β : exprime la combinaison de la similarité et de l'adaptabilité dans le style coût de transfert pour la réutilisation. $\Re \times \Re \rightarrow \Re$.

Π : est l'ensemble des fonction de propagation $\pi_n : \Re^E \rightarrow \Re$.

Pour chaque noeud $n \in E \cup C$

La description graphique est donnée par un graphe comportant les nœuds $E \cup C$ et les arcs dirigés étiquetés par $\sigma(e_i,e_j)$ entre le IEs e_i et e_j et ρ(e,c) du IEs e au nœud cas c.

Définition 4 : ***(L'activation d'un AGRN)***

L'activation d'un AGRN $N = [E,C,\sigma,\rho,\beta,\Pi]$ est une fonction

$\alpha : E \cup C \rightarrow \Re$

α(e) exprime l'importance de l'IE e vis à vis du problème posé (la requête).

Définition 5' : ***(Le processus de propagation dans un AGRN)***

Considérons un AGRN $N = [E,C,\sigma,\rho,\beta,\Pi]$ avec $E = \{e_1,\ldots,e_s\}$ et soit

$\alpha_t : E \cup C \rightarrow \Re$ la fonction d'activation à l'instant t. L'activation des IEs nœuds e $\in E$ à l'instant t+1 est donnée par :

$\alpha_{t+1}(e) = \pi_e(\sigma(e_1,e) \bullet \alpha_t(e_1) ,\ldots, \sigma(e_s,e) \bullet \alpha_t(e_s))$

et l'activation d'un noeud $c \in C$ à l'instant t+1 est donnée par :

$\alpha_{t+1}(c) = \pi_c(\rho(e_1,c) \bullet \beta(\alpha_t(e_1) ,\mu(e_1)) ,\ldots, \rho(e_s,c) \bullet \beta(\alpha_t(e_s),\mu(e_s)))$

Lorsqu'une requête est posée, l'activation initiale des noeuds est donnée par :

$\alpha_0(e) = \begin{cases} 1: & \text{pour les IEs e décrits dans la requête.} \\ 0 & : \text{sinon} \end{cases}$

Pour une prise en compte plus subtile des requêtes, α_0 pourrait assigner des poids à des IEs spécifiques ou un contexte d'initialisation.

Le processus de recherche des cas par propagation des activations se fera en trois étapes :

Etape 1 – Activation initiale :

α_0 est déterminé pour les IEs relativement à la requête.

Etape 2 – Propagation des similarités :

α_0 est propagée à tous les noeuds IEs e_i du réseaux :

$$\alpha_1(e) = \pi_e(\sigma(e_1,e) \bullet \alpha_0(e_1) ,\ldots, \sigma(e_s,e) \bullet \alpha_0(e_s))$$

Etape 3 – Propagation des pertinences :

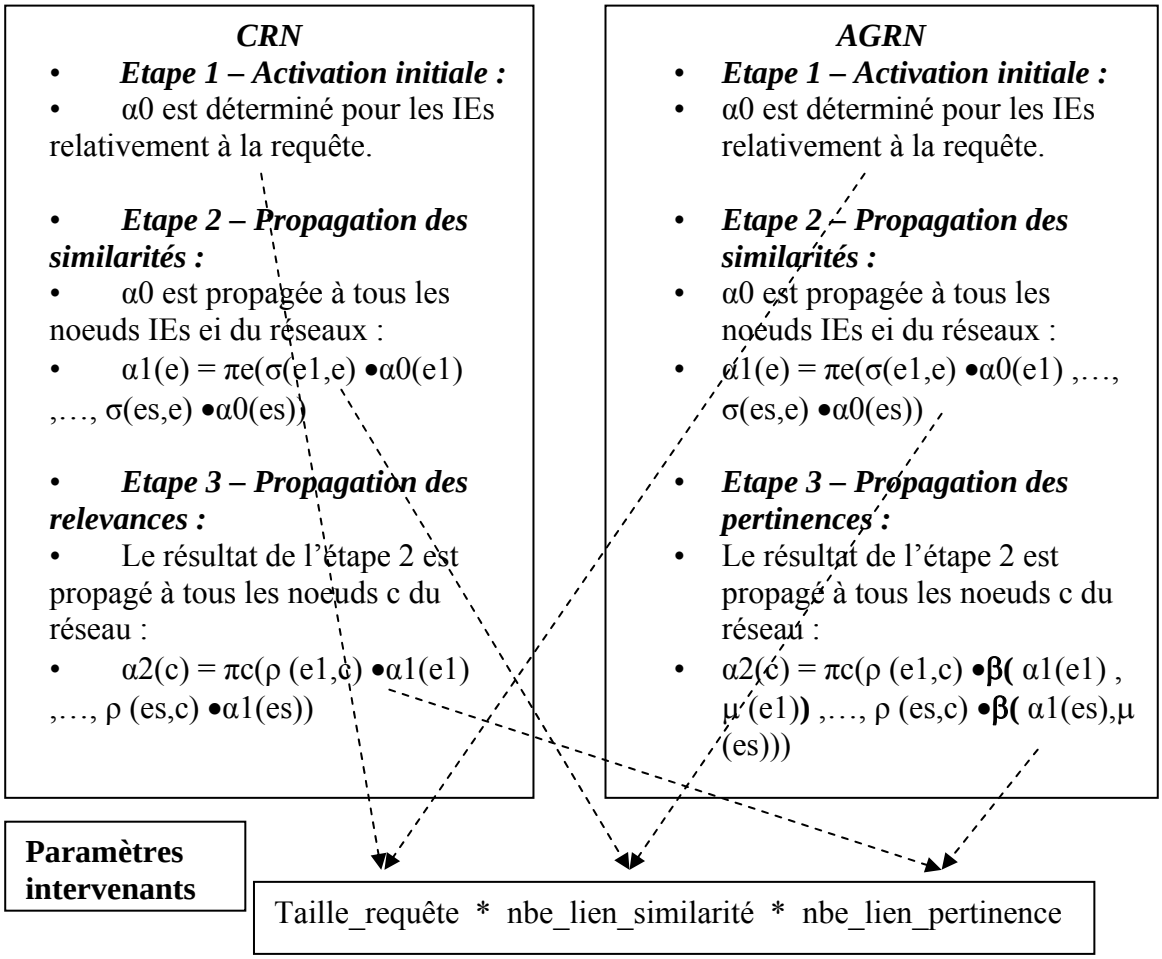
Le résultat de l'étape 2 est propagé à tous les noeuds c du réseau :

$$\alpha_2(c) = \pi_c(\rho\ (e_1,c) \bullet \beta(\ \alpha_1(e_1)\ ,\ \mu\ (e_1))\ ,\ldots,\ \rho\ (e_s,c) \bullet \beta(\ \alpha_1(e_s),\mu\ (e_s)))$$

Il s'en suit que les nœuds c collecteront non plus une mesure de similarité par rapport au cas de la requête mais une appréciation de l'utilité d'un cas c par rapport au problème posé.

7 Complexité du modèle

Notons que, le coût d'adaptabilité $\mu(e)$ est calculé relativement au problème cible. Et qu'il est, par ailleurs injecté dans le réseau, diffusé et transporté en même temps que la similarité. Les mécanismes d'activation et de propagation du CRN de base restent inchangés. Aucune complexité supplémentaire n'a été introduite. Nous pouvons d'ores et déjà nous prononcer sur les performances des AGRNs relativement à celles des CRNs .



Nous constatons bien que les mêmes paramètres interviennent aussi bien dans le modèle CRN que le modèle AGRN. La complexité est donc la même.

Elle n'est d'ailleurs pas calculable puisqu'elle dépend de plusieurs paramètres.

7 Conclusion

Dans la section 2 nous avons mis en évidence deux problèmes majeurs de l'hypothèse de similarité: la négligence du lien entre recherche et adaptation et la distorsion de la distance géométrique en passant de l'espace problème à l'espace solution. Dans la section 3, nous avons présenté un point de vue dual qui ramène la recherche à une exploration de l'espace solution plutôt que l'espace problème.

La section 4 nous a permis de réunir ces deux points de vue à travers une mise en équation de la réutilisation. Une utilisation pragmatique de cette équation a été faite dans le modèle de mémoire présenté en section 5.

Il est important de noter que le modèle de mémoire proposé constitue un cadre général d'étude. Nous ne nous sommes imposés ni restriction de domaine d'application ni contrainte de tâche de résolution. Si ce n'est le fait que le processus de résolution doit intégrer une étape d'adaptation (ce qui n'est pas toujours le cas dans les systèmes CBR) et ce pour justifier une recherche orientée adaptabilité.

Pour situer notre travail relativement à l'approche du système « Déjà-Vu » (qui constitue la référence dans les travaux AGR), Rappelons que :

- Dans « Déjà-Vu », comme dans notre approche le critère d'adaptabilité s'appuie sur une estimation rapide du coût d'adaptation.
- Dans « Déjà-Vu », l'estimation du coût d'adaptabilité passe par l'analyse des parties conditions des règles d'adaptation applicables. Ce qui donne, au moment de la remémoration, une indication supplémentaire sur l'étape d'adaptation. Cette démarche peut induire un coût de calcul supplémentaire. Or dans notre approche, la détection des relations de dépendances se fait dans une étape de prétraitement, puisque ces dépendances sont inhérentes à la description des cas sources indépendamment de tout cas cible. Elles serviront au calcul des scores au moment opportun.
- Dans « Déjà-Vu », l'approche développée pour la remémoration des cas repose non pas sur leur ressemblance avec un problème cible, mais sur une évaluation de leur facilité d'adaptation. C'est uniquement le critère d'adaptabilité qui guide la remémoration. Contrairement à notre approche, qui utilise les deux critères à la fois.
- Dans « Déjà-Vu », le processus de remémoration est réalisé en 4 étapes. Alors que dans notre approche, les deux critères sont transportés en même temps et en une passe lors de l'étape de recherche.

Chapitre 4 :

Approches d'Extraction de Connaissances d'Adaptabilité

"…Changes have to be made to a memory of the past in order to make it fit to the mold of the present…" K.J. Hammond dans **[Ham 1992]**

1 Introduction

Cette étape s'appuie sur des connaissances spécifiques. La suite de ce chapitre présente différentes techniques d'acquisition[1] de connaissances d'adaptation à travers la littérature puis Nous proposons trois approches d'extraction de connaissances d'adaptabilité.

Une étude sur l'Acquisition des Connaissances d'Adaptation (ACA) a été réalisée dans **[Lie 2004]**. L'article décrit brièvement quelques travaux du domaine, encore peu exploré, de l'acquisition des connaissances d'adaptation pour le CBR et procède à leur comparaison. Dans ce qui suit nous reprenons l'essentiel de cette synthèse pour mieux situer nos propres approches.

2 Travaux dans le Domaine

Les travaux en ACA résumés ci-dessous le sont dans l'ordre chronologique de publication de l'article de référence. En effet, les critères de comparaison plus sophistiqués qui permettraient de les ordonner sont multiples et ne seront décrits qu'à la section suivante. Par ailleurs, ces travaux sont nommés par leur premier auteur.

Hammond décrit l'approche de planification à partir de cas utilisée par le système CHEF pour la conception de recettes de cuisine. Trois types d'apprentissage sont décrits. Ils relèvent tous du *learning by remembering*, qui consiste à indexer en mémoire des expériences de résolution de problèmes de planification. Un de ces types — le *critic learning* —concerne l'ACA: il consiste en l'apprentissage automatique de moyens de réparer des échecs de plan. L'adaptation de CHEF utilise aussi un ensemble de règles d'adaptation, mais l'acquisition de ces règles n'est pas étudiée.

[1] Bien que nous n'ayons pas fait de distinction entre le terme 'extraction' (relevant de la fouille) et acquisition' (relevant de l'apprentissage), il est important de noter que les techniques présentées peuvent appartenir à l'une ou l'autre des disciplines.

Hastings décrit une technique intégrant CBR et raisonnement à partir de modèles pour la prédiction du comportement de systèmes biologiques. Il s'applique au système CARMA pour la prédiction des dégâts causés par les invasions de sauterelles. L'adaptation s'appuie sur l'influence de chaque attribut sur la solution. Si ces influences sont connues qualitativement, grâce aux experts, leurs valeurs numériques sont estimées par minimisation de l'erreur quadratique moyenne sur un échantillon de cas entre la prédiction effectuée par le système paramétré par ces valeurs et la prédiction d'un expert. Cette ACA consiste donc à raffiner le processus d'adaptation en apprenant ses paramètres.

Leake décrit le processus d'adaptation du planificateur à partir de cas : DIAL, qui s'appuie sur des principes similaires à ceux de CHEF et dont le domaine est la proposition de plans en réaction à des désastres (séismes, crues, etc.). Cette adaptation s'appuie à la fois sur des règles et des cas d'adaptation. Un cas d'adaptation est constitué d'un problème d'adaptation - le triplet (source, Sol(source), cible) - et des connaissances mises en oeuvre pour le résoudre. Deux approches d'ACA sont proposées et consistent toutes deux à acquérir des cas d'adaptation.

L'ACA automatique consiste à mémoriser les adaptations effectuées automatiquement. L'ACA supervisée consiste à mémoriser les adaptations effectuées de façon interactive avec l'utilisateur, quand l'adaptation automatique échoue.

Hanney décrit une approche d'ACA automatique s'appuyant principalement sur des couples de cas sources similaires: le résultat est une règle d'adaptation dont la prémisse représente la variation entre les problèmes comparés et la conclusion représente la variation entre les solutions comparées. Une généralisation (optionnelle) de l'ensemble de telles règles est également décrite ; elle s'appuie sur des techniques de généralisation, tel le *closing interval rules*. L'approche est testée dans deux domaines simples (estimation du prix d'un logement et prédiction du temps de montée d'un servomécanisme) ; le système de CBR muni des connaissances d'adaptation ainsi acquises donne de meilleurs résultats qu'une simple remémoration sans adaptation. Des prolongements de cette approche faisant intervenir d'autres sources de connaissances sont également décrits.

Wilke propose un cadre général pour l'ACA automatique s'appuyant sur la notion de «réservoirs de connaissances ». Comme ce travail ne décrit pas une approche particulière, il ne sera pas systématiquement comparé aux autres approches.

Anand propose une approche pour construire automatiquement une base de cas et des règles d'adaptation à partir d'une base de données (les tests se font pour une base de logements avec leurs caractéristiques et leurs prix). Des regroupements dans cette base

permettent de construire des concepts dans lesquels sont choisis des cas représentatifs, qui forment la base de cas. Les connaissances d'adaptation sont construites à partir de cette même base de données grâce à des techniques de génération de règles, en particulier un algorithme de découverte de règles d'association.

Cho décrit une approche d'ACA automatique très proche de celle de Hanney (mais sans y faire référence). Elle est testée sur le domaine de l'estimation du prix d'un voyage.

Corchado utilise un réseau de neurones artificiel de type RBF (*radial basis function*) pour effectuer l'adaptation: le réseau est entraîné par un ensemble de cas sources sélectionnés lors de la phase de remémoration (par la méthode des *k* plus proches voisins) et est appliqué à la résolution du problème cible: *cible* est donné à l'entrée du réseau et Sol(*cible)* est récupérée à sa sortie. Le domaine d'application est la prédiction de la température de l'océan à une profondeur fixée, pour les prochains kilomètres parcourus par un navire. On peut considérer que ce processus effectue bien une ACA: la fonction d'adaptation est bien apprise, et ceci, à partir des cas sources. La différence fondamentale avec les autres travaux étudiés dans cette section est que le résultat de cet apprentissage n'est pas retenu: il ne sert qu'au niveau d'une session particulière de CBR.

Lieber présente une approche d'ACA auprès d'experts, dans le cadre de l'application KASIMIR d'aide à la décision thérapeutique en cancérologie. Cette approche s'appuie en particulier sur des comptes-rendus informels d'adaptations effectuées par des experts et permet notamment de mettre en évidence des schémas d'adaptation. Un apport de ce travail est la décomposition d'une adaptation complexe et particulière en adaptations simples et généralisables.

Jarmulak décrit une approche d'ACA automatique à partir de la base de cas. Elle est appliquée et testée sur un problème de conception de cachets (quelle proportion des ingrédients -produit actif, excipient, etc. - choisir pour un nouveau médicament ?). Comme Leake, elle apprend des cas d'adaptation, plutôt que des règles d'adaptation générales. La description d'un cas d'adaptation construit à partir de deux cas sources contient une représentation des dissimilarités entre leurs problèmes et leurs solutions, ainsi que le problème et la solution d'un des deux, considéré comme le cas de référence.

Lee décrit une approche d'ACA automatique très proche de celle de Cho (et donc, de Hanney, mais sans y faire référence non plus). Elle est testée sur un *benchmark* pour l'estimation du prix d'une automobile.

D'Aquin présente un travail préliminaire pour l'ACA automatique à partir de la base de cas qui s'appuie sur des principes d'extraction des connaissances des bases de données et qui

sera testé pour le système KASIMIR. L'étape centrale de fouille de données applique la technique d'extraction de motifs fréquents dans les bases de données, qui permet notamment de construire des règles d'association. Elle est précédée d'un formatage qui traduit les couples de cas en ensembles de propriétés booléennes. Elle peut être suivie d'une étape d'interprétation (interactive) avec l'expert, pour produire des règles plus générales et dans un langage plus expressif.

Comparaison de travaux en ACA

Le but de la comparaison ci-dessous est de répondre aux trois questions suivantes, pour chacun des travaux en ACA étudiés :

– Quelles sources de connaissances sont utilisées?

– Quelles sont les hypothèses sur la représentation des cas?

– Quels sont les types de connaissances d'adaptation acquises?

Comparaison selon les sources de connaissances utilisées

Dans cette section, **[Lie 2004]** compare les ACA selon les sources de connaissances utilisées. Par *source de connaissance*, nous entendons n'importe quelle source permettant la construction de connaissances, pas nécessairement une base de connaissances. Cela peut être, par exemple, une base de données. **[Bar 1999]** présente quatre *réservoirs de connaissances* (*knowledge containers*) pour le CBR : le vocabulaire (V), la base de cas (BC), les connaissances de remémoration (CR) et les connaissances d'adaptation (CA). Nous définissons informellement le vocabulaire comme étant l'ensemble des éléments de représentation atomiques utilisés pour représenter les cas de la base de cas.Pour acquérir des connaissances d'adaptation, il est difficile de ne pas utiliser, d'une façon ou d'une autre, ce vocabulaire de description des cas. Quoi qu'il en soit, aucune des approches étudiées ne s'en passe, ce n'est donc pas un critère de différenciation de ces approches. BC est utilisé de deux façons très différentes par les approches d'ACA étudiées : soit pour fournir des couples de cas (BC. BC), soit pour fournir des cas sources comme références de comparaison à un problème cible (BC*réf.*). CR contient essentiellement des connaissances utiles à l'estimation de la différence (similarité et dissimilarité) entre deux cas; cette source de connaissances peut être une mesure de similarité. CA représente les connaissances d'adaptation déjà disponibles, avant le processus d'ACA (dont l'objectif est d'enrichir CA). Certaines approches tirent partie de connaissances pour la résolution directe d'un problème, ou de connaissances sur la *qualité* d'une solution vis-à-vis d'un problème (CRP/Q) ; ces deux sources de connaissances lient

l'espace des problèmes et l'espace des solutions de façon globale (indépendamment de cas particuliers), c'est pourquoi nous les avons regroupées. Enfin, une sixième source de connaissances est également à considérer: c'est l'expert (E). Nous appellerons ACA *supervisée*, celle qui s'appuie sur E et ACA *automatique*, une ACA qui n'utilise pas E. La table.IV.1 présente les travaux résumés auparavant selon les sources de connaissances exploitées
par le processus d'ACA.

	Hammond	Hastings	Leake/a	Leake/s	Hanney	Anand	Cho	Corchado	Lieber	Jarmulak	Lee	d'Aquin
BC×BC	−	−	−	−	++	−[a]	++	++[b]	−	++	++	++
BC*réf.*	−	+	+	+	−	−	−	++[b]	+	−	−	−
CR	−	+[c]	−	−	+	+	+[d]	+	−	+	+	±
CA	−	+[e]	++	−	±	±	−	−	−	−	−	±
CRP/Q	++	−	+	−	−	−	−	−	−	+	−	−
E	−	++[f]	−	++	−	−	−	−	++	−	−	±
autre	−	−	−	−	±[g]	++[h]	−	−	−	−	−	−

++ : source de connaissances principale pour cette approche
+ : source de connaissances utilisée par cette approche
± : source de connaissances pouvant être utilisée ou pas par cette approche (« optionnelle »)
− : source de connaissances non utilisée par cette approche

TAB .IV.1. Travaux en ACA et sources de connaissances de [Lie 2004]

*a: m*ais l'ACA s'appuie sur la base de données dont la base de cas est un sous-ensemble.

*b: i*l est difficile, pour de distinguer BCx BC de BC*réf.*. On fait, en tout cas, appel à plusieurs cas de la base de cas pour un processus d'adaptation qui peut être considéré en même temps comme une ACA.

c: l 'ACA s'appuie sur les poids de la mesure de similarité, mais ceux-ci sont également estimés automatiquement.

d: défend l'idée qu'il ne s'appuie que sur la base de cas pour mettre en évidence les connaissances d'adaptation. Néanmoins, il utilise une distance entre cas. Même si celle-ci est qualifiée de *syntaxique*—par opposition à une distance qui serait *sémantique*, i.e., qui tiendrait compte, en particulier, de la pertinence relative des descripteurs —nous avons considéré qu'il y avait bien prise en compte de connaissances de remémoration.

*e: l*e processus d'adaptation dépend de l'ensemble de paramètres à estimer.

f: l'expert fournit un jeu de test.

*g: u*n prolongement de l'apprentissage intégrant des « connaissances du domaine » est étudié. Outre CA, il y a les connaissances exprimées sous forme de règles exprimant la non pertinence d'attributs, des dépendances contextuelles et des interactions entre attributs.

*h: l*a base de données dont est extraite la base de cas (cf. note *a*).

Notons que la majorité des approches étudiées s'appuient sur des « cas d'adaptation » (à l'exception de Hammond, Hastings, Anand et Corchado). Cependant, alors que ces cas

d'adaptation sont la description de processus d'adaptation pour les approches Leake et Lieber, ce ne sont, pour les approches automatiques, Hanney, Cho, Jarmulak, Lee et D'Aquin que des reconstitutions de cas d'adaptation à partir de la base de cas: on prend deux cas de la base, un que l'on considère comme cas source et l'autre comme cas cible, et on considère que le cas d'adaptation est le processus à apprendre qui permet de passer du cas source au cas cible et dont on ne connaît que l'état initial *(source, Sol(source))* et l'état final *(cible, Sol(cible))*.

Comparaison selon la représentation des cas

Certaines approches d'ACA automatique supposent que les cas sont représentés sous la forme de n-uplets de valeurs simples (on a un ensemble *A* d'attributs, chaque cas est représenté par un ensemble de couples *(a,v)*, pour tout *a ϵ A* où ´ *v* est une valeur d'un type simple, i.e., énuméré ou numérique). C'est le cas pour Hasting, Hanney, Cho, Corchado, Jarmulak et Lee. C'est presque le cas pour Anand, sauf que l'ensemble des attributs peut varier d'un cas à l'autre (mais est toujours sélectionné dans un ensemble d'attributs prédéfini correspondant à des colonnes de relations de la base de données).

L'approche D'Aquin suppose qu'il existe une procédure de formatage permettant de traduire chaque cas sous la forme d'un ensemble de propriétés booléennes. Une telle traduction est proposée pour un formalisme de représentation analogue à une logique de descriptions simple. Les approches de Hammond et Leake/a utilisent le formalisme de représentation des connaissances par des *memory organisation packets* (MOPs).

Les approches supervisées Leake/s et Lieber ne s'appuient pas sur des hypothèses concernant le formalisme de représentation des cas. Pour Lieber, c'est même l'inverse: cette approche permet de mettre en évidence des schémas d'adaptation dont l'implantation induit de nouveaux besoins en représentation des cas.

Types de connaissances acquises

Le cadre général d'ACA automatique proposé par Wilke n'impose pas un type de connaissances d'adaptation. Cela peut être, par exemple, le nombre de cas à remémorer afin d'être adaptés/combinés pour résoudre le problème cible (on pourrait discuter du fait que ce nombre constitue une connaissance d'adaptation). L'approche de Hastings consiste à apprendre les paramètres numériques d'un processus d'adaptation. L'approche de Corchado consiste à entraîner un réseau de neurones artificiel RBF, ce qui implique l'apprentissage à la fois de poids de ce réseau et d'éléments de sa structure. Pour les autres travaux étudiés, les connaissances d'adaptation sont sous la forme de règles (ou de reformulations). Les

connaissances d'adaptation acquises chez Leake et Jarmulak, sont sous la forme de cas d'adaptation (l'adaptation est elle-même un processus CBR, selon le principe du *recursive* CBR), mais ces cas d'adaptation peuvent être vus comme des règles particulières. La forme générale d'une règle d'adaptation est :

Si *C(source,Sol(source),cible)*

Alors *Sol(cible)* ← *M(source, Sol(source), cible)*

A priori, *C* et *M* dépendent des trois paramètres *source, Sol(source),* et *cible*. Les approches supervisées (Leake/s et Lieber) ne font pas d'hypothèses supplémentaires. En revanche, les approches automatiques supposent généralement que, pour la condition *C*, le couple *(source, cible)* est modélisé par la dissimilarité et la similarité (aussi appelée, chez Hanney, le « contexte ») entre ces deux problèmes. Concernant la dissimilarité, toutes les approches d'ACA automatique étudiées en tiennent compte, mais cela, de deux façons différentes. Soit la dissimilarité consiste simplement à indiquer les éléments à rajouter ou à supprimer de *source* pour obtenir *cible* (nous la noterons sous la forme $x \rightarrow y$, pour dénoter le remplacement d'une valeur *x* par une valeur *y*). Soit la dissimilarité est « abstraite », au sens où elle décrit la transformation dans un langage différent qui permet l'expression de conditions plus larges (par exemple, la dissimilarité $6 \rightarrow 4$ sera représentée par la dissimilarité abstraite - *2*, qui pourra rendre compte d'autres dissimilarités $x \rightarrow y$, telles que $8 \rightarrow 6y$. Pour *M*, les approches automatiques étudiées sont soit « numériques » (addition ou soustraction d'un nombre à une valeur numérique d'attribut) soit « nominales » (qu'on peut voir comme l'ajout ou la suppression de variables propositionnelles dans une conjonction). La table .IV.2 présente les travaux d'ACA automatique étudiés selon ce critère (à l'exception de Leake/a pour laquelle l'auteur dans **[Lie 2004]** manque d'information). Notons enfin que pour Anand et D'Aquin, il est possible en plus d'associer une mesure de confiance aux règles acquises (ces deux approches s'appuient sur l'extraction de règles d'association, technique de fouille de données qui utilise une telle mesure).

