Extraction de connaissances d'adaptation en raisonnement à partir de cas

Fadi Badra

Département de formation doctorale en informatique **École doctorale IAEM Lorraine**
UFR STMIA

Extraction de connaissances d'adaptation en raisonnement à partir de cas

THÈSE

présentée et soutenue publiquement le 20 novembre 2009

pour l'obtention du

Doctorat de l'université Henri Poincaré – Nancy 1

(spécialité informatique)

par

Fadi Badra

Composition du jury

Rapporteurs :	Maguelonne Teisseire	Directeur de recherche Cemagref Montpellier
	Marie-Christine Jaulent	Directeur de recherche INSERM Paris
Examinateurs :	Alain Mille	Professeur Université Claude Bernard Lyon 1
	Bernard Girau	Professeur Université Henri Poincaré Nancy 1
Directeurs :	Jean Lieber	Maître de Conférences, Université Henri Poincaré Nancy 1
	Amedeo Napoli	Directeur de recherche CNRS Nancy

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

Table des matières

Table des figures

Liste des tableaux

Introduction

Confiture d'abricots secs et de pommes

Préparation : 20 mn - Cuisson : 1 h - Trempage : 24 h

250 g d'abricots secs / 750 g de pommes préparées / 2 kg de sucre cristallisé / 2 cuil. à soupe de noisettes hachées

Faire tremper les abricots dans 50 cl d'eau pendant 24 h. Couper les pommes en dés. Verser l'eau de macération et le sucre dans la bassine à confitures. Porter à ébullition sur feu modéré. Ajouter les abricots et les pommes. Laisser cuire pendant 1 heure à partir de la reprise de l'ébullition, en remuant avec une cuillère en bois. Ajouter les noisettes au contenu dans la bassine. Retirer la bassine du feu. Mettre la confiture en pots. Couvrir ceux-ci à chaud.

La **confiture d'abricots secs et d'ananas** se fait de la même manière. Il suffit de remplacer les pommes par de l'ananas, dans les mêmes proportions. De plus, supprimer les noisettes.

FIG. 1 – La recette de la confiture d'abricots secs et de pommes, d'après le livre [Perrier-Robert, 2004], p. 29.

LE livre [Perrier-Robert, 2004] propose un ensemble de recettes de confitures. La fiche recette reproduite dans la figure 1 est extraite de ce livre.

Le raisonnement à partir de cas

La fiche recette donnée en figure 1 peut être décomposée en deux parties A et B distinctes (figure 2). La partie A de la fiche décrit la recette de la confiture d'abricots secs et de pommes, puis dans la partie B, l'auteur précise que la recette de la confiture d'abricots secs et d'ananas peut être obtenue à partir de la recette de confiture d'abricots secs et de pommes en remplaçant les pommes par de l'ananas, dans les mêmes proportions, et en supprimant les noisettes. De cette observation on peut conclure que si l'on dispose (1) de la recette de confiture d'abricots

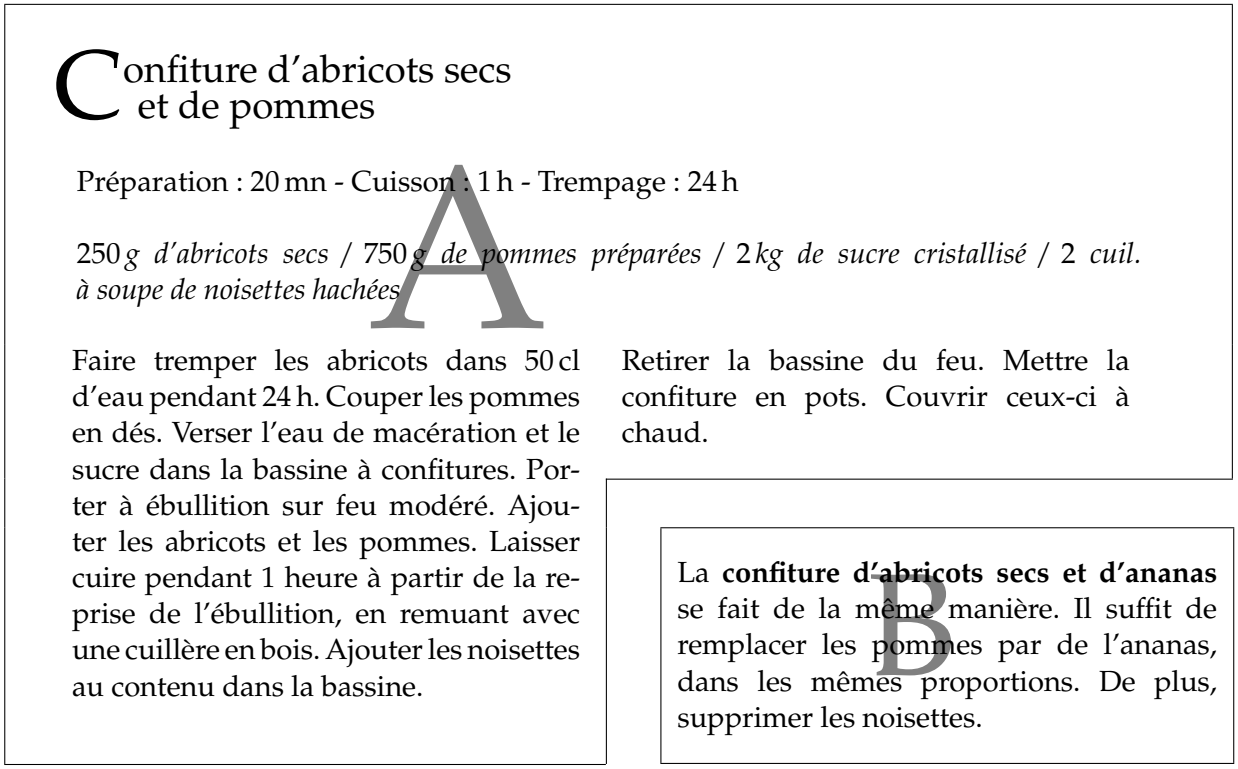


FIG. 2 – Décomposition en deux parties A et B de la fiche recette issue du livre [Perrier-Robert, 2004], p. 29.

secs et de pommes donnée dans la partie A de la figure 2 et (2) de la connaissance donnée dans la partie B de la figure 2, alors il est possible de résoudre le problème suivant, exprimé en langue naturelle :

Problème : « Je veux une recette de confiture d'abricots secs et d'ananas. »

en la solution suivante :

Solution : « Suivre la recette donnée dans la partie A de la figure 2, mais en utilisant de l'ananas au lieu des pommes, dans les mêmes proportions, et ne pas mettre de noisettes. »

Ce raisonnement qui consiste, face à un problème que l'on ne sait pas résoudre, à remémorer un problème déjà résolu et à en adapter la solution, est un raisonnement à partir de cas.

Pour mener à bien un tel raisonnement, le système a besoin de disposer de connaissances comme celles données par l'auteur du livre de recettes qui indiquent comment des recettes existantes peuvent être adaptées pour produire d'autres recettes. On appelle ce genre de connaissances des *connaissances d'adaptation*. La connaissance d'adaptation donnée dans la partie B de la figure 2 est très spécifique puisqu'elle ne peut être utilisée que pour adapter une seule recette. Mais si le système possède la connaissance plus générale selon laquelle l'ananas s'associe mal

avec la noisette, ou que dans les recettes de confitures, la proportion pomme/sucre et ananas/-sucre est la même, il pourra l'utiliser pour adapter d'autres recettes de confiture et résoudre plus de problèmes. Ces connaissances d'adaptation doivent donc être acquises.

L'acquisition de connaissances d'adaptation

Différents types d'approches ont été proposées pour acquérir des connaissances d'adaptation, qui peuvent être distinguées suivant la méthode d'acquisition utilisée et le moment où l'acquisition est effectuée :

- Les approches d'acquisition manuelle auprès d'experts s'effectuent hors ligne et directement auprès d'un expert du domaine. L'expert est invité à formuler les connaissances qu'il utilise pour résoudre les problèmes et est assisté d'un ingénieur de la connaissance pour formaliser les connaissances acquises de manière à ce qu'elles soient utilisables par un système informatique.
- Les approches automatiques consistent à appliquer un algorithme d'apprentissage automatique pour obtenir des connaissances d'adaptation à partir d'un ensemble de données ou de connaissances. La validation des connaissances acquises se fait alors quantitativement en mesurant les gains en performance du système. Lorsqu'il est déclenché en ligne, l'algorithme d'apprentissage a pour but de générer à la volée les connaissances d'adaptation qui seront utilisées dans le raisonnement.
- Les approches semi-automatiques auprès d'experts utilisent des techniques d'extraction de connaissances à partir de données pour générer des éléments de connaissances à partir de diverses sources de données. L'apprentissage est réalisé hors ligne. Les éléments de connaissances extraits sont présentés à un expert du domaine pour validation.
- Les approches interactives auprès d'utilisateurs sont réalisées en ligne. Dans ces approches, le système acquiert des connaissances de ses interactions avec l'utilisateur.

Nous proposons dans cette thèse des méthodes pour acquérir des connaissances d'adaptation par des techniques d'extraction de connaissances. Un processus d'extraction de connaissances est proposé. Il s'appuie sur une représentation des variations entre cas pour apprendre des connaissances d'adaptation à partir de la base de cas. L'étude aboutit à la proposition d'un nouveau type d'approche pour l'acquisition de connaissances d'adaptation dans lequel le processus d'extraction de connaissances est déclenché de manière opportuniste et interactive au cours d'une session particulière de résolution de problèmes. L'apprentissage a pour but d'acquérir les connaissances nécessaires à la résolution d'un problème donné.

Plan du mémoire

Le chapitre 1 présente le raisonnement à partir de cas et identifie les différentes approches proposées dans la littérature pour l'étape d'adaptation. Nous motivons alors l'obtention de connaissances d'adaptation pour un système de RàPC et dressons un état de l'art des différentes approches rencontrées dans la littérature pour acquérir des connaissances d'adaptation.

Le chapitre 2 présente Taaable, une application qui utilise un système de raisonnement à partir de cas dans le domaine culinaire pour proposer à ses utilisateurs des recettes de cuisine. Taaable sera le contexte applicatif de l'étude et servira d'illustration des différents concepts introduits dans la thèse.

Au chapitre 3 est défini un langage générique de représentation des variations entre cas. Le langage proposé est d'abord appliqué à un formalisme de représentation des cas de type

attributs-contraintes, qui généralise le formalisme attribut-valeur fréquemment utilisé en raisonnement à partir de cas. Puis nous montrons comment ce langage peut être utilisé pour représenter les variations entre cas dans l'application TAAABLE.

Au chapitre 4, nous montrons que l'introduction d'un langage de représentation des variations entre cas permet d'aboutir à une nouvelle modélisation de l'étape d'adaptation en raisonnement à partir de cas. Nous montrons que se donner un tel langage permet de représenter dans un même formalisme les similarités et dissimilarités entre cas et les connaissances d'adaptation utilisées dans le raisonnement pour les résoudre. Dans cette modélisation, tout élément du langage de représentation des variations entre cas peut s'interpréter comme une règle d'adaptation.

Au chapitre 5 est présenté un processus d'extraction de connaissances appelé CABAMAKA, qui permet d'acquérir des connaissances d'adaptation de manière semi-automatique à partir de la base de cas. L'apprentissage se fait par généralisation à partir d'une représentation des variations entre cas. Nous montrons comment ce processus d'extraction de connaissances peut être appliqué dans le contexte de l'application TAAABLE pour extraire des opérateurs de substitution d'ingrédients à partir de l'analyse de la base de recettes. Puis nous présentons une application de CABAMAKA dans le domaine de la cancérologie, dans laquelle le processus d'extraction de connaissances est utilisé pour fouiller des référentiels de traitement du cancer dans une perspective d'acquisition de connaissances d'adaptation semi-automatique auprès d'experts.

S'ensuit alors au chapitre 6 une discussion portant sur la place que doit avoir le processus d'extraction de connaissances dans la tâche d'acquisition de connaissances d'adaptation. Nous mettons en évidence l'apport des techniques d'extraction de connaissances pour acquérir des connaissances d'adaptation puis montrons les limites d'un déclenchement du processus d'extraction de connaissances hors-ligne, c'est-à-dire en dehors de tout contexte de résolution de problèmes. Nous montrons en particulier les difficultés rencontrées pour valider les éléments de connaissances produits par le processus d'extraction de connaissances. Nous nous tournons alors vers les approches interactives et plaidons pour un déclenchement opportuniste du processus d'extraction de connaissances au cours d'une session de résolution de problèmes.

Nous étudions alors au chapitre 7 comment le processus d'extraction de connaissances peut être déclenché de manière opportuniste et interactive au cours d'une session particulière de résolution de problèmes. Puis l'approche proposée pour l'extraction opportuniste de connaissances d'adaptation est appliquée pour acquérir des connaissances d'adaptation dans TAAABLE.

Cette thèse a également donné lieu à des travaux dans le domaine du Web Sémantique. Ces travaux ont été effectués dans la continuité du travail de thèse de Mathieu d'Aquin, dont l'objet était d'appliquer des technologies relatives au Web Sémantique pour la construction d'un portail dédié à la gestion des connaissances en cancérologie [d'Aquin, 2005]. Au cours de la thèse, le portail sémantique a été entièrement réécrit pour tenir compte des avancées technologiques et gagner en généricité. Une large part de ce travail a été consacré à la réécriture du serveur de connaissances du portail sémantique ainsi qu'au développement d'EDHIBOU, une interface graphique générée automatiquement à partir d'une ontologie qui permet d'éditer des connaissances à l'aide de formulaires Web dynamiques. EDHIBOU s'intègre dans le portail sémantique et implémente une approche novatrice de génération automatique d'une interface graphique à partir d'une ontologie. EDHIBOU est présenté dans l'annexe A.

1

Le raisonnement à partir de cas

Ce chapitre présente le raisonnement à partir de cas et donne une large part à l'étude de l'étape d'adaptation. Nous présentons une analyse des différentes approches proposées dans la littérature pour effectuer l'adaptation puis détaillons le déroulement de l'étape d'adaptation dans trois systèmes de raisonnement à partir de cas. Nous motivons alors l'obtention de connaissances d'adaptation pour un système de raisonnement à partir de cas et présentons un état de l'art des principales approches rencontrées dans la littérature pour leur acquisition. L'étude met en évidence six différents types d'approches rencontrées pour acquérir des connaissances d'adaptation, qui se différencient par la méthode utilisée pour l'acquisition (manuelle, automatique ou semi-automatique) et par le moment où cette acquisition est réalisée (en ligne ou hors ligne), ainsi que quatre principales sources de connaissances d'adaptation (les connaissances du domaine, la base de cas, l'opérateur humain et le Web).

Dans ses travaux fondateurs sur le raisonnement à partir de cas, Roger Schank met en avant dans [Riesbeck et Schank, 1989] le rôle dans la résolution de problèmes de la mémoire et de la capacité à expliquer les situations rencontrées. L'idée générale est que la capacité d'un humain à agir en présence d'une situation inédite est liée à sa capacité à l'expliquer et ce processus d'explication met en jeu ses connaissances et sa mémoire. Face à un problème inédit à résoudre, des connaissances sont tirées des expériences passées et sont utilisées pour interpréter ce problème et diriger la recherche d'une méthode de résolution. Le raisonnement doit donc être guidé par l'expérience.

La recherche sur le raisonnement à partir de cas a vu l'émergence de plusieurs sous-disciplines qui se distinguent par les caractéristiques des systèmes mis au point.

Lorsque l'utilisateur est fortement impliqué dans le raisonnement, le raisonnement à partir de cas est qualifié d'*interactif* [Aha et Muñoz-Avila, 2001]. C'est le cas de nombreuses approches de raisonnement à partir de cas *conversationnel*, dans lesquelles les cas représentent des extraits de conversations et sont classiquement constitués d'un descriptif textuel, d'un ensemble de couples question-réponse et d'un ensemble d'actions [Aha *et al.*, 2001].

Les systèmes de raisonnement à partir de cas pour la recommandation de produits, ou *systèmes de recommandation*, ont pour but de suggérer un ensemble de produits à un utilisateur à partir d'une évaluation de ses préférences. La recherche d'un produit s'apparente à une requête dans une base de données, mais utilise des techniques d'analyse de données (typiquement l'algorithme des k-plus proches voisins) pour assurer que l'ensemble des réponses soit non-vide [McSherry, 2006]. Les systèmes de recommandation utilisent une représentation des cas assez fruste (typiquement attribut-valeur), peu de connaissances supplémentaires mais opèrent généralement sur des bases de cas assez volumineuses. Par exemple, dans [Baccigalupo et Plaza, 2006], un système de recommandation est proposé qui suggère à un utilisateur un ensemble de listes de lecture musicales à partir du titre d'une chanson et d'une spécification de la longueur de la liste. La recommandation est déterminée en exploitant une base de données contenant 300 000 listes de lectures.

Les systèmes de raisonnement à partir de cas *textuels* se caractérisent quant à eux par l'utilisation d'expériences dont la description est contenue dans des documents textuels non-structurés ou semi-structurés [Lamontagne et Lapalme, 2002]. Ces systèmes mettent en œuvre des approches qui s'appuient sur des techniques de recherche d'information, d'apprentissage automatique ou de traitement automatique des langues.

1.1 Définitions

Raisonner à partir de cas consiste à résoudre un nouveau problème, appelé *problème cible*, en utilisant un ensemble de problèmes déjà résolus [Riesbeck et Schank, 1989, Kolodner, 1993]. Un *cas source* désigne un épisode passé de résolution de problèmes et une *base de cas* un ensemble de cas sources. Un cas source $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ est un couple composé d'un problème source $\texttt{srce}$ accompagné de sa solution $\texttt{Sol}(\texttt{srce})$.

Dans la suite, on suppose que les problèmes sont décrits dans un langage $\mathcal{L}_{\texttt{pb}}$ et que les solutions sont décrites dans un langage $\mathcal{L}_{\texttt{sol}}$. $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ et $(\texttt{cible}, \texttt{Sol}(\texttt{cible}))$ sont deux cas sources tels que $\texttt{srce} \in \mathcal{L}_{\texttt{pb}}$, $\texttt{Sol}(\texttt{srce}) \in \mathcal{L}_{\texttt{sol}}$, $\texttt{cible} \in \mathcal{L}_{\texttt{pb}}$ et $\texttt{Sol}(\texttt{cible}) \in \mathcal{L}_{\texttt{sol}}$.

Le processus de raisonnement à partir de cas est classiquement composé de trois opérations principales : la remémoration, l'adaptation et l'apprentissage. L'étape de *remémoration* consiste à sélectionner un cas $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ de la base de cas. Le but de l'étape d'*adaptation* est de modifier $\texttt{Sol}(\texttt{srce})$ de façon à construire une solution $\texttt{Sol}(\texttt{cible})$ de $\texttt{cible}$. Un *problème*

d'adaptation est alors un triplet $(\texttt{srce}, \texttt{Sol(srce)}, \texttt{cible}) \in \mathcal{L}_{\texttt{pb}} \times \mathcal{L}_{\texttt{sol}} \times \mathcal{L}_{\texttt{pb}}$ et se résoud en une solution `Sol(cible)`. L'étape d'*apprentissage* consiste à mettre à jour les connaissances du système à l'issue du raisonnement. Elle est habituellement réalisée en mémorisant dans la base de cas le cas (`cible`, `Sol(cible)`) (une fois la solution `Sol(cible)` validée). Une *solution candidate* à la résolution de `cible` sera notée $\widetilde{\texttt{Sol}}(\texttt{cible})$. Une solution candidate est une solution proposée par le système pour résoudre `cible` mais qui n'a pas encore été validée par l'utilisateur.

Le raisonnement à partir de cas est une forme de raisonnement par analogie. Pour cette raison, un problème d'adaptation et sa solution sont souvent représentés par un carré d'analogie. Le carré d'analogie formé pour la résolution du problème donné en introduction est donné en figure 1.1.

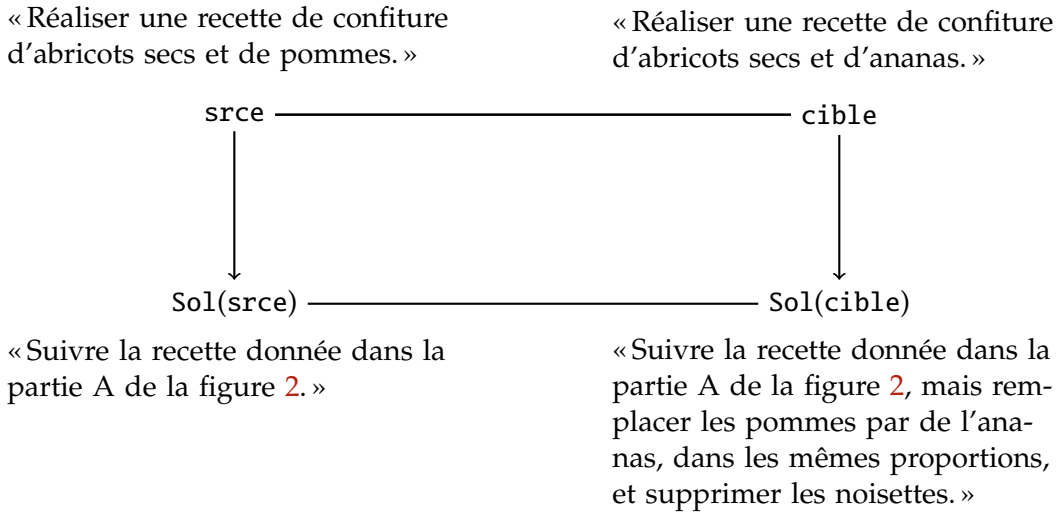


FIG. 1.1 – Le carré d'analogie formé pour les deux recettes de confiture.

Dans cette thèse, nous nous intéresserons particulièrement à l'étape d'adaptation en raisonnement à partir de cas.

1.2 L'adaptation en raisonnement à partir de cas

Cette section présente différentes approches qui ont été proposées dans la littérature pour réaliser l'étape d'adaptation en raisonnement à partir de cas. Pour plus de clarté, chacune des approches sera présentée dans un tableau de la forme suivante :

Nom de l'approche.

Description	Description de l'approche.
Références	Références bibliographiques et principaux systèmes mettant en œuvre l'approche.
Connaissances d'adaptation	Type de connaissances d'adaptation utilisées.

La section 1.2.1 présente un ensemble de stratégies générales pour l'adaptation (premier niveau de la figure 1.2), que l'on retrouve dans de nombreuses approches. La section 1.2.2 décrit alors trois approches particulières d'adaptation par décomposition, puis la section 1.2.3 décrit quatre approches d'adaptation par satisfaction de contraintes.

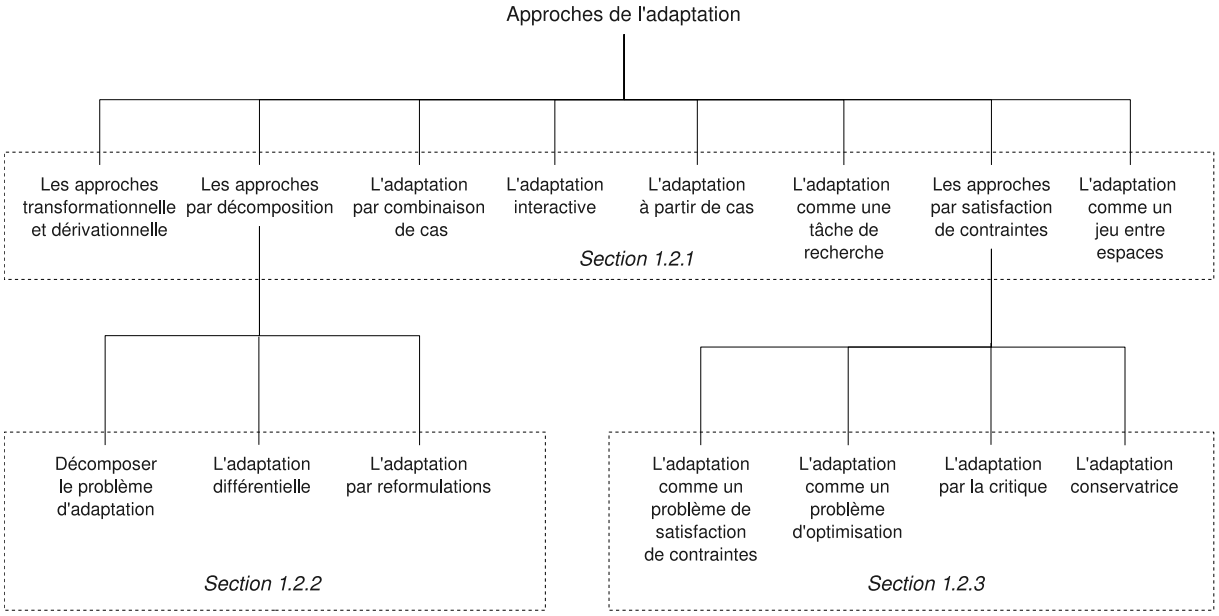


FIG. 1.2 – Une taxonomie de différentes approches de l'adaptation.

1.2.1 Les stratégies générales de l'adaptation

Les approches transformationnelle et dérivationnelle de l'adaptation

Description — Parmi les approches qui ont été proposées pour modéliser l'adaptation, on peut distinguer deux grands types d'approches : l'approche « par transformation » et l'approche « par dérivation ». L'approche par transformation de l'adaptation consiste à déterminer une modification à appliquer à la solution du (ou des) cas source(s) remémoré(s). L'approche par dérivation de l'adaptation consiste à adapter la méthode utilisée pour obtenir cette solution.

Références — [Carbonell, 1983a] [Carbonell, 1983b]

Les approches par décomposition

Description — L'adaptation est elle-même une tâche de résolution de problèmes. Une stratégie générale de résolution de problèmes consiste à décomposer un problème en sous-problèmes plus simples. L'idée des approches de l'adaptation par décomposition est donc de décomposer la tâche d'adaptation en sous-tâches plus simples. Différentes approches d'adaptation par décomposition sont présentées en section 1.2.2.

L'adaptation par combinaison de cas

Description — L'adaptation par combinaison de cas est une forme d'adaptation dans laquelle plusieurs cas sources sont remémorés pour un problème d'adaptation donné puis une solution est composée à partir des solutions `Sol(srce)` des différents cas sources remémorés.

Références — CELIA [Redmond, 1990]
IDIOM [Smith *et al.*, 1995]
COMPOSER [Purvis et Pu, 1998]
CREEK [Aamodt, 2004]

L'adaptation interactive

Description — L'adaptation interactive consiste à impliquer l'utilisateur dans l'étape d'adaptation. L'intervention de l'utilisateur peut être utilisée pour :
- lui demander d'effectuer l'adaptation manuellement (comme dans le système DIAL),
- affiner un modèle des préférences de l'utilisateur qui est utilisé dans l'adaptation (comme dans le système WebAdapt), ou encore
- affiner, dans un processus itératif, une solution proposée par le système (comme dans les systèmes IDIOM, FrakaS et IakA). Lorsqu'elle est couplée à une acquisition de connaissances, comme dans les systèmes FrakaS et IakA, l'interaction avec l'utilisateur est utilisée pour incorporer dans le système de raisonnement à partir de cas de nouvelles connaissances qui sont ensuite utilisées pour regénérer une solution corrigée.

Références — IDIOM [Smith *et al.*, 1995]
DIAL [Leake *et al.*, 1996, Leake *et al.*, 1997, Kinley, 2001]
WebAdapt [Leake et Powell, 2007, Leake et Powell, 2008]
FrakaS et IakA [Cordier, 2008]

L'adaptation à partir de cas

Description	L'adaptation à partir de cas consiste à appliquer un processus de raisonnement à partir de cas à la tâche d'adaptation elle-même. L'adaptation est alors réalisée en raisonnant sur un ensemble d'expériences passées d'adaptation.
Références	DIAL [Leake *et al.*, 1996, Leake *et al.*, 1997, Kinley, 2001] [Craw *et al.*, 2001] [Wiratunga *et al.*, 2002] [Craw *et al.*, 2006]
Connaissances d'adaptation	Les connaissances d'adaptation sont des expériences passées d'adaptation. La trace gardée d'une adaptation est appelée *cas d'adaptation*.

L'adaptation comme une tâche de recherche

Description

L'adaptation peut également être formulée comme une tâche de recherche dans l'espace des solutions `Solutions`, où l'état initial est la solution `Sol(srce)` d'un cas source remémoré et l'état final une solution `Sol(cible)` pour le problème cible. Cette recherche s'effectue par l'application d'*opérateurs d'adaptation*, qui sont des transformations effectués dans l'espace des solutions. Plusieurs types d'opérateurs d'adaptation sont utilisés dans la littérature pour modifier `Sol(srce)` :

- des opérateurs *de copie*, qui ne réalisent aucune transformation mais se contentent de recopier `Sol(srce)`,
- des opérateurs *d'ajustement*, qui modifient certaines valeurs de paramètres entrant en jeu dans `Sol(srce)`,
- des opérateurs *de substitution*, qui modifient `Sol(srce)` en ajoutant, supprimant ou substituant certains de ses constituants,
- des opérateurs *de transformation*, qui opèrent une modification structurelle sur la solution `Sol(srce)` (en changeant l'ordre des composants par exemple),
- des opérateurs *de généralisation-spécialisation*, qui exploitent une structuration hiérarchique de la base de cas pour généraliser tout ou partie de `Sol(srce)` puis respécialiser en une solution pour le problème cible,
- des *reformulations*, qui représentent un ensemble de liens entre l'espace des problèmes et l'espace des solutions (voir page 13 pour une présentation de l'approche d'adaptation par reformulations).

Certains systèmes comme JUDGE [Bain, 1896] utilisent un petit ensemble d'opérateurs prédéfinis. Mais dans beaucoup de systèmes, une recherche en mémoire est effectuée pour déterminer les informations nécessaires à l'instanciation d'un type opérateur particulier (voir [Kolodner, 1993, chap 11] pour une description de différentes méthodes de recherche en mémoire). Par exemple, dans le système de planification PLEXUS [Alterman, 1986], une recherche par abstraction/spécialisation est effectuée localement dans une hiérarchie d'abstractions pour trouver une substitution à une étape d'un plan remémoré lorsque cette étape est inapplicable dans le contexte du problème cible.

Cette recherche permet à PLEXUS d'adapter un plan utilisé dans le métro de San Francisco afin de prendre celui de New York. Dans le plan adapté, l'étape d'achat d'un ticket à la machine est substituée par un achat de ticket au guichet de la gare car aucune machine n'est disponible à New York et ces deux actions sont des sous-concepts du concept « accéder aux voies » dans la hiérarchie. Une autre façon de déterminer les informations nécessaires à l'instanciation d'un opérateur de substitution est de chercher ces informations dans des cas similaires de la base de cas. On parle alors de *substitution à partir de cas*. Par exemple, dans le domaine culinaire, JULIA [Hinrichs, 1992, Kolodner, 1993] détermine un substitut végétarien pour des lasagnes dans un plat de pâtes en recherchant des recettes de pâtes similaires dans la base de cas. Le système trouve un cas similaire contenant du manicotti et propose de substituer les lasagnes par du manicotti dans la recette. Plus récemment, le système WebAdapt [Leake et Powell, 2007] effectue également des substitutions à partir de cas en recherchant sur le Web les informations nécessaires à l'instanciation d'un opérateur de substitution.

Références

Une étude réalisée en 1995 et présentée d'abord dans [Hanney *et al.*, 1995] puis dans [Hanney, 1997] recense les types d'opérateurs d'adaptation utilisées par 42 systèmes de raisonnement à partir de cas. Cette étude montre que les opérateurs autres que l'opérateur de copie sont utilisés principalement lorsque le système effectue des tâches de configuration, de planification ou de prédiction. Néanmoins, dans les tâches de prédiction, seuls les opérateurs de substitution et de transformation sont utilisés. Quelques types d'opérateurs de copie sont présentés dans [Lieber, 2002a].

Connaissances d'adaptation

Les connaissances d'adaptation sont données par un ensemble d'opérateurs d'adaptation. La mémoire du système et les connaissances du domaine peuvent être considérées comme des sources de connaissances d'adaptation lorsqu'une recherche est effectuée en mémoire dans le but de trouver les informations nécessaires à l'instanciation d'un opérateur donné.

Les approches par satisfaction de contraintes

Description

L'adaptation peut aussi être vue comme la recherche dans un espace des solutions `Solutions` d'une solution `Sol(cible)` qui satisfasse un certain nombre de contraintes. Différentes approches d'adaptation par satisfaction de contraintes sont présentées en section 1.2.3.

L'adaptation comme un jeu entre espaces

Description	L'adaptation peut être formulée comme un jeu entre un espace des remémorations et un espace des adaptations. L'*espace des remémorations* désigne un ensemble de correspondances entre le problème `cible` et les problèmes sources (`srce`, `Sol(srce)`) de la base de cas, et l'*espace des adaptations* désigne un ensemble d'opérateurs qui peuvent être utilisés pour modifier la solution `Sol(srce)` du problème source remémoré de façon à produire une solution `Sol(cible)` pour `cible`.
Références	Déjà Vu [Smyth et Keane, 1993] [Melis *et al.*, 1998] [Lieber et Napoli, 1998] [Lieber, 1999]
Connaissances d'adaptation	Les connaissances d'adaptation sont définies comme des connaissances permettant de lier l'espace des remémorations à l'espace des adaptations.

1.2.2 Les approches par décomposition

Décomposer le problème d'adaptation

Description	Une façon d'appliquer cette stratégie à l'adaptation consiste à successivement (1) décomposer le problème d'adaptation à résoudre, (2) résoudre indépendamment les différents sous-problèmes d'adaptation, puis (3) combiner les solutions partielles obtenues pour proposer une solution `Sol(cible)` pour `cible`. Cette stratégie n'est pas toujours applicable car elle suppose qu'il y ait peu d'interactions entre les différents éléments entrants en jeu dans la description des problèmes.
Références	Déjà Vu [Smyth *et al.*, 2001] DzCBR [d'Aquin *et al.*, 2005] [Jarmulak *et al.*, 2001], [Wiratunga *et al.*, 2002], [Craw *et al.*, 2006]
Connaissances d'adaptation	Les connaissances qui permettent de décomposer le problème d'adaptation sont elle-mêmes des connaissances d'adaptation. Dans le système DzCBR, la décomposition du problème d'adaptation est donnée à l'avance et correspond aux points de vues de différents spécialistes en cancérologie (experts en chirurgie, hormonothérapie, etc.). Dans le système Déjà Vu, les connaissances d'adaptation qui permettent de décomposer le problème d'adaptation sont contenues dans la base de cas. La décomposition du problème d'adaptation se fait au cours de la remémoration à l'aide d'un ensemble de cas abstraits appelés cas de décomposition (en anglais, *decomposition cases*).

L'adaptation différentielle

Description

L'adaptation différentielle est une forme d'adaptation par décomposition de l'adaptation qui applique à l'adaptation la stratégie du calcul différentiel. En calcul différentiel, les différentielles $dx = (dx_1, \ldots, dx_n)$ et $dy = (dy_1, \ldots, dy_n)$ en un point $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$ sont liées aux dérivées partielles $\frac{\partial y_i}{\partial x_i}$ dans la formule suivante :

$$dy_i = \sum_i \frac{\partial y_i}{\partial x_i} dx_i$$

L'idée de l'adaptation différentielle est de s'inspirer de cette formule pour modéliser localement les variations entre cas à partir d'un ensemble de dépendances entre descripteurs de problèmes et descripteurs de solutions.

Références

[Fuchs *et al.*, 2000] [Fuchs *et al.*, 2006]
IakA-NF [Cordier *et al.*, 2008, Cordier, 2008]

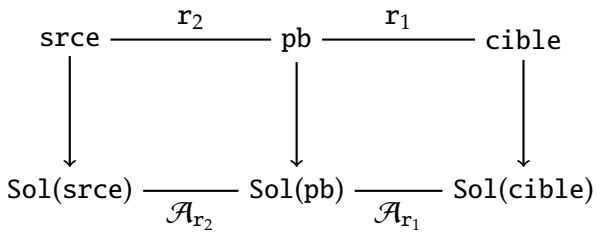
Connaissances d'adaptation

Dans l'adaptation différentielle, les connaissances d'adaptation sont données par un ensemble de dépendances entre descripteurs de problèmes et descripteurs de solutions. Ces dépendances expriment localement des liens entre l'espace des remémorations et l'espace des adaptations (voir l'approche de l'adaptation comme un jeu entre espaces).

L'adaptation par reformulations

Description

L'adaptation par reformulations combine adaptation par décomposition et recherche dans un espace d'états. Une reformulation est un couple $(\mathtt{r}, \mathcal{A}_\mathtt{r})$ dans lequel `r` est une relation binaire entre problèmes et $\mathcal{A}_\mathtt{r}$ est une fonction d'adaptation telle que si `pb r pb'` alors toute solution `Sol(pb)` de `pb` peut être adaptée par $\mathcal{A}_\mathtt{r}$ en une solution `Sol(pb')` de `pb'`. L'application de reformulations permet de décomposer l'adaptation en introduisant des problèmes virtuels comme le problème `pb` dans la figure suivante :



Références

[Melis *et al.*, 1998] [Lieber et Napoli, 1998] [Lieber, 1999]
[Lieber *et al.*, 2001] [Lieber *et al.*, 2008]
RESYN/RàPC [Lieber et Napoli, 1996]
IakA [Cordier *et al.*, 2008, Cordier, 2008]
Taaable [Badra *et al.*, 2008a]

Connaissances d'adaptation
Dans l'approche d'adaptation par reformulations, les connaissances d'adaptation sont données par un ensemble de reformulations $(\mathtt{r}, \mathcal{A}_{\mathtt{r}})$. Une reformulation représente un ensemble de liens entre l'espace des remémorations et l'espace des adaptations (voir l'approche de l'adaptation comme un jeu entre espaces).

1.2.3 Les approches par satisfaction de contraintes

L'adaptation comme un problème de satisfaction de contraintes

Description
L'adaptation peut être abordée comme un problème de satisfaction de contraintes : une solution `Sol(cible)` doit satisfaire un certain nombre de contraintes qui sont exprimées par le problème `cible` lui-même, par le (ou les) cas source(s) (`srce`, `Sol(srce)`) remémoré(s) et par les connaissances du domaine. Un *problème de satisfaction de contraintes* est la donnée d'un ensemble de variables, chacune prenant ses valeurs sur un ensemble fini de valeurs, et d'un ensemble de contraintes qui spécifient les combinaisons autorisées de valeurs prises par ces variables [Bessière *et al.*, 2007]. Une solution d'un problème de satisfaction de contraintes est une affectation des variables qui satisfait les contraintes énoncées. Les contraintes sont généralement modélisées par un ensemble de relations entre des sous-ensembles des ensembles de valeurs que peuvent prendre les variables.

Références
JULIA [Hinrichs, 1992, Kolodner, 1993]
IDIOM [Smith *et al.*, 1995]
COMPOSER [Purvis et Pu, 1998]
[Schmidt *et al.*, 2001][Schmidt et Vorobieva, 2005]

Connaissances d'adaptation
Les connaissances d'adaptation sont données par un ensemble de contraintes que doit satisfaire la solution `Sol(cible)`. Un domaine qui se prête particulièrement bien à l'application de contraintes est le domaine culinaire, où celles-ci peuvent exprimer des préférences sur le contenu d'un plat, des interactions entre ingrédients lors de la préparation du plat, ou encore des contraintes temporelles sur les différentes étapes de préparation du plat. En médecine, ces contraintes sont typiquement introduites pour exprimer des contre-indications qui réduisent le champ d'application de certaines traitements.

L'adaptation comme un problème d'optimisation

Description	L'adaptation peut aussi être vue comme un problème de maximisation d'une fonction Q de qualité de la solution. La fonction $Q : \texttt{Problèmes} \times \texttt{Solutions} \longrightarrow \mathbb{R}$ est une fonction définie sur l'ensemble $\texttt{Problèmes} \times \texttt{Solutions}$, où `Problèmes` est l'ensemble des problèmes et `Solutions` est l'ensemble des solutions. L'adaptation consiste donc à trouver un déplacement dans l'ensemble $\texttt{Problèmes} \times \texttt{Solutions}$, en partant du cas source remémoré (`srce`, `Sol`(`srce`)), et qui maximise la fonction Q.
Références	[Bergmann et Wilke, 1998] [Stahl et Bergmann, 2000]

L'adaptation par la critique

Description	L'adaptation par la critique (en anglais, *critic-based adaptation*) est une approche itérative de l'adaptation dans laquelle le système anticipe les problèmes rencontrés dans la solution proposée et tente de les corriger.
Références	CHEF [Hammond, 1986] Déjà Vu [Smyth et Cunningham, 1992]
Connaissances d'adaptation	Les connaissances d'adaptation sont données par un ensemble de critiques. Une critique identifie un ensemble de problèmes susceptibles d'être rencontés dans la solution proposée et y associe des stratégies de réparation. Par exemple, une critique indique dans CHEF que les crevettes doivent être décortiquées avant d'être utilisées. Lorsqu'une solution (recette) contient des crevettes, CHEF applique une stratégie de réparation qui consiste à rajouter une étape de préparation dans laquelle les crevettes sont décortiquées. Les stratégies d'adaptation utilisées dans le système Déjà Vu peuvent également être considérées comme des critiques dans la mesure où elles résolvent des interactions entre les différents composants de la solution proposée.