		Hanney	Anand	Cho	Jarmulak	Lee	d'Aquin
$\mathcal{C}$	dissimilarité $x \rightarrow y$	+	−	+	−	+	+
	dissimilarité abstraite	+	+	−	+	−	−[b]
	similarité	−[a]	+	−	+	−	+
	Sol(srce)	−	−	−	+	−	+
$\mathcal{M}$	numérique	+	+	+	+	+	−[b]
	nominal	+	−	−	+	−	+

TAB.IV. 2. Informations apparaissant dans les règles d'adaptation acquises par les approches d'ACA automatique étudiées générant de telles règles (extrait de [Lie 2004]).

a : La prise en compte de la similarité n'est pas complètement absente, mais elle suppose une source de connaissances supplémentaire : les connaissances sur les « dépendances contextuelles », qui sont représentées sous la forme d'un ensemble de « contraintes contextuelles ».

b : La phase d'interprétation interactive devrait permettre d'avoir des règles plus expressives, notamment, avec dissimilarité abstraite et modifications numériques.

Dans **[Lie 2004]**, trois critères de comparaison sont utilisés. Le premier compare les sources de connaissances utilisées par les différentes approches. Il apparaît que deux sources principales émergent et séparent les approches supervisées des approches automatiques : les premières s'appuient (évidemment) sur l'expert, les secondes, en général, sur la base de cas.

Le deuxième critère porte sur la représentation des cas. Il semblerait que, en général, les approches automatiques étudiées supposent que les cas sont soit représentés dans un formalisme simple (du type ensemble de couples attribut-valeur simple) soit, dans une phase préliminaire, traduits dans un formalisme simple (exceptions : Hammond et Leake qui utilisent le formalisme des MOPs).

Les approches supervisées, en revanche, ne nécessitent pas d'hypothèse sur la représentation des cas; elles relèvent en effet plutôt de méthodologies que de technologies, alors que les approches automatiques sont des techniques.

Le troisième critère est le type de connaissances acquises. Elles sont en général sous la forme de règles d'adaptation, qui peuvent être *a priori* quelconques pour les approches supervisées. Pour les approches automatiques, la prémisse d'une telle règle contient toujours des informations sur la variation entre les problèmes comparés (leur dissimilarité) et parfois (pour certaines approches) des informations sur le contexte (i.e., sur ce que ces problèmes doivent partager pour que la règle s'applique), voire des informations sur la solution du problème résolu.

Rappelons que nous cherchons à détecter les relations de dépendances entre problème et solution dans les cas sources. Il s'agit bien d'une détection sans quantification ni qualification aucune (du moins à ce stade de la recherche).

Dans ce qui suit, nous abordons cette détection de trois manières différentes :

3 Approche Heuristique

Elle est fondée sur l'analyse des règles d'adaptation, afin de déterminer les relations de dépendances existant entre partie condition et partie conséquence.

Exemple :

On se propose d'analyser quelques règles d'adaptation de l'exemple cité en chapitreIII. Soient :

Règle1 Si (source.jeu > 7) et (cible.jeu <3)
Alors solution.supprimer(Joystick)

Règle2 Si (source.musique <3) et (cible.musique >7)
Alors solution.ajouter(carte_son_haut_de_gamme)

Règle3 Si (source.puissance < 3) et (cible.puissance >7)
Alors solution.modifier(processeur.vitesse,delta_puissance,+)

La Règle 1 établit qu'une variation de la valeur du descripteur 'jeu' entre source et cible implique la suppression de 'joystick'. Ce qui signifie en clair que le descripteur solution 'joystick' dépend du descripteur problème 'jeu'.

La Règle 2 établit qu'une variation de la valeur du descripteur 'musique' entre source et cible implique l'ajout de 'carte son'. Ce qui signifie en clair que le descripteur solution 'carte son' dépend du descripteur problème 'musique'.

La Règle 3 établit qu'une variation de la valeur du descripteur 'puissance' entre source et cible implique la modification de 'vitesse processeur'. Ce qui signifie en clair que le descripteur solution 'vitesse processeur' dépend du descripteur problème 'puissance'.

Nous remarquerons que la nature de la variation (croissante ou décroissante) nous indique le type d'action à entreprendre (ajout, suppression ou modification).

4 Approche Analyse de Données

Si l'on considère la description d'un cas comme étant un vecteur de couples attributs/valeurs. Ces couples peuvent soient appartenir à la description du problème ou à la description de la solution. L'étude de la relation de dépendance entre un descripteur problème et un descripteur solution revient à analyser les dépendances entre les valeurs correspondant à chacun des descripteurs.

En termes d'analyse de données, il s'agirait d'établir un lien ou une explication entre deux variables (nos deux descripteurs) moyennant leurs valeurs. La variable correspondant à la description du problème jouera le rôle de la variable explicative. La variable correspondant à la description de la solution jouera le rôle de la variable expliquée

Le choix de la méthode d'analyse dépendra du type des variables intervenant. Le tableau tab.IV.3. donne un panorama des méthodes utilisables

Variable explicative \ *Variable expliquée*	V2 Nominale	V2 Numérique
V1 Nominale	Correspondance Tableaux d'effectifs (tableaux croisés) Carte AFC Test chi²	Comparaison tableau de moyennes analyse de la variance Test de Fisher
V1 Numérique	Comparaison tableau de moyennes analyse de la variance Test de Fisher	Corrélation modèle de régression nuage de points Test de corrélation

Tab.IV.3. Tests de l'analyse bivariée (extrait de [Mas91]).

Nous trouvons cette approche de la question intéressante puisqu'elle ne s'appuie sur aucune connaissance experte comme c'est le cas pour l'approche heuristique qui suppose l'existence préalable des règles d'adaptation. C'est pour cette raison que nous lui avons réservé une partie de nos expérimentations (Application III du chapitre V).

5 Approche Base de données

On peut considérer les relations de dépendance issues du carré d'analogie comme étant équivalente à la notion de dépendance fonctionnelle dans la théorie des bases de données **[Sma 1994]**. Rappelons qu'en base de données, si un attribut B dépend fonctionnellement d'un attribut A (A→B), on veillera alors à ce que deux tuples de relation ayant la même valeur pour A aient la même valeur pour B (le concept de contrainte d'intégrité). Ces dépendances fonctionnelles sont par ailleurs, utilisées pour la conception de schémas relationnels normalisés permettant d'éviter les redondances pouvant nuire à la cohérences des données. Elles peuvent être sémantiques (le type d'un avion détermine sa capacité et son constructeur) ou être le fruit de contraintes plus ou moins artificielles (Par exemple, le numéro d'un avion détermine tous les attributs définis dans la relation avion).

Ces dépendances sont généralement résumées dans un graphe de dépendances fonctionnelles (GDF) qui établie toutes les dépendances entre attributs, qu'elles soient sémantiques ou artificielles.

Pour notre part, ce sont uniquement les dépendances sémantiques qui nous intéressent et lorsqu'elles existent entre un attribut appartement à la description du problème et un attribut appartenant à la description de la solution.

En effet, si l'on considère, cette fois-ci, la description d'un cas comme étant une relation entre attributs Ces attributs peuvent soient appartenir à la description du problème ou à la description de la solution. L'étude de la relation de dépendance entre un descripteur problème et un descripteur solution revient à construire le sous graphe de dépendances fonctionnelles entre les attributs problèmes et les attributs solution.

Dans la littérature, on retrouve de multiples travaux portant sur les techniques de construction automatique des GDF. Nous envisageons de récupérer la sortie de l'une de ces techniques pour en extraire le sous graphe qui nous intéresse.

6 Conclusion

Il est important de noter que la nature des connaissances recherchées dans l'approche proposée en chapitre III, diffère de celle présentée dans la section 2.

Par ailleurs, nous avons opté pour un style d'adaptation transformationnelle qui rappelons le, fonctionne comme suit : Des éléments de la solution du cas retrouvé sont modifiés, supprimés ou ajoutés ; selon des écarts de contexte observés entre cas source et cas cible, et grâce à un ensemble de règles d'adaptation.

L'adaptation générative quant à elle, travaille sur la trace du« raisonnement » ayant mené à la solution. On substitue les éléments de contexte du raisonnement retrouvé par les éléments différents du contexte du cas nouveau. On « rejoue » le raisonnement dans ce nouveau contexte. Ce style ne nous intéresse pas, puisqu'il s'agit de l'adaptation non pas de la solution mais de la démarche ayant mené à sa génération.

Pour l'heure, les approches proposées en section 3, 4 et 5 se limitent à mettre en évidence l'existence ou non des relations de dépendance, leur quantification ne nous intéresse pas. Cependant, cette connaissance pourrait donner lieu à d'autres utilisations possibles telle que la pondération des descripteurs.

Chapitre 5 :

Evaluation et Résultats

Comme tout modèle doit être expérimenté, le présent chapitre constitue un cadre d'expérimentation et d'argumentation des chapitres 3 et 4.

1 Plan d'évaluation

Pour mieux évaluer notre approche aussi bien dans son ensemble que dans ses parties, nous allons d'abord rappeler les objectifs de modélisation fixés :

- Un modèle de mémoire de cas supportant les critères de similarité et d'adaptation à la fois.
- Fournir un coût d'adaptation sans effectuer l'adaptation.
- L'estimation du coût d'adaptation se fera moyennant l'extraction de connaissances expertes à partir de la base de cas elle-même

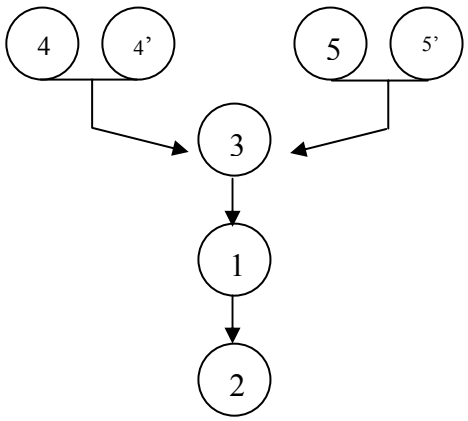
A partir de là se sont dégagés les objectifs d'évaluation suivants :

1. Implémenter un modèle de mémoire CRN.
2. Implémenter l'extension du CRN en AGRN.
3. Implémenter un outil de datamining

Le tout devant être réalisé dans un cadre applicatif particulier :

4. une tâche de résolution donnée
5. un domaine d'application précis

Nous avons ordonnancé ces objectifs comme suit :



Remarques :

- Nous avons opté pour deux variantes de la tâche et du domaine, d'où 4' et 5'.

- L'étape 3 a été appliquée à d'autre data set en dehors de ceux collectés dans 5 et 5'
- L'étape 1 a été implémentée séparément pour pouvoir démontrer les performances similaires de l'AGRN et du CRN.

Dans la suite de ce chapitre, nous allons détailler les objectifs d'évaluation à travers trois applications[1] :

1. Une application dans le domaine du choix de configuration d'ordinateur PC_CONFIG **[Nou 2005a]**.
2. Une application dans le domaine de la diététique : GUIDIETE **[Nou 2005b]**.
3. Une application de datamining pour l'extraction du coût d'adaptabilité : AdaX **[Nou 2005c]**.

Cette dernière devrait un outil utilisé pour les deux premières applications. Cependant, nous la présenterons en dernier pour qu'il n'y ait pas de redondances dans la présentation de la matière utilisée.

2 Application I : PC_Config.

2.1 Méthode et Matière

Le système PC_CONFIG permet de rendre la tâche de choix d'un PC une tâche accessible pour une gamme d'utilisateurs (novices, experts…), qui expriment souvent des besoins divers.

PC_CONFIG est un support de vente intelligent des ordinateurs où le rapport entre la description de l'utilisateur et la solution recommandée est en termes de mesures de similarité et d'adaptabilité. Son fonctionnement s'appuie sur :

- une base de connaissances (des composants de PC).
- une recherche des produits avec un recouvrement à base de cas à travers un réseau de recherche de cas.
- personnalisation de la recommandation des produits.

Les entrées /sorties du système :

[1] La complexité de chaque objectif fait qu'il ne peut être présenté qu'à travers une application.

L'utilisateur spécifie les applications qu'ils souhaitent exploiter par un système de notation qui indique ses préférences : ex une note de 10/10 pour jeux signifie que la configuration doit être orientée jeux. Par contre une note de 4/10 pour programmation signifie que l'on cherche une satisfaction moyenne de ce critère.

Par ailleurs, d'autres caractéristiques techniques peuvent être données en entrées notamment si le client est expert, telles que : Vitesse CPU, Taille RAM,…

A la sortie, le système fournit une liste de produits recommandés et adaptés de sorte à satisfaire le client.

Le Cas :

Dans ce système le cas est une offre de configuration d'ordinateur, il décrit le PC avec tous ce qui peut le caractériser.

Un cas est constitué d'un ensemble de descripteurs au nombre 36, ces descripteurs sont partagés en deux parties :

1. la description du problème :

Cette partie est composée d'un ensemble de descripteurs reliés aux besoins des utilisateurs telle que : bureautique, jeux, musique, traitement d'image, multimédia, programmation et le prix.

Le prix : c'est un descripteur de valeur numérique limité dans l'intervalle [374,3790] On répartie notre échantillon en quatre intervalle en utilisant la terminologie du domaine : Gamme faible, Gamme moyenne, Gamme chère, Gamme très chère

2. la description de la solution :

Elle est faite en terme de matériel nécessaire à la configuration tel que mentionné dans le tableau V.1.

Le système utilise la structure de CRN/AGRN. Les descripteurs dans le CRN/AGRN sont aux nombres 18, à savoir :

Bureautique, multimédia, jeux, traitement d'image, montage vidéo, programmation, Musique, Internet, la marque de PC , vitesse de CPU, Marque CPU,carte mère, carte graphique, carte son, taille de RAM, taille de disque, type d'écran, taille d'écran et le prix.

Descripteur de cas	Les valeurs
Carte mère	Asus A7N8X-X, Gigabyte GA-I8 PE, P5GD1-VM…
Marque de processeur	AMD, INTEL
Vitesse de processeur	2, 2.2 ,2.4, 2.5, 2.6, 2.8, 3, 3.2, 3.4, 3.6
Carte graphique	ATI Radeon 9600 Xt 128Mo, Sapphire ATI Radeon 9800 256mo Geforce Fx 5200 128Mo, Geforce 6800 GT 256Mo…
Carte son	Intégrée à la carte mère, high definition audio, M_audio delta audio phile, codec audio AC '97,…
Type De Ram	Pc4200, PC3200, PC 2700, PC 2100…
Taille de RAM	256 Mo, 512 Mo, 1Go
Taille de disque	40, 60, 80, 120, 160, 200,250 GO
Marque de Pc	IBM, HP Compaq, SONY, Packard BELL, HEWLLETT PACKERD, FIJUTSU SIEMENS, assemble…
Mémoire cache	256mo, 512mo, 1go
Type d'écran	LCD, CRT
Taille d'écran	15, 17, 19
Clavier	Logitech ultra flat, Standard, sans File
Souris	Logitech MX510, Standard, fournis avec le clavier…
Modem	Modem interne Olitec PCI V92
Lecteur disquette	Lecteur disquette1.44MO
Lecteur CD	Lecteur CD 52X, Lecteur CD 48X
Lecteur DVD	Lecteur DVD TOSHIBA, Lecteur DVD plextor,…
Type de Graveur	Graveur DVD IDE Multimédia 3540, Graveur CD …
Type de d'imprimante	Imprimante Laser N/D, imprimante canon PiXma5100…
Type de scanner	Scanner canon Lide 510,…
Option jeux	Joystick, tactique ball…
2éme disque	2x74, 160
Type de châssis	Moyenne, mini, micro, tout en un, grande
Enceintes	Al Tec Lansing 221 5.1, altec Lansing 212 2.1, logitech 510 5.1 …
Autres	Carte tuner TV carte d'acquisition vidéo…

Tab.V.1. Descripteurs solution pour PC_CONFIG.

Les connaissances utilisées:

Les connaissances de similarité:

Concernant les descripteurs numériques, comme la distance entre deux valeurs reflète par dualité la similarité entre elles, nous utiliserons donc la métrique de distance dont l'expression est la suivante: se base sur la somme pondérée des distances entre descripteurs :

$$D(cas_i, cas_j) = \sum_{k=1}^{N} \alpha_k \cdot d_k(des_k^i, des_k^j) \quad (1)$$

où :

N : nombre total de descripteur du cas.

α_k : poids reflétant l'importance du k^ième descripteur dans un cas.

des_k^i : valeur du k^ième descripteur pour le cas i.

des_k^j : valeur du k^ième descripteur pour le cas j.

$d_k(des_k^i, des_k^j)$: similarité entre les descripteur des_k^i et des_k^j.

Il s'agit de la distance partielle entre deux cas du point de vue d'un descripteur particulier.

$$d_k(des_k^i, des_k^j) = \left| des_k^i - des_k^j \right| \quad (2)$$

Grower (dans **[Did 1982]**) propose l'utilisation d'un indice de similarité normalisé : (exprimé en terme de cas).

$$d_k(des_k^i, des_k^j) = 1 - \frac{\left| des_k^i - des_k^j \right|}{R_k} \quad (3)$$

où :

R_k : est l'écart maximum entre les valeurs prises par le descripteur des_k. Ainsi, toutes les distances d_k prennent des valeurs comprises entre 0 et 1 et on peut affecter à tous les α_k la valeur 1. Cette dernière expression (3) de la distance, nous évitera l'utilisation du poids α_k dans la mesure de ressemblance, au risque qu'une imprécision de poids n'entraîne une imprécision dans la ressemblance. Ainsi, plus la valeur de $D(cas_i, cas_j)$ est grande plus les cas_i et cas_j sont proches.

Par ailleurs, pour les descripteurs symboliques des heuristiques de similarité ont été extraites du domaine. Nous en citons un exemple ici :

Les heuristiques dégagées pour les carte mère : on doit utiliser 8 paramètres:

-Chipset

-Socket

-Ram max

-Type de réseau

- Les ports (PCI, USB, SATA, DMA).

On se propose de pondérer ses paramètres comme suit:

α (Chipset) = 0.4

Si chipset. source= chipset.cible **alors** Sim(source, cible) =1.

Sinon Sim(source, cible) =0.

On a affecte ce poids au chipset car il est un composant principale dans la carte mère dont son rôle est d'interconnecter et de faire dialoguer les différents composantes d'extension. Il conditionne notamment les performances.

α (Socket) = 0.2 car il détermine la compatibilité du CPU à la carte mère.

α (Ram_max)= 0.1

α (Type_Reseau)=0.1.

ET les 0.20 restants seront partagées entre les ports suivants : USB, PCI, SATA, DMA.

D'autres heuristiques sont citées en annexe 2.

Les Connaissances de Relevance :

Dans le modèle de CRN/AGRN, la relevance exprime le degré d'importance d'une entité dans la détermination d'un cas. Plus l'entité est importante plus la relevance est élevée. Pour notre système, et vu le manque d'information nécessaire pour déterminer les relevances, on choisi d'associer une relevance binaire égale 1 si l 'entité appartient au cas et 0 sinon.

Les connaissances d'adaptation :

Nous avons utiliser une base de règles qui couvre les besoins d'utilisateur ainsi les modifications possibles des composants. La liste complète est donnée en Annexe.2. Voici quelque exemples

- Si (Cible. Multimédia > 7) Et (Source. Multimédia <3) alors Modifier (Carte graphique avec TV Out).

- **Si** (Cible. Bureautique > 8) **ET** (Source. Bureautique <4) **alors** :
  - Ajouter (Modem interne).
  - Modifier (Ecran 17).
  -Ajouter (Souris optique).
  -Ajouter Imprimante.
- **Si** Ajouter (Pièce) **alors** Prix PC= Ancien Prix+Prix Pièce.

Les connaissances d'adaptabilité:

A prés avoir dégager les connaissances d'adaptation, on doit déterminer les coûts d'adaptabilité. Qui sont des scores établi sur la base du nombre d'opérateurs d'adaptation applicables qui indique la dépendance entre le problème et la solution. Voici quelques exemples illustratifs:

Pour l'attribut bureautique le coût d'adaptabilité égale à 4 d'après la règle 1 vue précédemment.

Pour l'attribut multimédia le coût d'adaptabilité égale à 6

Pour l'attribut jeux le coût d'adaptabilité égale à 4

Pour l'attribut musique le coût d'adaptabilité égale à 3

Pour l'attribut Internet le coût d'adaptabilité égale à 2

Un tableau récapitulatif des dépendances extraites à partir des règles d'adaptation est donné en Annexe.2.

2.2 Résultats et Interprétation

Les résultats qui suivent (de Fig.V.1. à Fig.V.9), illustrent la distribution des cas par rapport aux descripteurs du problème en d'autres termes la qualité de la couverture de la base de cas utilisée. Notons que la base actuelle contient 108 cas.

Concernant l'attribut Bureautique, la distribution des cas est de qualité moyenne

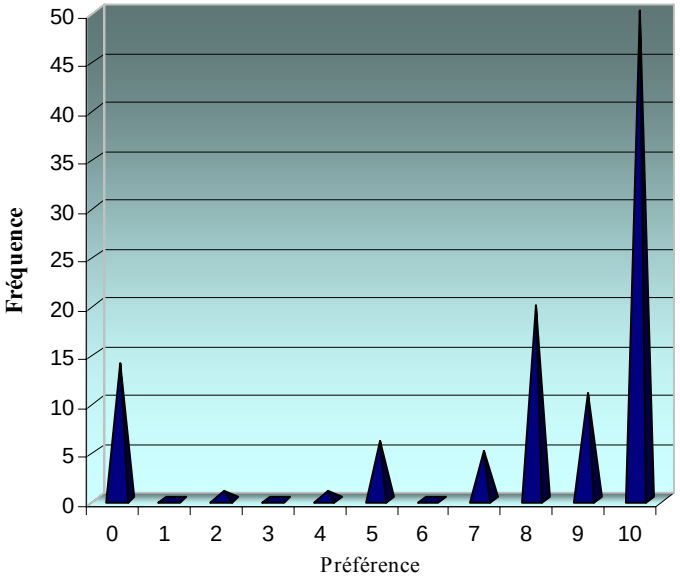


Fig.V.1.Répartition de la base de cas par rapport à l'attribut bureautique

De même que pour l'attribut multimédia.

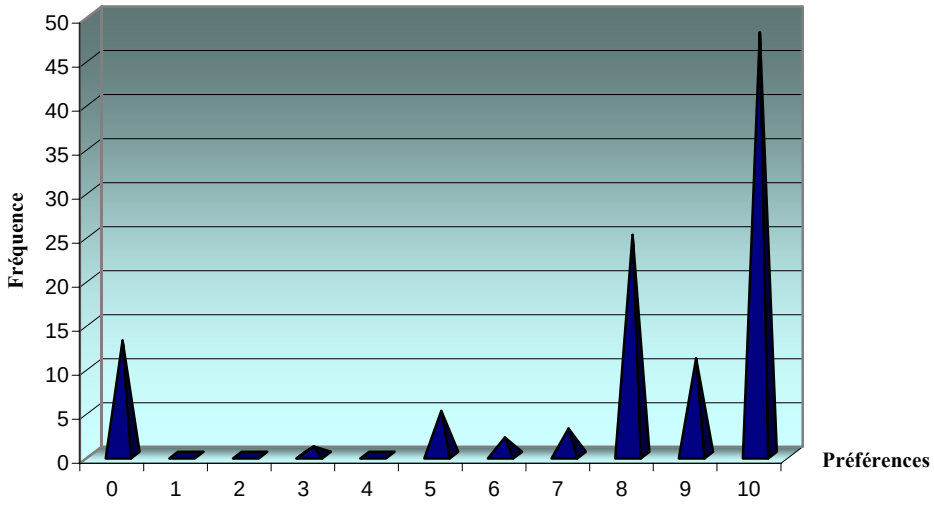


Fig.V.2.Répartition de la base de cas par rapport à l'attribut multimédia

Pour l'attribut montage vidéo, la couverture est jugée de très bonne qualité.