L'adaptation conservatrice

Description	L'adaptation conservatrice fait appel à la théorie AGM de la révision pour produire un cas (`cible`, `Sol`(`cible`)) par changement minimal sur le cas source (`srce`, `Sol`(`srce`)) remémoré tout en restant cohérent avec les connaissances du domaine.
Références	[Lieber, 2007] [Cojan et Lieber, 2008] FrakaS [Cordier *et al.*, 2007]

Connaissances d'adaptation	Les connaissances d'adaptation sont données par un opérateur de révision sur des bases de connaissances. Lorsque l'opérateur de révision est défini à partir d'une distance, ce qui est habituellement le cas, l'adaptation conservatrice peut être exprimée comme un problème d'optimisation sous contraintes.

1.2.4 L'étape d'adaptation dans trois systèmes de RàPC

Cette section décrit l'étape d'adaptation de trois systèmes de raisonnement à partir de cas : CHEF, Déjà Vu et DIAL. Ces systèmes ont été choisis car chacun d'eux a été novateur dans sa façon d'aborder l'adaptation. De plus, chacun de ces systèmes combine plusieurs des approches de l'adaptation présentées dans la section précédente. Par ailleurs, les systèmes CHEF et DIAL ont eu une grande influence sur les choix effectués lors de la mise au point du processus d'acquisition de connaissances d'adaptation présenté au chapitre 7. Chaque système sera décrit à l'aide d'un tableau qui prend la forme suivante :

Nom du système	
Domaine d'application	Le domaine d'application du système.
Processus d'adaptation	Le déroulement de l'étape d'adaptation.
Connaissances d'adaptation	Les connaissances d'adaptation utilisées.
Approches de l'adaptation	Liens avec les différentes approches de l'adaptation présentées à la section 1.2.
Références	Références bibliographiques.

CHEF	
Domaine d'application	CHEF est un système de planification à partir de cas dans le domaine de la cuisine du Sichuan. Comme le système Taable présenté au chapitre 2, CHEF a pour but de proposer des recettes qui satisfont un ensemble de contraintes données par l'utilisateur. Dans CHEF, Ces contraintes portent sur les goûts, les textures, les ingrédients et les types de plats. Un exemple de problème `cible` exprimé en langue naturelle est : « Je veux une recette de bœuf aux broccolis ». Une solution `Sol(cible)` à ce problème est un plan décrivant les étapes de préparation d'une recette satisfaisant les contraintes énoncées. Le processus de remémoration de CHEF sélectionne parmi les 20 plans présents dans la base de cas un plan qui satisfait le plus possible des contraintes énoncées dans `cible` et le réinstancie.

Processus d'adaptation	Le processus d'adaptation de CHEF consiste à appliquer des opérateurs de substitution au plan réinstancié pour satisfaire au mieux les contraintes énoncés dans le problème cible puis à anticiper et corriger les éventuels problèmes rencontrés dans la solution proposée. L'adaptation comporte ainsi deux étapes : 1. le système simule une première adaptation dans laquelle un ou plusieurs opérateurs de substitution sont appliqués pour modifier le plan remémoré. Par exemple, un opérateur de substitution est appliqué pour obtenir une recette de bœuf aux brocolis à partir d'une recette de bœuf aux pois. L'application de cet opérateur consiste à ajouter des brocolis à la recette et à enlever les pois. 2. un modèle causal est utilisé à la fois pour détecter et pour expliquer les problèmes susceptibles d'être rencontrés dans la solution proposée puis le système utilise un ensemble de critiques pour résoudre ces problèmes. Dans l'exemple précédent, une étape de la recette obtenue consiste à frire tous les ingrédients. Or les brocolis ne pourront pas être frits convenablement car ils ont été ajoutés entiers. Le système détecte donc un problème dans la recette obtenue. Pour résoudre ce problème, CHEF utilise une critique qui stipule que tout légume doit être coupé en morceaux avant d'être frit. CHEF rajoute donc une étape de préparation dans laquelle les brocolis sont coupés en morceaux.
Connaissances d'adaptation	Trois types de connaissances entrent en jeu dans l'adaptation : – un ensemble d'opérateurs de substitution prédéfinis, – un modèle causal, qui est utilisé pour anticiper des problèmes susceptibles d'être rencontrés dans la solution proposée, et – un ensemble de critiques, qui définissent des stratégies de réparation à appliquer pour résoudre ces problèmes.
Approches de l'adaptation	adaptation dérivationnelle, adaptation par la critique, adaptation comme une tâche de recherche
Références	[Hammond, 1986] [Hammond, 1989]

Déjà Vu

Domaine d'application	Déjà Vu est un planificateur pour le contrôle de déplacements de véhicules autonomes dans des environnements industriels. Un problème de Déjà Vu est une tâche de déplacement de véhicule et une solution est un programme informatique qui spécifie les actions à entreprendre pour effectuer un tel déplacement.
Processus d'adaptation	Déjà Vu introduit deux nouvelles techniques pour l'adaptation : la remémoration guidée par l'adaptabilité (en anglais, *adaptation-guided retrieval*) et le raisonnement à partir de cas hiérarchique (en anglais, *hierarchical case-based reasoning*). La remémoration guidée par l'adaptabilité consiste à utiliser des connaissances d'adaptation pour guider la remémoration. Pour ce faire, la procédure de remémoration dans Déjà Vu inclut une évaluation de l'adaptabilité des différents cas sources de la base de cas. La technique du raisonnement à partir de cas hiérarchique consiste quant à elle à exploiter une structuration hiérarchique de la base de cas pour composer une solution pour le problème cible à partir de parties de solutions prises pour différents cas sources et à différents niveaux d'abstraction. La base de cas de Déjà Vu contient deux types de cas sources : les cas de conception (en anglais, *design cases*) et les cas de décomposition (en anglais, *decomposition cases*). Les cas de conception contiennent des bouts de programmes informatiques qui peuvent être composés pour former la solution finale. Les cas de décomposition sont des opérateurs abstraits, organisés en hiérarchie, et qui regroupent implicitement un ensemble de cas de conception. Par exemple, l'opérateur de collecte et l'opérateur de livraison sont des cas de décomposition qui sont regroupés en un opérateur abstrait « transfert de contenu ».
Connaissances d'adaptation	Déjà Vu utilise deux types de connaissances d'adaptation : les spécialistes d'adaptation (en anglais, *adaptation specialists*) et les stratégies d'adaptation (en anglais, *adaptation strategies*). Les spécialistes d'adaptation sont des opérateurs qui permettent d'effectuer des transformations locales sur une solution. Par exemple, un spécialiste de la vitesse pourra réaliser un ajustement de la vitesse du véhicule d'un cas source remémoré pour correspondre à la vitesse requise dans le problème cible. Les stratégies d'adaptation résolvent les interactions entre les différents spécialistes d'adaptation. Par exemple, un véhicule dont la vitesse est augmentée consomme plus d'énergie, ce qui peut entraîner une panne lors du déplacement donc pour résoudre cette interaction le plan de déplacement pourra inclure une étape de recharge du véhicule.
Approches de l'adaptation	adaptation transformationnelle, décomposition du problème d'adaptation, adaptation par combinaison de cas, adaptation comme une tâche de recherche, adaptation comme un jeu entre espaces
Références	[Smyth et Cunningham, 1992] [Smyth et Keane, 1993] [Smyth, 1996] [Leake, 1996a] [Smyth *et al.*, 2001]

DIAL

Domaine d'application	Le domaine d'application du système DIAL est la planification d'interventions en réponse à des catastrophes naturelles ou d'origine humaine (tremblements de terre, asphyxies accidentelles, etc.). L'étape d'adaptation prend en entrée un plan d'intervention qui a été instancié à l'issue de l'étape de remémoration ainsi que la description de ce qui dans ce plan nécessite une adaptation. Par exemple, en réponse à des difficultés respiratoires manifestées par les occupants d'un immeuble, DIAL instancie un plan d'intervention en environnement industriel qui recommande d'informer les syndicats des victimes mais ce plan n'est pas applicable tel quel lorsque les victimes sont des militaires (car il n'existe pas de syndicats militaires) et doit donc être adapté.
Processus d'adaptation	Pour adapter le plan, DIAL tente d'appliquer successivement : 1. Un méthode d'adaptation à partir de cas, qui consiste à se remémorer un cas d'adaptation appliqué avec succès pour résoudre un problème similaire et à le réappliquer pour adapter le plan d'intervention remémoré. 2. Une méthode d'adaptation à partir de règles : lorsqu'aucun cas d'adaptation n'a pu être appliqué, DIAL tente de générer une règle d'adaptation à appliquer suivant le principe *adaptation = tranformations + exploration de la mémoire*. Ce principe consiste à sélectionner une règle abstraite puis à rechercher dans la mémoire les informations nécessaires à son instanciation. Dans l'exemple précédent, DIAL associe au problème d'adaptation une transformation par substitution puis recherche en mémoire par quoi la recommandation d'informer les syndicats des victimes pourrait être remplacée. Le système propose alors d'appliquer le plan d'intervention remémoré mais d'informer les officiers, qui sont d'autres représentants des victimes. 3. Une méthode d'adaptation manuelle, lorsqu'aucune des méthodes précédentes n'a réussi. Une interface graphique permet alors à l'utilisateur de sélectionner une transformation à appliquer et de naviguer dans la mémoire du système pour déterminer les informations nécessaires à son instanciation.
Connaissances d'adaptation	DIAL utilise deux types de connaissances d'adaptation : des cas d'exploration de la mémoire (en anglais, *memory search cases*) et des cas d'adaptation. Les cas d'exploration de la mémoire contiennent des informations sur les étapes du processus d'exploration de la mémoire. Les cas d'adaptation comprennent la transformation utilisée pour l'adaptation ainsi que des pointeurs vers les cas d'exploration de la mémoire qui ont servi à trouver les informations pour l'instancier.
Approches de l'adaptation	adaptation dérivationnelle, adaptation interactive, adaptation à partir de cas, adaptation comme une tâche de recherche

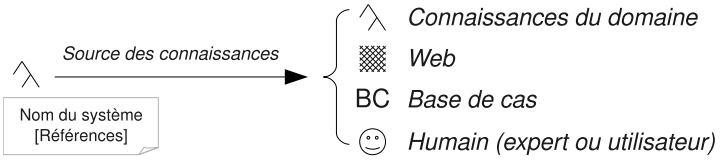
Références	[Leake *et al.*, 1996] [Leake, 1996b] [Kinley, 2001]

L'étape d'adaptation est considérée comme un goulet d'étranglement pour les concepteurs de systèmes de raisonnement à partir de cas car elle nécessite des connaissances spécifiques au domaine d'application qui doivent être acquises. La section suivante dresse un état de l'art des différentes approches d'acquisition de connaissances d'adaptation en raisonnement à partir de cas.

1.3 L'acquisition de connaissances d'adaptation

Peu d'études ont été consacrées à ce jour à l'analyse des différentes approches mises en œuvre pour l'acquisition de connaissances d'adaptation en raisonnnement à partir de cas. Dans [Hanney, 1997], quelques travaux sont recencés qui sont régroupés selon qu'ils exploitent des connaissances du domaine, la base de cas ou l'expert du domaine pour apprendre des connaissances d'adaptation. Une étude différente est réalisée dans [Lieber *et al.*, 2004] où les critères d'évaluation sont les sources de connaissances utilisées, les hypothèses faites sur le formalisme de représentation des cas et les types de connaissances d'adaptation acquises. Cette étude permet de distinguer deux grands types d'approches que sont les approches manuelles et les approches semi-automatiques. Les approches manuelles s'appuient sur l'interaction avec l'expert, ne font pas d'hypothèses sur le formalisme de représentation des cas et permettent en général d'apprendre des règles d'adaptation. Les approches semi-automatiques quant à elles exploitent en général la base de cas, utilisent plutôt un formalisme simple de représentation des cas, de type attribut-valeur, et permettent d'apprendre des règles d'adaptation dont la prémisse contient des informations sur la variation entre les problèmes comparés.

Cette section présente un état de l'art des différentes approches d'acquisition de connaissances d'adaptation (ACA). Ici le terme « acquisition » est utilisé indistinctement pour désigner toute forme d'obtention et de représentation de connaissances d'adaptation, que ce soit auprès d'experts ou d'utilisateurs ou de manière automatique ou semi-automatique à partir de diverses sources de connaissances. Chacune des approches est décrite par le nom du système de raisonnement à partir de cas qui implémente l'approche, une ou plusieurs références bibliographiques et un ensemble de figurines placées au dessus du nom de l'approche précisant les sources de connaissances d'adaptation utilisées : les connaissances du domaine, le Web, la base de cas BC ou l'opérateur humain. Chacune des approches est donc représentée de la façon suivante :



Dans la sous-section 1.3.1, nous distinguons six types d'approches qui se différencient par la méthode utilisée pour l'acquisition (manuelle, automatique ou semi-automatique) et par le moment où cette acquisition est réalisée (en ligne ou hors ligne). Puis nous analysons dans la sous-section 1.3.2 les différentes sources de connaissances utilisées pour acquérir des connaissances d'adaptation. Finalement, la sous-section 1.3.3 présente un résumé des principales approches d'acquisition de connaissances d'adaptation.

1.3.1 Différents types d'approches

Dans cette section, nous distinguons six types d'approches pour l'acquisition de connaissances d'adaptation (numérotées ①, ②, ③, ④, ⑤ et ⑥ sur la figure 1.4) qui se différencient par la méthode utilisée pour l'acquisition (manuelle, automatique ou semi-automatique) et par le moment où cette acquisition est réalisée (en ligne ou hors ligne) :

① Les approches d'**acquisition manuelle auprès d'experts** s'effectuent hors ligne et directement auprès d'un expert du domaine. Des sessions d'acquisition de connaissances sont organisées dans lesquelles un « transfert d'expertise » est réalisé entre un ingénieur de la connaissance et un expert du domaine. Durant ces sessions d'acquisition de connaissances, l'expert est mis en situation et est encouragé à formuler les connaissances qu'il utilise pour résoudre les problèmes. Les connaissances acquises sont alors formalisées de façon à être utilisables par un système informatique. Ce genre d'approche est très coûteuse car elle prend beaucoup de temps et nécessite de réunir à la fois un expert du domaine et un ingénieur de la connaissance.

② et ⑤ Les **approches automatiques** consistent à appliquer un algorithme d'apprentissage automatique pour obtenir des connaissances d'adaptation à partir d'un ensemble de données ou de connaissances. L'expert du domaine ne fait pas partie du processus. La validation des connaissances acquises se fait alors quantitativement en mesurant les gains en performance du système. Lorsqu'il est déclenché en ligne (type d'approche ⑤), l'algorithme d'apprentissage a pour but de générer à la volée les connaissances d'adaptation qui seront utilisées dans le raisonnement. Ce type d'apprentissage, lorsqu'il est réalisé au cours du raisonnement, est caractéristique de la façon dont est appréhendé la résolution de problèmes en raisonnement à partir de cas : le système est doté pour résoudre des problèmes d'un ensemble de données et de connaissances dans lesquelles il recherche « à la demande » les informations nécessaires à la résolution d'un nouveau problème. Pour cette raison, les systèmes de raisonnement à partir de cas sont parfois qualifiés de systèmes « paresseux » de résolution de problèmes (en anglais, *lazy problem-solving*) [Aha, 1998].

③ Les approches **semi-automatiques auprès d'experts** utilisent des techniques d'extraction de connaissances à partir de données pour générer des éléments de connaissances à partir de diverses sources de données. L'apprentissage est réalisé hors ligne. Les éléments de connaissances extraits sont présentés à un expert du domaine pour validation. À notre connaissance, seul le système CabamakA inclut une étape de validation des connaissances extraites auprès d'un expert du domaine (voir chapitre 6).

④ Les approches **interactives auprès d'utilisateurs** sont réalisées en ligne. Dans ces approches, le système acquiert des connaissances de ses interactions avec l'utilisateur.

⑥ Dans les approches **semi-automatiques auprès d'utilisateurs**, un algorithme d'apprentissage est déclenché en ligne pour acquérir les connaissances d'adaptation nécessaires à la résolution d'un problème donné. L'apprentissage est guidé par le problème à résoudre et son paramétrage résulte d'un ensemble d'interactions avec l'utilisateur du système. L'utilisateur intervient également pour valider les connaissances acquises. Le chapitre 7 présente une méthode d'acquisition semi-automatique de connaissances d'adaptation auprès d'utilisateurs dans laquelle le système CabamakA est déclenché en ligne de manière opportuniste et interactive.

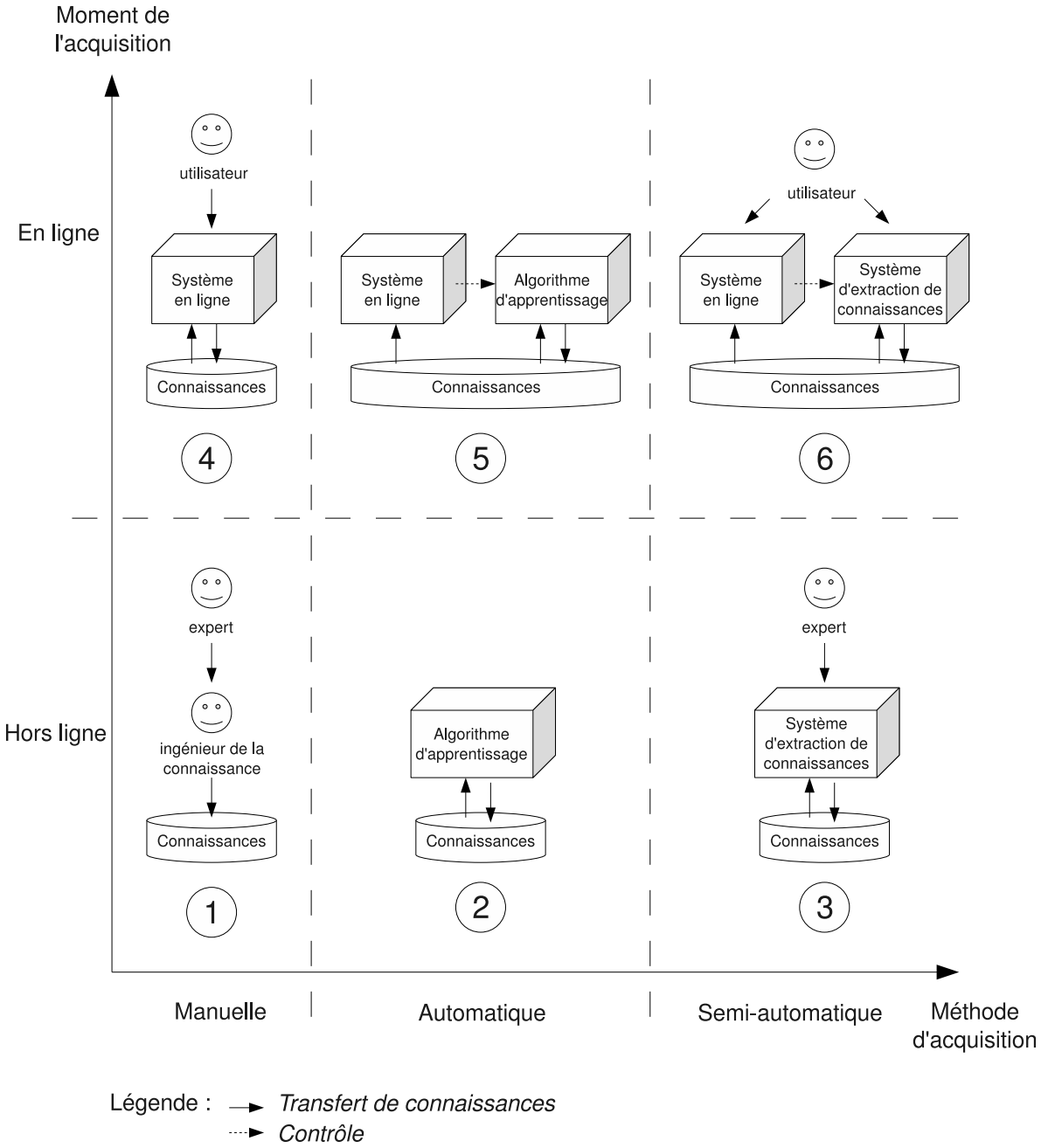


FIG. 1.3 – Différents types d'approches pour l'acquisition de connaissances d'adaptation.

1.3.2 Différentes sources de connaissances d'adaptation

Les connaissances d'adaptation peuvent être apprises à partir de différentes sources. Michael Richter identifie dans [Richter, 1998] quatre conteneurs de connaissances qui sont utilisés par un système de raisonnement à partir de cas : le vocabulaire, les connaissances de similarité, la base de cas et les connaissances d'adaptation. Dans [Wilke *et al.*, 1997], les auteurs proposent alors de distinguer deux types d'approches d'acquisition de connaissances d'adaptation : les approches à moindre coût d'acquisition de connaissances, ou *knowledge-light approaches*, qui ne nécessitent pas d'acquisition de connaissances préalable mais exploitent les connaissances déjà représentées dans le système, et les approches qui requièrent des connaissances externes au système, comme des connaissances du domaine ou des connaissances acquises auprès de l'expert, et sont qualifiées pour cela de *knowledge-intensive approaches*. Les approches à moindre coût d'acquisition de connaissances ne prennent pour source de connaissances que les conteneurs utilisés par le système de raisonnement à partir de cas et réalisent un transfert de connaissances d'un ou plusieurs conteneurs de connaissances vers le conteneur de connaissances d'adaptation.

ACA à partir des connaissances du domaine. Certains systèmes de raisonnement à partir de cas comme DIAL [Leake *et al.*, 1996], CHEF [Hammond, 1986] ou CASEY [Koton, 1988] utilisent des connaissances du domaine comme sources de connaissances d'adaptation. Par exemple, lorsque DIAL applique le principe *adaptation = transformation + exploration de la mémoire* pour générer une règle d'adaptation, des heuristiques sont appliquées pour acquérir à partir des connaissances du domaine les informations nécessaires à l'instanciation d'une règle abstraite. La principale hypothèse faite dans de telles approches est que le système est pourvu de suffisamment de connaissances du domaine pour faire face à tout nouveau problème d'adaptation. Ce type d'approche est particulièrement intéressant lorsque le système fait déjà usage d'un ensemble de connaissances du domaine pour représenter les cas. Les connaissances du domaine peuvent alors être réutilisées pour en dériver des connaissances d'adaptation. C'est le cas dans Taaable, où à partir de l'ontologie $\mathcal{O}$, qui est utilisée pour représenter les recettes, on dérive un ensemble de substitutions entre ingrédients qui sont ensuite utilisées lors de l'adaptation. Mais les travaux effectués sur le système Taaable ont montré que l'utilisation des connaissances du domaine pour dériver des connaissances d'adaptation reste limité : les connaissances d'adaptation obtenues de cette façon sont assez simples. Pour réaliser des adaptations complexes, le système a besoin de connaissances d'adaptation additionnelles. Acquérir ces connaissances à partir des connaissances du domaine nécessiterait de réaliser une acquisition préalable de connaissances du domaine (en affinant l'ontologie de Taaable par exemple), ce qui substituerait à le problématique d'acquisition de connaissances d'adaptation une problématique d'acquisition de connaissances du domaine.

ACA réalisée à partir de bases de connaissances sur le Web. Pour réduire la tâche d'acquisition de connaissances du domaine, David Leake et Jay Powell ont proposé dans le système WebAdapt [Leake et Powell, 2007, Leake et Powell, 2008] d'acquérir des connaissances d'adaptation à partir de bases de connaissances disponibles sur le Web. Les travaux effectués sur WebAdapt suggèrent que pour que la méthode soit efficace, les sources de connaissances externes utilisées doivent être suffisamment riches, mais également que l'utilisation de multiples sources de connaissances ou de sources de connaissances trop nombreuses ou à la sémantique mal définie nuit aux performances du système.

Pour exploiter pleinement les connaissances disponibles sur le Web, une direction de recherche qui semble prometteuse consiste à s'appuyer sur les langages du Web Sémantique

(RDF(S), OWL, SPARQL) pour intégrer les ressources disponibles sur le Web avec les connaissances du système de raisonnement à partir de cas. Quelques travaux ont déjà été effectués en ce sens, comme par exemple le travail de Mikael Kirkeby Fidjeland [Fidjeland, 2006] dont le but est de doter le système CREEK d'un vocabulaire OWL, la proposition d'un vocabulaire RDF appelé CaseML [Chen et Wu, 2003] pour décrire les connaissances utilisées dans un système de RÀPC, ou encore l'ontologie CBROnto [Díaz-Agudo et González-Calero, 2002] pour représenter les tâches et méthodes utilisées en raisonnement à partir de cas.

ACA à partir de la base de cas. Pour faciliter la tâche d'acquisition de connaissances d'adaptation, Kathleen Hanney et Mark T. Keane ont proposé dans [Hanney, 1997] d'apprendre des connaissances d'adaptation à partir de différences entre cas dans la base de cas. La principale hypothèse faite dans ce type d'approche est que les différences entre les cas de la base de cas sont souvent représentatives des différences qui seront rencontrées entre un nouveau problème à résoudre et la base de cas. La méthode d'apprentissage par généralisation de règles d'adaptation présentée dans le chapitre 5 suit cette approche : les règles d'adaptation sont apprises à partir des différences entre cas dans la base de cas.

ACA auprès de l'opérateur humain. Une autre source de connaissances d'adaptation est l'opérateur humain. Lorsque l'apprentissage est réalisé hors ligne, l'opérateur est un expert du domaine qui est sollicité pour formuler les connaissances qu'il utilise dans le raisonnement. Lorsque l'apprentissage est réalisé en ligne, c'est l'utilisateur du système qui est sollicité pour valider la solution proposée par le système ou pour formuler les connaissances nécessaires à sa réparation, ce qui suppose de se doter d'une interface homme-machine qui permette une telle acquisition.

1.3.3 Résumés des principales approches d'ACA

Cette section présente une description succinte des principales approches d'acquisition de connaissances d'adaptation. Un panorama des principales approches est donné par la figure 1.4

Dans la suite, chaque approche sera décrite par le nom du système dans laquelle elle est implémentée, par une ou plusieurs références bibliographiques, ainsi que par le type d'approche d'acquisition de connaissances mise en œuvre (numérotées ①, ②, ③, ④, ⑤ et ⑥ dans la section 1.3.1) et par les sources de connaissances à partir desquelles les connaissances d'adaptation sont acquises. Ces informations sont résumées dans un tableau de la forme suivante :

Nom du système [Références]	**Type d'approche** : le type d'approche. **Sources de connaissances** : les sources de connaissances d'adaptation. Une description de l'approche.

Pour présenter ces approches, nous suivrons l'ordre qui a été introduit pour présenter les types d'approches d'acquisition de connaissances (numérotées de ① à ⑥).

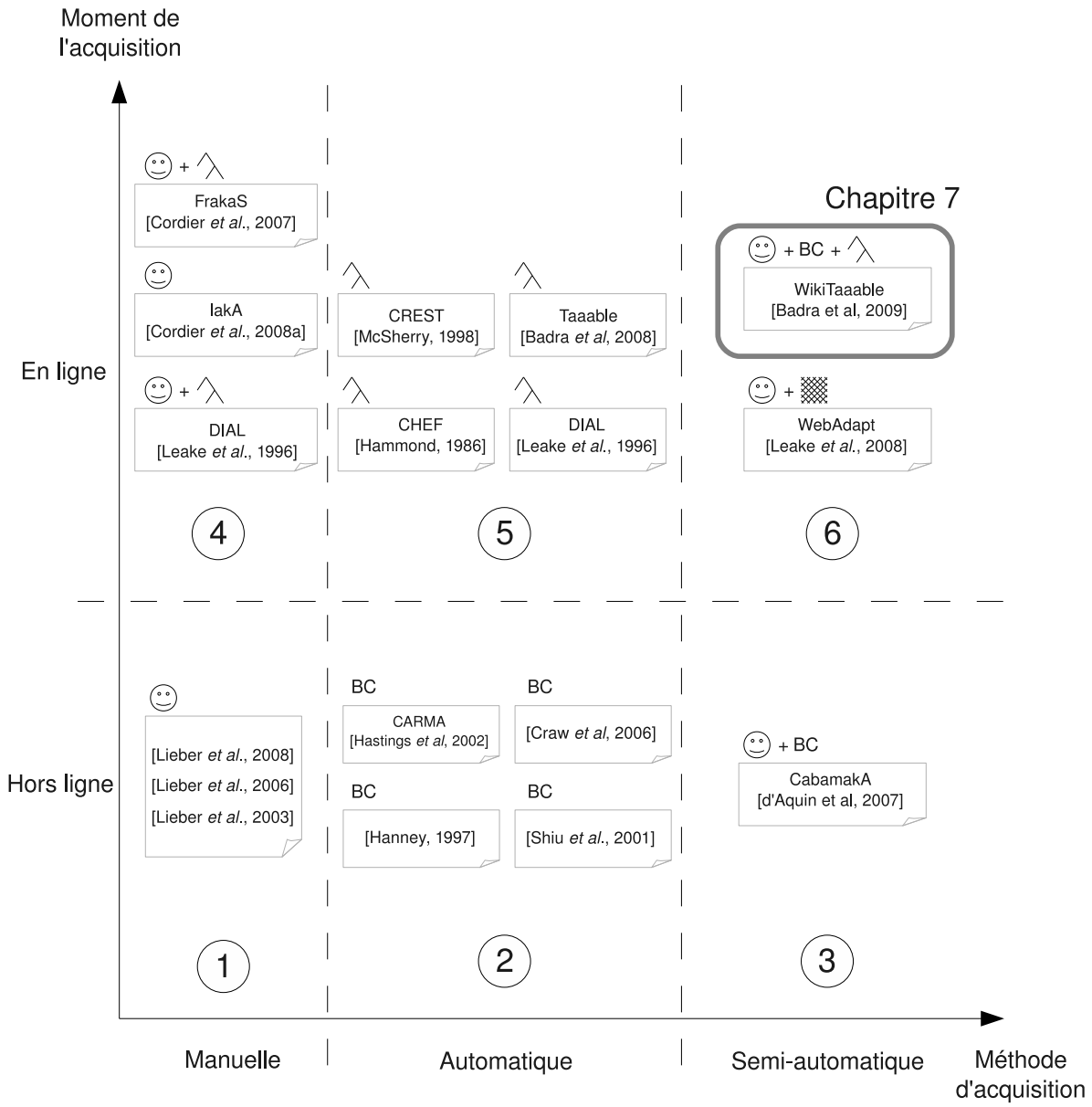


FIG. 1.4 – Panorama des principales approches d'acquisition de connaissances d'adaptation.

[Lieber *et al.*, 2008]
[Lieber *et al.*, 2006]
[Lieber *et al.*, 2003]

Type d'approche : ① acquisition manuelle auprès d'experts
Sources de connaissances : ☺ l'expert du domaine

[Lieber *et al.*, 2003], [d'Aquin *et al.*, 2006b] et [Lieber *et al.*, 2008] présentent des sessions d'acquisition de connaissances réalisées aurpès d'experts dans le domaine de la cancérologie. Ces sessions ont permis de mettre en évidence un petit nombre de stratégies générales d'adaptation appelées *patrons d'adaptation*. Un patron d'adaptation est une connaissance abstraite applicable dans de nombreuses situations lorsqu'elle est correctement instanciée. De ce point de vue un patron d'adaptation peut être rapproché de la notion de cas de décomposition introduite dans le système Déjà Vu (voir section 1.2.4). Néanmoins la forme que prend la connaissance une fois instanciée est différente puisque l'instanciation d'un patron d'adaptation donne une reformulation.

CARMA
[Hastings *et al*, 2002]

Type d'approche : ② acquisition automatique hors ligne
Sources de connaissances : `BC` la base de cas

CARMA [Hastings *et al.*, 2002] est un système de prédiction dans le domaine du traitement des invasions de sauterelles dans l'ouest des États-Unis. Dans CARMA, la base de cas est composée d'un ensemble de cas prototypiques représentant chacun un scénario d'invasion de sauterelles. L'adaptation se fait en ajustant le traitement effectué dans un cas prototypique remémoré pour prendre en compte les différences entre le nouveau problème et le problème traité dans le cas prototypique. Par exemple, la prolifération des sauterelles augmente avec la température donc l'estimation des pertes en fourrage dans le cas prototypique remémoré devront être ajustées à la baisse si la température est inférieure dans le nouveau problème. Pour déterminer comment ajuster chacun des paramètres lors de la prédiction, un ensemble de poids d'adaptation est appris à partir de la base de cas. Les poids d'adaptation sont appris en évaluant la façon dont le cas cible obtenu pourrait être adapté pour prédire les pertes en fourrage pour les différents problèmes traités dans les cas prototypiques de la base de cas. Un algorithme de recherche incrémental de type *Hill Climbing* ajuste progressivement les poids d'adaptation de manière à minimiser l'erreur quadratique entre la prédiction réalisée par CARMA par adaptation du cas cible obtenu pour chacun des problèmes traités dans les cas prototypiques et l'estimation qui en est faite par l'expert.

[Hanney, 1997]

Type d'approche : ② acquisition automatique hors ligne
Sources de connaissances : BC la base de cas

[Hanney, 1997] présente une méthode d'apprentissage automatique hors ligne de régles d'adaptation dans le domaine de la prédiction de prix d'appartements. L'apprentissage est réalisé à partir des différences entre cas dans la base de cas. Un ensemble de couples de cas sources de la base de cas est sélectionné et chaque couple de cas sources est considéré comme une règle d'adaptation spécifique. Les différences entre problèmes constituent l'antécédent de la règle et les différences entre solutions constituent le conséquent. L'algorithme de fermeture d'intervalle proposé par Ryszard S. Michalski dans [Michalski, 1983] est alors utilisé pour généraliser les antécédents des règles.

[Shiu *et al.*, 2001]

Type d'approche : ② acquisition automatique hors ligne
Sources de connaissances : BC la base de cas

Dans [Shiu *et al.*, 2001], l'idée d'apprendre des connaissances d'adaptation à partir des différences entre cas a été appliquée avec succès à la tâche de maintenance de la base de cas. En raisonnement à partir de cas, on parle de maintenance de la base de cas pour désigner un ensemble de méthodes qui permettent de mettre à jour l'organisation ou le contenu de la base de cas dans le but d'améliorer les performances du système. Dans [Shiu *et al.*, 2001], un algorithme de construction d'arbres de décision flous est appliqué pour apprendre un ensemble de règles d'adaptation à partir de couples de cas sources similaires de la base de cas. Ces règles d'adaptation sont ensuite utilisées pour générer toute la base de cas à partir d'un ensemble de cas sources représentatifs. Les auteurs ont ainsi montré qu'il était possible de réduire la taille de la base de cas en transférant certaines connaissances de la base de cas vers les connaissances d'adaptation.

[Craw *et al*, 2006]

Type d'approche : ② acquisition automatique hors ligne
Sources de connaissances : `BC` la base de cas

[Jarmulak *et al.*, 2001], [Wiratunga *et al.*, 2002] et [Craw *et al.*, 2006] présentent une approche d'acquisition automatique de cas d'adaptation à partir des différences entre cas dans la base de cas. L'approche est appliquée à une tâche de configuration dans le domaine de la formulation de médicaments. Un cas d'adaptation associe une action d'adaptation à une représentation des différences entre deux cas sources. L'approche proposée est à moindre coût d'acquisition de connaissances (*knowledge-light*) dans la mesure où les connaissances d'adaptation sont acquises à partir de la base de cas uniquement. Des algorithmes d'apprentissage comme C4.5 [Quinlan, 1993] ou RISE [Domingos, 1996] sont appliqués pour obtenir des connaissances d'adaptation plus générales à partir d'un ensemble de cas d'adaptation.

IakA
[Cordier *et al.*, 2008a]

Type d'approche : ④ acquisition interactive auprès d'utilisateurs
Sources de connaissances : ☺ l'expert du domaine

IakA [Cordier *et al.*, 2008, Cordier, 2008] est une approche d'acquisition interactive de connaissances d'adaptation. L'acquisition est déclenchée de manière opportuniste pendant la phase de test de la solution proposée par le système. Son but est d'acquérir auprès de l'utilisateur les connaissances nécessaires à la réparation d'une adaptation qui a échoué. La modélisation générale de l'adaptation utilisée dans IakA est celle de l'adaptation par reformulations. Les connaissances acquises sont des reformulations, appelées « opérateurs d'adaptation » dans ces travaux. L'approche est implémentée dans le prototype IakA-NF dans le domaine de l'approximation de fonctions numériques. Dans IakA-NF, les opérateurs d'adaptation acquis auprès de l'utilisateur prennent la forme d'une approximation de valeur de la dérivée de la fonction à approcher au voisinage d'un point. Un tel opérateur correspond à la notion de dépendance entre descripteurs problèmes et descripteurs solutions introduite dans la théorie de l'adaptation différentielle.

FrakaS
[Cordier *et al.*, 2007]

Type d'approche : ④ acquisition interactive auprès d'utilisateurs
Sources de connaissances : ☺ l'expert du domaine
⩓ les connaissances du domaine

FrakaS [Cordier *et al.*, 2007] est un système de raisonnement à partir de cas interactif qui apprend de certains échecs de l'adaptation. La modélisation de l'adaptation utilisée dans FrakaS est celle de l'adaptation conservatrice [Lieber, 2007]. Dans FrakaS, l'utilisateur peut interagir avec le système pour exprimer des incohérences vis-à-vis des connaissances de l'expert dans la solution proposée. Les connaissances du domaine acquises auprès de l'utilisateur du système sont utilisées lors de l'étape d'adaptation pour corriger l'adaptation proposée. Les systèmes IakA et FrakaS sont deux exemples de systèmes implémentant l'approche générale FIKA [Cordier, 2008] pour l'acquisition de connaissances interactive et opportuniste en raisonnement à partir de cas.

DIAL
[Leake *et al.*, 1996]

Type d'approche : ④ acquisition interactive auprès d'utilisateurs
⑤ acquisition automatique en ligne
Sources de connaissances : ☺ l'expert du domaine
⩓ les connaissances du domaine

Le système DIAL [Leake *et al.*, 1996] acquiert des connaissances d'adaption sous la forme de cas d'adaptation. Un cas d'adaptation constitue une trace du raisonnement effectué lors de l'adaptation qui peut être réutilisé pour résoudre de nouveaux problèmes d'adaptation. Les cas d'adaptation sont acquis de deux façons différentes. La première consiste à mémoriser un cas d'adaptation chaque fois qu'une adaptation a réussi. La deuxième consiste à acquérir des cas d'adaptation à l'issue d'une adaptation manuelle réalisée par l'utilisateur *via* une interface graphique dédiée.

CHEF
[Hammond, 1986]

Type d'approche : ⑤ acquisition automatique en ligne
Sources de connaissances : ⋀ les connaissances du domaine

Le système CHEF [Hammond, 1986] apprend des connaissances d'adaptation de ses échecs de raisonnement. Les connaissances d'adaptation apprises par CHEF sont des traces d'échecs d'adaptation et des critiques. Les traces d'échecs d'adaptation sont mémorisés dans le modèle causal de CHEF, ce qui permet au système d'anticiper par la suite de nouveaux échecs d'adaptation. Par exemple, le système détecte un échec d'adaptation lorsqu'il effectue une recette de soufflé aux fraises car il prédit que les fraises vont lâcher de l'eau à la cuisson et que le soufflé ne lèvera pas. Pour éviter de rencontrer à nouveau un tel échec d'adaptation, CHEF modifie son modèle causal pour ajouter une règle selon laquelle les fraises sont susceptibles de rendre de l'eau à la cuisson et que ceci a pour conséquence d'empêcher un soufflé de lever. Comme ce type d'échec est susceptible de se produire pour tout type de fruit, la règle est ensuite généralisée à tous les types de fruits. Chaque fois qu'une adaptation a été réparée avec succès, le système génère également une nouvelle critique qui reflète la réparation effectuée. CHEF décide alors de la mémoriser ou non selon sa propension à être utilisée pour résoudre de nouveaux problèmes.

CREST
[McSherry, 1998]

Type d'approche : ⑤ acquisition automatique en ligne
Sources de connaissances : ⋀ les connaissances du domaine

Le système CREST [McSherry, 1998, McSherry, 1999] apprend des connaissances d'adaptation à partir des différences entre cas dans la base de cas dans le but de réaliser des substitutions à partir de cas. CREST sélectionne lors de la remémoration un cas source $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ à l'aide d'un algorithme de recherche des plus proches voisins puis résoud chacune des différences entre `srce` et `cible` en cherchant dans la base de cas des couples de cas sources exhibant les mêmes différences. La solution $\texttt{Sol}(\texttt{srce})$ du cas source est alors mise à jour pour refléter les différences entre les solutions des cas sources extraits de la base de cas.