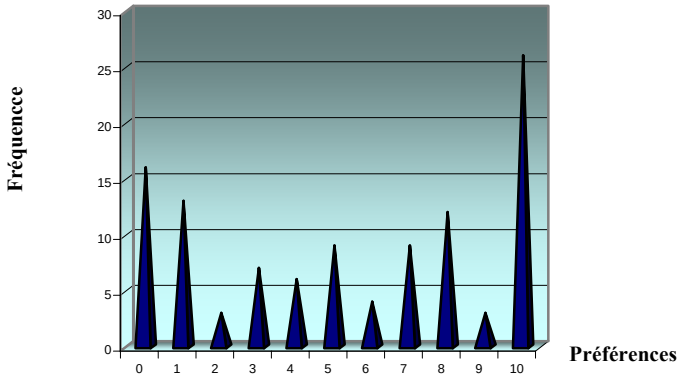


Fig.V.3.Répartition de la base de cas par rapport à l'attribut montage vidéo

Pour l'attribut jeux, elle est satisfaisante.

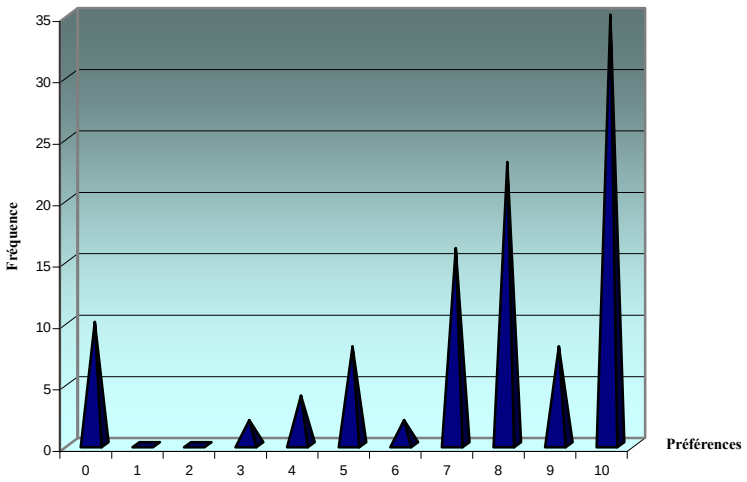


Fig.V.4.Répatrition de la base de cas par rapport de l'attribut jeux

L'attribut programmation est bien repésenté dans cette base.

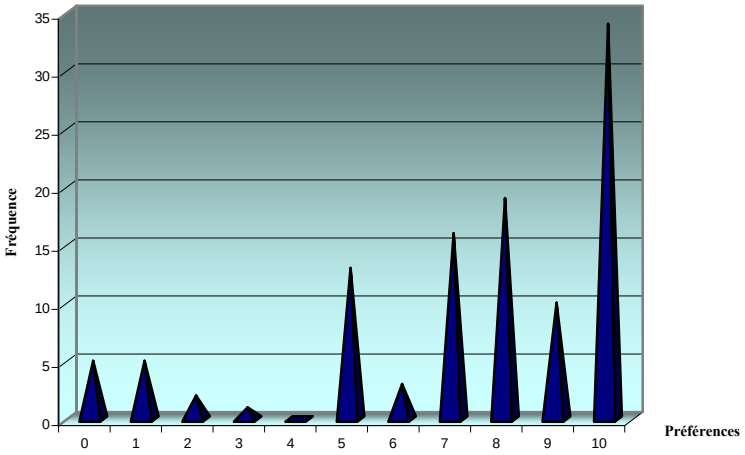


Fig.V.5.Répartition de la base de cas par rapport de l'attribut programmation

Quant à l'attribut internet, sa couverture est de très bonne qualité.

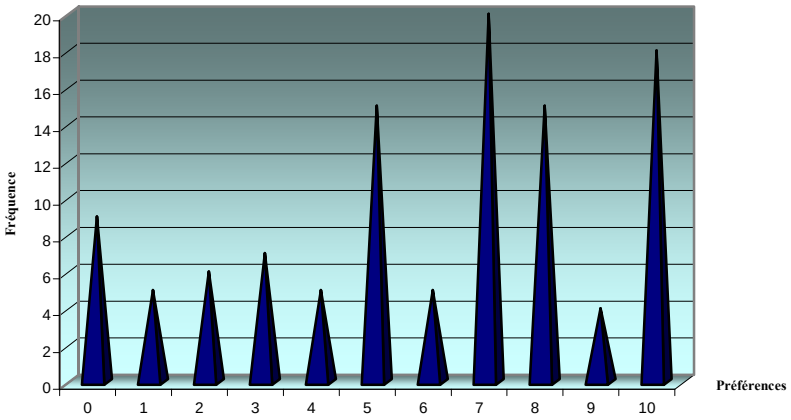


Fig.V.6.Répartition de la base de cas par rapport à l'attribut inertnet

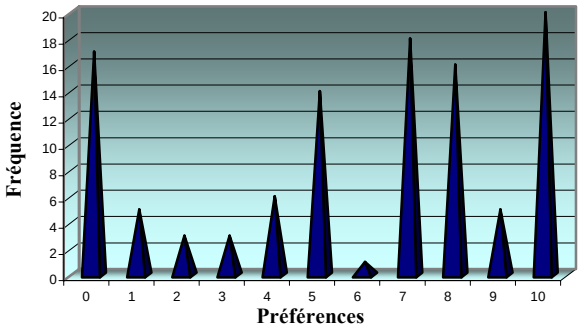


Fig.V.7.Répartition de la base de cas par rapport à l'attribut musique

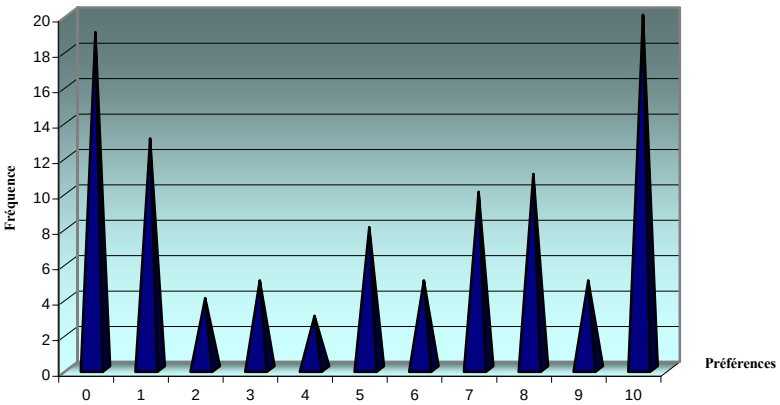


Fig.V.8.Répartition de la base de cas par rapport à l'attribut traitement d'image

De même que pour les attributs muique et traitement d'image.

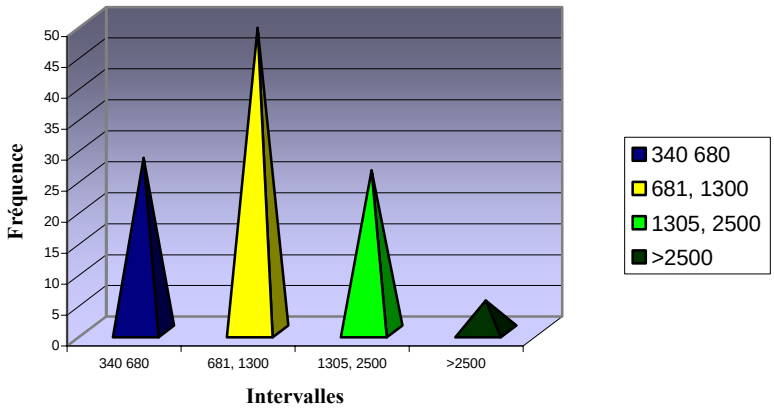


Fig.V.9.Répatirion de la base de cas par rapport au prix

Quant au prix, seule la gamme très chère a une fréquence < 5. Ce qui est logique.

Une des caractéristiques des CRN est que plus la taille de la requête est importante plus le taux d'activation du réseau augmente. Nous avons décelé la même caractéristique pour les AGRN. Le graphe de la Fig. V.10. ullistre cette constatation.

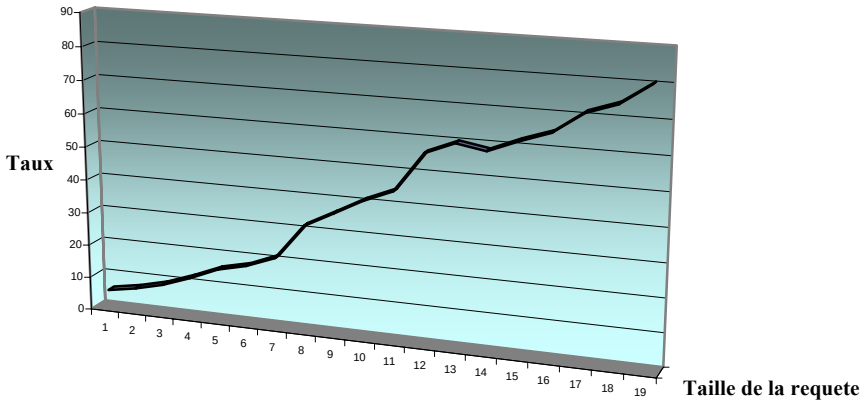


Fig.V.10.Taux d'activation du CRN/AGRN par rapport de la taille de la requete

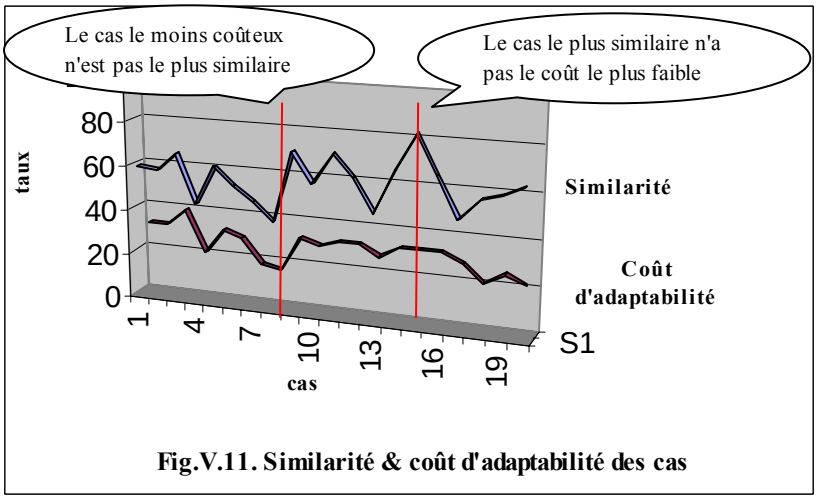


Fig.V.11. Similarité & coût d'adaptabilité des cas

Cette constatation nous permet de noter un premier trait de similarité comportementale entre le CRN et l'AGRN. Le second trait porte sur l'hypothèse émise en chapitre III concernant les performances.

En effet, la Fig.V.12. nous confirme cette hypothèse. Et c'est là le résultat le plus important de cette étude.

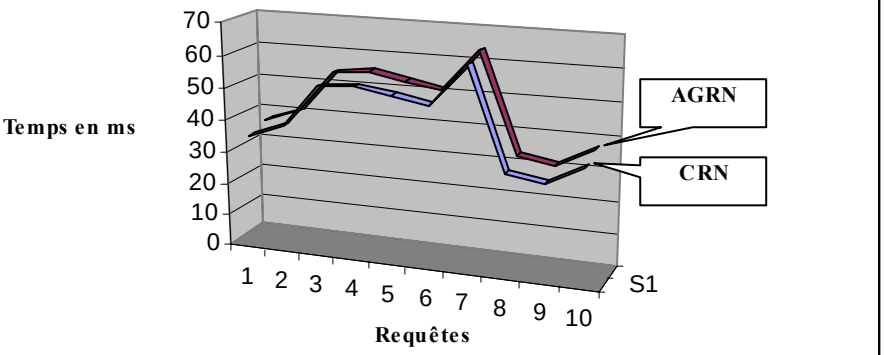


Fig.V.12. Performances similaires CRN/AGRN

Notons par ailleurs, que nous n'avons pas pris en considération certains aspects de l'application, tels que qualité de couverture de la base relativement aux descripteurs techniques jugés secondaires au départ. Il s'avère, malheureusement, que c'est une erreur de jugement comme nous l'expliquerons lors de la présentation de l'application III.

3 Application II : Diététique.

3.1 Méthode et Matière

Il s'agit du d'une application permettant de proposer un menu diététique à une personne selon son profil : age, sexe, dépense énergétique, IMC (indice de masse corporelle).

Cette recommandation est faite en adaptant des cas constitués d'anciens profils accompagnés de leurs menus.

GUIDIETE est un système qui guide l'utilisateur vers une nutrition bien équilibrée et qui lui propose des menus et des recommandations selon le cas en entrée. Il présente au patient un formulaire contenant un ensemble de descripteurs (Age, Sexe, Poids, Taille, Activité), aussi un ensemble de maladies (pour déterminer les antécédents_ médicaux de l'utilisateur) qui influencent les régimes alimentaires ou affectent le système gastrique (Voir Fig.V.13.).

Fig.V.13. Les entrées de GUIDIETE

A partir de ces informations, GUIDIETE détermine l'état nutritionnel de l'utilisateur (en calculant l'IMC et la dépense énergétique et en considérant les antécédents médicaux), pour pouvoir fournir un calendrier d'une semaine (de trois repas/jour) contenant des menus

convenables à l'état de l'utilisateur, aussi, il propose un ensemble de recommandations sur le régime, les aliments à privilégier ou à éviter, la façon de cuisiner…etc.

Description de cas :

Le cas est écrit en termes de dimensions ou descripteurs (attribut), chaque descripteur peut prendre des valeurs élémentaires, on trouve :

- *Age* : l'âge de l'individu qui sera classé ensuite dans l'un des intervalles suivants : [1,3], [4,6], [7,12], [13,20], [21,45], [46,115].

- *Sexe* : le sexe de l'individu, masculin ou féminin

- *IMC* : l'indice de masse corporelle de l'individu : [10,20.5[, [20.5, 25],] 25,28], [28,40[

- - *Dépense énergétique (DER)* : le nombre des calories dépensées par l'individu : [500,1200],] 1200,2500],] 2500, 3000],] 3000,5000].

- *Activité* : l'activité pratiquée par l'individu. Soit en repos, activités moyennes, à grand effort, ou à effort intense.

- *Antécédents médicaux* : prennent les valeurs 1 pour existants et 0 pour non existants.

- *Menu* : un calendrier d'une semaine qui comporte trois repas par jour: Petit_déjeuner, Déjeuner, Dîner.

Chacun de ces repas est constitué à partir d'un ensemble d'aliments, que l'on catégorise comme suite :

- Les féculents.
- Les produits laitiers.
- Les boissons.
- Fruits.

- Légumes.
- Matière grasse.
- Viande/Œuf
- Poissons.
- Sucreries.
- Divers : Pizza, Sandwich, …

Chaque catégorie contient un ensemble d'aliments classés selon ses constituants en nutriments et leurs valeurs énergétiques. Cette structure facilite l'adaptation par la suite, et permet la construction de nouveaux menus et repas en choisissons les aliments préférés et conseillés pour l'état présenté.

Exemple de cas :

	Cas 1	Cas 2	Cas 3
Sexe	M	F	F
Age	2	30	58
IMC	22	27	24
Dépenses énergétiques	750	2800	1200
Activité	Repos	Grand effort	Moyenne
Diabète	0	1	0
Ulcère	0	0	0
Hypercholestérolémie	1	1	0
Goutte	0	0	0
Gastrite	0	0	0
Constipation	0	0	0
Intolérance au gluten	0	0	1
Ballonnements intestinaux	0	0	0
Grossesse	0	0	0
Colite spasmodique	0	0	0
Menu	1	3	4
Recommandations	2	4	9

La base de cas initiale :

La base des cas de GUIDIETTE intègre100 cas. Elle est structurée en CRN/AGRN. Cette organisation permet de faire des accès plus rapides et d'accélérer la recherche car elle évite les indirections de l'organisation hiérarchique et de faire une recherche selon la similarité et l'adaptabilité à la fois.

Connaissances de similarité :

L'évaluation de ressemblances entre deux cas passe par l'évaluation des similarités entre les entités, de même type, qui les caractérisent. Pour GUIDIETE, on a :

- Similarité heuristique : dans le cas du sexe; dans les premiers âge (de un à 12 ans) il n'y a pas d'influence du sexe sur l'état nutritionnel de la personne, donc on peut la négliger.
- Similarité entre valeurs logiques : nulle; dans le cas des antécédents médicaux.
- Similarité entre intervalle (valeurs numériques) : qu'on applique dans le cas de l'âge, la dépense énergétique, l'IMC. On a des intervalles réels continus, et comme la distance entre deux valeurs médianes dans les deux intervalles reflète la similarité entre eux, la métrique utilisée est donc celle de Gower (voir Application I, équation .3.)

Les connaissances de relevance :

Nous avons choisi d'associer une relevance binaire égale à 1 si l'entité appartient au cas et 0 sinon.

Connaissances d'adaptation :

L'adaptation du cas en cours d'étude s'effectue sur le menu proposé et la recommandation associée à l'attribut en question, en appliquant des règles prédéfinies conçues pour fournir les équivalences des aliments en qualité ou en quantités.
Dans ce qui suit, nous donnons un exemple illustratif. Pour plus de détails voir Annexe 3.

- ✓ **Heuristique de d'adaptation pour l'attribut *Diabète*:**

Il faudrait penser à l'équilibre alimentaire qui est la base du traitement. Pour cela, il faut respecter les répartitions journalières des aliments pour maintenir une bonne glycémie, donc il faut modifier le menu de sorte qu'il préserve sa valeur énergétique.

Exemple:

Si (Maladie = "Diabète") **A**lors ***S**ubstituer* [(Prod_lait**.** Lait gélifié) par (Prod_lait**.** Lait demi écrémé)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Prod_lait**.** Concentré sucré) par (Prod_lait**.** Ecrémé)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Viande**.** Panées) par (Viande**.** Toutes les viandes)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Viande**.** Rissolettes) par (Viande**.** Toutes les viandes)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Chausson) par (Féculent**.** Pain blanc)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Pain au chocolat) par (Féculent**.** Pain seigle)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Fruits au sirop) par (Fruits**.** Fruits frais)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Fruits confits) par (Fruits**.** Fruits surgelés)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Mat_grass**.** Friture) par (Mat_grass**.** Beurre)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Mat_grass**.** Charcuterie) par (Mat_grass**.** Huile)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Boisson**.** Limonade) par (Boisson**.** Eau ordinaire)].

Si (Maladie = "Diabète") **A**lors **S**ubstituer [(Boisson**.** Boissons fruités) par (Boisson**.** Eau Minérale)].

Le menu proposé n'est plus le cas extrait, mais il a la même structure et la même quantité énergétique fournie.

✓ Heuristique de d'adaptation pour l'attribut *IMC*:

Il faut savoir qu'il n'existe aucun traitement ou régime standard, en raison de l'hétérogénéité clinique et biologique de l'affection liée à la surcharge pondérale : les attitudes thérapeutiques doivent donc être personnalisées, et un régime visant à une perte importante de poids ne doit être entrepris que sous suivi médical et psychologique.

Exemple:

Si (IMC>29) **A**lors **S**ubstituer [(Viande**.** Entrecôtes) par (Viande**.** Bœuf)].

Si (IMC>29) **A**lors **S**ubstituer [(Viande**.** côtelettes) par (Viande**.** Jambon dégraissé)].

Si (IMC>29) **A**lors **S**ubstituer [Féculent**.** Pain au chocolat) par (Féculent**.** Pain blanc)].

Si (IMC>29) **A**lors **S**ubstituer [Féculent**.** Brioche) par (Féculent**.** Pain complet)].

Les connaissances d'adaptabilité :

Le modèle de mémoire de GUIDIETE est un CRNs doté d'une nouvelle information qui est le coût d'adaptabilité.

Le coût est calculé à partir des règles d'adaptation, où chaque attribut doit avoir un coût total résultant du cumul des coûts des ses entités d'informations.

Exemple de calcul de coût :

Supposons que le patient est diabétique :

Maladie = « diabète »

Règle1 **S**i (source. Maladie = ‘‘diabète’’)
Alors solution. ***S****upprimer* (Sucreries. Chocolat)

Règle2 **S**i (source. Maladie = ‘‘diabète’’)
Alors solution. ***A****jouter* (Sucreries. Pain complet)

Règle3 **S**i (source. Maladie = ‘‘diabète’’)
Alors solution. ***S****ubstituer* [(Prod_lait**.** Lait gélifié) par (Prod_lait**.** Lait demi écrémé)].

La Règle 1 établit qu'une variation de la valeur du descripteur 'Maladie = *Diabète*' entre source et cible implique la suppression de l'aliment '*Chocolat*' de la catégorie '*Sucreries*'. Ce qui signifie en clair que le descripteur solution '*Chocolat*' dépend du descripteur problème '*Diabète*', ce qui nous donne le coût 1 pour l'attribut 'maladie'.

La Règle 2 établit qu'une variation de la valeur du descripteur 'Maladie = *Diabète*' entre source et cible implique l'ajout de l'aliment '*Pain complet*' de la catégorie '*Féculents*'. Ce qui

signifie en clair que le descripteur solution '*Pain complet*' dépend du descripteur problème '*Diabète*'. On en déduit le coût de l'attribut 'maladie', égale à 1, par rapport à la partie solution modifiée.

La Règle 3 établit qu'une variation de la valeur du descripteur 'Maladie = *Diabète*' entre source et cible implique la substitution de l'aliment '*Lait gélifié*'. Ce qui signifie en clair que le descripteur solution '*Lait gélifié*' dépend du descripteur problème '*Diabète*'. De même que la deuxième règle, on obtient un coût égal à 1pour le même attribut 'maladie'.

On obtiendra le coût total de l'attribut 'maladie', en cumulant les coûts obtenus de toutes les parties solutions qui dépendent de l'attribut 'maladie = diabète'.

Nous remarquerons que la nature de la variation (croissante ou décroissante) nous indique le type d'action à entreprendre (ajout, suppression ou substitution).

L'étude complète des coûts d'adaptabilité est donnée en Annexe 3.

3.2 Résultats et Interprétation

La qualité de couverture de la base de cas est globalement moyenne (voir de Fig.V.14. à Fig.V.18)

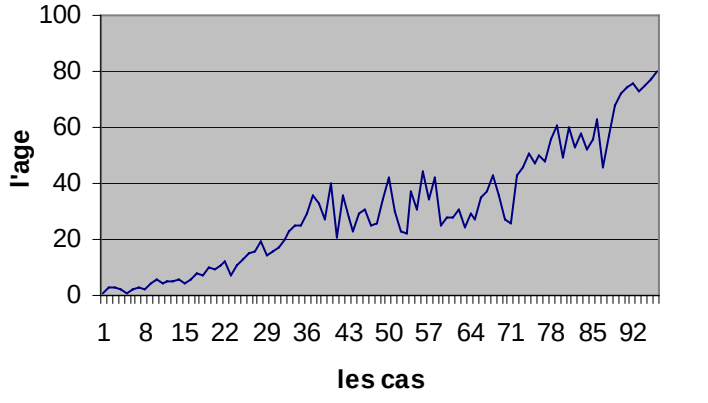


Fig.V.14. Répartition des cas selon AGE

On note cependant l'absence de personne centenaire.

La répartition selon la dépense énergitique est acceptable.

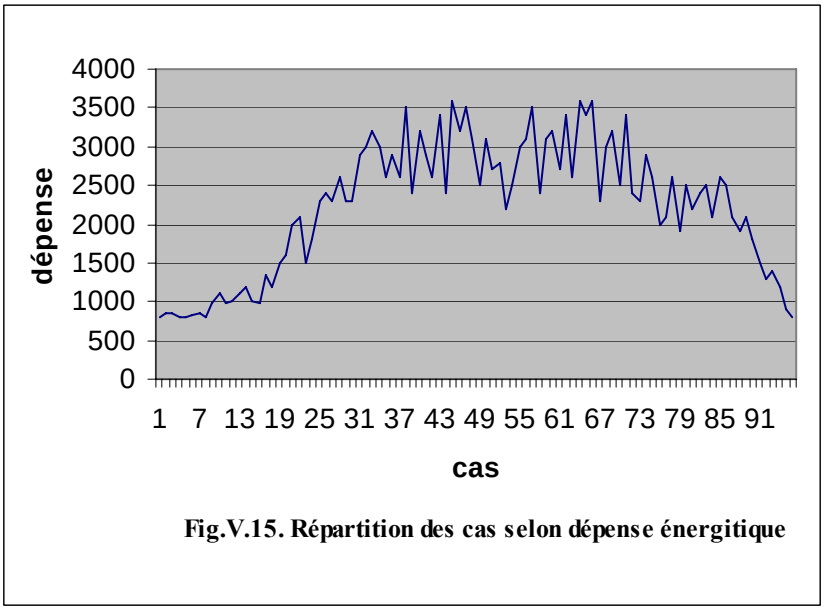


Fig.V.15. Répartition des cas selon dépense énergitique

On note la présence d'un Pic, au niveau de la répartition selon IMC. Il s'agit d'un cas présentant une obésité extrême.

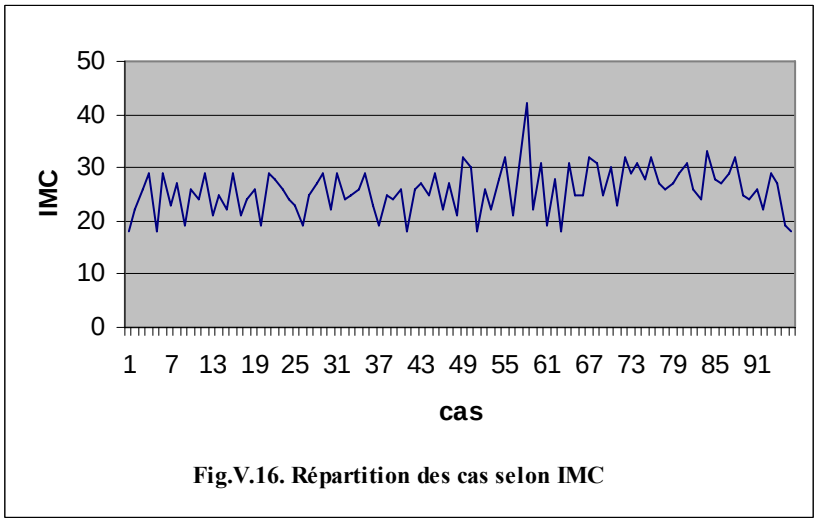


Fig.V.16. Répartition des cas selon IMC

La répartition selon le sexe est équilibrée.

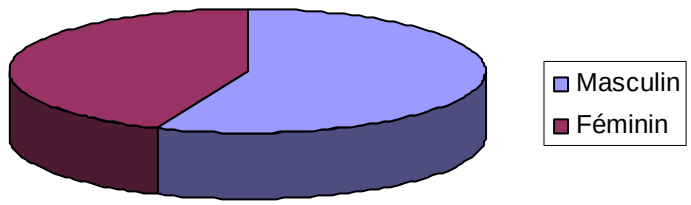


Fig.V.17. Répartition des cas selon sexe

Les antécédents Médicaux les plus fréquents dans cette base sont : le diabète et l'ulcère avec 15% de fréquence. La il y a un effort à fournir quand à l'acquisition de nouveaux cas.

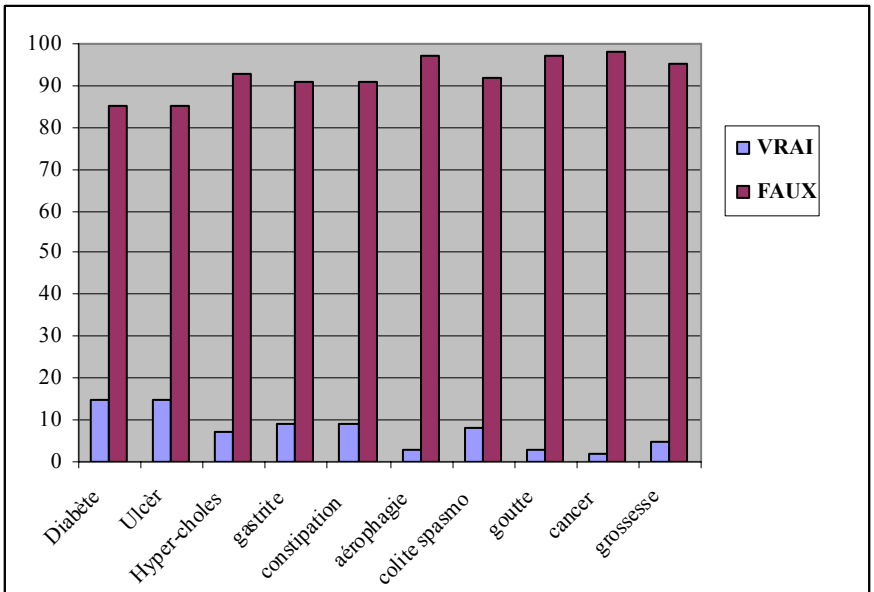


Fig.V.18. Répartition des cas selon Antécédents Médicaux

Rappelons que le but essentiel de cette application est de confirmer l'hypothèse que « étant donnée que l' AGRN est une extension du CRN pour laquelle aucune complexité supplémentaire n'a été introduite, AGRN et CRN devraient avoir des performances similaires ». Pour GUIDIETE, comme pour PC_CONFIG, cette hypothèse à été confirmée (voir FigV.19.).

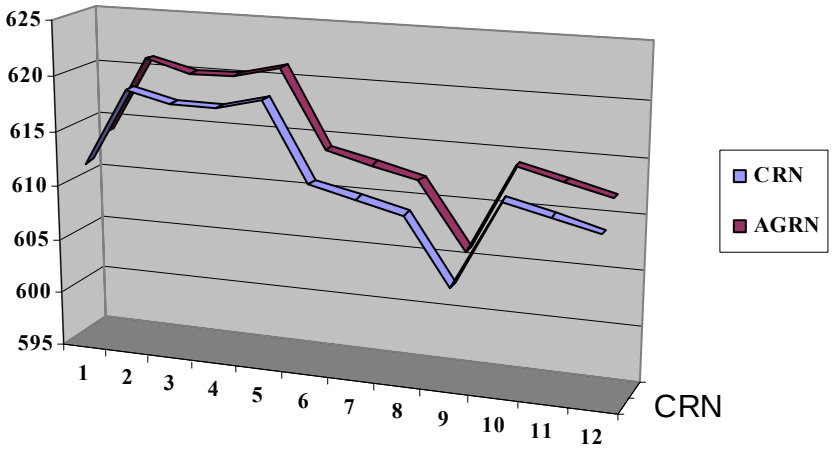


Fig.V.19. Performances similaires CRN/AGRN

4 Application III : AdaX.

4.1 Méthode et Matière

Rappelons que l'outil « AdaX » se situe à une étape de prétraitement relativement à un système CBR. Il permet de mettre en évidence l'existence ou non de relations de dépendances entre descripteur problème et descripteur solution. Et ce afin de faciliter le calcul du coût d'adaptabilité. Son utilisation intervient lors de la construction de l'AGRN.

L'outil est conçu selon une approche analyse de données selon les méthodes d'analyse bivariées telles que vues en chapitre IV.

La nature des descripteurs utilisés détermine la méthode à appliquer. En effet pour des descripteurs :

- Numérique/numérique : on utilise la corrélation. La corrélation va traiter un nuage de points représentant les valeurs des deux variables, d'une façon approximative en l'ajustant par une fonction mathématique simple connue, telle que la droite, la parabole, l'hyperbole, l'exponentielle.
- Symbolique/symbolique : on utilise , on fait la correspondance entre les deux variables en se basant sur un tableau de contingence et test de χ^2. Le test d'indépendance du chi-deux vise à déterminer si deux variables observées sur un

échantillon sont indépendantes (Hypothèse Nulle) ou non (Hypothèse Alternative). Les variables étudiées sont des variables qualitatives. Il s'effectue sur la base d'une table de contingence.

- Numérique/symbolique : Quand on a l'un des descripteur est de type quantitatif et l'autre de type qualitatif, on aura recours au **test de Fisher** si la taille de chaque échantillon est >30, sinon on pourra employer le **test de Kruskal-Wallis.**

4.2 Résultats et Interprétation

Pour procéder à une évaluation méthodique des résultats fournis par « AdaX », nous nous sommes placés dans le contexte du système « PC-Config ».

Les dépendances détectées par la méthode heuristique (voir Annexe 2) vont servir de références pour tester l'efficacité de l'outil AdaX.

Etudions la dépendance du descripteur problème'multimédia' avec le descripteur solution 'carte graphique'. Il s'agira du test du Chi-2. Le résultat sera :

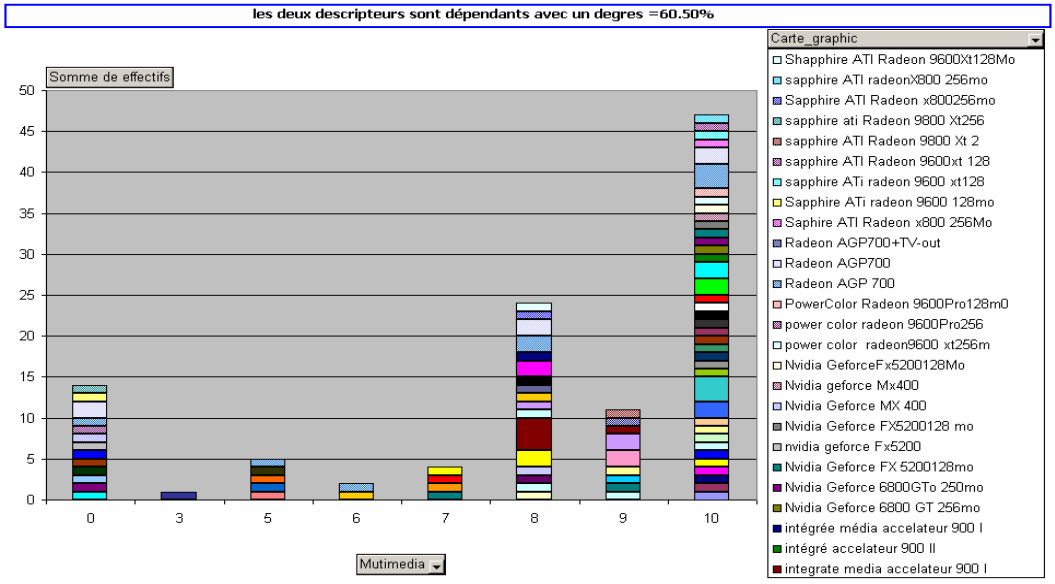


Fig.V.20. Dépendance multimédia/carte graphique

On détecte une dépendance avec un degrés de 60.50%. Ce qui confirme le résultat de l'approche heuristique. D'un point de vue graphique, cette dépendance est représentée par la faiblesse de distribution d'une couleur (modalité solution).

Etudions à présent, la dépendance du descripteur problème‘programmation’ avec le descripteur solution ’Vitesse CPU’. Il s’agira du test du Fisher. Le résultat sera :

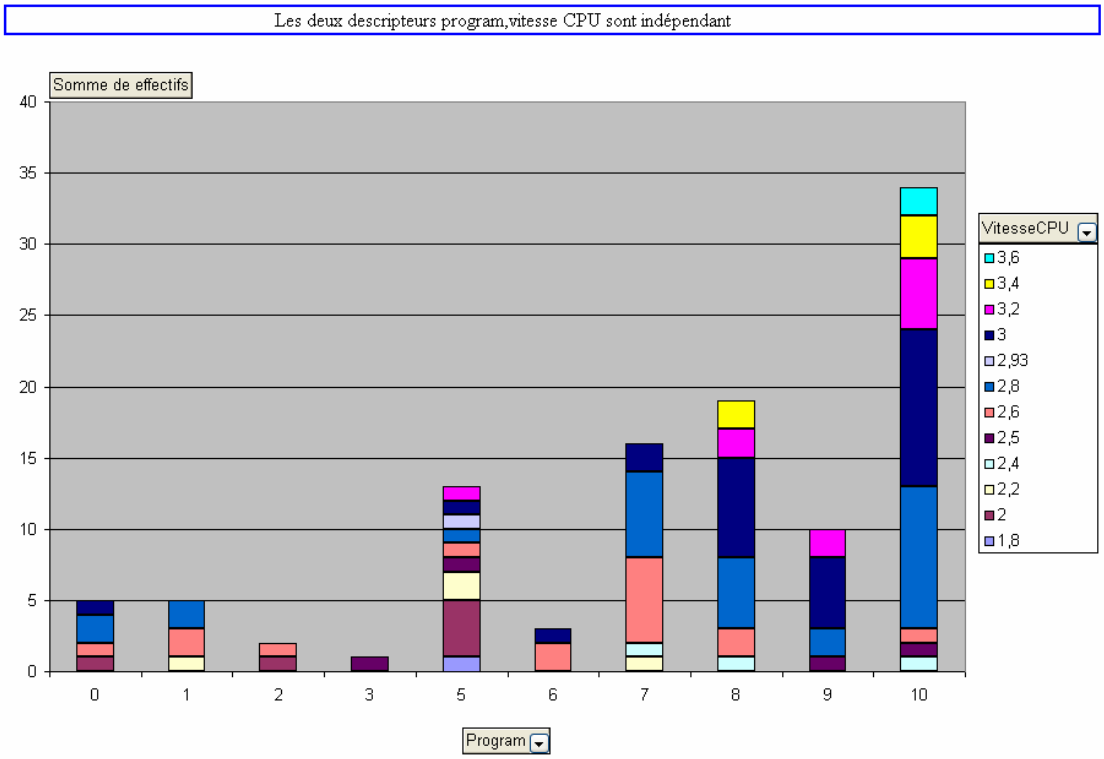


Fig.V.21. Dépendance programmation/vitesse CPU

On ne détecte pas la dépendance. Ce qui contredit le résultat de l’approche heuristique. Cependant, la dépendance est détectée graphiquement par la faiblesse de distribution de 7 couleurs parmi 11 couleurs.

Dans ce cas de figure, le test de Fisher était non significatif à cause de sa sensibilité à la normalité des données ainsi qu’à la taille de l’échantillon.

Etudions la dépendance du descripteur problème‘programmation’ avec le descripteur solution ’Marque CPU’. Il s’agira du test du Chi-2. Le résultat sera :

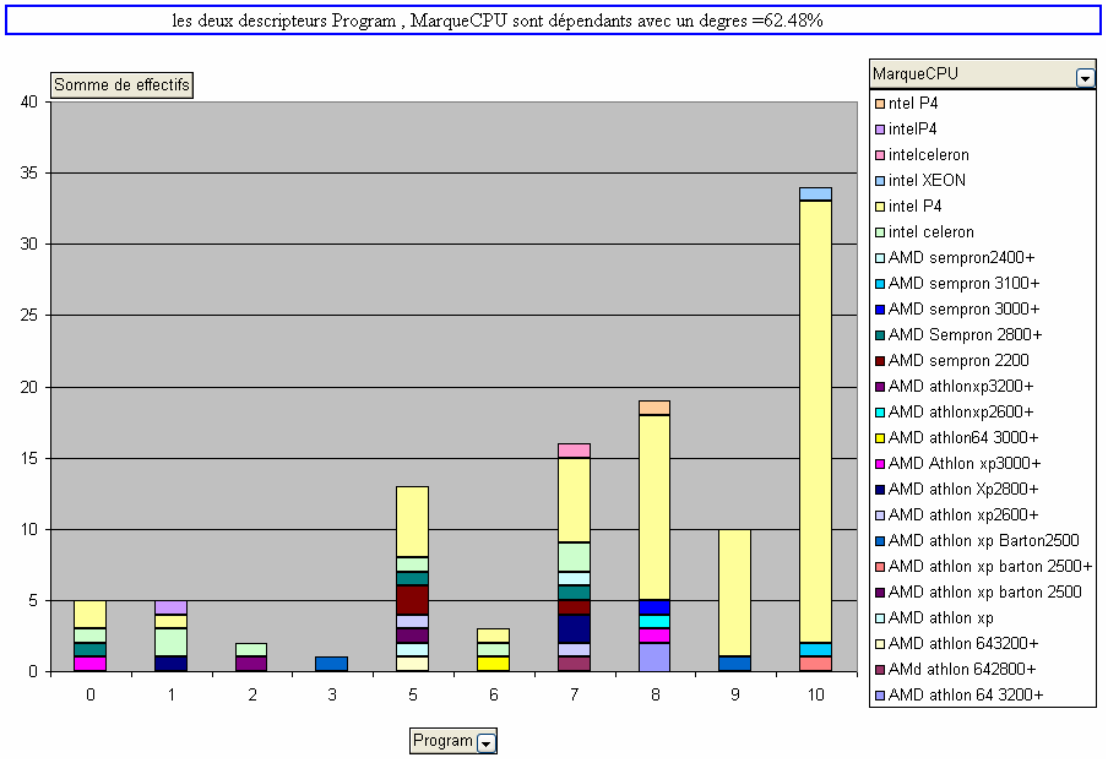


Fig.V.22. Dépendance Programmation/Marque CPU

On détecte une dépendance avec un degrés de 62.48%. Ce qui confirme le résultat de l'approche heuristique. D'un point de vue graphique, cette dépendance est représentée par la faiblesse de distribution des couleurs (modalités solutions).

Testons à présent, la dépendance entre le descripteur « Prix » considéré cette fois-ci comme descripteur problème et le descripteur « vitesse CPU». On appliquera la corrélation

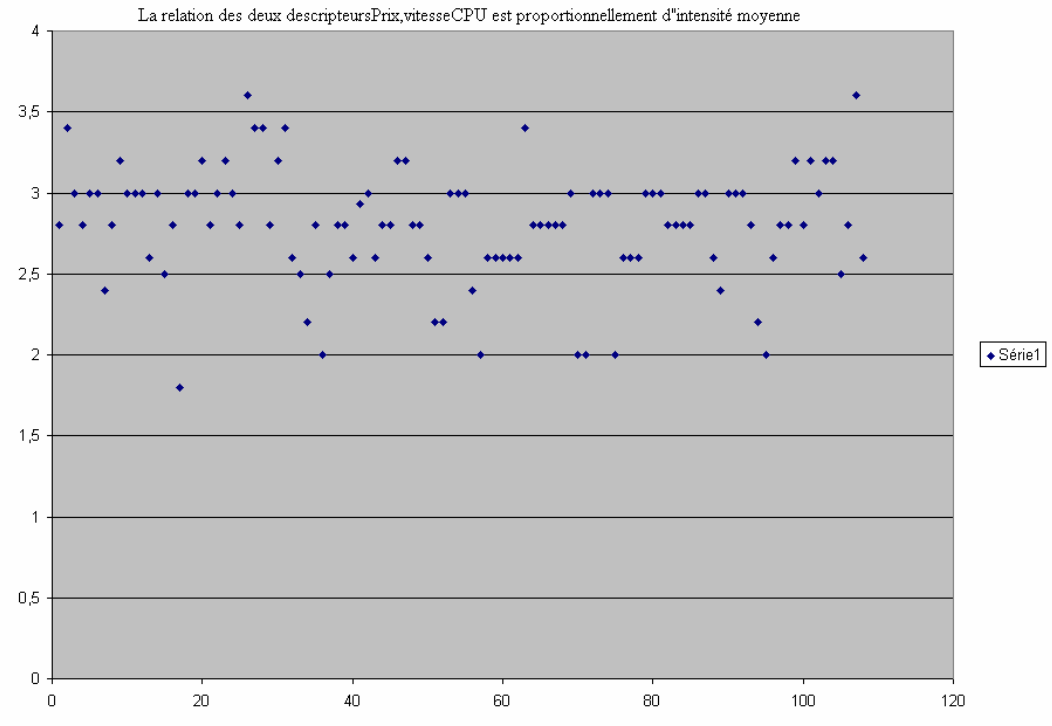


Fig.V.23. Dépendance Prix/Vitesse CPU

On détecte une dépendance avec un degré de 55%. D'un point de vue graphique, cette dépendance est représentée par nuage de points de densité moyenne.

Testons, la dépendance entre le descripteur « Prix » considéré cette fois-ci comme descripteur problème et le descripteur « Lecteur DVD». Nous utiliserons le test de Fisher.

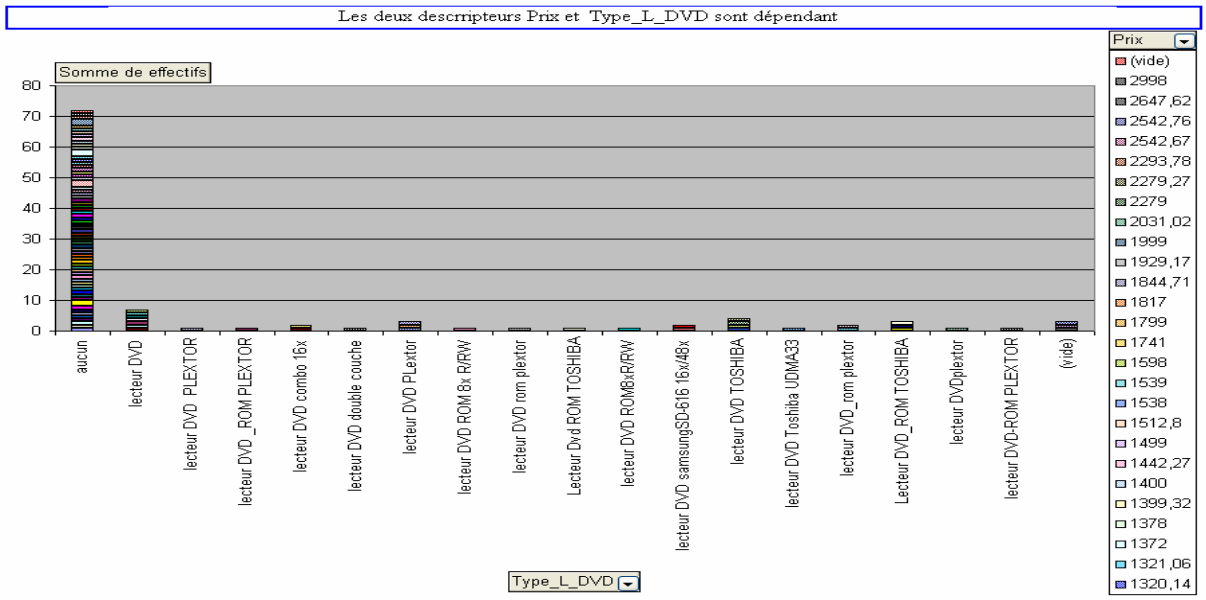


Fig.V.24. Dépendance Prix/Type L DVD

On détecte une dépendance forte. D'un point de vue graphique, cette dépendance est représentée par la faiblesse de distribution d'une couleur (modalité solution).

En résumé, récapitulons les dépendances détectées :

Pbm / Sol	Multimedia	bureautique	Jeux	Programmation	Trt Images	musique	Intenet	Montage Vidéo	Prix
Carte graphique	++		++		++			++	--
Lecteur DVD	++							++	-+
DD				+-					-+
Imprimante		++		++					-+
Graveur				++					-+
Marque CPU			++	++					--
Vitesse CPU			+-	+-					-+
Taille RAM			+-	+-					-+
Type RAM			++	++					-+
Scanner					++				-+
Modem							++		--
Nombre de pouces		+-							-+
Souris		++							-+
Carte audio						++		++	-+
Prix	+-	+-	+-	+-	++	++	++	++	

Tab.V.2. Comparaison dépendance Heuristique/AdaX

++ :détecté par approche heuristique et approche AdaX

+- :détecté par approche heuristique et non détecté par approche AdaX.

-+ :non détecté par approche heuristique et détecté approche AdaX.

Nous pouvons déduire une efficacité de 68% relativement au dépendance détectée par l'approche heuristique. Sur le nombre de dépendances non détectées 100% ont pu être expliquées.

Explication: Le système n'a pas pu détecter la dépendance dans certain cas (Où il utilisé le test de Fisher) car la taille de la plus part des modalités est inférieur à 30 et les valeurs ne suivent pas la loi normale qui est une condition nécessaire dans ce test.

En contre partie, l'approche heuristique ne peut pas prendre en charge l'étude du prix comme descripteur problème, et détecter par la même, sa dépendance avec les descripteurs solutions.

5 Conlusion

L'objectif de ce chapitre était de prouver de manière expérimentale des hypothèses émises en théorie, aussi bien concernant les performances analogues de l'AGRN et du CRN que pour l'extraction des relations de dépendances à partir de la base de cas.

La première hypothèse a été confirmée à deux reprises par PC_CONFIG et GUIDIETE. Ce qui constitue en soit un résultat très important.

Quant à l'extraction des relations de dépendances, les méthodes utilisées par AdaX ont montré leurs limites à cause de leur sensibilité à la taille des données et à leur distribution. En effet, ce que nous avions jugé satisfaisant comme couverture de la base de PC_CONFIG s'avère être insuffisant pour permettre à AdaX de fournir des résultats.

Bien que testé sur d'autres data set non mentionnés ici, AdaX n'a pu être appliqué à la base de GUIDIETE à cause de la nature particulière de la solution. Il s'agit en fait d'un menu décrit en termes de catégories et sous catégories d'aliments. AdaX ne s'applique qu'à une description linéaire.

Chapitre 6 :

Autres Travaux de Modélisation

1 Introduction

Les approches que nous allons présenter dans ce chapitre sortent du cadre de notre problématique de thèse tout en y restant liées. Néanmoins, elles n'en sont pas démunies d'intérêt dans la mesure où chacune d'elles offre une façon inédite d'envisager la modélisation en CBR.

Leur genèse n'est pas fortuite, elle découle du même processus de réflexion ayant mené au raffinement de notre problématique de base qui, rappelons le :

- a comme premier objectif la modélisation du processus CBR. Et de là est venu l'idée de proposer une modélisation basée sur un paradigme multi- agents (approche présentée en section.2.) révolutionnant ainsi la vision des choses en CBR.
- De cet objectif principal découle un second objectif qui est la modélisation de la mémoire de cas. Et de là est venu l'idée d'utiliser un algorithme génétique pour la construction du voisinage d'un problème cible puis d'accéder à ce voisinage à travers une mémoire associative (présentée en section.3.). Là aussi, cette approche se démarque de la vision classique de la recherche en CBR.

Chacune des approches a été située par rapport à la littérature. Nous ne prétendons pas avoir fait une étude exhaustive, mais avoir au moins, déblayé le terrain.

2 Agentification du CBR

2.1 Introduction

Les systèmes multi-agents (SMA) offrent une nouvelle façon de concevoir des applications IA. Leur apport se situe à différents niveaux **[Bri 2000]** : ils sont particulièrement appropriés pour représenter des problèmes qui peuvent être abordés selon de multiples méthodes de résolution, de multiples perspectives et/ou de multiples résolveurs. Ces systèmes possèdent les avantages traditionnels de la résolution distribuée et concurrente de problèmes comme la modularité, la vitesse (grâce au parallélisme), et la fiabilité (grâce à la redondance). Ils héritent aussi des bénéfices envisageables de l'IA comme le traitement symbolique (au niveau des connaissances), la facilité de maintenance, la réutilisation et la portabilité, mais

surtout ils ont l'avantage de faire intervenir des schémas d'interaction sophistiqués (coopération, coordination et négociation).

En bref ils offrent un outil puissant de modélisation de comportement de systèmes aussi bien naturels et notamment biologiques qu'artificiels.