WebAdapt
[Leake *et al.*, 2008]

Type d'approche : ⑥ acquisition semi-automatique auprès d'utilisateurs
Sources de connaissances : ☺ l'expert du domaine
▒ le Web

Comme dans le système DIAL, WebAdapt aborde l'adaptation comme un problème de recherche d'informations nécessaires à l'instanciation d'une transformation abstraite. Néanmoins, dans WebAdapt, les informations sont recherchées dans un ensemble de bases de connaissances préexistantes sur le Web comme OpenCyc ou Wikipedia. Une première évaluation de la méthode permet de mettre en évidence la nécessité de guider la recherche d'informations et de sélectionner soigneusement les sources de connaissances utilisées par le système pour générer des règles d'adaptation. Pour pallier à ces limitations, une amélioration de WebAdapt est présentée dans [Leake et Powell, 2008] dans laquelle le système est pourvu d'un modèle des préférences de l'utilisateur qui permet de guider la recherche. L'utilisateur intervient pour sélectionner une ou plusieurs solutions parmi celles proposées par le système et pour demander des explications sur la façon dont ces solutions ont été trouvées. Les interactions avec l'utilisateur sont utilisées pour enrichir le modèle de préférences utilisé par le système pour guider la recherche. Néanmoins, l'approche développée pour l'acquisition interactive de connaissances auprès d'utilisateurs n'est ni itérative ni opportuniste (contrairement à l'approche FIKA proposée par Amélie Cordier dans [Cordier, 2008]) : les connaissances sont acquises pendant l'étape d'adaptation et l'adaptation n'est pas corrigée pour tenir compte des interactions avec l'utilisateur.

2

Contexte applicatif : le système de RàPC culinaire Taaable

Ce chapitre présente Taaable*, une application qui utilise un système de raisonnement à partir de cas dans le domaine culinaire pour proposer à ses utilisateurs des recettes de cuisine.* Taaable *constitue le domaine applicatif de l'étude. Ce système a été développé pour participer en 2008 au* Computer Cooking Contest *(CCC), une compétition qui a eu lieu en marge de la 9ème conférence européenne sur la raisonnement à partir de cas (ECCBR'08) et pour laquelle le système a obtenu le titre de vice-champion d'Europe. La modélisation de l'adaptation utilisée dans* Taaable *est celle de l'adaptation par reformulations. Dans la version du système proposée en 2008 pour participer au CCC, les connaissances d'adaptation sont acquises en ligne et de manière automatique. Le système génère automatiquement des connaissances d'adaptation à partir des connaissances du domaine. Nous montrons dans ce chapitre les limites de la méthode et motivons l'acquisition de connaissances d'adaptation dans* Taaable*.*

❄

[Badra *et al.*, 2009a] F. Badra, J. Cojan, A. Cordier, J. Lieber, T. Meilender, A. Mille, P. Molli, E. Nauer, A. Napoli, H. Skaf-Molli et Y. Toussaint. Knowledge acquisition and discovery for the textual case-based cooking system WikiTaaable. Dans L. McGinty et D. C.Wilson, rédacteurs, *8th International Conference on Case-Based Reasoning - ICCBR 2009, Workshop Proceedings*. 2009.

[Badra *et al.*, 2008a] F. Badra, R. Bendaoud, R. Bentebibel, P.-A. Champin, J. Cojan, A. Cordier, S. Després, S. Jean-Daubias, J. Lieber, T. Meilender, A. Mille, E. Nauer, A. Napoli et Y. Toussaint. Taaable : Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. Dans Martin Schaaf, rédacteur, *9th European Conference on Case-Based Reasoning - ECCBR 2008*, p. 219–228. Trier Allemagne, 2008.

Le système Taaable [Badra *et al.*, 2008a] est un système de raisonnement à partir de cas culinaire. Dans le domaine culinaire, le raisonnement à partir de cas a pour but de répondre à une requête en utilisant un ensemble de recettes. Pour répondre à une requête, le système sélectionne une recette dans sa base de recettes et l'adapte pour produire une recette qui satisfait la requête. Le système Taaable a été proposé pour participer au *Computer Cooking Contest* (CCC) en 2008 [Schaaf, 2008] en marge de la 9ème conférence européenne sur le raisonnement à partir de cas (ECCBR) et a obtenu le titre de vice-champion d'Europe. Une copie d'écran de l'interface graphique de l'application Taaable proposée à cette occasion est donnée par la figure 2.1. Le système Taaable est accessible en ligne à l'URL `http://taaable.fr`.

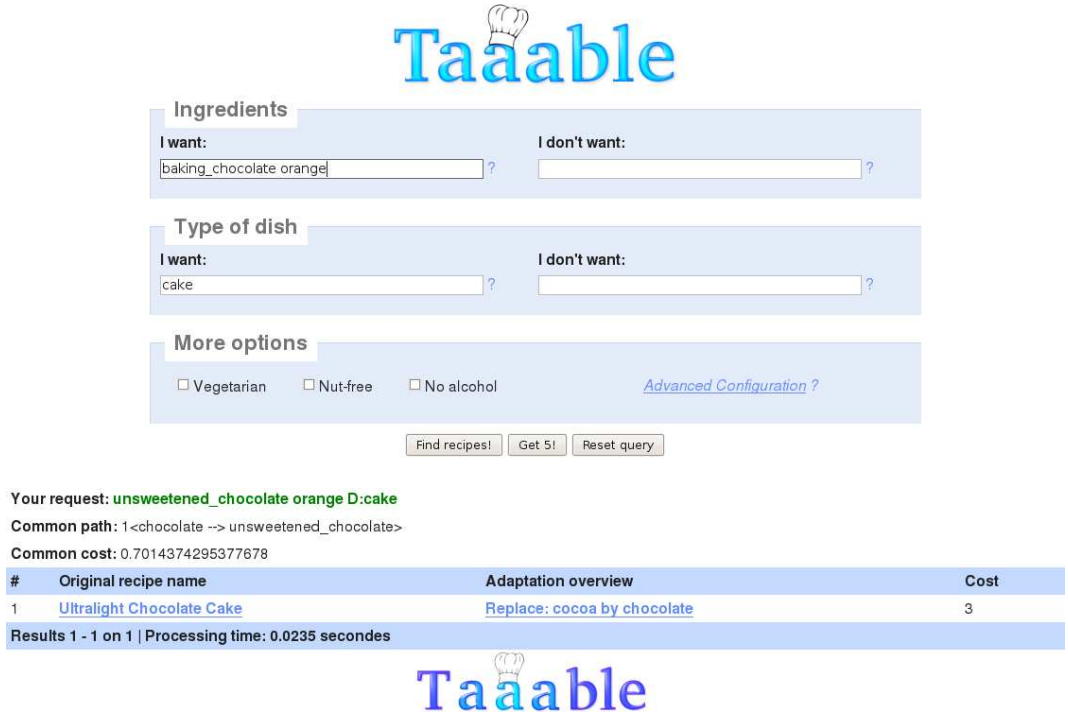


Fig. 2.1 – Copie d'écran de l'interface graphique de Taaable (septembre 2008).

Lors de cette compétition, les requêtes sont données en langage naturel et expriment un ensemble de contraintes que la recette désirée doit satisfaire. Ces contraintes portent sur les ingrédients qui doivent être inclus ou non dans la recette (comme de la viande ou des oranges), sur la pratique diététique que la recette doit respecter (comme une recette qui s'accorde avec un régime sans noix), sur le type de plat (comme une soupe ou un dessert), ou encore sur le type de cuisine (comme méditerranéenne ou chinoise). Un exemple de requête est :

« Je veux une soupe chinoise contenant des poireaux mais sans huile d'arachide. »

Les recettes sont données au format XML et incluent une liste d'ingrédients et une description en texte libre de la préparation de la recette.

2.1 Les connaissances du domaine

Le système utilise une ontologie culinaire O représentée en logique propositionnelle. Chaque concept de O correspond à une variable propositionnelle prise dans un ensemble fini $\mathcal{V}$ de variables propositionnelles. O est composée d'un ensemble de concepts organisés en hiérarchie, ce qui correspond, en logique propositionnelle, à un ensemble d'implications $\mathtt{a} \Rightarrow \mathtt{b}$. Par exemple, l'axiome $\mathtt{poireau} \Rightarrow \mathtt{allium}$ représente le fait que toute recette avec du poireau est une recette avec un allium (le poireau étant un allium).

2.2 Représentation des cas

En logique propositionnelle, les formules sont construites sur un ensemble fini $\mathcal{V}$ de variables propositionnelles. Une interprétation $\mathcal{I}$ est une fonction définie sur $\mathcal{V}$ et à valeurs dans l'ensemble des booléens $\mathbb{B} = \{\mathtt{vrai}, \mathtt{faux}\}$. $\mathcal{I}$ est étendue à l'ensemble des formules de la manière habituelle (par exemple si f et g sont deux formules, $(f \wedge g)^{\mathcal{I}} = \mathtt{vrai}$ ssi $f^{\mathcal{I}} = \mathtt{vrai}$ et $g^{\mathcal{I}} = \mathtt{vrai}$). Un modèle d'une formule f est une interprétation $\mathcal{I}$ telle que $f^{\mathcal{I}} = \mathtt{vrai}$. L'ensemble des modèles d'une formule f est noté $\mathtt{Mod}(f)$. Pour deux formules f et g, f implique g, noté $f \models g$ si $\mathtt{Mod}(f) \subseteq \mathtt{Mod}(g)$. Finalement, f entraîne g étant donné l'ontologie O, noté $f \models_O g$, si $f \wedge O \models g$.

Dans Taaable, un problème $\mathtt{srce} \in \mathcal{L}_{\mathtt{pb}}$ représente une requête et une solution $\mathtt{Sol}(\mathtt{srce}) \in \mathcal{L}_{\mathtt{sol}}$ représente une recette. Le langage $\mathcal{L}_{\mathtt{pb}}$ de représentation des problèmes est un fragment de la logique propositionnelle défini sur un ensemble de variables propositionnelles $\mathcal{V}$. Les formules de $\mathcal{L}_{\mathtt{pb}}$ sont des conjonctions de littéraux $l_1 \wedge \cdots \wedge l_n$ où chaque littéral l_i est soit une variable propositionnelle (littéral positif) soit la négation d'une variable propositionnelle (littéral négatif). Le langage $\mathcal{L}_{\mathtt{sol}}$ de représentation des solutions est défini de manière similaire. Par exemple, la représentation $\mathtt{cible} \in \mathcal{L}_{\mathtt{pb}}$ de la requête mentionnée ci-dessus est :

$$\mathtt{cible} = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{poireau} \wedge \neg\, \mathtt{huile\_arachide}$$

Dans cette requête, $\mathtt{soupe}$ représente la classe des recettes de soupe, $\mathtt{chinoise}$ représente la classe des recettes chinoises et $\mathtt{poireau}$ représente la classe des recettes contenant du poireau.

La base de cas $\mathtt{BC}$ contient un ensemble de recettes. Chaque recette est indexée dans la base de cas par une formule $\mathrm{Idx}(\mathtt{R}) \in \mathcal{L}_{\mathtt{sol}}$. Par exemple, l'index $\mathrm{Idx}(\mathtt{R})$ de la recette de la soupe Wonton est :

$$\mathrm{Idx}(\mathtt{R}) = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{oignon\_vert} \wedge \mathtt{bok\_choy} \wedge \mathtt{huile\_arachide} \wedge \textit{Rien d'autre}$$

Rien d'autre dénote une conjonction de littéraux négatifs $\neg\, \mathtt{a}$ pour tout $\mathtt{a} \in \mathcal{V}$ tel que $\mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{oignon\_vert} \wedge \mathtt{bok\_choy} \wedge \mathtt{huile\_arachide} \not\models_O \mathtt{a}$. Cette sorte d'« hypothèse du monde clos » établit explicitement que pour toute variable propositionnelle $\mathtt{a} \in \mathcal{V}$, soit $\mathrm{Idx}(\mathtt{R}) \models_O \mathtt{a}$ (la recette contient l'ingrédient représenté par $\mathtt{a}$) soit $\mathrm{Idx}(\mathtt{R}) \models_O \neg\, \mathtt{a}$ (la recette ne contient pas l'ingrédient représenté par $\mathtt{a}$).

Chaque index de recette $\mathrm{Idx}(\mathtt{R})$ peut être vu comme la représentation d'un ensemble de cas sources : $\mathrm{Idx}(\mathtt{R})$ représente l'ensemble des cas sources $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ tels que $\mathtt{Sol}(\mathtt{srce}) = \mathrm{Idx}(\mathtt{R})$ et $\mathtt{srce}$ est résolu par $\mathrm{Idx}(\mathtt{R})$, c'est-à-dire que $\mathtt{srce}$ vérifie $\mathrm{Idx}(\mathtt{R}) \models_O \mathtt{srce}$.

2.3 Représentation des connaissances d'adaptation

Dans Taaable, les connaissances d'adaptation sont données par un ensemble de reformulations $(\mathtt{r}, \mathcal{A}_{\mathtt{r}})$, où $\mathtt{r}$ est une relation binaire entre problèmes et $\mathcal{A}_{\mathtt{r}}$ est une fonction d'adaptation

associée à $\mathtt{r}$ [Melis *et al.*, 1998]. La sémantique de ces reformulations correspond à celle présentée dans l'approche de l'adaptation par reformulations (section 1.2.2) : si deux problèmes $\mathtt{pb}_1$ et $\mathtt{pb}_2$ sont liés par $\mathtt{r}$ — noté $\mathtt{pb}_1\ \mathtt{r}\ \mathtt{pb}_2$ — alors pour chaque recette $\mathtt{Sol}(\mathtt{pb}_1)$ qui résout la requête $\mathtt{pb}_1$, $\mathcal{A}_\mathtt{r}(\mathtt{pb}_1, \mathtt{Sol}(\mathtt{pb}_1), \mathtt{pb}_2) = \widetilde{\mathtt{Sol}}(\mathtt{pb}_2)$ résout la requête $\mathtt{pb}_2$.

Dans TAAABLE, les relations binaires $\mathtt{r}$ sont données par des substitutions de la forme $\gamma = \alpha \rightsquigarrow \beta$, où α et β sont des littéraux (positifs ou négatifs). Par exemple, la substitution $\gamma = \mathtt{poireau} \rightsquigarrow \mathtt{allium}$ généralise les poireaux en alliums.

Les fonctions d'adaptation $\mathcal{A}_\mathtt{r}$ sont données par des substitutions de la forme $\sigma = \mathtt{A} \rightsquigarrow \mathtt{B}$, dans lesquelles $\mathtt{A}$ et $\mathtt{B}$ sont des conjonctions de littéraux. Par exemple, la substitution $\sigma = \mathtt{soupe} \wedge \mathtt{poivre} \rightsquigarrow \mathtt{soupe} \wedge \mathtt{gingembre}$ établit que le poivre peut être substitué par du gingembre dans les recettes de soupe.

2.4 Sources des connaissances d'adaptation

La principale source de connaissances d'adaptation est l'ontologie $\mathcal{O}$. Une substitution $\gamma = \mathtt{a} \rightsquigarrow \mathtt{b}$ est générée automatiquement à partir de chaque axiome $\mathtt{a} \Rightarrow \mathtt{b}$ de $\mathcal{O}$ et correspond à une *substitution par généralisation*. Une substitution $\gamma = \mathtt{a} \rightsquigarrow \mathtt{b}$ peut être appliquée à une requête $\mathtt{pb}$ si $\mathtt{pb} \vDash_\mathcal{O} \mathtt{a}$. γ génère une nouvelle requête $\gamma(\mathtt{pb})$ dans laquelle la variable propositionnelle $\mathtt{a}$ a été substituée par la variable propositionnelle $\mathtt{b}$. Par exemple, la substitution $\gamma = \mathtt{poireau} \rightsquigarrow \mathtt{allium}$ est générée automatiquement à partir de l'axiome $\mathtt{poireau} \Rightarrow \mathtt{allium}$ de $\mathcal{O}$. γ peut être appliquée à la requête $\mathtt{cible}$ pour produire la requête

$$\gamma(\mathtt{cible}) = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{allium} \wedge \neg\,\mathtt{huile\_arachide}$$

dans laquelle $\mathtt{poireau}$ a été substitué par $\mathtt{allium}$. Pour chaque variable propositionnelle $\mathtt{a}$ de $\mathcal{V}$, une substitution additionnelle de la forme $\gamma = \neg\,\mathtt{a} \rightsquigarrow \emptyset$ est également générée. Une telle substitution peut être appliquée à un problème $\mathtt{pb}$ si $\mathtt{pb} \vDash_\mathcal{O} \neg\,\mathtt{a}$ et génère un nouveau problème $\gamma(\mathtt{pb})$ dans lequel le littéral négatif $\neg\,\mathtt{a}$ a été supprimé. Cela a pour effet de relâcher certaines contraintes imposées sur une requête, par exemple en omettant dans la requête une contrainte sur l'absence d'un ingrédient. Par exemple, la substitution $\neg\,\mathtt{huile\_arachide} \rightsquigarrow \emptyset$ est appliquée à $\mathtt{cible}$ pour générer la requête

$$\gamma(\mathtt{cible}) = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{poireau}$$

dans laquelle la condition sur l'absence de l'ingrédient $\mathtt{poireau}$ a été omise.

Une substitution σ peut être générée automatiquement à partir d'une substitution γ : σ est formée par généralisation/spécialisation à partir de la substitution γ^{-1} en utilisant un axiome de l'ontologie $\mathcal{O}$. Par exemple, une substitution $\sigma = \mathtt{oignon\_vert} \rightsquigarrow \mathtt{poireau}$ peut être formée à partir de la substitution $\gamma = \mathtt{poireau} \rightsquigarrow \mathtt{allium}$ et du fait que $\mathtt{oignon\_vert} \vDash_\mathcal{O} \mathtt{allium}$ lorsque la recette remémorée $\mathtt{Sol}(\mathtt{srce})$ contient des oignons verts (figure 2.2). Lorsque γ est de la forme $\neg\,\mathtt{a} \rightsquigarrow \emptyset$, une substitution $\sigma = \emptyset \rightsquigarrow \neg\,\mathtt{a}$ peut être générée à partir de γ. Par exemple, une substitution $\sigma = \emptyset \rightsquigarrow \neg\,\mathtt{huile\_arachide}$ peut être générée à partir de la substitution $\gamma = \neg\,\mathtt{huile\_arachide} \rightsquigarrow \emptyset$.

Néanmoins, lorsque $\mathcal{O}$ est la seule source de connaissances d'adaptation, le système est seulement capable de réaliser des adaptations simples dans lesquelles les modifications effectuées sur $\mathtt{Sol}(\mathtt{srce})$ correspondent à une séquence de substitutions qui peuvent être utilisées pour transformer $\mathtt{srce}$ en $\mathtt{cible}$. Pour cette raison, une base de connaissances d'adaptation additionnelle $\mathtt{CA}$ est introduite. $\mathtt{CA}$ contient un ensemble de reformulations (γ, σ) qui peuvent servir à modéliser des stratégies d'adaptation plus complexes.

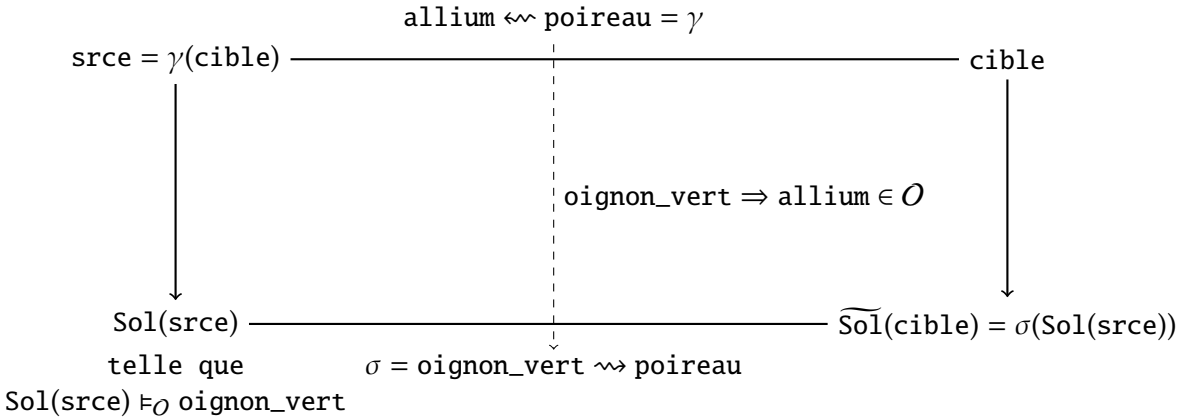


FIG. 2.2 – Génération automatique de la substitution σ = oignon_vert ⇝ poireau à partir de la substitution par généralisation γ = poireau ⇝ allium en utilisant l'axiome oignon_vert ⇒ allium de O.

2.5 Remémoration

L'algorithme de remémoration s'appuie sur une *classification élastique* dans une hiérarchie d'index [Lieber, 2002b]. Un tel algorithme a pour but de déterminer un ensemble de modifications à appliquer à cible de manière à obtenir une requête modifiée srce qui résolve au moins une recette Sol(srce) de la base de cas. L'algorithme construit un *chemin de similarité*, qui est une composition de substitutions ChSim = $\gamma_q \circ \gamma_{q-1} \circ \cdots \circ \gamma_1$ telle qu'il existe au moins une recette Sol(srce) qui satisfasse la requête modifiée srce = $\gamma_q(\gamma_{q-1}(\cdots \gamma_1(\text{cible})\cdots))$, c'est-à-dire qui vérifie Sol(srce) $\models_O$ srce. Un chemin de similarité ChSim peut donc s'écrire :

$$\text{Sol(srce)} \models_O \text{srce} \xleftarrow{\gamma_q} \xleftarrow{\gamma_{q-1}} \cdots \xleftarrow{\gamma_1} \text{cible}$$

Par exemple, pour résoudre la requête cible ci-dessus, le système génère un chemin de similarité ChSim = $\gamma_2 \circ \gamma_1$ avec :

$$\begin{aligned}
\text{cible} &= \text{soupe} \wedge \text{chinoise} \wedge \text{poireau} \wedge \neg\,\text{huile_arachide} \\
\gamma_1 = \neg\,\text{huile_arachide} &\rightsquigarrow \emptyset \qquad \gamma_2 = \text{poireau} \rightsquigarrow \text{allium} \\
\text{srce} &= \text{soupe} \wedge \text{chinoise} \wedge \text{allium} \\
\text{Sol(srce)} &= \text{soupe} \wedge \text{chinoise} \wedge \text{oignon_vert} \wedge \text{bok_choy} \wedge \text{huile_arachide} \wedge \textit{Rien d'autre}
\end{aligned}$$

Dans ce chemin de similarité, Sol(srce) est la représentation de la recette de la soupe Wonton. Comme l'ontologie O contient l'axiome oignon_vert ⇒ allium, la requête modifiée srce = $\gamma_2 \circ \gamma_1$ (cible) vérifie Sol(srce) $\models_O$ srce.

La modélisation du processus de raisonnement à partir de cas présentée ici diffère légèrement de celle qui a été présentée au chapitre 1. En effet, le problème srce n'est pas représenté tel quel dans la base de cas mais est choisi dynamiquement lors de la remémoration parmi l'ensemble des problèmes que Sol(srce) résoud. Le choix de srce s'effectue en fonction de cible, des connaissances d'adaptation et des coûts attribués lors de la remémoration aux différentes substitutions par généralisation γ applicables à cible.

2.6 Adaptation

À un chemin de similarité ChSim est associé un *chemin d'adaptation* ChAd. Un chemin d'adaptation est une composition de substitutions $\texttt{ChAd} = \sigma_1 \circ \sigma_2 \circ \cdots \circ \sigma_q$ telle que la recette modifiée $\widetilde{\texttt{Sol}}(\texttt{cible}) = \sigma_1(\sigma_2(\cdots\sigma_q(\texttt{Sol}(\texttt{srce}))\cdots))$ satisfait la requête cible initiale, c'est-à-dire vérifie $\widetilde{\texttt{Sol}}(\texttt{cible}) \vDash_O \texttt{cible}$. Un chemin d'adaptation peut donc s'écrire :

$$\texttt{Sol}(\texttt{srce}) \xrightarrow{\sigma_q} \xrightarrow{\sigma_{q-1}} \cdots \xrightarrow{\sigma_1} \widetilde{\texttt{Sol}}(\texttt{cible}) \vDash_O \texttt{cible}$$

Un chemin d'adaptation ChAd est construit à partir du chemin de similarité ChSim en associant une substitution σ_i à chaque substitution γ_i. Pour déterminer quelle substitution σ_i associer à une substitution γ_i, le système cherche d'abord dans la base d'adaptation additionnelle CA. Pour une substitution $\gamma_i = \alpha \rightsquigarrow \beta$, le système recherche dans CA une substitution $\sigma = \texttt{A} \rightsquigarrow \texttt{B}$ telle que $\texttt{A} \vDash_O \beta$ et $\texttt{B} \vDash_O \alpha$. Par exemple, si $\gamma_2 = \texttt{poireau} \rightsquigarrow \texttt{allium}$ est utilisé dans ChSim et CA contient la reformulation (γ, σ) avec $\gamma = \gamma_2$ et $\sigma = \texttt{oignon\_vert} \rightsquigarrow \texttt{poireau} \wedge \texttt{gingembre}$, σ sera sélectionnée pour constituer la substitution σ_2 dans ChAd car $\texttt{oignon\_vert} \vDash_O \texttt{allium}$ et $\texttt{poireau} \wedge \texttt{gingembre} \vDash_O \texttt{poireau}$. Si aucune substitution σ n'est trouvée dans CA pour une substitution γ_i donnée, alors σ_i est générée automatiquement à partir de γ_i.

Dans l'exemple précédent, CA est supposée vide donc σ_1 et σ_2 sont générées automatiquement à partir des substitutions γ_1 et γ_2 : $\sigma_1 = \emptyset \rightsquigarrow \neg\,\texttt{huile\_arachide}$ car $\gamma_1 = \neg\,\texttt{huile\_arachide} \rightsquigarrow \emptyset$ et $\sigma_2 = \texttt{allium} \rightsquigarrow \texttt{poireau}$ car $\gamma_2 = \texttt{poireau} \rightsquigarrow \texttt{allium}$. En utilisant l'axiome $\texttt{oignon\_vert} \Rightarrow \texttt{allium}$ de O, la substitution σ_i est spécialisée en la substitution $\texttt{oignon\_vert} \rightsquigarrow \texttt{poireau}$ et il est proposé à l'utilisateur de remplacer les oignons verts par des poireaux dans la recette de la soupe Wonton et d'enlever l'huile. Le chemin d'adaptation généré est $\texttt{ChAd} = \sigma_1 \circ \sigma_2$ (figure 2.3), avec :

$$\begin{aligned}
\texttt{Sol}(\texttt{srce}) &= \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{oignon_vert} \wedge \texttt{bok_choy} \wedge \texttt{huile_arachide} \wedge \textit{Rien d'autre} \\
\sigma_2 = \texttt{oignon_vert} \rightsquigarrow \texttt{poireau} &\qquad \sigma_1 = \emptyset \rightsquigarrow \neg\,\texttt{huile_arachide} \\
\widetilde{\texttt{Sol}}(\texttt{cible}) &= \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{poireau} \wedge \texttt{bok_choy} \wedge \textit{Rien d'autre} \\
\texttt{cible} &= \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{poireau} \wedge \neg\,\texttt{huile_arachide}
\end{aligned}$$

La solution $\widetilde{\texttt{Sol}}(\texttt{cible})$ inférée satisfait la requête initiale cible : $\widetilde{\texttt{Sol}}(\texttt{cible}) \vDash_O \texttt{cible}$.

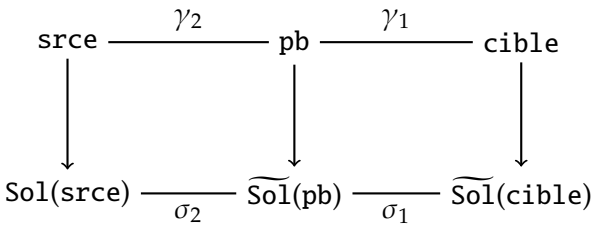


Fig. 2.3 – Un chemin de similarité et le chemin d'adaptation associé.

2.7 Pourquoi acquérir des connaissances d'adaptation dans Taaable ?

Dans la version du système Taaable qui a été proposé pour participer au *Computer Cooking Contest*, $\texttt{CA} = \emptyset$ donc les connaissances d'adaptation sont inférées uniquement à partir de l'ontologie O. Le principal avantage de cette approche est sa simplicité : le système acquiert des

connaissances d'adaptation « à la demande » à partir de ses connaissances du domaine et est capable de proposer une solution à tout problème cible. Néanmoins, les capacités d'adaptation du système sont très limitées (de simples substitutions) et l'utilisateur n'a aucun moyen d'intervenir pour donner son avis sur la qualité de l'adaptation proposée.

Par exemple, la substitution $\sigma_1 = \emptyset \rightsquigarrow \neg$ `huile_arachide` suggère de supprimer l'huile d'arachide dans la recette remémorée, or l'huile est utilisée dans la recette pour sauter le bok choy. La recette obtenue devient pratiquement infaisable. Une meilleure adaptation consisterait à remplacer l'huile d'arachide par un autre type d'huile comme par exemple l'huile de sésame. Cette adaptation peut être modélisée par la substitution $\sigma_1 =$ `huile_arachide` $\rightsquigarrow$ `huile_sésame`. Pour générer cette substitution automatiquement à partir de l'ontologie, le système pourrait effectuer une recherche locale dans la hiérarchie d'ingrédients, à la manière de ce qui est fait dans le système PLEXUS [Alterman, 1986]. Le système pourrait exploiter le fait que les concepts `huile_arachide` et `huile_sésame` sont tous deux des sous-concepts du concept `huile_végetale` dans $\mathcal{O}$. Mais alors, des connaissances supplémentaires seraient nécessaires au système pour exprimer le fait que l'huile d'arachide devrait être remplacée par de l'huile de sésame et non par de l'huile d'olive ou de l'huile de tournesol, car `huile_olive` et `huile_tournesol` sont également deux sous-concepts de `huile_végetale` dans $\mathcal{O}$. De plus, il serait souhaitable de pourvoir le système de la connaissance selon laquelle cette substitution n'est valable que dans la cuisine asiatique, ce qui est modélisé par la substitution $\sigma_1 =$ `asiatique` $\wedge$ `huile_arachide` $\rightsquigarrow$ `asiatique` $\wedge$ `huile_sésame`.

Par ailleurs, la deuxième substitution $\sigma_2 =$ `oignon_vert` $\rightsquigarrow$ `poireau` suggère de remplacer les oignons verts émincés dans la recette par du poireau cru. Mais les oignons verts sont utilisés dans la recette de la soupe Wonton pour la garniture, donc l'adaptation proposée est susceptible de déplaire fortement à l'utilisateur car les poireaux crus altèrent fortement le goût de la soupe. Une meilleure adaptation consisterait à frire les poireaux avec du tempeh et des poivrons rouges pour préparer la garniture. Cette adaptation peut être modélisée par la substitution $\sigma_2 =$ `oignon_vert` $\rightsquigarrow$ `poireau` $\wedge$ `tempeh` $\wedge$ `poivron_rouge`. Cette substitution, qui traduit un savoir-faire culinaire, ne peut pas être générée automatiquement à partir de l'ontologie.

Ces exemples montrent l'utilité pour améliorer les capacités d'adaptation du système de disposer d'un ensemble de règles d'adaptation représentant des stratégies d'adaptation plus complexes. Ces exemples montrent aussi que le rôle de l'expert humain est primordial dans l'acquisition de connaissances d'adaptation et que dans le domaine culinaire, les règles d'adaptation sont souvent très contextuelles.

Dans ce contexte, continuer à réaliser l'acquisition de connaissances d'adaptation de manière automatique à partir des connaissances du domaine supposerait d'enrichir les connaissances du domaine du système. Une autre direction de recherche, privilégiée dans cette thèse, consiste à utiliser d'autres sources de connaissances d'adaptation pour TAAABLE que les connaissances du domaine. En particulier, le chapitre 5 montre comment la base de cas peut être utilisée comme une source additionnelle de connaissances d'adaptation pour le système. La méthode proposée au chapitre 5 pour l'apprentissage de connaissances d'adaptation à partir de la base de cas est ensuite combinée dans le chapitre 7 avec une méthode d'acquisition de connaissances interactive auprès d'utilisateurs. Ainsi, le système bénéficie de deux sources de connaissances supplémentaires (la base de cas et l'utilisateur).

3

Représentation des variations entre cas

Dans ce chapitre est défini un langage générique de représentation des variations entre cas. Le langage proposé est d'abord appliqué à un formalisme de représentation des cas de type attribut-contrainte, qui généralise le formalisme attribut-valeur fréquemment utilisé en raisonnement à partir de cas. Nous montrons comment ce langage peut être utilisé pour représenter les variations entre cas dans l'application Taaable.

❄

[Badra et Lieber, 2008] F. Badra et J. Lieber. Representing Case Variations for Learning General and Specific Adaptation Rules. Dans Amedeo Cesta et Nikos Fakotakis, rédacteurs, *Fourth Starting AI Researcher's Symposium (STAIRS 2008)*, p. 1–11. IOS Press, Patras Grèce, 2008.

[Badra et Lieber, 2007b] F. Badra et J. Lieber. Une approche pour représenter les variations entre cas – Vers une application à l'extraction de connaissances d'adaptation. Dans Amélie Cordier, rédacteur, *15ème atelier sur le raisonnement à partir de cas - RàPC-07*, p. 47–56. Grenoble France, 2007.

3.1 Définition d'un langage de représentation des variations entre cas

Dans cette section est défini un langage générique $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ de représentation des variations entre cas, où $\mathcal{L}_{\Delta\mathtt{pb}}$ est un langage de représentation des variations entre problèmes et $\mathcal{L}_{\Delta\mathtt{sol}}$ est un langage de représentation des variations entre solutions.

Définition de $\mathcal{L}_{\Delta\mathtt{pb}}$. Dans $\mathcal{L}_{\Delta\mathtt{pb}}$, une variation entre problèmes est la représentation $\mathtt{r}$ d'une relation binaire entre problèmes. Dans ce langage, la sémantique d'une relation $\mathtt{r} \in \mathcal{L}_{\mathtt{pb}}$ est donnée par son extension :

$$\mathrm{Ext}(\mathtt{r}) \subseteq \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}}$$

$\mathcal{L}_{\Delta\mathtt{pb}}$ contient la relation $\top_{\Delta\mathtt{pb}}$ et la relation $\langle \mathtt{srce}, \mathtt{cible} \rangle$ pour chaque couple $(\mathtt{srce}, \mathtt{cible})$ de $\mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}}$. La sémantique de ces relations est donnée par :

$$\mathrm{Ext}(\top_{\Delta\mathtt{pb}}) = \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}}$$
$$\mathrm{Ext}(\langle \mathtt{srce}, \mathtt{cible} \rangle) = \{(\mathtt{srce}, \mathtt{cible})\}$$

Soit $\vDash$ la relation d'inférence sur $\mathcal{L}_{\Delta\mathtt{pb}}$ (si $\mathtt{r}, \mathtt{s} \in \mathcal{L}_{\Delta\mathtt{pb}}$, $\mathtt{r} \vDash \mathtt{s}$ ssi $\mathrm{Ext}(\mathtt{r}) \subseteq \mathrm{Ext}(\mathtt{s})$) et $\equiv$ la relation d'équivalence sur $\mathcal{L}_{\Delta\mathtt{pb}}$ ($\mathtt{r} \equiv \mathtt{s}$ ssi $\mathtt{r} \vDash \mathtt{s}$ et $\mathtt{s} \vDash \mathtt{r}$). $\vDash$ (resp., $\equiv$) est supposée être calculable. Pour $\mathtt{r} \in \mathcal{L}_{\Delta\mathtt{pb}}$, la *clôture déductive* de $\mathtt{r}$ est l'ensemble

$$\mathtt{Clôture}(\mathtt{r}) = \{\mathtt{s} \in \mathcal{L}_{\Delta\mathtt{pb}} \mid \mathtt{r} \vDash \mathtt{s}\}$$

Définition de $\mathcal{L}_{\Delta\mathtt{sol}}$. La représentation des variations entre solutions est réalisée de manière similaire : $\mathcal{L}_{\Delta\mathtt{sol}}$ est le langage de représentation des variations entre solutions qui contient $\top_{\Delta\mathtt{sol}}$ et $\langle \mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible}) \rangle$; $\vDash$, $\equiv$ et $\mathtt{Clôture}$ sont définis de manière similaire sur $\mathcal{L}_{\Delta\mathtt{sol}}$.

Représentation des variations entre cas. Le langage de représentation des variations entre cas est $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$. La relation d'inférence $\vDash$ définie sur $\mathcal{L}_{\Delta\mathtt{pb}}$ et sur $\mathcal{L}_{\Delta\mathtt{sol}}$ est étendue à $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ de la façon suivante : pour $(\mathtt{r}_1, \mathtt{R}_1), (\mathtt{r}_2, \mathtt{R}_2) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$, $(\mathtt{r}_1, \mathtt{R}_1) \vDash (\mathtt{r}_2, \mathtt{R}_2)$ ssi $\mathtt{r}_1 \vDash \mathtt{r}_2$ et $\mathtt{R}_1 \vDash \mathtt{R}_2$. Etant donné un élément $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$, la clôture déductive de $(\mathtt{r}, \mathtt{R})$ est l'ensemble $\mathtt{Clôture}((\mathtt{r}, \mathtt{R})) = \{(\mathtt{r}', \mathtt{R}') \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}} \mid (\mathtt{r}, \mathtt{R}) \vDash (\mathtt{r}', \mathtt{R}')\}$.

Hypothèse de symétrie. L'hypothèse de symétrie est optionnelle. Elle consiste à supposer que pour tout $\mathtt{r} \in \mathcal{L}_{\Delta\mathtt{pb}}$, il existe une relation $\mathtt{r}^{-1} \in \mathcal{L}_{\Delta\mathtt{pb}}$ telle que, pour chaque $(\mathtt{srce}, \mathtt{cible}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{pb}}$, $(\mathtt{srce}, \mathtt{cible}) \in \mathrm{Ext}(\mathtt{r})$ ssi $(\mathtt{cible}, \mathtt{srce}) \in \mathrm{Ext}(\mathtt{r}^{-1})$. En particulier, ${\top_{\Delta\mathtt{pb}}}^{-1} \equiv \top_{\Delta\mathtt{pb}}$ et $\langle \mathtt{srce}, \mathtt{cible} \rangle^{-1} \equiv \langle \mathtt{cible}, \mathtt{srce} \rangle$. Une hypothèse similaire peut être faite sur $\mathcal{L}_{\Delta\mathtt{sol}}$.

Choix des relations utilisées dans $\mathcal{L}_{\Delta\mathtt{pb}}$ et $\mathcal{L}_{\Delta\mathtt{sol}}$. Dans cette formalisation, les langages $\mathcal{L}_{\Delta\mathtt{pb}}$ et $\mathcal{L}_{\Delta\mathtt{sol}}$ sont définis par la donnée d'un ensemble de relations binaires entre problèmes et entre solutions. Néanmoins toute relation binaire n'est pas pertinente pour résoudre un type de problème donné. Le choix des relations à prendre en compte dépend du domaine d'application et doit être réalisé en concertation avec l'expert du domaine. Ce choix constitue donc une tâche d'acquisition de connaissances auprès d'experts.

3.2 Application à un formalisme attribut-contrainte

Dans cette section, le langage générique qui a été défini à la section 3.1 pour représenter les variations entre cas est appliqué dans le cas où les problèmes et les solutions sont représentés dans un formalisme attribut-contrainte qui étend le formalisme attribut-valeur fréquemment utilisé en raisonnement à partir de cas [Bergmann *et al.*, 2005].

Représentation des cas. Une instance de problème est décrite par les valeurs qu'elle prend pour certains attributs. Les problèmes sont définis par la spécification des ensembles de valeurs que peuvent prendre ces attributs. Par exemple, dans le domaine culinaire, le problème

$$\texttt{pb} = (\texttt{pommes-qté}, \{3\}) \wedge (\texttt{sucre-qté}, [500; +\infty[)$$

représente la classe des recettes qui contiennent 3 pommes et plus de 500 grammes de sucre.