C'est justement ce dernier aspect du paradigme qui nous intéresse.

2.2 Objectifs et Motivations

Notre objectif est de présenter une vision uniforme[2] et différente du processus de raisonnement basé cas à travers le paradigme multi–agents.

Les travaux existants sur la modélisation du raisonnement basé cas présentaient le cas comme une entité passive dans l'attente d'être trouvée pour offrir une aide à la décision ou à la résolution de problème. Nous proposons l'utilisation de cas actifs organisés en réseaux, en d'autres termes des agents dans une société qui résolvent les problèmes en s'adaptant éventuellement.

Une vision globale et homogène du processus nous garantirait ainsi une meilleure maîtrise de la modélisation.

Néanmoins, dans toute démarche de modélisation SMA certaines questions sont à satisfaire (voir **[Bri 2000]**) parmi les quelles :

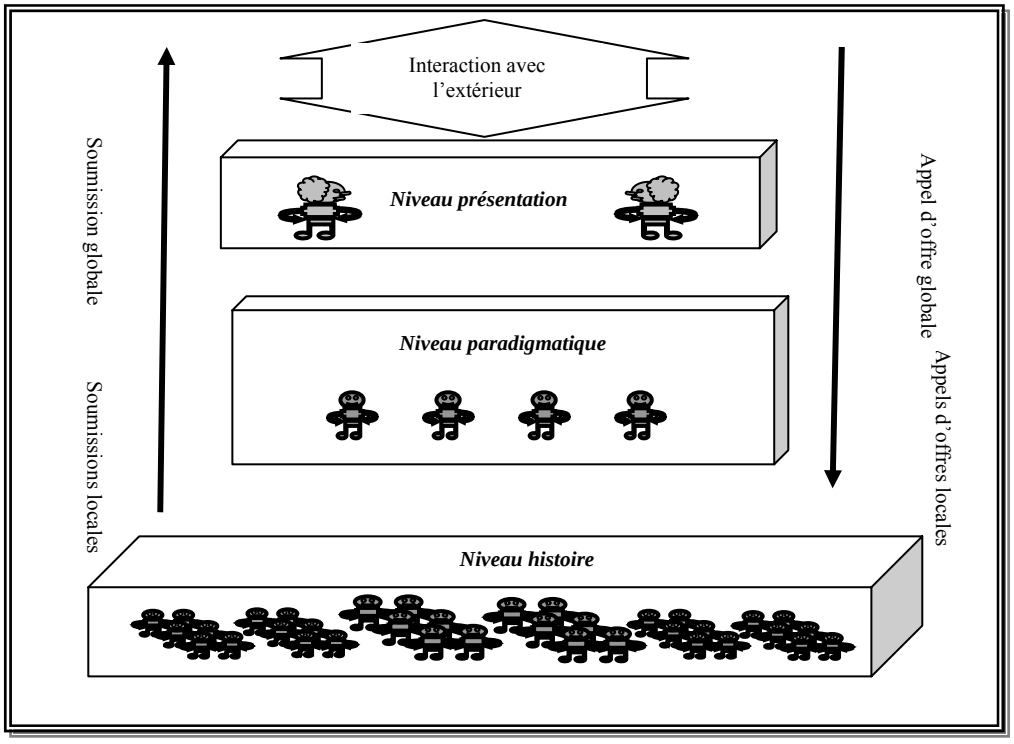
- Comment formuler, décrire, décomposer, allouer les problèmes et synthétiser les résultats ?
- Comment permettre aux agents de communiquer et d'interagir ? Quoi et quand communiquer ?
- Comment assurer que les agents agissent de manière cohérente en prenant leur décisions ou actions, en gérant les effets non locaux de leur décisions non locales et en évitant les interactions nuisibles ?
- Comment permettre aux agents individuels de représenter et raisonner sur les actions, plans et connaissances des autres agents afin de se coordonner avec eux ?

Pour répondre à ces questions, une première ébauche d'agentification du CBR a été initiée dans **[Nou 2004a]**. La présentation qui suit est une consolidation de cette ébauche avec une application dans le domaine de la réutilisation logicielle.

[2] moyennant un formalisme unique d'expression.

2.3 Organisation de la Société d'Agents

Dans une modélisation classique du CBR, l'entité de base pour l'inférence est le cas, qui est structuré en dimensions ou descripteurs relatifs soit à la partie problème soit à la partie solution. Il intègre la connaissance nécessaire à la résolution d'un problème donné d'une manière précise (il s'agit notamment de la description d'un cas histoire). Malheureusement, cette aptitude ou plus exactement cette autonomie de résolution (en termes d'agents) n'est effectivement exploitée qu'après une étape de recherche qui peut ne pas être exhaustive, notamment dans les mémoires de cas hiérarchisées (**[Len 1998]**). Il s'avère que le stockage des cas dans des bases de cas induit des problèmes de remémoration ou rappel qui peuvent constituer un handicap majeur pour une bonne résolution. D'où l'idée de contourner ce problème en dotant le cas d'une aptitude à se proposer lui même pour résoudre un problème s'il rentre dans le cadre de ses compétences (proactivité en termes d'agent).



FigVI.1. Une société d'agents

Au lieu de parler d'une mémoire de cas, on parlera alors de réseau de cas actifs répartis sur différents niveaux d'abstraction ou encore d'une société hiérarchisée d'agent (voir Fig.VI.1.). On retrouvera trois niveaux d'abstraction selon le type de tâche à effectuer.

2.3.1. Niveau présentation.

C'est un niveau permettant l'interfaçage avec l'extérieur. D'une part, il récupère la requête de l'utilisateur, l'analyse et la traduit en termes d'appel d'offre à transmettre au niveau inférieur. D'autre part il se charge de la présentation à l'utilisateur, de la solution offerte par le niveau inférieur. Pour se faire deux types d'agents sont nécessaires : l'agent INTERPRETEUR et l'agent SYNTHETISEUR.

Les deux agents travaillent en étroite collaboration pour assurer une tâche globale d'interfaçage entre l'utilisateur et le système. En gros, l'interpréteur incarne la sémantique de l'étape « présentation » du cycle CBR (voir **[Kol 1993]** et **[Wat 1995]**). Quant au synthétiseur, il assure la sémantique de rappel et réutilisation, éventuellement adaptation de cas multiple.

L'architecture interne des deux agents peut être décrite comme suit :

Pour l'agent interpréteur, un module IHM (Fig.VI.2.) assure la communication avec l'utilisateur et la récupération de la requête qui sera traitée par le module d'interprétation puis transmise via le module de communication au synthétiseur. Le module de communication interne assure aussi la récupération de la solution et sa transmission au module IHM qui la présentera à l'utilisateur.

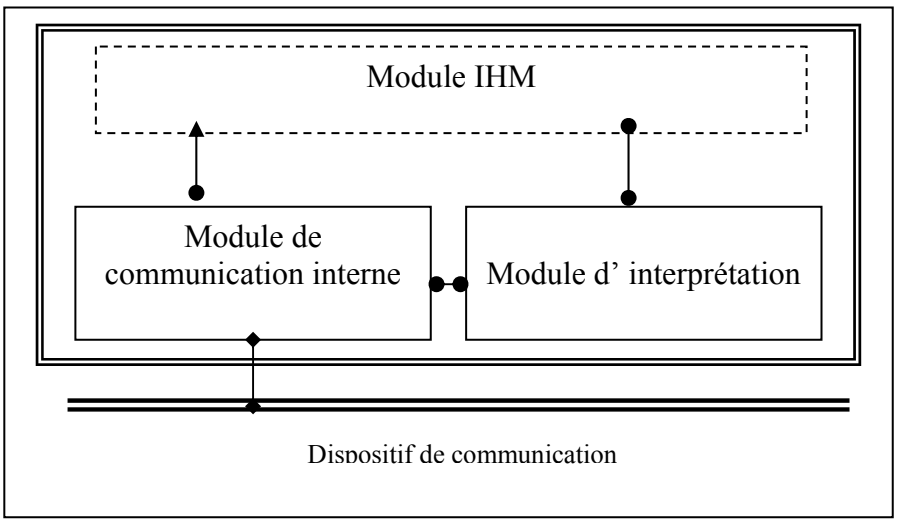


Fig.VI.2. Architecture de l'agent interpréteur

L'agent synthétiseur quant à lui, dispose d'un module de formulation et de diffusion d'appel d'offre via le module de communication aux agents paradigmatiques (Fig.VI.3.).

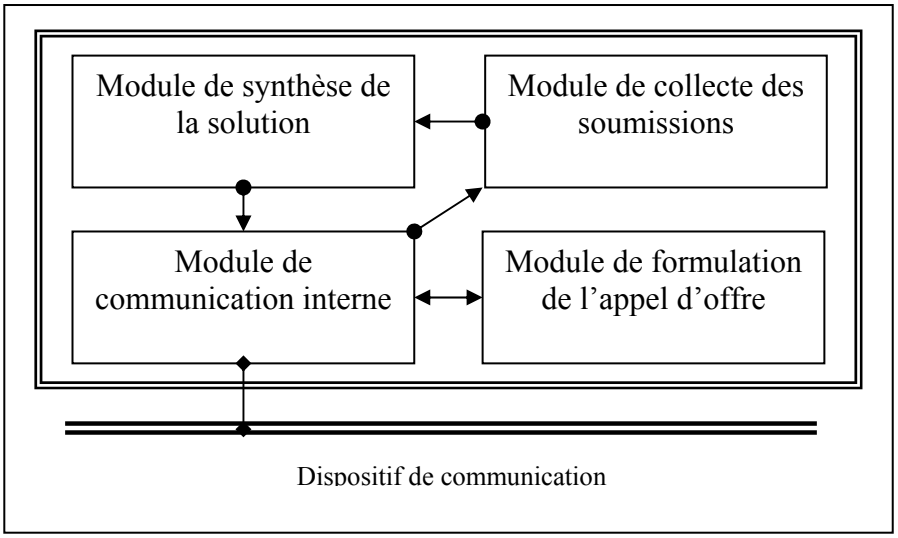


Fig.VI.3. Architecture de l'agent synthétiseur

Le module de collecte des soumissions s'assure de la conformité des soumissions vis à vis de l'offre (est-ce que tous les critères sont saturés ?).

Le module de synthèse procèdera à la construction proprement dite de la solution qui sera transmise via le module de communication à l'interpréteur.

2.3.2. Niveau Paradigmatique.

Il est constitué d'agents paradigmatiques. Au sens CBR du terme, un cas paradigmatique est un cas relatif à un ensemble de situations (tel un proverbe). Chaque agent paradigmatique est associé à un ensemble d'agents histoires répondant à une certaine description paradigmatique et présentant donc des similarités de comportement de résolution.

En sémantique CBR cet agent assure l'extraction des cas les plus proches du cas en entrée et procède éventuellement à la mise à jour de la base de cas en intégrant de nouvelles histoires.

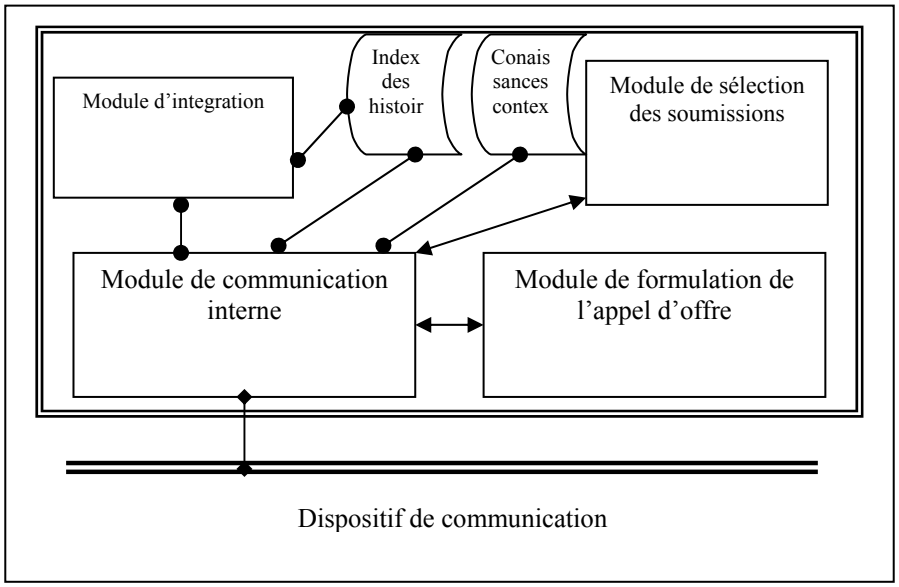


Fig.VI.4. Architecture de l'agent paradigmatique

L'agent paradigmatique dispose d'un module de formulation de l'appel d'offre locale dont le rôle est d'extraire du problème à résoudre (qui est l'appel global) la partie qui le concerne (voir Fig.VI.4.). Il diffuse ensuite cet appel selon son propre carnet d'adresse (l'index des histoires) via le module de communication interne. Le module de sélection lui permet d'évaluer puis de choisir la meilleure soumission reçue de la part des agents histoires, qu'il transmettra au niveau supérieur. Le module d'intégration quant à lui, permet l'évaluation des requêtes d'enrôlement et leur satisfaction éventuelle en mettant à jour l'index des histoires.

2.3.3. Niveau Histoire.

Dans ce niveau, on retrouve une population importante « ***d' agents histoires*** » correspondant à des cas histoires en termes CBR. Concrètement, il s'agit de la base de cas constituée d'un ensemble de cas histoires. Pour qu'il puisse soumissionner de manière efficace, l'agent histoire dispose de connaissances contextuelles se rapportant à la situation de résolution

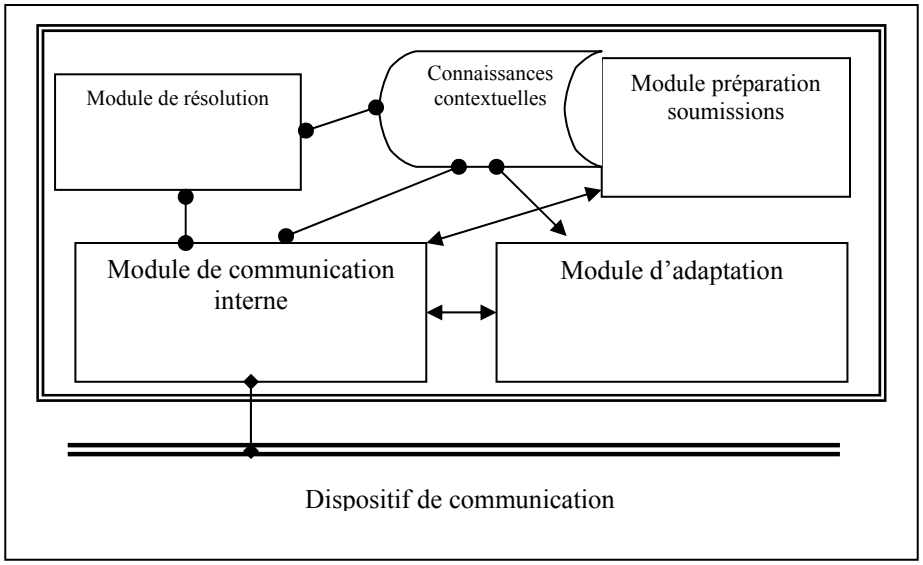


Fig.VI.5. Architecture de l'agent histoire

Un module de préparation de la soumission lui permet, à partir de l'appel local d'établir sa soumission (Fig.VI.5.). Le module résolveur, incorpore les aptitudes de résolution de problème en tenant compte du contexte. Il arrive que la résolution ne soit pas directe mais que l'agent, après analyse de l'appel et du contexte, éprouve le besoin d'une adaptation de ses aptitudes. On parlera alors de ***Mutation***. Ce qui sera pris en charge par le module d'adaptation. La ***Mutation*** consiste en fait en une duplication de l'agent histoire avec ajustement de ses aptitudes au contexte actuel.

Aussi bien le cas mutant que le cas obtenu par croisement (lors de la synthèse de la solution), tous les deux sont de nouveaux cas histoires. Ils formuleront leur requête d'enrôlement et les soumettront respectivement au cas paradigmatique associé à leur origine ou aux différents cas paradigmatiques à partir desquels ils ont été synthétisés. Le ou les cas paradigmatiques concernés analyseront la requête et conformément ou non à un critère d'utilité future accepte ou refuse l'enrôlement. Le cas échéant, l'acceptation consiste en l'ajout du cas histoire candidat à l'index des histoires du cas paradigmatique concerné.

On peut ainsi trouver un même cas histoire obtenu par synthèse, indexé dans différents cas paradigmatiques selon différentes facettes de son rôle complexe.

Il est important de noter que le nouveau cas histoire, dès sa naissance, prend l'initiative de demander l'enrôlement. Et que si sa requête est rejetée, il se suicide.

2.3.4. Schéma d'interaction.

Pour un agent, interagir avec un autre constitue à la fois la source de sa puissance et l'origine de ses problèmes (**[Fer 1997]**). C'est en effet parce qu'ils coopèrent que des agents peuvent espérer que leur satisfaction sera meilleure que s'ils agissaient de manière isolée.

Les agents doivent être capables, par le biais de la communication, de transmettre des informations, mais surtout d'induire chez l'autre un comportement spécifique. Communiquer est donc une forme d'action particulière qui, au lieu de s'appliquer à la transformation de l'environnement, tend à une modification de l'état mental du destinataire. La forme de communication utilisée dans notre architecture est de type intentionnelle puisqu'elle permet à des agents intelligents organisés en réseau d'interagir au mieux, contrairement à la communication réactive qui n'est applicable que dans des contextes robotiques ou biologiques.

Par ailleurs, mis à part le niveau interface, on ne retrouve aucune communication intra niveau. Le schéma le plus riche est donc vertical, inter- niveau. En numérotant les niveaux de haut en bas (dans la Fig.VI.1.), on retrouve :

Niveau1-niveau2 : il y a transmission de l'appel d'offre globale ainsi que les connaissances contextuelles dans un sens. Et récupération des soumissions dans l'autre.

Niveau2-niveau3 : il y a transmission de l'appel d'offre locale ainsi que les connaissances contextuelles dans un sens. Et récupération des soumissions et requête d'enrôlement dans l'autre.

Notons qu'il n'y a aucune interaction directe entre les niveaux 1 et 3, tout passe par le niveau intermédiaire.

Seul le niveau 1 est doté d'une communication interne entre agent interpréteur et agent synthétiseur. Puisque les deux coopèrent pour la réalisation de la même tâche d'interfaçage.

Par ailleurs, seuls les agents histoires associés à un même agent paradigmatique peuvent être en concurrence puisqu'il soumissionnent pour le même appel d'offre local. La gestion de conflit se fera au niveau de l'agent paradigmatique qui décidera de la meilleure soumission relativement à l'appel global.

2.4 Application à la réutilisation de composant logiciels

Dans la communauté « Génie Logiciel », la réutilisation est une pratique systématique. Elle consiste à développer du logiciel en partant systématiquement d'un stock de briques de construction. Elle a pour objectif l'amélioration de la qualité et de la productivité, soit l'industrialisation de la production logicielle. Dans **[Erz 1999]**, l'auteur présente des exemples concrets de réutilisation. Il s'agit des chiffres correspondant aux bénéfices obtenus par de grandes entreprises qui ont adopté une stratégie de réutilisation. Nous en reprenons quelques uns dans l'encadré suivant :

Fujitsu :
Proportion de projet respectant les délais : augmentation de 20% à 70%
Effort d'adaptation aux besoins spécifiques du client : réduction de 30 mois-homme à 4 jour-homme.

Hewlett-Packard : sur les deux projets pilotes considérés
Erreurs : diminutions de 24% et 76%.
Productivité : augmentation de 40% et 57%
Délais de livraison : réduction de 42% sur l'un des deux projet.

NEC (Nippon Electric Company) :
Productivité : 6,7 fois plus élevée.
Qualité : 2,8 fois meilleure.

Trêve d'arguments, l'intérêt de la réutilisation en génie logiciel n'est donc plus à démontrer. Ce qui a motivé notre premier travail dans ce domaine **[Nou 1999]**. Il s'agit du système CEPROC (pour Case basEd PROgram Constructor). CEPROC est un outil intégré dans un environnement de développement de programmes écrits en Pascal, orienté vers le calcul scientifique. Lors de l'édition de son code, un utilisateur peut émettre un besoin en calcul style produit de deux matrices ou tri d'un tableau. Sa requête formulée dans un langage semi-contraint, est analysée puis reformulée sous forme d'arbre abstrait. Une recherche sera ensuite lancée dans la base de cas pour trouver le ou les cas de la base dont l'arbre abstrait est soit identique soit une sous structure de l'arbre de la requête. Le ou les cas trouvés passeront ensuite par une étape d'adaptation qui consiste à ajuster le code du cas trouvé relativement à la requête ou à combiner puis réajuster les codes des cas trouvés. Le code ainsi généré est ensuite intégré dans l'éditeur de programme (aussi bien partie déclarations que partie instructions).

Le système CEPROC avait donc été réfléchi et conçu dans une vision classique du CBR.

Si à l'époque nous disposions du modèle agent du CBR, à quoi ressemblerai CEPROC ?

Nous avons donc envisagé un scénario.

La distribution des rôles sera comme suit :

Rôle de l'INTERPRETEUR:

* prend en charge l'analyse du problème en entrée : la requête formulée dans un langage semi contraint sera traduite en arbre abstrait constitué d'opérateurs et d'opérandes.

* fournit au synthétiseur une représentation de ce problème

* Procède à l'intégration du code généré par le SYNTHETISEUR dans sa partie déclarative ainsi que dans sa partie instruction.* *Tâche de fond : scrute les entrées.*

Rôle du SYNTHETISEUR:

* reformule la représentation de l'INTERPRETEUR en termes d'APPEL D'OFFRE GLOBAL *:*

{opérateurs impliqués dans l'arbre abstrait de la requête+ contexte}

* communique l'appel au niveau inférieur.

* collecte les soumissions.

* synthétise la solution à partir des soumissions en tenant compte de l'ordre des opérateurs.**Tâche de fond : scrute les entrées pour récupérer le contexte de la requête.*

Rôle de l'agent paradigmatique:

* étudie l'appel pour déceler les opérateurs qui le concernent (opérateurs scalaires, matriciels,..).

* formule l'appel d'offre local en récupérant le sous arbre abstrait le concernant.

* collecte les soumissions locales et sélectionne la plus optimale en fonction du contexte (par exemple : le meilleur algorithme de tri en fonction des besoins de l'utilisateur).

* établit une soumission.

* intègre de nouvelles histoires

**Tâche de fond : scrute les appels.*

Rôle de l'agent histoire:

* analyse l'appel local et formule sa soumission sous forme de compétences.

* est apte à résoudre une situation particulière. Exemple : tri à bulle d'un tableau, somme de deux matrices, …

* peut s'adapter soit par mutation localement soit par croisement au niveau de l'agent synthétiseur.

* peut formuler des requêtes d'enrôlement.

**Tâche de fond : scrute l'appel d'offre locale.*

Il est à noter que le contexte de la requête est constitué de toute information susceptible de donner une indication sur les contraintes d'utilisation du composant logiciel requis (autrement dit compétences requises de l'agent histoire). Il peut s'agir par exemple de type ou taille des données, performances de temps ou taille mémoire,…

Par ailleurs, l'adaptation des cas histoires peut se faire par :

- Mutation : l'agent histoire peut subir dans sa partie code (module de résolution) un renommage de variables ou un ajustement de la taille des structures de données par exemple.
- Croisement : deux agents histoires peuvent combiner leurs compétences pour répondre à une requête. Exemple : si la requête formulée porte sur le calcul de l'inverse d'une matrice. Il y aura donc calcul du déterminant et calcul de la transposée ainsi que produit scalaire par matrice. Ces trois agent subiront un croisement (deux à deux en cascade) qui donnera lieu un nouvel agent histoire : calcul de la matrice inverse.

2.5 Travaux Similaires et Discussion

La majeure partie des travaux sur l'agentification du CBR s'est focalisée sur les aspects communication et coopération entre agent dans le contexte des bases de cas distribuées (on retrouve notamment **[Pla 1997]**).

Néanmoins, il existe des approches abordant la question différemment :

Huang (dans **[Hua 1996]**) présente une mémoire basée agent pour effectuer une tâche de classification. Il cherche à classer de nouveau cas à partir d'un ensemble de cas préclassifiés. Les objectifs du modèle sont : dans un premier temps, le maintien d'une petite base représentative de cas. Secondairement, le modèle doit pouvoir s'ajuster à la lumière de nouveau résultats de classification. Ces objectifs sont atteints par optimisation des seuils de distances entre agent, de leurs poids ainsi qu'en permettant aux agents de changer de localisation dans l'espace des cas. Le concept conventionnel de cas est étendu au concept d'agent mémoire capable d'effectuer une liste de tâches telles que : se proposer pour la résolution d'un problème, s'ajuster à la lumière des résultats de l'offre et changer de

localisation dans l'espace des cas en fonction de l'offre. La mémoire de cas conventionnelle devient un ensemble d'agents coopératifs.

Morisbak et Tessem dans **[Mor 2001]** décrivent un outil de recherche basé agent dans le domaine du génie logiciel. La technologie agent est utilisée durant le processus de recherche de manière hiérarchique. L'agent Gestionnaire prend en charge les interactions avec l'utilisateur et contrôle le processus de recherche. Les agents coordinateurs sont des packages de cas, un regroupement logique de cas. Un agent ouvrier est un cas individuel avec sa propre description. Les agents ouvriers ont la capacité de se comparer à une description de cas reçue à partir de l'agent gestionnaire. Selon le résultat de la comparaison, ils peuvent décider s'ils se proposent comme cas potentiels à réutiliser ou attendre que la description du cas en cours de résolution soit plus riche, ou encore décider de mourir. L 'agent gestionnaire contrôle la compétition entre agent ouvriers et décide lequel sera offert à l'utilisateur, lequel attendra une seconde chance et lequel mourra.

Murdock et Goel dans **[Mur 2001]** présentent un mécanisme dans lequel des modèles qualitatifs sont utilisés pour proposer efficacement un ensemble de conceptions alternatives pour des éléments spécifiques dans un méta-cas. Ils ont construit une plate-forme de raisonnement ayant la capacité d'exécuter et d'adapter des agents représentés dans un modèle TMK (Task Method Knowledge).

Si l'on tente de situer notre approche relativement à ces travaux, on notera que : dans l'approche de Huang comme dans la notre, le cas est une entité active. Dans l'architecture de Morisbak, la hiérarchisation permet de mieux gérer la concurrence. On retrouve la même philosophie dans notre travail.

Contrairement à ces deux travaux et de même que dans **[Mur 2001]**, notre architecture à été projetée générique sans contrainte du domaine.

Par ailleurs, d'un point de vue distribution de la base de cas, nous pouvons constater qu'elle est maximale dans notre travail, puisque chaque cas est un agent.

En dernier lieu, nous envisageons une extension en termes de multi-expertise. En effet, en dupliquant le niveau interface et en dotant chaque interface d'une ontologie propre à un domaine particulier, nous aurons un résolveur générique pouvant répondre à différentes requêtes dans divers domaines

3 Modélisation Génétique de la mémoire de cas

Le CBR est un paradigme IA qui peut être combiné synergiquement avec d'autres approches pour palier à de multiples problèmes **[Mar 2002]**, **[Pal 2004]**.

Parmi ces possibilités, nous allons présenter une approche permettant d'améliorer les performances, du processus de rappel dans une mémoire associative en utilisant un algorithme génétique.

L'idée principale est le calcul du voisinage d'un nouveau problème à résoudre, par un algorithme génétique. Ce voisinage va délimiter notre espace de recherche dans la base de cas. Une structure associative reliée à la base nous permettra d'accéder directement aux cas similaires dont la description problème a été générée dans le voisinage.

3.1 La Mémoire proposée

Pour fonctionner correctement, le CBR utilise des cas stokes dans une mémoire de cas. Celle-ci est supposée être représentative de l'ensemble des problèmes rencontrés dans le domaine. Plus elle est riche en cas plus l'inférence est puissante. Or plus sa taille croît plus la recherche devient coûteuse. C'est pour cela que le choix des algorithmes de recherche est une tâche importante dans ce mode de raisonnement.

Il existe différentes organisations de la mémoire pour lesquelles des algorithmes de recherche ont été mis au point **[Kol 1993]**, **[Aam 1994]** et **[Wat 1997]** Les modèles de mémoire les plus couramment utilisés sont basés sur une stratégie descendante. Ils présentent les caractéristiques communes suivantes **[Len 1998]** :

- Ils supportent une structuration des données par regroupement des objets liés.
- La recherche est efficace moyennant les algorithmes traditionnels de parcours d'arbre.
- Le parcours descendant de l'arborescence est assuré par la réponse aux questions des noeuds internes pour déterminer le chemin à empreinter. Ce qui requiert un certain ordre dans les questions. Dans le cas d'information manquante, ceci peut induire le choix d'un chemin erroné.
- Une fois un certain cluster atteint au niveau des feuilles de l'arbre, il sera difficile d'accéder aux clusters voisins contenant des cas similaires.

Pour toutes ses raisons, nous exposons dans ce qui suit, une vision de la recherche basée sur la construction du voisinage d'un problème par algorithme génétique.

La mémoire de cas est en fait, une structure plate chapotée par une structure de réseau. Les nœuds sont de deux types : les nœuds « valeur » et les noeuds « cas ». Un nœud valeur représente une valeur particulière de descripteur problème disponible dans le domaine (voir Fig.VI.6.) ce nœud est lié à tous les nœuds cas où il apparaît.

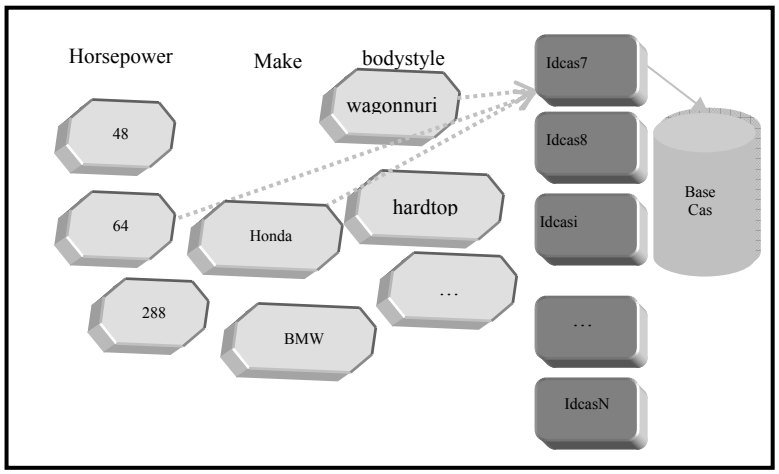


Fig.VI.6. La Mémoire Associative

Le noeud cas pointe vers la description complète du cas stocké dans la base.

Les particularités de cette structure sont :

- Le cas est accessible par son contenu (principe des mémoires associatives)
- Toute la structure peut être construite automatiquement en compilant les données de la base de cas.

3.2 L'Architecture Générale

La recherche d'un cas utilisable peut être formulée en : Comment extraire à partir de l'espace de recherche, le sous espace de cas similaires au cas cible. Ce sous-espace de candidat est ce que nous convenons d'appeler voisinage du problème cible. Il est traditionnellement obtenu par une stratégie de parcours de la structure de mémoire.Notre idée est de calculer ou construire ce voisinage par un algorithme évolutionniste (Fig.VI.7.).

Par la suite, l'accès au voisinage se fera à travers le réseau de manière associative.

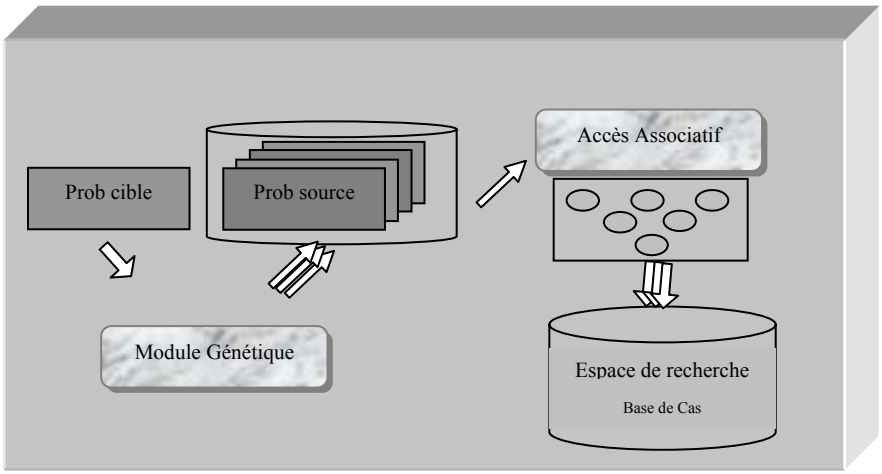


Fig.VI.7. Architecture globale

Chaque problème source calculé par l'algorithme évolutionniste, sera directement pointé dans l'espace de recherche via le réseau.

Nous allons nous intéresser à présent au module évolutionniste.

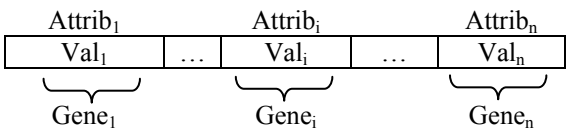
Rappelons qu'un cas est une entité au sein de la quelle sont regroupées des informations sur une situation passée, le terme « situation » étant très général.

Un cas est constitué d'un ensemble de descripteurs ou dimensions répartis sur deux espaces : l'espace problème et l'espace solution.

La recherche du cas source similaire au cas cible se fait à travers les descripteurs problème. Puisqu'en effet, le cas cible ne dispose que de sa partie problème. Aussi, c'est cette partie qui nous intéresse lors du chois du codage.

Le Codage

Lorsqu'un problème est posé, la requête pour trouver le cas le plus similaire est formulée aux termes des descripteurs problème (Fig.VI.8.).



FigVI.8. Description du problème comme un chromosome

Le codage du problème sera donc:

Description Problem: pbm	Chromosome
Descripteurs : d_i	Genes
Valeur Descripteur val_j	Alleles

Tab.VI.1.Mise en correspondence entités CBR et entités EC

Pbm = { d_i } : un vecteur de descripteurs.
d_i = (Attribi,valij) : un couple d'attribut/valeur
$val_{ij} \in Dom_j$: chaque valeur appartenant a un domaine spécifique pouvant être symbolique ou numérique.

Pour notre expérimentation, nous avons utilisé un sous ensemble de « Auto Import Database » un data set de UCML repository.
Nous nous sommes intéréssés à la description suivante :

description problème =<make, bodystyle, horsepower>

$Domaine_{make}$ = {alfa-romero, audi, bmw, chevrolet, dodge, honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen, Volvo} discret

Dans le cas d'un codage Décimal, nous avons:
Make $\in$ [1,22] → 2 digits

Pour un codage binaire:
Make $\in$ [00000,10110] → 5 bits

$Domaine_{bodystyle}$ = { hardtop, wagon, sedan, hatchback, convertible } discret

Dans le cas d'un codage Décimal, nous avons:
Bodystyle $\in$ [1,5] → 1 digit

Pour un codage binaire:

bodystyle ∈ [000,101] → 3 bits

$Domaine_{horsepower}$ = [48 , 288] continu

Dans le cas d'un codage Décimal, nous avons:

Horsepower ∈ [48,288] → 3 digits

Pour un codage binaire:

horsepower∈ [000110000,100100000] → 9 bits

Un exemple de chromosome sera:

Pour un codage decimal

1	3	2	0	6	8

Le codage binaire étant plus long:

01101 010 001001000

Ce qui correspond à la description du problème suivant:

Prob=<nissan, wagon, 68>

La population initiale est générée aléatoirement.

L'étape de selection est basée sur une stratégie de similarité à la requête (nouveau problème). La forme générale en est :

$D(G,Gk) = \sum wi\ di(G,Gk)$ pou $i \in [1,3]$

Où: wi est le poids du descripteur i (gene i) et di est la distance partielle:

$di = 1-(|X^i-X^i_k|/$ écart maximal)

La reproduction est essentiellement effectuée par:

- Cross-over: Aussi bien pour le codage binaire que le codage décimal nous avons deux points de rupture. Ce sont les séparations entre gênes.
- La mutation des gênes: La mutation du chromosome correspond à la perturbation de la description du problème en entrée afin d'obtenir des problèmes voisins.

La fonction fitness est basée sur l'estimation de la similarité entre le problème en entrée et le chromosome actuel. Elle est de la forme :

Maxmiser $\sum D(G,Gk)$ pour k=1 à cardinalité de la population.

L'algorithme globale étant:

1. Initialiser une population de chromosomes.
2. Evaluer chaque chromosome de la population.
3. Créer nouvelle génération de problèmes par cross over puis par mutation de la génération courante.
4. Evaluer nouvelle population.
5. Si <critère d'arrêt > satisfait
   Alors stop
   Sinon aller à 3.

critère d'arrêt = population stable or max Temps

Pour le problème en entrée:

1	3	2	0	6	8

Un exemple de population (avec card =5):

Pour le codage décimal:

0	9	4	1	1	6	0.76
1	1	2	0	7	0	0.96
0	4	3	1	6	0	0.76
2	2	1	2	0	7	0.58
1	4	3	1	1	0	0.84

La dernière colonne représente la fonction de selection.

Une première simulation pour un codage décimal mène aux résultats présentés en Fig.VI.9.

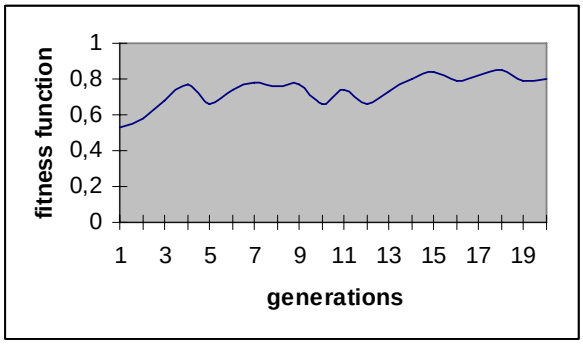


Fig.VI.9. Résultats de simulation

Il est à noter que dans notre étude, aussi bien la fonction de sélection que la fonction fitness exprime la même sémantique puisque l'objectif est de trouver les problèmes les plus similaires au problème en entrée.

2.4 Conclusion

Différents modèles de mémoire ont été proposés dans la littérature (voir **[Kol 1993]**). Cependant, l'approche évolutionniste semble intéressante à plus d'un titre **[Gol 1994]** :

- Représentation des connaissances flexible.
- Bonne performance de calcul.
- Large éventail d'applicabilité.

Jusqu'à présent leur application dans les CBR s'est limité à la tâche d'adaptation. Une approche évolutionniste d'adaptation est présentée en **[Gom 1999]**. En **[Pur 1997]**, l'adaptabilité des cas est augmentée par algorithme génétique.

L'approche que nous proposons met en œuvre une mémoire accessible par le contenu. Flexible, de construction facile et utilisant une représentation des connaissances uniforme relativement au module évolutionniste.

Il est important de noter que l'approche proposée constitue un modèle général. Et que pour une application spécifique il faudra procéder au paramétrage du système pour assurer la convergence.

Puisque la base de notre travail reste la recherche guidée similarité et adaptabilité à la fois, il serait intéressant d'envisager une extension intégrant le critère d'adaptabilité dans la fonction objective.

4 Conclusion

Une voie de recherche naît parfois d'une simple idée pourvu que l'on se donne le temps de l'explorer méthodiquement pour en faire une problématique potentielle. C'est cela même l'objectif de ce chapitre : ouvrir de nouvelles voies.

Conclusion et Perspectives

Le plus difficile dans une conclusion est de reprendre, sans se répéter, en quelques paragraphes, ce sur quoi nous nous sommes étalés tout au long des cent et quelque pages précédentes.

Aussi, pour rendre cette tâche facile pour l'auteur et agréable pour le lecteur, nous allons procéder avec méthode : la problématique de base sera présentée à travers l'intitulé même de la thèse à savoir : « un modèle de mémoire de cas pour une réutilisation optimale ». Chaque mot clé de ce titre sera expliqué, justifié, placé dans un contexte de modélisation puis dans un contexte d'expérimentation et enfin évalué et critiqué pour donner lieu à de nouvelles perspectives.

En raisonnement à base de cas la réutilisation est une tâche centrale au processus d'inférence, elle est constituée de deux sous tâches : la copie et l'adaptation. En générale, il existe une nette séparation entre cette tâche et celle de la remémoration qui est dans la majeure partie du temps, guidée par un critère de similarité. De ce fait, et comme il a été établi de manière expérimentale (voir chapitre V), ce ne sont pas toujours les cas les plus similaires qui sont les plus faciles à adapter.

Si l'on veut donc optimiser la réutilisation des cas, il faut intégrer une connaissance sur leur potentialité à être adaptés dans le processus de rappel. Ce que nous avons proposé sous forme d'une équation de réutilisation. Cette équation exprime l'équilibre entre les éléments constitutifs de la réutilisation, en tenant compte de l'aspect dual de la copie/adaptation d'une part et de la similarité /dissimilarité d'autre part.

Cette équation a été intégrée par la suite, dans un modèle de mémoire, pour mettre en œuvre le processus de rappel.

Nous avons donc étudié les notions liées à la mémoire dans les systèmes de raisonnement à partir de cas. Nous avons détaillé les différents modèles de mémoire utilisés.

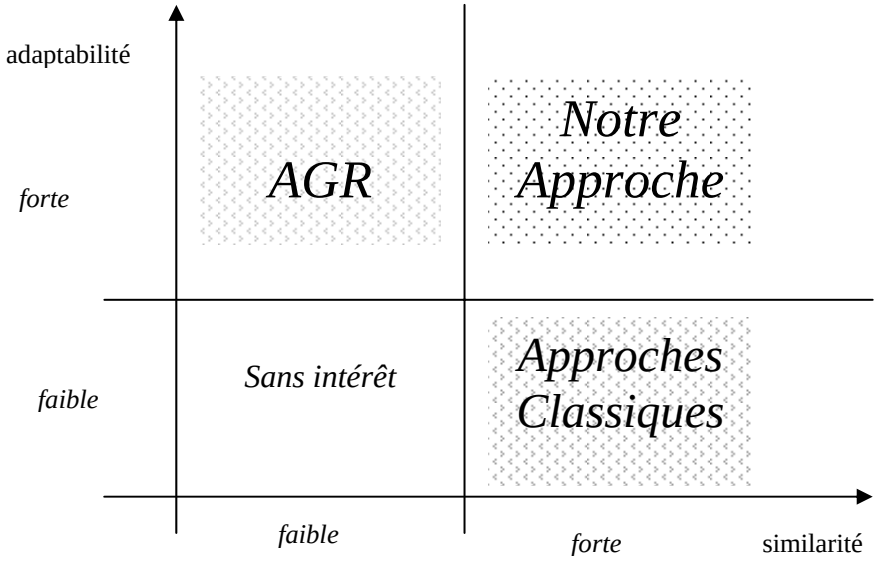
Nous avons ensuite proposé un modèle de mémoire qui intègre le critère de similarité et le critère d'adaptabilité il s'agit de l'AGRN, et ce en étendant un modèle déjà existant et très performant le CRN.

Nous avons démontré les avantages d'une telle combinaison par rapport aux deux critères pris séparément.

Nous avons finalement évalué expérimentalement ce modèle sur les bases de deux applications.

Il est important de noter que le modèle de mémoire proposé constitue un cadre général d'étude. Nous ne nous sommes imposés ni restriction de domaine d'application ni contrainte de tâche de résolution. Si ce n'est le fait que le processus de résolution doit intégrer une étape d'adaptation et ce pour justifier une recherche orientée adaptabilité.

Si l'on devait situer notre travail par rapport aux travaux sur les mémoires de cas et l'approche AGR, la figure suivante synthétise parfaitement notre position :



De multiples extensions des BCRNs ont été proposées par Lenz et son équipe (voir **[Len 1996a]** et **[Len 1996b]** ainsi que **[Len 1999]**). Parmi les quelles deux approches nous intéressent tout particulièrement :

Approche : diffusion paresseuse de l'activation (Lazy spreading activation)

Il s'agit d'une extension permettant une amélioration du processus de recherche dans la mesure où la portée de l'activation des IEs qui habituellement se fait des plus similaires vers les moins similaires, sera inhibée en cours de route dès l'instant où les IEs jusque là activé nous suffisent (il y a satisfaction du seuil de similarité). Cela permet l'activation d'un sous

réseau au lieu de la totalité du réseau. Relativement à l'extension orientée adaptabilité, ce processus de diffusion paresseuse n'est pas applicable car il procèderait à une réduction de l'espace de recherche par un critère de similarité sans tenir compte du critère d'adaptabilité.

Approche : OCRN pour Object Case Retrieval Net

Il s'agit d'un enrichissement dans la structure du cas. Au lieu d'utiliser une structure plate uniquement, un modèle objet du cas va sus planter la couche Entités d'information offrant ainsi une orientation lors du choix des entités à activer. L'application du critère d'adaptabilité à l'OCRN est une idée intéressante à plus d'un titre :

- Une description objet du cas est nettement plus riche que celle trouvée dans les CRN de base.
- L'évaluation du coefficient d'adaptabilité pourrait se faire au niveau objet et non au niveau IE. Ce qui dans la pratique serait plus réaliste dans la mesure où une solution est constituée d'un ensemble d'objet et non d'un ensemble d'IEs.

Parmi les organisations de mémoire guidées similarité celle de Fish & Shrink est particulièrement intéressante à étudier dans l'éventualité d'enrichir ses facettes par une connaissance d'adaptabilité. Ce sera là une autre opportunité d'intégrer l'équation de réutilisation.

Par ailleurs, il serait intéressant d'envisager une recherche à critères multiples où l'on trouverai non seulement le critère classique de 'similarité' ou encore 'l'adaptabilité' mais aussi un 'critère d'utilité'. L'idée étant partiellement introduite dans 'similarity versus diversity' **[Smy 2001]**. Enrichir notre modèle par ce troisième critère et déterminer le cadre de son utilisation, constituent nos objectifs futurs.

Secondairement à cette problématique, nous avons dégagé les grandes lignes pour l'extraction des connaissances d'adaptabilité. Sur les trois approches proposées, deux d'entre elles ont été expérimentées.

L'approche heuristique est jugée simple de mise en œuvre. Son inconvénient majeur vient du fait que son exhaustivité dépend de la complétude des règles d'adaptation. Un oubli dans les règles biaiserait indubitablement les dépendances extraites.

L'approche AdaX quant à elle, est totalement indépendante des règles d'adaptation et s'inscrit dans une tradition de data mining. A l'exception qu'elle n'extrait pas de règles à partir des données (comme c'est le cas pour les techniques présentées en chapitre IV), mais elle extrait des dépendances. Malheureusement, son expérimentation nous a montré ses limites quant à sa sensibilité à la taille et à la distribution des données.

Elle présente aussi une limite d'applicabilité lorsque le cas est décrit de manière hiérarchisée.

En perspective, une expérimentation de l'approche base de données est incontournable.

Par ailleurs, un outil intégrant les trois approches gagnerait en efficacité et en qualité de réponse.

Comme il est souvent difficile de brider les pensées de l'esprit, deux autres approches de modélisation se sont offertes à nous :

- L'agentification du CBR a été abordée sur un plan purement modélisation. Nous oeuvrerons dans le futur au développement d'une plate forme agent supportant le paradigme CBR. Il y a certes beaucoup de travail encore mais il s'agit là d'une ouverture de recherche incommensurable.
- De même que celle offerte par la modélisation génétique de la mémoire. Bien que abordé sur le plan modélisation et expérimentation, ce modèle ne nous a pas encore livré tous ses secrets.

A travers cette thèse, nous avons tenté de retracer non seulement, comment la problématique avait été initialement posée, mais aussi comment a-t-elle évoluée jusqu'à arriver à maturation ou à bifurquer sur d'autres approches.

L'objectif n'étant pas uniquement de montrer une modélisation et son expérimentation mais plus encore, le méta raisonnement qui y a mené avec ses succès et surtout ses échecs pour que cela serve aux recherches futures.

Références bibliographiques

[Aam 1994]: A.Aamodt, E. Plaza, "Case-Based Reasoning : Fondational Issues", AI-Communications, Vol.7, N°1, pp.39-59, 1994.

[All 1994]: B. Allen «Case based reasoning : business applications». C-ACM, vol37, n°3, pp40-42, March 1994.

[Ash 1988]: K. D. Ashley, E. L. Rissland « A Case Based Approach to Modelling legal expertise», IEEE Expert, n°3, vol 3, pp:70-77, 1988.

[Bar 1992]: R. Bareiss, J. A. King, «Similarity Assessment in CBR», DARPA Workshop on CBR, 1992.

[Bar 1999]: B. Bartsch-Spörl & al, "Case-Based Reasoning Surveys and Future Direction", ed Springer, 1999.

[Ber 1998]: R.Bergmann, K. Althoff, « Methodology for Building Case-Based Reasoning Applications » in LNAI 1400, ed. Springer 1998.

[Bic 1994]: I. Bichindaritz, «A Case Based Reasoning System Using a Control Case Base», ECAI, Wiley & son, 1994.

[Bri 2000]: collectif d'auteurs dirigé par J-. P. Briot et Y. Demazeau, « Principes et architecture des systèmes multi-agents », édition Hèrmes 2000.

[Bro 1994]: M. Brown, « A Memory Model for Case Retrieval by Activation Passing », PhD Dissertation, Manchester University, United Kingdom, 1994.

[Cap 2002]: G. Caplat, « Modélisation cognitive et résolution de problème », éd. Presses polytechniques et universitaires Romandes, 2002.

[Col 1997]: B. Collins, P. Cunningham, "Adaptation-Guided Retrieval in EBMT: A case based approach to machine translation", in: Proceedings of EWCBR'96, LNAI 198, ed. Springer, 1997.

[Did 1982]: E. Diday & al, «Eléments d'analyse de données», ed. Dunod, 1982.

[Erz 1999]: M. Erzan & al, "Réutilisation logicielle: Guide pratique et retour d'expérience", ed. Eyrolles, 1999.

[Fer 1997]: J. Ferber, « les systèmes multi-agents : un aperçu général », Revue Techniques et Sciences Informatiques. Vol. 16 N°8, pp :979-10012. ed. Hermès 1997.

[Fuc 1997]: B. Fuch, « Représentation des connaissances pour le raisonnement à partir de cas. Système Rocade », thèse de Doctrat de l'université Jean Monnet de Saint_Etienne, France, 1997.

[Fuc 2000]: B. Fuchs, J. Lieber, A. Mille, A. Napoli, "An Algorithm for Adaptation in Case-based Reasoning", in: proceedings of ECAI 2000, pp:45-49, 2000.

[Gol 1994]: D. E. Goldberg, « Genetic algorithms », ed. Addison Wesley, 1994.

[Gom 1999]: A. Gomez de Silva Garza, M.L. Maher « An Evolutionary Approach to Case Adaptation », 3rd ICCBR'99, in LNAI 1650 , Germany, July 1999.