Plus formellement, une *instance de problème* est un élément du produit cartésien $\texttt{Inst}_{\texttt{pb}} = \texttt{V}_1 \times \cdots \times \texttt{V}_m$, où $\texttt{V}_i$ est un ensemble de valeurs. Dans les exemples, $\texttt{V}_i$ est pris parmi les ensembles $\mathbb{R}$ (l'ensemble des nombres réels), $\mathbb{Z}$ (l'ensemble des entiers relatifs), $\mathbb{B} = \{\texttt{vrai}, \texttt{faux}\}$ (l'ensemble des booléens) ou un ensemble énuméré (par exemple {rouge,vert,bleu}). L'*attribut* $\texttt{a}_i$ ($i \in \{1, 2, \ldots, m\}$) est la $i^{\text{ème}}$ projection de $\texttt{Inst}_{\texttt{pb}}$ sur $\texttt{V}_i$:

$$\texttt{a}_i(x_1, \ldots, x_m) = x_i$$

À chaque $\texttt{a}_i$ est associé un langage de contraintes $\mathcal{L}_{\texttt{a}_i}$: $\texttt{C} \in \mathcal{L}_{\texttt{a}_i}$ est interprété comme un sous-ensemble Ext(C) de $\texttt{V}_i$. Par exemple, la contrainte $[50, +\infty[$ établie sur $\texttt{V}_i = \mathbb{Z}$ représente l'ensemble des entiers relatifs qui sont supérieurs ou égaux à 50. $\mathcal{L}_{\texttt{a}_i}$ est supposé (1) contenir les contraintes $\top_i$ et $\bot_i$ telles que $\text{Ext}(\top_i) = \texttt{V}_i$ et $\text{Ext}(\bot_i) = \emptyset$, (2) être clos pour la conjonction (si $\texttt{C}, \texttt{D} \in \mathcal{L}_{\texttt{a}_i}$ alors il existe $\texttt{E} \in \mathcal{L}_{\texttt{a}_i}$ tel que $\text{Ext}(\texttt{C}) \cap \text{Ext}(\texttt{D}) = \text{Ext}(\texttt{E})$). De plus, on suppose que deux contraintes distinctes ne peuvent dénoter le même ensemble d'objets : si $\text{Ext}(\texttt{C}) = \text{Ext}(\texttt{D})$ alors $\texttt{C} = \texttt{D}$.

Un *descripteur de problèmes* est un couple $\texttt{d} = (\texttt{a}_i, \texttt{C})$ dans lequel $i \in \{1, \ldots, m\}$ et $\texttt{C} \in \mathcal{L}_{\texttt{a}_i}$. Un descripteur de problèmes est interprété comme un ensemble d'instances de problèmes dont la valeur pour le $i^{\text{ème}}$ attribut $\texttt{a}_i$ est dans l'extension de C :

$$\text{Ext}((\texttt{a}_i, \texttt{C})) = \texttt{a}_i^{-1}(\text{Ext}(\texttt{C})) = \{x \in \texttt{Inst}_{\texttt{pb}} \mid \texttt{a}_i(x) \in \text{Ext}(\texttt{C})\}$$

Finalement, un problème est une expression de la forme $\texttt{pb} = \texttt{d}_1 \wedge \cdots \wedge \texttt{d}_p$ où les $\texttt{d}_i$ sont des descripteurs de problèmes. Une conjonction est interprétée par une intersection :

$$\text{Ext}(\texttt{d}_1 \wedge \cdots \wedge \texttt{d}_p) = \text{Ext}(\texttt{d}_1) \cap \ldots \cap \text{Ext}(\texttt{d}_p)$$

Un problème pb est satisfiable si $\text{Ext}(\texttt{pb}) \neq \emptyset$. Un problème pb est mis sous forme normale si $\texttt{pb} = \texttt{d}_1 \wedge \cdots \wedge \texttt{d}_m$ où, pour chaque $i \in \{1, \ldots, m\}$, il existe un $\texttt{C} \in \mathcal{L}_{\texttt{a}_i}$ tel que $\texttt{d}_i = (\texttt{a}_i, \texttt{C})$. Il peut être montré qu'un problème mis sous forme normale est satisfiable ssi il ne contient aucune occurrence de $\bot_i$. De plus, tout problème pb satisfiable est équivalent à un unique problème pb' mis en forme normale : $\text{Ext}(\texttt{pb}) = \text{Ext}(\texttt{pb}')$. Si $\texttt{pb} = (\texttt{a}_1, \texttt{C}_1) \wedge \cdots \wedge (\texttt{a}_m, \texttt{C}_m)$ est satisfiable et mis sous forme normale, alors $\texttt{a}_i(\text{Ext}(\texttt{pb})) = \text{Ext}(\texttt{C}_i)$, ce qui justifie la notation $\texttt{a}_i(\texttt{pb}) = \texttt{C}_i$. Pour des besoins de simplicité, un descripteur de problème de la forme $(\texttt{a}_i, \top_i)$ pourra être omis dans la notation : $\texttt{pb} = (\texttt{a}_1, \texttt{C}_1) \wedge (\texttt{a}_2, \top_2) = (\texttt{a}_1, \texttt{C}_1)$.

Les solutions sont définies de manière analogue par un ensemble d'attributs $\texttt{A}_j$ avec $j \in \{1, \ldots, n\}$.

Représentation des variations entre cas. Dans cette section, les problèmes et les solutions sont supposées être satisfiables et mis en forme normale. Le langage $\mathcal{L}_{\Delta\mathtt{pb}}$ de représentation des variations entre problèmes est défini par cinq constructeurs. Les constructeurs $\top_{\Delta\mathtt{pb}}$, $\bot_{\Delta\mathtt{pb}}$ et $\langle \mathtt{srce}, \mathtt{cible} \rangle$ ont déjà été définis dans le cadre général (cf. section 3.1). Le quatrième constructeur est la conjonction : si $\mathtt{r}, \mathtt{s} \in \mathcal{L}_{\Delta\mathtt{pb}}$ alors $\mathtt{r} \wedge \mathtt{s} \in \mathcal{L}_{\Delta\mathtt{pb}}$ et $(\mathtt{pb}_1, \mathtt{pb}_2) \in \mathrm{Ext}(\mathtt{r} \wedge \mathtt{s})$ ssi $(\mathtt{pb}_1, \mathtt{pb}_2) \in \mathrm{Ext}(\mathtt{r})$ et $(\mathtt{pb}_1, \mathtt{pb}_2) \in \mathrm{Ext}(\mathtt{s})$. Le dernier constructeur est $\mathtt{a}_i^\delta$, où $\mathtt{a}_i$ est un attribut et δ représente une relation binaire sur $\mathcal{L}_{\mathtt{a}_i}$; δ est choisi dans un langage $\mathcal{L}_{\Delta\mathtt{a}_i}$. La sémantique de $\mathtt{a}_i^\delta$ est la suivante : deux problèmes `srce` et `cible` sont reliés par $\mathtt{a}_i^\delta$ si $\mathtt{a}_i(\mathtt{srce})$ et $\mathtt{a}_i(\mathtt{cible})$ sont reliés par δ. Plus formellement :

$$\mathrm{Ext}(\mathtt{a}_i^\delta) = \{(\mathtt{srce}, \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid (\mathtt{a}_i(\mathtt{srce}), \mathtt{a}_i(\mathtt{cible})) \in \mathrm{Ext}(\delta)\}$$

Par exemple, considérons le formalisme $\mathcal{L}_{\mathtt{pb}}$ fondé sur les attributs $\mathtt{a}_1 = \mathtt{type\text{-}de\text{-}plat}$ et $\mathtt{a}_2 = \mathtt{pommes\text{-}qté}$ ($\mathtt{V}_1 = \{\mathtt{tarte}, \mathtt{confiture}\}$ et $\mathtt{V}_2 = \mathbb{Z}$). Tous les $\mathcal{L}_{\Delta\mathtt{a}_i}$ partagent les relations $=$ et $\neq$. Ce formalisme est supposé être un formalisme attribut-*valeur*, dans le sens où $\mathcal{L}_{\Delta\mathtt{a}_i}$ contient seulement $\top_i$, $\bot_i$, et les singletons $\{x\}$, pour $x \in \mathtt{V}_i$. $\mathcal{L}_{\Delta\mathtt{pommes\text{-}qté}}$ contient les relations $\delta \in \{<, \leqslant, \geqslant, >\}$ sur les singletons définies par $\{x\}\ \delta\ \{y\}$ si $x\ \delta\ y$ (par exemple, $\{3\} < \{4\}$ car $3 < 4$). Si

$$\begin{aligned} \mathtt{srce} &= (\mathtt{type\text{-}de\text{-}plat}, \{\mathtt{tarte}\}) \wedge (\mathtt{pommes\text{-}qté}, \{3\}) \\ \mathtt{cible} &= (\mathtt{type\text{-}de\text{-}plat}, \{\mathtt{tarte}\}) \wedge (\mathtt{pommes\text{-}qté}, \{5\}) \end{aligned}$$

sont deux problèmes décrits dans ce formalisme, alors

$$\langle \mathtt{srce}, \mathtt{cible} \rangle \vDash \mathtt{type\text{-}de\text{-}plat}^{=} \wedge \mathtt{pommes\text{-}qté}^{\neq} \wedge \mathtt{pommes\text{-}qté}^{<} \wedge \mathtt{pommes\text{-}qté}^{\leqslant}$$

$\mathcal{L}_{\Delta\mathtt{sol}}$ est défini à l'aide de constructeurs similaires.

Exemples de relations δ. La définition de $\mathcal{L}_{\Delta\mathtt{pb}}$ a été réduite ci-dessus à la définition d'un $\mathcal{L}_{\Delta\mathtt{a}_i}$ pour chaque attribut $\mathtt{a}_i$. Bien que la définition de $\mathcal{L}_{\Delta\mathtt{a}_i}$ soit un problème d'acquisition de connaissances, quelques exemples de relations $\delta \in \mathcal{L}_{\Delta\mathtt{a}_i}$ qui peuvent être utiles sont présentés ici.

Lorsqu'à un ensemble $\mathtt{V}_i$ est associée une structure algébrique, cette dernière peut être réutilisée sur les singletons de l'ensemble. Par exemple, la relation d'ordre $\leqslant$ sur $\mathbb{Z}$ peut être utilisée comme une relation entre singletons d'entiers, comme cela a déjà été mentionné dans l'exemple précédent ($\{3\} < \{5\}$). Un autre exemple est lié à la loi $+$ sur $\mathbb{Z}$, qui peut être utilisée pour définir la relation binaire $\delta = \mathtt{ajouter}(\alpha)$ sur $\mathbb{Z}$ (pour chaque $\alpha \in \mathbb{Z}$) : $x\ \mathtt{ajouter}(\alpha)\ y$ si $x + \alpha = y$.

Dans la mesure où δ relie deux (representations d')ensembles, les relations binaires classiques entre ensembles ($\subsetneq$,$\subseteq$,$=$,$\supseteq$,$\supsetneq$) peuvent être suggérées pour les éléments de $\mathcal{L}_{\Delta\mathtt{a}_i}$. De plus, soit la relation ($\subseteq\mathtt{C}\supseteq$) définie, pour $\mathtt{C}_i \in \mathcal{L}_{\Delta\mathtt{a}_i}$ par $\mathtt{C}_1$ ($\subseteq\mathtt{C}\supseteq$) $\mathtt{C}_2$ si $\mathrm{Ext}(\mathtt{C}_1) \subseteq \mathrm{Ext}(\mathtt{C}) \supseteq \mathrm{Ext}(\mathtt{C}_2)$: $\mathtt{C}_1$ et $\mathtt{C}_2$ partagent la contrainte $\mathtt{C}$. Une autre relation est $\ominus$ (resp., $\oplus$) définies par $\mathtt{C} \ominus \mathtt{D}$ si $\mathtt{C} \neq \top_i$ et $\mathtt{D} = \top_i$ (resp., $\mathtt{C} = \top_i$ et $\mathtt{D} \neq \top_i$). Notons que $\mathtt{a}_i^{\ominus} \vDash \mathtt{a}_i^{\subsetneq}$ et $\mathtt{a}_i^{\oplus} \vDash \mathtt{a}_i^{\supsetneq}$. Ces relations peuvent être appliquées en particulier quand $\mathcal{L}_{\Delta\mathtt{a}_i} = 2^{\mathtt{V}_i}$, où $\mathtt{V}_i$ est un ensemble énuméré. Elles peuvent également être appliquées lorsque $\mathcal{L}_{\Delta\mathtt{a}_i}$ est un ensemble fini de contraintes atomiques organisées en une hiérarchie de racine $\top_i$. Finalement, elles peuvent être appliquées à des intervalles, par exemple de $\mathbb{Z}$ ou $\mathbb{R}$.

D'autres relations entre intervalles peuvent être définies en réutilisant les relations de Allen sur des intervales temporels [Allen, 1983]. Par exemple, si $\mathtt{C}_1 = [x_1, y_1]$ et $\mathtt{C}_2 = [x_2, y_2]$ sont

deux intervalles clos sur $\mathbb{R}$, $\mathtt{C}_1\ b\ \mathtt{C}_2$ si $y_1 < x_2$, $a = b^{-1}$, $\mathtt{C}_1\ m\ \mathtt{C}_2$ si $y_1 = x_2$, etc. (b, a, et m sont les initiales des termes anglais *before*, *after*, et *meets*). Ces relations *qualitatives* peuvent être complétées par des relations *quantitative* comme $\mathtt{ajouter\text{-}aux\text{-}bornes}(\alpha;\beta)$ définies par $\mathtt{C}_1\ \mathtt{ajouter\text{-}aux\text{-}bornes}(\alpha;\beta)\ \mathtt{C}_2$ si $x_1 + \alpha = x_2$ et $y_1 + \beta = y_2$.

3.3 Application au formalisme utilisé dans TAAABLE

Dans cette section, le langage générique qui a été défini à la section 3.1 pour représenter les variations entre cas est appliqué au formalisme de représentation des cas de TAAABLE.

Représentation des cas. Le formalisme utilisé pour représenter les cas dans TAAABLE est celui décrit à la section 2.2 page 35.

Représentation des variations entre problèmes. Le langage $\mathcal{L}_{\Delta\mathtt{pb}}$ de représentation des variations entre problèmes est défini à l'aide des relations $\top_{\Delta\mathtt{pb}}$ et $\langle \mathtt{srce}, \mathtt{cible} \rangle$ définies dans la section 3.1 ainsi que de la conjonction $\wedge$: si $\mathtt{r}, \mathtt{s} \in \mathcal{L}_{\Delta\mathtt{pb}}$ alors $\mathtt{r} \wedge \mathtt{s} \in \mathcal{L}_{\Delta\mathtt{pb}}$ et $(\mathtt{pb}_1, \mathtt{pb}_2) \in \mathrm{Ext}(\mathtt{r} \wedge \mathtt{s})$ ssi $(\mathtt{pb}_1, \mathtt{pb}_2) \in \mathrm{Ext}(\mathtt{r})$ et $(\mathtt{pb}_1, \mathtt{pb}_2) \in \mathrm{Ext}(\mathtt{s})$. À ces constructeurs s'ajoutent pour chaque variable propositionnelle $\mathtt{a}$ de $\mathcal{V}$ six relations $\mathtt{a}^-$, $\mathtt{a}^+$, $\mathtt{a}^=$, $(\neg\,\mathtt{a})^-$, $(\neg\,\mathtt{a})^+$, $(\neg\,\mathtt{a})^=$ dont la sémantique est donnée par :

$$\begin{aligned}
\mathrm{Ext}(\mathtt{a}^-) &= \{(x,y) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid x \vDash_O \mathtt{a} \text{ and } y \nvDash_O \mathtt{a}\} \\
\mathrm{Ext}(\mathtt{a}^+) &= \{(x,y) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid x \nvDash_O \mathtt{a} \text{ and } y \vDash_O \mathtt{a}\} \\
\mathrm{Ext}(\mathtt{a}^=) &= \{(x,y) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid x \vDash_O \mathtt{a} \text{ and } y \vDash_O \mathtt{a}\} \\
\mathrm{Ext}((\neg\,\mathtt{a})^-) &= \{(x,y) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid x \vDash_O \neg\,\mathtt{a} \text{ and } y \nvDash_O \neg\,\mathtt{a}\} \\
\mathrm{Ext}((\neg\,\mathtt{a})^+) &= \{(x,y) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid x \nvDash_O \neg\,\mathtt{a} \text{ and } y \vDash_O \neg\,\mathtt{a}\} \\
\mathrm{Ext}((\neg\,\mathtt{a})^=) &= \{(x,y) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{pb}} \mid x \vDash_O \neg\,\mathtt{a} \text{ and } y \vDash_O \neg\,\mathtt{a}\}
\end{aligned}$$

Par exemple, si

$$\begin{aligned}
\mathtt{srce} &= \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{allium} \\
\mathtt{cible} &= \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{poireau} \wedge \neg\,\mathtt{huile_arachide}
\end{aligned}$$

sont deux problèmes de $\mathcal{L}_{\mathtt{pb}}$ et si $\mathtt{poireau} \vDash_O \mathtt{allium}$ alors

$$\langle \mathtt{srce}, \mathtt{cible} \rangle = \mathtt{soupe}^= \wedge \mathtt{chinoise}^= \wedge \mathtt{allium}^= \wedge \mathtt{poireau}^+ \wedge (\neg\,\mathtt{huile\_arachide})^+$$

Une variation $\mathtt{r} \in \mathcal{L}_{\Delta\mathtt{pb}}$ est interprétée comme une substitution de la forme $\alpha \rightsquigarrow \beta$, où α et β sont des conjonctions de littéraux tels que :

$$\begin{cases}
\alpha \vDash_O \mathtt{a} \text{ et } \beta \nvDash_O \mathtt{a} \text{ si } \mathtt{r} \vDash \mathtt{a}^- \\
\alpha \nvDash_O \mathtt{a} \text{ et } \beta \vDash_O \mathtt{a} \text{ si } \mathtt{r} \vDash \mathtt{a}^+ \\
\alpha \vDash_O \mathtt{a} \text{ et } \beta \vDash_O \mathtt{a} \text{ si } \mathtt{r} \vDash \mathtt{a}^= \\
\alpha \vDash_O \neg\,\mathtt{a} \text{ et } \beta \nvDash_O \neg\,\mathtt{a} \text{ si } \mathtt{r} \vDash (\neg\,\mathtt{a})^- \\
\alpha \nvDash_O \neg\,\mathtt{a} \text{ et } \beta \vDash_O \neg\,\mathtt{a} \text{ si } \mathtt{r} \vDash (\neg\,\mathtt{a})^+ \\
\alpha \vDash_O \neg\,\mathtt{a} \text{ et } \beta \vDash_O \neg\,\mathtt{a} \text{ si } \mathtt{r} \vDash (\neg\,\mathtt{a})^=
\end{cases}$$

Par exemple, la variation $\langle \texttt{srce}, \texttt{cible} \rangle$ entre les problèmes `srce` et `cible` ci-dessus s'interprète comme la substitution $\alpha \rightsquigarrow \beta$ avec :

$$\alpha = \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{allium}$$
$$\beta = \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{allium} \wedge \texttt{poireau} \wedge \neg\, \texttt{huile\_arachide}$$

Représentation des variations entre solutions. $\mathcal{L}_{\Delta\texttt{sol}}$ est défini à l'aide de constructeurs similaires, mais en utilisant trois relations $\texttt{a}^-$, $\texttt{a}^+$ et $\texttt{a}^=$ dont la sémantique est donnée par :

$$\begin{aligned} \text{Ext}(\texttt{a}^-) &= \{(x,y) \in \mathcal{L}_{\texttt{sol}} \times \mathcal{L}_{\texttt{sol}} \mid x \vDash_O \texttt{a} \text{ and } y \nvDash_O \texttt{a}\} \\ \text{Ext}(\texttt{a}^+) &= \{(x,y) \in \mathcal{L}_{\texttt{sol}} \times \mathcal{L}_{\texttt{sol}} \mid x \nvDash_O \texttt{a} \text{ and } y \vDash_O \texttt{a}\} \\ \text{Ext}(\texttt{a}^=) &= \{(x,y) \in \mathcal{L}_{\texttt{sol}} \times \mathcal{L}_{\texttt{sol}} \mid x \vDash_O \texttt{a} \text{ and } y \vDash_O \texttt{a}\} \end{aligned} \tag{3.1}$$

Par exemple, si

$$\begin{aligned} \texttt{Sol}(\texttt{srce}) &= \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{oignon\_vert} \wedge \texttt{bok\_choy} \wedge \texttt{huile\_arachide} \wedge \textit{Rien d'autre} \\ \texttt{Sol}(\texttt{cible}) &= \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{poireau} \wedge \texttt{bok\_choy} \wedge \textit{Rien d'autre} \end{aligned}$$

sont deux solutions de $\mathcal{L}_{\texttt{sol}}$ et si $\texttt{oignon\_vert} \vDash_O \texttt{allium}$ et $\texttt{poireau} \vDash_O \texttt{allium}$ alors

$$\begin{aligned} \langle \texttt{Sol}(\texttt{srce}), \texttt{Sol}(\texttt{cible}) \rangle = \texttt{soupe}^= \wedge \texttt{chinoise}^= \wedge \texttt{allium}^= \wedge \texttt{oignon\_vert}^- \wedge \texttt{poireau}^+ \\ \wedge\, \texttt{bok\_choy}^= \wedge \texttt{huile\_arachide}^- \end{aligned}$$

Une variation $\texttt{R} \in \mathcal{L}_{\Delta\texttt{sol}}$ est interprétée comme une substitution de la forme $\texttt{A} \rightsquigarrow \texttt{B}$, où `A` et `B` sont des conjonctions de littéraux tels que :

$$\begin{cases} \texttt{A} \vDash_O \texttt{a} \text{ et } \texttt{B} \nvDash_O \texttt{a} \text{ si } \texttt{R} \vDash \texttt{a}^- \\ \texttt{A} \nvDash_O \texttt{a} \text{ et } \texttt{B} \vDash_O \texttt{a} \text{ si } \texttt{R} \vDash \texttt{a}^+ \\ \texttt{A} \vDash_O \texttt{a} \text{ et } \texttt{B} \vDash_O \texttt{a} \text{ si } \texttt{R} \vDash \texttt{a}^= \end{cases} \tag{3.2}$$

Par exemple, la variation $\langle \texttt{Sol}(\texttt{srce}), \texttt{Sol}(\texttt{cible}) \rangle$ entre les solutions $\texttt{Sol}(\texttt{srce})$ et $\texttt{Sol}(\texttt{cible})$ ci-dessus s'interprète comme la substitution $\texttt{A} \rightsquigarrow \texttt{B}$ avec :

$$\texttt{A} = \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{oignon\_vert} \wedge \texttt{bok\_choy} \wedge \texttt{huile\_arachide}$$
$$\texttt{B} = \texttt{soupe} \wedge \texttt{chinoise} \wedge \texttt{poireau} \wedge \texttt{bok\_choy}$$

Le littéral `allium` n'est présent ni dans `A` ni dans `B` car il est inutile : $\texttt{oignon\_vert} \vDash_O \texttt{allium}$ et $\texttt{poireau} \vDash_O \texttt{allium}$.