[Ham 1992]: K. Hammond, «Adaptation of cases», DARPA workshop on CBR, 1992.

[Hat 1991]: J.P. Haton & al «le raisonnement en IA». Interédition , 1991.

[Hat 1993]: J-.P Haton, «Le raisonnement dans les systèmes à base de connaissances», Le courrier du CNRS, n° 80, 1993.

[Hua 1996]: Y. Huang, « An Evolutionary Agent model of Case-Based Classification » in Proceedings of 3rd EWCBR'96, LNAI 1168, ed. Springer Verlag 1996.

[Hub 2000]: A. Hubner, M. Lenz et al , « Last minute travel Application » in AI Magazine Vol. 21, N°4, pp : 58-62, Winter 2000.

[Kod 1993]: Y. Kodratoff, «L'apprentissage», Le Courrier du CNRS N°80, Février 1993.

[Kol 1991]: J. Kolodner, «improving human decision through case based decision aiding », AI magazine, 1991.

[Kol 1992]: J. Kolodner, « Judging which is the best case for a case based reasoner », in DARPA Workshop on CBR, 1992.

[Kol 1993]: J. Kolodner, "Case Based Reasoning", ed. Morgan Kauffmann, 1993.

[Len 1996a]: M. Lenz, H.- D. Burkhard, "Lazy propagation in Case Retrieval Nets", in: proceedings of ECAI, p:127-131, ed. John Wiley and sons, 1996.

[Len 1996b]: M. Lenz, H.- D. Burkhard, "Case Retrieval Nets : basic Ideas and extentions", in: LNAI 1137, ed. Springer Verlag, 1996.

[Len 1998]: M. Lenz et al, « Diagnosis and decision support » in LNAI 1400, éd. Springer 1998.

[Len 1999]: M. Lenz, « Case Retrieval Nets as a Model for Building Flexible Information Systems », PhD Dissertation, Humboldt University, Berlin, Germany, 1999.

[Lie 2004]: J. Lieber & al , « Une étude comparative de quelques travaux sur l'acquisition de connaissances d'adaptation pour le raisonnement à partir de cas », 12 Atelier RàPC, Villetneuse, Paris, France, 26 Mars2004.

[Mal 1996]: M. Malek, « Un Modèle Hybride de Mémoire Pour le Raisonnement à Partir de Cas», thèse de Doctrat de l'université Joseph Fourier, Grenoble, France, 1997.

[Mar 1993]: C. Martin, «In dexing Using Complex Features», DARPA Workshop on CBR, 1993.

[Mar 2002]: C.Marling et al., « Case-Based reasoning Integrations », in AI Magazine, vol. 23, N°1, Spring 2002.

[Mas 1991]: J. Mascarola, «Réussir une enquête. Analyse univariée et bi-variée », éd. Vuibert, 1991.

[Mor 2001]: S. I. Morisbak, B. Tessem, « Agents for Case-Based Software Reuse » in Applied AI, Vol. 15, N°3, pp:297-332, March 2001.

[Mur 2001]: J..W. Murdock, A. K. Goel, « Meta-Case-Based reasoning : using functional models to adapt Case-Based Agents » in Proceedings of 4th ICCBR'2001,pp: 407-421, LNAI 2080, ed. Springer Verlag 2001.

[Nou 1999]: N. Nouaouria-Amri**,** N.Bounour, H. Bouraoui, I. Mebarki « CEPROC : système basé cas d'aide à laconstruction de programme"**,** Rapport de recherche interne, GRIA016/99, juin 1999**.**

[Nou 2001a]: N. Nouaouria-Amri, A. Bendjedid, M, Benchetioui, «TRAVAGE aide au choix de séjour touristique par CBR/CRN », Rapport de recherche interne, GRIA10/2001, juin 2001.

[Nou 2001b]: N. Nouaouria-Amri, T. Seridi, H. Boussaha «PROJESTIME : estimation du coût de développement d'un logiciel par CBR/CRN », Rapport de recherche interne, GRIA12/2001, juin 2001.

[Nou 2002]: N. Nouaouria-Amri, «PEDIAGNO : aide au dignostic en pédiatrie par CBR/CRN », Rapport de recherche interne, GRIA03/2002, juin 2002.

[Nou 2003]: N. Nouaouria-Amri, Med Tayeb Laskri « Towards a formal model of case based reasoning from the roots", proceedings CESA'03, du 9 au 11 Juillet 2003 à Lille

[Nou 2004a]: N. Nouaouria-Amri**,** Med Tayeb Laskri « Vers un modélisation multi-agent du raisonnement à base de cas"**,** SETIT, Sousse, Tunisie, pp :145, 15-20 Mars **2004.**

[Nou 2004b]: N. Nouaouria, M.T. Laskri « Adaptability versus Similarity : Why not the two ? », in proceedings of 9th UKWCBR, Cambridge, UK, December 13th , 2004.

[Nou 2005a]: N. Nouaouria-Amri, M. Menaa, H. Mira « Un outil Basé Cas pour la Recommandation de Configuration PC", Rapport de recherche interne, GRIA04/2005, juin 2005.

[Nou 2005b]: N. Nouaouria-Amri, F. Nacer, N. Zerguine « GUIDIETE :Un Système Basé Cas pour la Recommandation en Diététique", Rapport de recherche interne, GRIA05/2005, juin 2005.

[Nou 2005c]: N. Nouaouria-Amri, S. Achiri, S. Gharbi « AdaX : Un outil d'extraction de dépendances Problème/Solution", Rapport de recherche interne, GRIA06/2005, juin 2005.

[Pal 2004]: S. K. Pal, S. C. Shiu, « Foundations of Soft Case-Based Reasoning », ed. John Wiley & Sons Inc, 2004.

[Pla 1997]: E. Plaza, J. L. Arcos, F. Martin, « Coopérative Case-Based Reasoning » in Distribueted artificial intelligence meets Machine learning, éd. Springer Verlag 1997.

[Pur 1997]: L. Purvis, S. Athalye « Towards Improving Case Adaptability with a Genetic Algorithm », 2rd ICCBR'97, in LNAI 1266 , USA, July 1997.

[Ric 2004]: M. Richter, « Similarity Based Retrieval », Research Report, University of Kaiserslautern, Germany, 2004.

[Rip 1995]: T. Rippol, A. Tricot, « Quelques points de repères sur l'évolution de l'étude du raisonnement en psychologie cognitive », Cahiers pédagogiques, 344-345, N°spécial « apprendre à raisonner ? », pp : 37-40, 1995.

[Ris 1989]: E. Rissland, D. Skalak, «Case Based Reasoning in a Rule Governed domain», Proceedings of the 5th Conference on AI Applications, 1989.

[Rou 1993]: J. Rousu, R. J. Aarts, "Adaptation Cost as a criterion for solution evaluation", in: Proceedings of EWCBR'96, LNAI 1168, ed. Springer, 1993.

[Rou 1994]: S. ROUGEGREZ-LORIETTE, «Prédiction de processus à partir de comportement obsevés : le système REBECAS», thèse de doctorat d'université, Institut Blaise Pascal Paris VI, LAFORIA TH94/05, Juillet 1994.

[Sch 1996] : J. W. Schaaf, « Fish and Shrink. A Next Step Towards Efficient Case Retrieval in Large Scaled Case Bases », in proceedings of 3th EWCBR'96, LNAI 1168, ed. I. Smith & B. Faltings, Lausane, Swizeland, November 1996.

[Sma 1994]: M. Smail, «Raisonnement à base de cas pour une recherche évolutived'information; Prototype Cabri-n. Vers la définition d'un cadre d'acquisition de connaissances.», thèse de doctorat d'université, Univ. Henri Pointcaré - Nancy I, Octobre 1994.

[Smy 1993]: B.Smyth., M. T. Keane, "Retrieving Adaptable Cases. The role of adaptation knowledge in case retrieval", in: proceedings of EWCBR'93, LNAI 837, ed. Springer, 1993.

[Smy 1995]: B.Smyth., M. T. Keane, "Experiments on Adaptation–Guided Retrieval In Case-Based Design", in: Proceedings of ICCBR'95, LNAI 1010, ed. Springer, 1995.

[Smy 1998]: B. Smyth, E. McKenna, « Modelling the competence of case bases » in LNAI 1488, pp: 208-220, ed. Springer 1998.

[Smy 1999]: B.Smyth., M. T. Keane, "Adaptation-Guided Retrieval: Questioning the Similarity Assumption in Reasoning", in Journal of Artificial Intelligence, ed. Elsevier, 102(2), pp. 249-293, 1999.

[Smy 2001]: B. Smyth, P. McCLave, "Similarity versus Diversity", in: Proceedings of ICCBR'2001, LNAI 2080, pp: 347-361, ed. Springer, 2001.

[Sta 2002]: A. Stahl, S. Schmitt, "Optimizing Retrieval in CBR by Introducing Solution Similarity", in: Proceedings of International Conference on Artificial Intelligence ICAI'2002, Las Vegas, USA, 2002.

[Wal 1992]: D. Waltz & al, « Trading MIPS and Memory for Knowledge Engineering», C-ACM, vol 35, n°8, August 1992.

[Wat 1995]: I. Watson, « An Introduction to Case-Based Reasoning » in LNAI 1020, éd. Springer 1995.

[Wat 1997]: I. Watson, "Applying Case-Based Reasoning : techniques for enterprise systems", editions Morgan Kaufman, 1997.

[Wat 1999]: I. Watson, "Case-Based Reasoning is a methodology not a technology", Knowledge Based Systems 12(1999) 303-308, ed. Elsevier Science, 1999.

Annexes

Annexe 1

Les Réseaux de Recherche de Cas

Le principe des CRNs est inspiré des réseaux de neurones et des modèles de mémoires associatives. L'idée est que le processus de rappel d'un cas ne se fait pas en parcourant un chemin dans une arborescence mais plutôt de façon reconstructive en récupérant graduellement les entités d'information constituants le cas.

Les connaissances de base dans les CRNs sont les entités d'information (IEs). Un cas est un ensemble de ces entités. Une mémoire de cas est alors un réseaux de nœuds correspondant aux IEs du domaine ainsi que de nœuds additionnels dénotant le cas. Les nœuds IEs sont connectés par des arcs de ***similarité*** et les nœuds cas sont accessibles à partir des IEs Ies constituants à travers des arcs de ***relevance***. Différents degrés de similarité et de relevance peuvent être exprimés en faisant varier le poids des arcs.

A partir de cette structure la recherche est faite en trois étapes :

1. Activation des IEs donnés par la requête (cas à résoudre).
2. propagation des activations correspondant à la similarité à travers le réseau des IEs,
3. collecte des activations finales dans les nœuds cas correspondants.

De manière formelle, on utilisera les définitions suivantes :

Définition 1 : ***(Entité d'information)***

Une entité d'information (IE) est une connaissance atomique dans le domaine. Elle présente la plus fine granularité dans la représentation des connaissances du cas et de la requête.

Définition 2 : ***(Le cas)***

Le cas consiste en un unique descripteur (ou identificateur) de cas et d'un ensemble d'IEs. La requête est constituée uniquement d'un ensemble d'IEs.

Définition 3 : ***(Le réseau basique pour la recherche de cas BCRN)***

Un réseau basique de recherche de cas (BCRN) est définie par la structure

$N = [E,C,\sigma,\rho,\Pi]$ avec :

E est un ensemble fini de nœuds IEs.

C est un ensemble fini de nœuds cas.

σ : est la fonction de similarité $\sigma : E \times E \rightarrow \Re$

elle décrit la similarité $\sigma(e_i,e_j)$ entre le IEs e_i et e_j.

ρ : est la fonction de relevance $\rho : E \times C \rightarrow \Re$

elle décrit la relevance ρ (e,c) du IEs e au nœud cas c.

Π :est l'ensemble des fonctions de propagation $\pi_n : \Re^E \rightarrow \Re$.

Pour chaque nœud $n \in E \cup C$

La description graphique est donnée par un graphe comportant les nœuds $E \cup C$ et les arcs dirigés étiquetés par $\sigma(e_i,e_j)$ entre le IEs e_i et e_j et ρ(e,c) du IEs e au nœud cas c.

Définition 4 : ***(L'activation d'un BCRN)***

L'activation d'un BCRN $N = [E,C,\sigma,\rho,\Pi]$ est une fonction

$\alpha : E \cup C \rightarrow \Re$

α(e) exprime l'importance de l'IE e vis à vis du problème posé (la requête).

Définition 5 : ***(Le processus de propagation dans un BCRN)***

Considérons un BCRN $N = [E,C,\sigma,\rho,\Pi]$ avec $E = \{e_1,\ldots,e_s\}$ et soit $\alpha_t : E \cup C \rightarrow \Re$ la fonction d'activation à l'instant t. L'activation des IEs nœuds $e \in E$ à l'instant t+1 est donnée par :

$$\alpha_{t+1}(e) = \pi_e(\sigma(e_1,e) \bullet \alpha_t(e_1) ,\ldots, \sigma(e_s,e) \bullet \alpha_t(e_s))$$

et l'activation d'un noeud $c \in C$ à l'instant t+1 est donnée par :

$$\alpha_{t+1}(c) = \pi_c(\rho\ (e_1,c) \bullet \alpha_t(e_1) ,\ldots, \rho\ (e_s,c) \bullet \alpha_t(e_s))$$

Lorsqu'une requête est posée, l'activation initiale des noeuds est donnée par :

$\alpha_0(e) =$ ⎧ 1 : pour les IEs e décrits dans la requête.

⎩ 0 : sinon

Pour une prise en compte plus subtile des requêtes, α_0 pourrait assigner des poids à des IEs spécifiques ou un contexte d'initialisation.

Le processus de recherche des cas par propagation des activations se fera en trois étapes :

Etape 1 – Activation initiale :

α_0 est déterminé pour les IEs relativement à la requête.

Etape 2 – Propagation des similarités :

α_0 est propagée à tous les noeuds IEs e_i du réseaux :

$$\alpha_1(e) = \pi_e(\sigma(e_1,e) \bullet \alpha_0(e_1) ,\ldots, \sigma(e_s,e) \bullet \alpha_0(e_s))$$

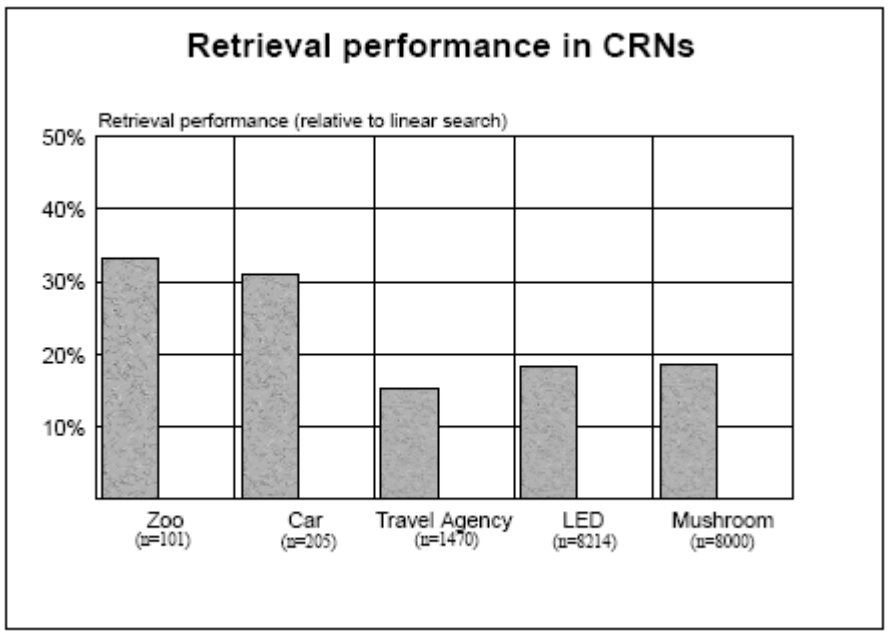
Etape 3 – Propagation des relevances :

Le résultat de l'étape 2 est propagé à tous les noeuds c du réseau :

$$\alpha_2(c) = \pi_c(\rho\ (e_1,c) \bullet \alpha_1(e_1) ,\ldots, \rho\ (e_s,c) \bullet \alpha_1(e_s))$$

Dans la pratique, considérer les IE comme des entités atomiques n'est pas une approche intéressante puisque cela nécéssiterait la spécification complète des fonctions de similarité. Par ailleurs, ce niveau de granularité peu ne pas être conviviale d'un point de vue représentation des cas pour des applications réelles. De ce fait, il serait plus commode d'envisager l'utilisation des CRN à micro-caractéristiques (microfeature CRNs). Les IEs dans cette extension de CRN ne sont plus des entités atomiques non décomposables mais leur granularité sera fixée selon l'application choisie.

Les performances des CRNs relativement à une recherche linéaire, comme le montre la figure suivante :



Sont entre 15% et 32% fois plus rapides.

Annexe 2

Connaissances utilisées dans PC_CONFIG

Heuristiques de similarité Utilisées dan PC_Config

- ✓ **Les heuristiques dégagées pour les carte mère** : on doit utiliser 8 paramètres savoir (2):

-Chipset

-Socket

-Ram max

-Type de réseau

- Les ports (PCI, USB, SATA, DMA).

On propose les heuristiques suivantes:

α (Chipset) = 0.4

Si Id .chipset. s= Id .chipset.c **alors** Sim(s, c) =1.

Sinon Sim(s, c) =0.

On a affecte ce poids au chipset car il est un composant principale dans la carte mère dont son rôle est d'interconnecter et de faire dialoguer les différents composantes d'extension.
Et conditionne notamment les performances et l'amélioration.

α (Socket) = 0.2 car il détermine la compatibilité du CPU à la carte mère.

α (Ram_max)= 0.1

α (Type_Reseau)=0.1.

ET les 0.20 restants seront partagées entre les ports suivants : USB, PCI, SATA, DMA.

- ✓ **Les heuristiques dégagées pour les cartes graphiques:**

Ont doit utiliser 6 paramètres pour le calcul de similarité globale (2) entre las cartes graphiques à travers les similarités locales entre ses paramètres à savoir :

α (CPU Graphique)= 0.3.

Si CPU Graphique.s = CPU Graphique.c **alors** Sim(s, c) = 1

Sinon Sim(s, c) = 0.

α(Type de carte) = 0.3

Si Source. Type de carte = Cible. Type. Carte **alors** Sim(s, c) = 1

Sinon Sim (s, c) = 0

α(Mémoire vidéo) = 0.2.

α (Interface physique) = 0.05

α (Entrée/Sortie) = 0.05

α (Interface d 'affichage) = 0.5

On a attribuer le poids (**α)** le plus élevé au type de carte et processeur graphique puisqu' ils indiquent les performance de la carte graphique, ainsi la compactibilité et infusent sur les autres caractéristiques.

Les règles d'adaptation.

1. Si (Cible. Multimédia > 7) Et (Source. Multimédia <3) alors Modifier (Carte graphique avec TV Out).

2. (cible. Multimédia ≥ 5) Et (Source. Multimédia <3) alors Ajouter (Lecteur DVD) .

3. Si (Cible. Bureautique <3) Et (Source. Bureautique > 7) alors Supprimer (imprimante)

4. Si (Cible. Jeux <3) Et (Source. Jeux> 7) alors Supprimer (Option de jeux).

5. Si (Cible. Jeux > 7) Et (Source. Jeux <3) alors Modifier (Carte graphique) , Modifier(Processeur) Et Ajouter (Option de jeux).

6. Si (Cible. Jeux ≥ 5) Et (Source. Jeux <3) alors Modifier (RAM) .

7. Si (Cible. Programmation > 7) Et (Source. Programmation <3) alors Modifier (Processeur) ET Modifier (RAM).

8. Si (Cible. Traitement Image > 7) Et (Source. Traitement Image <3) alors Ajouter (Carte graphique) , ajouter(scanner).

9. Si (Cible. Musique > 7) ET (Source .Musique <2) alors Ajouter (Carte d' acquisition audio).

10. Si (Cible. Bureautique > 8) ET (Source. Bureautique <4) alors :
- Ajouter (Modem interne).
- Modifier (Ecran 17) .
-Ajouter (Souris optique).
-Ajouter Imprimante.

11. Si (Cible. Programmation ≥ 9) Et (Source. Programmation <2) alors :
-Modifier (Processeur)
-Ajouter (RAM).
-Modifier (Disque dur).
-Ajouter (Graveur).

12- Si Ajouter (Pièce) alors Prix PC= Ancien Prix+Prix Pièce.

13. Si Supprimer (Pièce) alors Prix PC= Ancien Prix - Prix Pièce.

14. Si (Cible. Internet <= 1) Et (Source. Internet =0) alors :
-Ajouter (Modem).

15. Si (Cible. Montage vidéo <= 1) Et (Source. Montge vidéo =0) alors :
-Ajouter (carte graphique)
-Ajouter (carte son)
-Ajouter (lecteur DVD)

Les Dépendances problème/solution.

Pbm / Sol	Multimedia	Bureautique	Jeux	Programmation	Trt Images	musique	Intenet	Montage Vidéo
Carte graphique	+		+		+			+
Lecteur DVD	+							+
DD				+				
Imprimante		+		+				
Graveur				+				
Marque CPU			+	+				
Vitesse CPU			+	+				
Taille RAM			+	+				
Type RAM			+	+				
Scanner					+			
Modem							+	
Nombre de pouces		+						
Souris		+						
Carte audio						+		+
Prix	+	+	+	+	+	+	+	+

Tableau. Dépendance problème / solution dans PC_CONFIG

Annexe 3

Connaissances utilisées dans GUIDIETE

Heuristique d'adaptation:

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Prod_lait**.** Lait_gélifié) par (Prod_lait**.** Lait_demi écrémé)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Prod_lait**.**Concentré_sucré) par (Prod_lait**.**Ecrémé)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Prod_lait**.**Fromage blanc) par (Prod_lait**.**Yaourt nature)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Prod_lait**.**Yaourt suisse) par (Prod_lait**.**Fromages)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Prod_lait**.**Yaourt suisse) par (Prod_lait**.**Glaces et Entremets)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Viande**.**Panées) par (Viande**.**Toutes les viandes)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Viande**.**Rissolettes) par (Viande**.**Toutes les viandes)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Viande**.**Poissons panés) par (Viande**.**Fruits de mer)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Viande**.**Poissons panés) par (Viande**.**Cristacés)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.**Pain d'épice) par (Féculent**.** Pain complet)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Chausson) par (Féculent**.** Pain blanc)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Pain au chocolat) par (Féculent**.**Pain seigle)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Brioche) par (Féculent**.** Pain grillé)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Biscuiterie) par (Féculent**.** Biscottes)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Farine) par (Féculent**.** Pomme de terre)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Pâtisserie) par (Féculent**.** Riz)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Pâtisserie) par (Féculent**.** Pâtes)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Pâtisserie) par (Féculent**.**Céreales)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Féculent**.** Farine) par (Féculent**.** Légumes secs)]-(Selon les quantités autorisées)-

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Fruits au sirop) par (Fruits**.** Fruits frais)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Fruits confits) par (Fruits**.** Fruits surgelés)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Pâtes de fruit) par (Fruits**.** Fruits crus)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Fruits secs) par (Fruits**.** Fruits cuits sans sucre)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Fruits**.** Marrons glacés) par (Fruits**.** Fruits surgelés)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Mat_grass**.** Friture) par (Mat_grass**.** Beurre)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Mat_grass**.** Charcuterie) par (Mat_grass**.** Huile)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Mat_grass**.** Végetaline) par (Mat_grass**.** margarine)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Mat_grass**.** Plats en sauce) par (Mat_grass**.** Crème fraîche)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Boisson**.** Limonade) par (Boisson**.** Eau ordinaire)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Boisson**.** Boissons fruités) par (Boisson**.** Eau Minérale)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Boisson**.** Sirop de fruits) par (Boisson**.** Thé)].

***Si** (Maladie = "Diabète") **A**lors **S**ubstituer [(Boisson**.** Limonade) par (Boisson**.** Café)].

Le coût d'adaptabilité (Diabète) =32.

*Si (Maladie = "Goutte") Alors Substituer [(Prod_lait. Fromage très fermenté) par (Prod_lait. Lait)].
*Si (Maladie = "Goutte") Alors Substituer [(Prod_lait. Fromage très fermenté) par (Prod_lait. Yaourt naturel)].
*Si (Maladie = "Goutte") Alors Substituer [(Prod_lait. Fromage très fermenté) par (Prod_lait. Fromage blanc)].
*Si (Maladie = "Goutte") Alors Substituer [(Viande. Rognon) par (Viande. Viande de bœuf)].
*Si (Maladie = "Goutte") Alors Substituer [(Viande. Foie) par (Viande. Agneau)].
*Si (Maladie = "Goutte") Alors Substituer [(Viande. Charcuteries) par (Viande. Viande de bœuf)].
*Si (Maladie = "Goutte") Alors Substituer [(Viande. Volailles) par (Viande. Jambon)].
*Si (Maladie = "Goutte") Alors Substituer [(Viande. Extraits de viande) par (Viande. Œufs)].
*Si (Maladie = "Goutte") Alors Supprimer (Poisson. Gras).
*Si (Maladie = "Goutte") Alors Supprimer (Poisson. Anchois).
*Si (Maladie = "Goutte") Alors Supprimer (Poisson. Sardines).
*Si (Maladie = "Goutte") Alors Supprimer (Poisson. Hreng).
*Si(Maladie="Goutte") Alors Supprimer(Poisson.Truite).
*Si (Maladie = "Goutte") Alors Supprimer (Poisson. Carpe).
*Si (Maladie = "Goutte") Alors Supprimer (Poisson. Saumon).
*Si (Maladie = "Goutte") Alors Supprimer (Légumes. Asperges).
*Si (Maladie = "Goutte") Alors Supprimer (Légumes. Champignons).
*Si (Maladie = "Goutte") Alors Supprimer (Légumes. Légumes secs).
*Si (Maladie = "Goutte") Alors Supprimer (Légumes. Chou fleur).
*Si (Maladie = "Goutte") Alors Supprimer (Légumes. Oseille).
*Si (Maladie = "Goutte") Alors Substituer [(Sucreries. Chocolat) par (Sucreries. Tous les autres sucres)].
*Si (Maladie = "Goutte") Alors Substituer [(Sucreries. Chocolat) par (Sucreries. Céréales)].
*Si (Maladie = "Goutte") Alors Substituer [(Sucreries. Chocolat) par (Sucreries. Pâtes)].
*Si (Maladie = "Goutte") Alors Substituer [(Sucreries. Chocolat) par (Sucreries. Riz)].
*Si (Maladie = "Goutte") Alors Substituer [(Sucreries. Chocolat) par (Sucreries. Semoule)].
*Si (Maladie = "Goutte") Alors Substituer [(Boissons. Alcool) par (Boissons. Eau de vichy)].
*Si (Maladie = "Goutte") Alors Substituer [(Boissons. Alcool) par (Boissons. Eau de minérale)]

-Le coût d'adaptabilité (Goutte) =27.

*Si (Maladie = "Colite_Spasmodique") Alors Supprimer (Boissons. Alcool).
*Si (Maladie = "Colite_Spasmodique") Alors Supprimer (Boissons. Café).
*Si (Maladie = "Colite_Spasmodique") Alors Supprimer (Boissons. Boissons gazeuses).
*Si (Maladie = "Colite_Spasmodique") Alors Ajouter (Féculent. Céréales).
*Si (Maladie = "Colite_Spasmodique") Alors Ajouter (Prod_lait. Fromages frais).
*Si (Maladie = "Colite_Spasmodique") Alors Substituer [(Viande. Charcuterie) par (Viande. Viande blanche)].
*Si (Maladie = "Colite_Spasmodique") Alors Substituer [(Viande. Viande grasse) par (Viande. Poissons)].
*Si (Maladie = "Colite_Spasmodique") Alors Supprimer (Légume. Moutarde).
*Si (Maladie = "Colite_Spasmodique") Alors Supprimer [(Sucreries. Gâteaux) par (Féculent. Pain)].

*Si (Maladie = "Colite_Spasmodique") Alors Supprimer [(Sucreries. Pâtisseries) par (Féculent. Pâtes)].

*Si (Maladie = "Colite_Spasmodique") Alors Substituer [(Mat_grasse. Graisses d'origine animale) par (Mat_grasse. L'huile d'olive).

-Le coût d'adaptabilité (Colite_ Spasmodique) =11.

*Si (Maladie = "Ulcère") Alors Substituer [(Légumes. Choux) par (Légumes. Pomme de terre)].

*Si (Maladie = "Ulcère") Alors Substituer [(Légumes. Brocolis) par (Légumes. Courgettes)].

*Si (Maladie = "Ulcère") Alors Substituer [(Légumes. Poivrons) par (Légumes. Carotte)].

*Si (Maladie = "Ulcère") Alors Substituer [(Fruit. Figue) par (Fruit. Pommes)].

*Si (Maladie = "Ulcère") Alors Substituer [(Fruit. Orange) par (Fruit. Poires)].

*Si (Maladie = "Ulcère") Alors Substituer [(Fruit. Kiwi) par (Fruit. Bananes)].

*Si (Maladie = "Ulcère") Alors Substituer [(Fruit. Fruits acides) par (Fruit. Fruits cuits)].

*Si (Maladie = "Ulcère") Alors Substituer [(Viande. Viandes grasses) par (Viande. Viandes maigres)].

*Si (Maladie = "Ulcère") Alors Substituer [(Viande. Charcuterie) par (Viande. Viande blanche)].

*Si (Maladie = "Ulcère") Alors Substituer [(Poisson. Crustacés) par (Poisson. Poissons)].

*Si (Maladie = "Ulcère") Alors Substituer [(Boisson. Boisson gazeuse) par (Boisson. Eau minérale non gazeuse)].

*Si (Maladie = "Ulcère") Alors Substituer [(Boisson. Alcool) par (Boisson. Lait)].

*Si (Maladie = "Ulcère") Alors Substituer [(Mat_grasse. Graisses d'origine animale) par (Mat_grasse. L'huile d'olive)].

*Si (Maladie = "Ulcère") Alors Substituer [(Féculent. Cacao) par (Féculent. Biscottes)].

*Si (Maladie = "Ulcère") Alors Substituer [(Féculent. Pain complet) par (Féculent. Céréales)].

*Si (Maladie = "Ulcère") Alors Supprimer (Légume. Poivre).

*Si (Maladie = "Ulcère") Alors Supprimer (Légume. Moutarde).

*Si (Maladie = "Ulcère") Alors Supprimer (Boisson. Café).

*Si (Maladie = "Ulcère") Alors Supprimer (Légume. Bouillons cubes).

*Si (Maladie = "Ulcère") Alors Supprimer (Sucreries. Glaces).

*Si (Maladie = "Ulcère") Alors Ajouter (Prod_lait. Fromages).

*Si (Maladie = "Ulcère") Alors Ajouter (Prod_lait. Yaourts).

*Si (Maladie = "Ulcère") Alors Ajouter (Prod_lait. Fromages frais).

*Si (Maladie = "Ulcère") Alors Ajouter (Prod_lait. Lait).

-Le coût d'adaptabilité (Ulcère) =24.

*Si (Maladie = "Constipation") Alors Supprimer (Légume. Poivre).

*Si (Maladie = " Constipation") Alors Supprimer (Légume. Moutarde).

*Si (Maladie = "Constipation") Alors Supprimer (Légume. Citron).

*Si (Maladie = "Constipation") Alors Substituer [(Féculent. Farine d'avoine) par (Féculent. Céréales complets)].

*Si (Maladie = "Constipation") Alors Substituer [(Féculent. Riz) par (Féculent. Pain)].

*Si (Maladie = "Constipation") Alors Substituer [(Féculent. Semoule) par (Féculent. Céréales complets)].

***S**i (Maladie = "Constipation") **A**lors **S**ubstituer [(Mat_grasse**.** Graisses d'origine animale) par (Mat_grasse**.** L'huile d'olive)].

***S**i (Maladie = "Constipation") **A**lors **S**ubstituer [(Boisson**.** Alcool) par (Boissons**.** Eau minérale)].

***S**i (Maladie = "Constipation") **A**lors **S**ubstituer [(Boisson**.** Boissons gazeuses) par (Boissons**.** Eau minérale)].

***S**i (Maladie = "Constipation") **A**lors **S**upprimer (Sucreries**.** Crèmes renversées).

***S**i (Maladie = "Constipation") **A**lors **S**upprimer (Sucreries**.** Chocolat).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Prod_lait**.** Lait).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Prod_lait**.** Yaourts naturels).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Prod_lait**.** Fromages frais).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Légumes**.** Légumes verts).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Fruits**.** Fruits avec peau).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Viande**.** Viande).

***S**i (Maladie = " Constipation") **A**lors **A**jouter (Viande**.** Poissons).

-Le coût d'adaptabilité (Constipation) =18.

***S**i (Maladie = "Hyper_Cholestérolémie") **A**lors **S**ubstituer [(Prod_lait**.** Lait entier liquide en poudre) par (Prod_lait**.** Lait entier écrémé en poudre)].

***S**i (Maladie = "Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Lait écrémé)].

***S**i (Maladie = "Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Prod_lait**.** Petits suisse) par (Prod_lait**.**Yaourt naturel)].

***S**i (Maladie = "Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Prod_lait**.**Yaourt au lait entier) par (Prod_lait**.** Yaourts de 0% de MG)].

***S**i (Maladie = "Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Prod_lait**.** Fromages triple crème) par (Prod_lait**.** Fromage blanc)].

***S**i (Maladie = "Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Prod_lait**.** Crème fraîche) par (Prod_lait**.** Fromage blanc à 10% de MG)].

***S**i (Maladie = "Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Viande**.** Bœuf) par (Viande**.** Bœuf maigre)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Viande**.** Mouton) par (Viande**.** Cheval)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Viande**.** Jambon cuit) par (Viande**.** Poulet)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Viande**.** Charcuterie) par (Viande**.** Gibier)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Viande**.** Corned bœuf) par (Viande**.** Jambon à l'os maigre)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Poisson**.** Soupe de poisson) par (Poisson**.** Poisson cuisiné du commerce allégé)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Poisson**.** Panés) par (Poisson**.** Surgelé)].

***S**i (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Poisson**.** Poisson en conserve à l'huile végétale -sauce-) par (Poisson**.** Poisson en conserve naturel)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Poisson**.** Panés) par
(Poisson**.** Surgelé)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Féculent**.** Pâtes aux œufs) par (Féculent**.** Farine blanche)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Féculent**.** Muesli avec noix de coco) par (Féculent**.** Biscottes)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Féculent**.** Pain au fromage) par (Féculent**.** Pain complet)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Féculent**.** Brioche au beurre) par (Féculent**.** Pain Blanc)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Féculent**.** Biscuits apéritif) par (Féculent**.** Pain aux céréales)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Féculent**.** Croissant ordinaire) par (Féculent**.** Croissant à base de margarine ou tournesol)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Pâte chocolatée) par (Sucreries**.** Pâte aux fruits)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Chocolat au lait) par (Sucreries**.** Confiture)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Nougat) par (Sucreries**.** Marrons glacés)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Pâte d'amande) par (Sucreries**.** Fourré aux fruits)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Entremets aux œufs) par (Sucreries**.** Entremets ne contenant pas d'œufs)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Crème pâtissière) par (Sucreries**.** Crème de marrons)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Sucreries**.** Lait gélifié) par (Sucreries**.** Meringue)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Sucreries**.** Pâtisseries de maison contenant des ingrédients déconseillés).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Sucreries**.** Crème au beurre).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Sucreries**.** Toute sorte de chocolat).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Fruit**.** Pommes dauphine).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Fruit**.** Pommes noisette).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Fruit**.** Amande) par
(Fruit**.** Abricot sec)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Fruit**.** Noisette) par (Fruit**.** Raisin sec)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Fruit**.** Pistaches) par (Fruit**.** Pruneaux sec)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Fruit**.** Cacahuète)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Fruit**.** Noix de coco)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **A**jouter (Fruit**.** Fruits au sirop)

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors Substituer [(Légumes**.** Olives noires) par (Légumes**.** Olives verts)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Légumes**.** Purée au lait) par (Légumes**.** Pomme de terre)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Légumes**.** Olives noires) par (Légumes**.** Olives verts)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Mat_grass**.** Huile de palme).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Mat_grass**.** Margarine ordinaire).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Mat_grass**.** Beurre).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Mat_grass**.** Mayonnaise).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Mat_grass**.** Graisse d'oie).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Boisson**.** Lait entier) par (Boisson**.** Lait écrémé)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Boisson**.** Lait fermenté) par (Boisson**.** Lait écrémé)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Boisson**.** Chocolat) par (Boisson**.** Sirop de fruits)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**ubstituer [(Boisson**.** Lait de poule) par (Boisson**.** Café)].

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Boisson**.** Boissons alcoolisés).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **A**jouter (Boisson**.** Jus de fruits).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **A**jouter (Boisson**.** Bouillon de légumes).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **A**jouter (Boisson**.** Bouillon de viandes dégraissés).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **A**jouter (Boisson**.** Sodas).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **A**jouter (Boisson**.** Boissons à base de soja).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Thym).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Origan).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Poivre).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Persil).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Menthe).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Basilic).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Ail).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Moutarde).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Vinaigre).

***Si** (Maladie = " Hyper_Cholestérolémie ") **A**lors **S**upprimer (Ingrédients**.** Oignons).

-Le coût d'adaptabilité (Hyper_ Cholestérolémie) =67.

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Boissons**.** Boissons gazeuses).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Boissons**.** Alcools).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Boissons**.** Boissons gazeuses).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Féculents**.** Biscuits apéritifs).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Féculents**.** Cacao).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Ingrédients**.** Le sel).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Ingrédients**.** Les conserves).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Prod_lait**.** Fromages).

***Si** (Maladie = " Grossesse ") **A**lors **S**upprimer (Sucreries**.** Toutes les formes).

***Si** (Maladie = " Grossesse ") **A**lors **S**ubstituer [(Viande**.** Viandes grasses) par (Viande**.** Viandes maigres)].

***Si** (Maladie = " Grossesse ") **A**lors **S**ubstituer [(Mat_grass**.** Graisses d'origine animale) par (Mat_grass**.** L'huile d'olive)].

***Si** (Maladie = " Grossesse ") **A**lors **A**jouter (Prod_lait**.** Produits laitiers frais).

***Si** (Maladie = " Grossesse ") **A**lors **A**jouter (Légumes**.** Légumes verts).

***Si** (Maladie = " Grossesse ") **A**lors **A**jouter (Légumes**.** Légumes).

***Si** (Maladie = "Grossesse ") **A**lors **A**jouter (Féculents**.** Céréales).

***Si** (Maladie = "Grossesse ") **A**lors **A**jouter (Féculents**.** Les neufs).

***Si** (Maladie = "Grossesse ") **A**lors **S**ubstituer [(Fruits**.** Fruits secs) par (Fruits**.** Fruits frais)].

***Si** (Maladie = "Grossesse ") **A**lors **S**ubstituer [(Fruits**.** Fruits confits) par (Fruits**.** Fruits frais)].

-Le coût d'adaptabilité (Grossesse) =18.

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Féculent**.** Les épices).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Féculent**.** Aliments en conserve).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Féculent**.** Vinaigre).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Féculent**.** Aliments fumés).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Légume**.** Citron).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Légume**.** Choux).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Légume**.** Champignon).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Légume**.** Brocolis).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Légume**.** Cornichons).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Légume**.** Poivrons).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Fruit**.** Raisin).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Fruit**.** Figues).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Fruit**.** Melons).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Fruit**.** Pastèque).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Sucreries**.** Chocolat).

***Si** (Maladie = " Gastrite chronique") **A**lors **S**upprimer (Boissons**.** Bouillon de viande).

***Si** (Maladie = " Gastrite chronique ") **A**lors **S**ubstituer [(Mat_grass**.** Graisse d'origine animale) par (Mat_grass**.** L'huile d'olive)].

***Si** (Maladie = " Gastrite chronique ") **A**lors **S**ubstituer [(Boissons**.** Les alcools) par (Boissons**.** Eau minérale non gazeuse)].

***Si** (Maladie = " Gastrite chronique ") **A**lors **S**ubstituer [(Boissons**.** Boissons gazeuses) par (Boissons**.** Eau minérale non gazeuse)].

***Si** (Maladie = " Gastrite chronique") **A**lors **A**jouter (Féculent**.** Pain plutôt sec).

***Si** (Maladie = " Gastrite chronique") **A**lors **A**jouter (Féculent**.** Pâtes).

***Si** (Maladie = " Gastrite chronique") **A**lors **A**jouter (Féculent**.** Riz).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Féculent**.** Semoule).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Viande**.** Viandes maigre) et (Viande**.** Viande blanche).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Prod_lait**.** Lait).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Prod_lait**.** Yaourts).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Prod_lait**.** Fromages frais).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Légumes**.** Légumes verts).

***S**i (Maladie = " Gastrite chronique") **A**lors **A**jouter (Fruits**.** Fruits).

-Le coût d'adaptabilité (Gastrite chronique) =29.

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Préparation industrielles) par (Prod_lait**.** Lait de vache liquide)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Lait gélifié) par (Prod_lait**.** Lait de vache en poudre)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Fromages blancs)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Fromages blancs)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Yaourts)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Suisses)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Crème gruyère)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Viande**.** Viandes cuisiné du commerce) par (Viande**.** Viandes fraîches)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Viande**.** Viandes panées) par (Viande**.** Viandes fraîches surgelées)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Viande**.** Viandes en conserve) par (Viande**.** Jambon cuit)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Viande**.** Charcuterie) par (Viande**.** Œufs)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Poissons**.** Poissons cuisinés du commerce) par (Poissons**.** Poissons frais)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Poissons**.** Poissons Surgelés panés) par (Poissons**.** Poissons frais non panés) et (Poissons**.** Poissons frais non farinés)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Poissons**.** Sardine).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Poissons**.** Thon à l'huile naturel).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Poissons**.** Crustacés).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Poissons**.** Caviar).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Légumes**.** Légumes verts secs et cuisinés en conserve) par (Légumes**.** Légumes verts frais et non cuisinés en conserve au naturel)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Légumes**.** Potage de légumes du commerce) par (Légumes**.** Légumes secs)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Légumes**.** Sauce tomate du commerce) par (Légumes**.** Pomme de terre)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Légumes**.** Pomme de terre cuisinés en conserve) par (Légumes**.** Fécule de pomme de terre)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Fruits**.** Marron glacé) par (Fruits**.** Marrons frais)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Fruits**.** Crème de marron) par (Fruits**.** Marrons frais)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Fruits**.** Compote de fruits) par (Fruits**.** Sirop de fruits)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Fruits**.** Jus de fruits).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Fruits**.** Tous fruits au sirop).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Féculents**.** Blé seigle) par (Féculents**.** Maïs)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Féculents**.** Farines) par (Féculents**.** Farines de maïs)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Féculents**.** Pain blanc) par (Féculents**.** Pain sans gluten)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Féculents**.** Pain complet) par (Féculents**.** Pain sans gluten)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Féculents**.** Biscottes) par (Féculents**.** Semoules sans gluten)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Féculents**.** Couscous) par (Féculents**.** Pâtes sans gluten)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Féculents**.** Pâtisseries).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Féculents**.** Crème d'orge).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Féculents**.** Pâtes).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Féculents**.** Biscottes sans gluten).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Mat_grass**.** Margarine) par (Mat_grass**.** Beurre)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Mat_grass**.** Mayonnaise) par (Mat_grass**.** Crème fraîche)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Mat_grass**.** Mayonnaise du commerce) par (Mat_grass**.** Mayonnaise de maison)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Sucreries**.** Pâtisserie) par (Sucreries**.** Pâtisserie maison sans farine de blé)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Sucreries**.** Biscuits) par (Sucreries**.** Meringue)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Pain d'épice).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Pâte d'amandes).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Confiserie).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Pâtes de fruits).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Nougat).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Chocolat).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Sucreries**.** Poudres chocolatés).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Sucreries**.** Miel).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Sucreries**.** Glaces de maison).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Sucreries**.** Crème chantilly maison).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Sucreries**.** Gâteau de riz).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**ubstituer [(Divers**.** Epices en poudre) par (Divers**.** Epices en grain)].

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Boissons**.** Alcools).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Divers**.** Moutardes).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Divers**.** Levure chimique).

***S**i (Maladie = " Intolérance au gluten") **A**lors **S**upprimer (Divers**.** Poivre moulu).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Divers**.** Mayonnaise).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Divers**.** Champignons).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Divers**.** Tomate).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Divers**.** Viande rôtie).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Divers**.** Fines herbes).

***S**i (Maladie = " Intolérance au gluten") **A**lors **A**jouter (Divers**.** Persil).

-Le coût d'adaptabilité (Intolérance au gluten) =63.

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Prod_lait**.** Fromages fermentés) par (Prod_lait**.** Fromages fondus)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Prod_lait**.** Lait fermenté) par (Prod_lait**.** Lait écrémé)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Prod_lait**.** Crème chantilly) par (Prod_lait**.** Yaourts)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Viande**.** Viande cuisiné en sauce) par (Viande**.** Viandes de boucherie Jambon)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Viande**.** Viande en conserve) par (Viande**.** Viandes de boucherie Jambon)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Viande**.** Charcuterie) par (Viande**.** Viandes de boucherie Jambon)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Poissons**.** Poissons fumés) par (Poissons**.** Poissons frais)].

***S**i (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Poissons**.** Poissons Salé) par (Poissons**.** Poissons frais)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Poissons**.** Poissons en conserve) par (Poissons**.** Poissons surgelés)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Poissons**.** Poissons cuisinés en sauce) par (Poissons**.** Poissons coquillage)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Poissons**.** Fritures) par (Poissons**.** Crustacés)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Poissons**.** Œufs de poissons).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Poissons**.** Poissons panés).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Viandes**.** Œufs en sauce).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Viande**.** Mayonnaise).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Viande**.** Œufs frits).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Sucreries**.** Chocolat) par (Sucreries**.** Bonbon acidulés)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Sucreries**.** Pâte d'amande) par (Sucreries**.** Pâte de fruits)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Sucreries**.** Nougat).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Sucreries**.** Crème de marron).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Légumes**.** Légumes secs) par (Légumes**.** Légumes cuits)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Légumes**.** Purée de pomme de terre) par (Légumes**.** Potage)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Légumes**.** Lentilles).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Légumes**.** Choux).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Légumes**.** Ail).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Légumes**.** Oignon).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Légumes**.** Poivrons).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Légumes**.** Lentilles).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Fruits**.** Amandes) par (Fruits**.** Fruit cuit)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Fruits**.** Figue) par
(Fruits**.** Fruit en compote)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Fruits**.** Dattes) par
(Fruits**.** Fruit en sirop)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Fruits**.** Melon).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Fruits**.** Pruneaux).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Fruits**.** Cerises).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Féculents**.** Céréales complètes).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Féculents**.** Pâtes feuilletées).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Féculents**.** Macaron).

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**ubstituer [(Féculents**.** Pain frais) par
(Féculents**.** Pain grillé)].

***Si** (Maladie = " Ballonnements intestinaux") **A**lors **S**upprimer (Divers**.** Pâtisseries à la crème).

*Si (Maladie = " Ballonnements intestinaux") Alors Substituer [(Boissons. Jus) par (Boissons. Sirop)].
*Si (Maladie = " Ballonnements intestinaux") Alors Substituer [(Boissons. Alcool) par (Boissons. Eau)].
*Si (Maladie = " Ballonnements intestinaux") Alors Substituer [(Boissons. Gazeuse) par (Boissons. Bouillon d'agrumes)].
*Si (Maladie = " Ballonnements intestinaux") Alors Substituer [(Mat_grass. Cuites) par (Mat_grass. Crues)].
*Si (Maladie = " Ballonnements intestinaux") Alors Supprimer (Mat_grass. Sauces).
*Si (Maladie = " Ballonnements intestinaux") Alors Supprimer (Mat_grass. Saindoux).
*Si (Maladie = " Ballonnements intestinaux") Alors Supprimer (Divers. Epices).
*Si (Maladie = " Ballonnements intestinaux") Alors Supprimer (Divers. Plats cuisinés du commerce).
*Si (Maladie = " Ballonnements intestinaux") Alors Supprimer (Divers. Olives).
*Si (Maladie = " Ballonnements intestinaux") Alors Supprimer (Divers. Câpres).
*Si (Maladie = " Ballonnements intestinaux") Alors Ajouter (Divers. Sel).
*Si (Maladie = " Ballonnements intestinaux") Alors Ajouter (Divers. Aromates).
*Si (Maladie = " Ballonnements intestinaux") Alors Ajouter (Divers. Fines herbes).
*Si (Maladie = " Ballonnements intestinaux") Alors Ajouter (Divers. Vanille).
*Si (Maladie = " Ballonnements intestinaux") Alors Ajouter (Divers. Fleur d'oranger).

-Le coût d'adaptabilité (Ballonnements intestinaux) =55.

*Si (IMC>29) Alors Substituer [(Prod_lait. Lait gélifié) par (Prod_lait. Lait écrémé)].
*Si (IMC>29) Alors Substituer [(Prod_lait. Lait concentré sucré) par (Prod_lait. Yaourt nature)].
*Si (IMC>29) Alors Substituer [(Prod_lait. Lait Yaourt au lait entier) par (Prod_lait. Lait $1/2$ écrémé)].
*Si (IMC>29) Alors Substituer [(Viande. Entrecôtes) par (Viande. Bœuf)].
*Si (IMC>29) Alors Substituer [(Viande. côtelettes) par (Viande. Jambon dégraissé)].
*Si (IMC>29) Alors Substituer [(Viande. Viande cuisinée) par (Viande. Volaille sans peau)].
*Si (IMC>29) Alors Substituer [(Viande. Charcuterie) par (Viande. Dinde)].
*Si (IMC>29) Alors Substituer [(Poisson. Poisson cuisiné) par (Poisson. Poisson surgelé)].
*Si (IMC>29) Alors Substituer [(Poisson. Poisson cuisiné) par (Poisson. Crustacés)].
*Si (IMC>29) Alors Substituer [(Poisson. Poisson cuisiné) par (Poisson. Fruits de mer)].
*Si (IMC>29) Alors Substituer [(Poisson. Poisson cuisiné) par (Poisson. Conserve de poisson)].
*Si (IMC>29) Alors Substituer [Féculent. Pain au chocolat) par (Féculent. Pain blanc)].
*Si (IMC>29) Alors Substituer [Féculent. Brioche) par (Féculent. Pain complet)].
*Si (IMC>29) Alors Substituer [Féculent. Chausson au pommes) par (Féculent. Biscottes)].
*Si (IMC>29) Alors Substituer [Féculent. Pâtisserie) par (Féculent. Pain grillé)].
*Si (IMC>29) Alors Substituer [Féculent. Biscuit sucré) par (Féculent. Semoules)].
*Si (IMC>29) Alors Substituer [Féculent. Farine pour petit déjeuner) par (Féculent. Céréales)].
*Si (IMC>29) Alors Substituer [Féculent. Biscuit salé) par (Féculent. Légumes secs)].

-Le coût d'adaptabilité (IMC) =18.