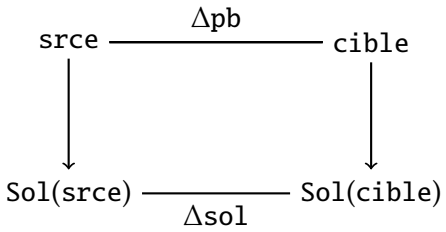
4

Application à la modélisation de l'adaptation

Dans ce chapitre, nous montrons que l'introduction d'un langage de représentation des variations entre cas permet d'aboutir à une nouvelle modélisation de l'étape d'adaptation en raisonnement à partir de cas. Nous montrons que se donner un tel langage permet de représenter dans un même formalisme les similarités et dissimilarités entre cas et les connaissances d'adaptation utilisées dans le raisonnement pour les résoudre. Dans cette modélisation, tout élément du langage de représentation des variations entre cas peut s'interpréter comme une règle d'adaptation. Nous montrons comment cette nouvelle modélisation de l'adaptation peut être appliquée pour représenter les connaissances d'adaptation dans Taaable.

Nous nous placerons dans cette thèse dans le cadre d'une approche transformationnelle de l'adaptation. Dans une telle approche, l'adaptation consiste à résoudre les différences qui existent entre le problème cible et un problème remémoré [Hanney *et al.*, 1995]. Une décomposition de l'adaptation est proposée qui s'appuie sur une représentation des variations entre problèmes et entre solutions.

4.1 Décomposition de l'adaptation



① Δ: (srce, cible) $\mapsto$ Δpb
② f_{CA} : (Δpb, Sol(srce)) $\mapsto$ Δsol
③ (Sol(srce), Δsol) $\mapsto$ Sol(cible)

FIG. 4.1 – Décomposition de l'adaptation suivant une analogie transformationnelle.

L'adaptation est composée de trois étapes qui sont résumées dans la figure 4.1. La première étape ① est une étape d'appariement. Elle consiste à identifier les similarités et dissimilarités entre le problème cible et le problème source remémoré. Une fonction d'appariement Δ est appliquée au couple (srce, cible) et produit une représentation Δpb de la variation entre srce et cible.

> Par exemple, le problème cible donné en introduction consistait à déterminer une recette de confiture d'abricots secs et d'ananas à partir d'un ensemble de recettes de confitures trouvées dans un livre de recettes. L'étape ① de l'adaptation consiste à apparier les deux problèmes srce et cible et à produire une représentation Δpb de la variation entre srce et cible :
>
> srce = « Je veux une recette de confiture d'abricots secs et de pommes. »
> cible = « Je veux une recette de confiture d'abricots secs et d'ananas. »
> Δpb = « pommes ↔ ananas. »

La deuxième étape ② consiste à utiliser des connaissances d'adaptation CA pour construire une variation Δsol entre solutions à partir de la variation Δpb.

> Dans l'exemple donné en introduction, la connaissance d'adaptation CA utilisée est la connaissance donnée par l'auteur du livre de recettes (partie B de la figure 2), et celle-ci permet de construire Δsol, qui représente le remplacement des pommes par l'ananas et la suppression des noisettes dans Sol(srce) :
>
> Sol(srce) = « Suivre la recette donnée dans la partie A de la figure 2 (page 2). »
> Δsol = « pommes et noisettes ↔ ananas. »

Enfin, la troisième étape ③ consiste à appliquer Δsol pour construire une solution Sol(cible) à partir de Sol(srce), c'est à dire à modifier Sol(srce) de telle façon que Δ(Sol(srce), Sol(cible)) = Δsol.

L'application de Δsol permet de modifier Sol(srce) pour construire la solution Sol(cible) suivante :

> Sol(cible) = « Suivre la recette donnée dans la partie A de la figure 2, mais en utilisant de l'ananas au lieu des pommes, dans les mêmes proportions, et ne pas mettre de noisettes. »

Le carré d'analogie formé à l'issue de la résolution de ce problème par le système est donné dans la figure 4.2.

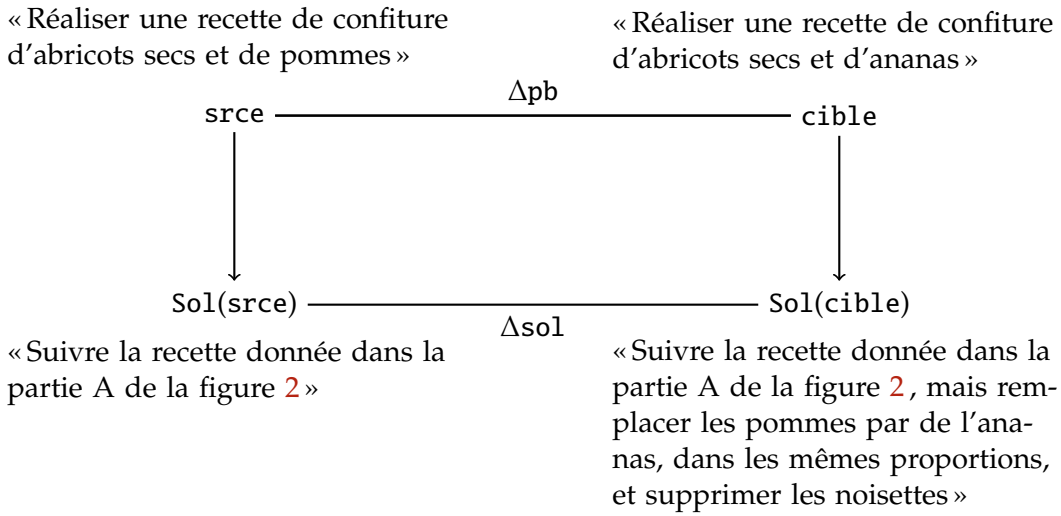


FIG. 4.2 – Décomposition de l'adaptation : l'exemple des recettes de confiture.

Dans cette modélisation, deux fonctions ont été introduites qui jouent un rôle essentiel :
- Δ est la fonction d'appariement qui permet de produire une représentation des variations entre problèmes et entre solutions.
- f_{CA} est la fonction d'adaptation qui permet d'associer une variation entre solutions à une variation entre problèmes.

Nous montrons alors que les langages $\mathcal{L}_{\Delta\text{pb}}$ et $\mathcal{L}_{\Delta\text{sol}}$ peuvent être utilisés pour définir les fonctions Δ et f_{CA}.

4.2 Définition de la fonction d'appariement

Apparier deux problèmes srce et cible dans $\mathcal{L}_{\Delta\text{pb}}$ consiste à identifier les relations $\texttt{r} \in \mathcal{L}_{\Delta\text{pb}}$ reliant srce à cible, c'est à dire l'ensemble Clôture($\langle$srce, cible$\rangle$). Comme toute relation $\texttt{r} \in \mathcal{L}_{\Delta\text{pb}}$ reliant srce à cible peut être déduite de la relation $\langle$srce, cible$\rangle$, l'appariement est défini dans le langage $\mathcal{L}_{\Delta\text{pb}}$ par la relation $\langle$srce, cible$\rangle$. De la même façon, l'appariement entre Sol(srce) et Sol(cible) est définie sur $\mathcal{L}_{\Delta\text{sol}}$ par la relation $\langle$Sol(srce), Sol(cible)$\rangle$.

Définition 1 (Fonction d'appariement Δ)**.** *Soit* ((*srce*, *Sol*(*srce*)), (*cible*, *Sol*(*cible*)) *un couple de cas sources. La fonction d'appariement* Δ *est définie sur* $\mathcal{L}_{\Delta\text{pb}}$ *par*

$$\Delta(\textit{srce}, \textit{cible}) = \langle \textit{srce}, \textit{cible} \rangle$$

De la même façon, Δ est définie sur $\mathcal{L}_{\Delta sol}$ par :

$$\Delta(Sol(srce), Sol(cible)) = \langle Sol(srce), Sol(cible)\rangle$$

L'étape ① de l'adaptation est donc réalisée par la représentation de la variation $\Delta\mathtt{pb}$ entre les problèmes `srce` et `cible` par la relation $\Delta(\mathtt{srce}, \mathtt{cible}) = \langle \mathtt{srce}, \mathtt{cible}\rangle \in \mathcal{L}_{\Delta\mathtt{pb}}$.

4.3 Définition de la fonction d'adaptation

Pour un cas source $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ remémoré, le rôle de la fonction d'adaptation f_{CA} est alors d'utiliser les connaissances d'adaptation CA pour produire une variation $\Delta\mathtt{sol}$ qui puisse être utilisée pour résoudre le problème d'adaptation $\mathcal{A}_{\mathtt{pb}} = (\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{pb}}$ en une solution $\mathtt{Sol}(\mathtt{cible})$. Du fait de la sémantique des relations $\langle \mathtt{srce}, \mathtt{cible}\rangle$ et $\langle \mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible})\rangle$, ce problème d'adaptation peut être résolu par la donnée de la relation $\langle \mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible})\rangle \in \mathcal{L}_{\Delta\mathtt{sol}}$. Le but de l'étape ② de l'adaptation est donc de déterminer la relation $\langle \mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible})\rangle$ ou de l'approcher par une relation $\mathtt{R}$ plus générale.

Définition 2 (Fonction d'adaptation f_{CA})**.** *La fonction d'adaptation f_{CA} associe à un problème d'adaptation $(srce, Sol(srce), cible) \in \mathcal{L}_{pb} \times \mathcal{L}_{sol} \times \mathcal{L}_{pb}$ une relation $R \in \mathcal{L}_{\Delta sol}$ vérifiant :*

$$\textit{il existe } Sol(cible) \in \mathcal{L}_{sol} \textit{ telle que } \Delta(Sol(srce), Sol(cible)) \vDash R$$

Une telle relation $\mathtt{R} \in \mathcal{L}_{\Delta\mathtt{sol}}$ est spécialisée en une relation $\langle \mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible})\rangle$ qui est utilisée dans l'étape ③ de l'adaptation pour construire la solution $\mathtt{Sol}(\mathtt{cible})$ de `cible`.

4.4 Représentation des connaissances d'adaptation

Chaque élément $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ peut être utilisé pour résoudre un certain nombre de problèmes d'adaptation $\mathcal{A}_{\mathtt{pb}} = (\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{pb}}$. Pour cette raison, un tel couple est appelé *règle d'adaptation*.

Définition 3 (Règle d'adaptation)**.** *Une règle d'adaptation est un couple $(r, R) \in \mathcal{L}_{\Delta pb} \times \mathcal{L}_{\Delta sol}$. Une règle d'adaptation permet de résoudre un ensemble de problèmes d'adaptation $(srce, Sol(srce), cible) \in \mathcal{L}_{pb} \times \mathcal{L}_{sol} \times \mathcal{L}_{pb}$ et s'interprète comme suit :*

$$\begin{array}{ll} \textbf{si} & \Delta(srce, cible) \vDash r \\ \textbf{alors} & Sol(cible) \textit{ est telle que } \Delta(Sol(srce), Sol(cible)) \vDash R \end{array} \tag{4.1}$$

De part sa sémantique, une règle d'adaptation peut donc être considérée comme une généralisation d'un ensemble de cas d'adaptation. S'il existe au plus une solution $\mathtt{Sol}(\mathtt{cible}) \in \mathcal{L}_{\mathtt{sol}}$ qui satisfait (4.1) pour tout problème d'adaptation $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{cible})$ alors la règle d'adaptation est dite *spécifique* et correspond à la donnée d'un cas d'adaptation particulier. Dans le cas contraire, elle est dite *générale*. Étant donné deux règles d'adaptation $\mathrm{RA}_1 = (\mathtt{r}_1, \mathtt{R}_1)$ et $\mathrm{RA}_2 = (\mathtt{r}_2, \mathtt{R}_2)$, si $\mathrm{RA}_1 \vDash \mathrm{RA}_2$ (RA_2 est dite plus générale que RA_1) alors RA_1 peut être appliquée à moins de problèmes d'adaptation mais est plus précise dans le sens où la contrainte sur $\mathtt{Sol}(\mathtt{cible})$ est plus forte. La règle d'adaptation la plus générale est $(\top_{\Delta\mathtt{pb}}, \top_{\Delta\mathtt{sol}})$.

Par ailleurs, si $\mathrm{RA} = (\mathtt{r}, \mathtt{R})$ et $\mathrm{RA}' = (\mathtt{r}', \mathtt{R}')$ sont deux règles d'adaptation alors $\mathrm{RA} \wedge \mathrm{RA}'$ dénote la règle d'adaptation $(\mathtt{r} \wedge \mathtt{r}', \mathtt{R} \wedge \mathtt{R}')$, ce qui est cohérent avec la sémantique des règles d'adaptation donnée par l'équation (4.1).

Sous l'hypothèse de symétrie sur $\mathcal{L}_{\Delta\mathtt{pb}}$ et $\mathcal{L}_{\Delta\mathtt{sol}}$ (voir section 3.1, chapitre 3), une règle d'adaptation inverse $\mathrm{RA}^{-1} = (\mathtt{r}^{-1}, \mathtt{R}^{-1})$ peut être associée à chaque règle $\mathrm{RA} = (\mathtt{r}, \mathtt{R})$.

4.5 Règle d'adaptation candidate

Une règle d'adaptation $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{pb}}$ extraite de la base de cas permet de résoudre un ensemble de problèmes d'adaptation. Inversement, pour résoudre un problème d'adaptation donné, seules certaines règles d'adaptation peuvent être utilisées. On cherchera des règles d'adaptation de la forme $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ telles que :

(1) $\mathtt{r}$ représente la variation entre $\mathtt{srce}$ et $\mathtt{cible}$, ou une variation plus générale,

(2) $\mathtt{R}$ permet, lorsqu'elle est appliquée à $\mathtt{Sol}(\mathtt{srce})$, de proposer au moins une solution $\mathtt{Sol}(\mathtt{cible})$ pour $\mathtt{cible}$.

La condition (1) imposée sur les connaissances à acquérir permet de piloter l'acquisition par le problème à résoudre et la condition (2) d'assurer l'adaptabilité du cas source remémoré. Plus formellement, une règle d'adaptation $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{pb}}$ est dite *candidate* à la résolution du problème d'adaptation $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{pb}}$ si ce problème d'adaptation peut être résolu par une spécialisation de $\mathtt{R}$.

Définition 4 (Règle d'adaptation candidate)**.** *Une règle d'adaptation* $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta pb} \times \mathcal{L}_{\Delta sol}$ *est candidate à la résolution du problème d'adaptation* $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{cible}) \in \mathcal{L}_{pb} \times \mathcal{L}_{sol} \times \mathcal{L}_{pb}$ *si*

$$\begin{cases} \Delta(\mathtt{srce}, \mathtt{cible}) \vDash \mathtt{r} \\ \textit{il existe } \mathtt{Sol}(\mathtt{cible}) \in \mathcal{L}_{sol} \textit{ telle que } \Delta(\mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible})) \vDash \mathtt{R} \end{cases} \tag{4.2}$$

4.6 Application à Taaable

Dans cette section, l'approche proposée pour modéliser l'adaptation est appliquée à Taaable. Les langages $\mathcal{L}_{\Delta\mathtt{pb}}$ et $\mathcal{L}_{\Delta\mathtt{sol}}$ de représentation des variations entre problèmes (requêtes) et entre solutions (recettes) sont ceux définis à la section 3.3 (page 45).

Décomposition de l'adaptation dans Taaable. Dans l'étape ① de l'adaptation, la variation $\Delta\mathtt{pb}$ entre les problèmes $\mathtt{srce}$ et $\mathtt{cible}$ est représentée par la relation $\Delta(\mathtt{srce}, \mathtt{cible}) = \langle \mathtt{srce}, \mathtt{cible} \rangle \in \mathcal{L}_{\Delta\mathtt{pb}}$. Par exemple, si

$$\begin{aligned} \mathtt{srce} &= \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{allium} \\ \mathtt{cible} &= \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{poireau} \wedge \neg\, \mathtt{huile\_arachide} \end{aligned}$$

sont deux problèmes de $\mathcal{L}_{\mathtt{pb}}$ et si $\mathtt{poireau} \vDash_O \mathtt{allium}$ alors

$$\Delta\mathtt{pb} = \langle \mathtt{srce}, \mathtt{cible} \rangle = \mathtt{soupe}^{=} \wedge \mathtt{chinoise}^{=} \wedge \mathtt{allium}^{=} \wedge \mathtt{poireau}^{+} \wedge (\neg\, \mathtt{huile\_arachide})^{+}$$

La deuxième étape ② consiste à utiliser des connaissances d'adaptation CA pour construire une variation $\Delta\mathtt{sol}$ entre solutions à partir de la variation $\Delta\mathtt{pb}$ et de $\mathtt{Sol}(\mathtt{srce})$. Supposons par exemple que $\mathtt{Sol}(\mathtt{srce})$ soit la recette de la soupe Wonton :

$$\mathtt{Sol}(\mathtt{srce}) = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{oignon\_vert} \wedge \mathtt{bok\_choy} \wedge \mathtt{huile\_arachide} \wedge \textit{Rien d'autre}$$

La fonction d'adaptation f_{CA} doit associer au problème d'adaptation $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{pb}}$ une relation $\mathtt{R} \in \mathcal{L}_{\Delta\mathtt{sol}}$ vérifiant :

$$\text{il existe } \mathtt{Sol}(\mathtt{cible}) \in \mathcal{L}_{\mathtt{sol}} \text{ telle que } \Delta(\mathtt{Sol}(\mathtt{srce}), \mathtt{Sol}(\mathtt{cible})) \vDash \mathtt{R}$$

puis la relation $\mathtt{R} \in \mathcal{L}_{\Delta\mathtt{sol}}$ est spécialisée en une relation $\Delta\mathtt{sol} = \langle \mathtt{Sol(srce)}, \mathtt{Sol(cible)} \rangle$. Dans l'exemple, supposons que la fonction d'adaptation f_{CA} ait permis de produire une variation R et que cette relation ait été spécialisée en la solution $\Delta\mathtt{sol}$ suivante :

$$\Delta\mathtt{sol} = \langle \mathtt{Sol(srce)}, \mathtt{Sol(cible)} \rangle = \mathtt{soupe}^{=} \wedge \mathtt{chinoise}^{=} \wedge \mathtt{allium}^{=} \wedge \mathtt{oignon\_vert}^{-} \wedge \mathtt{poireau}^{+} \wedge \mathtt{bok\_choy}^{=} \wedge \mathtt{huile\_arachide}^{-}$$

Dans l'étape ③ de l'adaptation, la variation $\Delta\mathtt{sol}$ obtenue entre les solutions Sol(srce) et Sol(cible) est interprétée comme une substitution $\mathtt{A} \rightsquigarrow \mathtt{B}$ puis appliquée pour modifier Sol(srce). Dans l'exemple, la variation $\Delta\mathtt{sol}$ est interprétée comme la substitution $\mathtt{A} \rightsquigarrow \mathtt{B}$ avec :

$$\mathtt{A} = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{oignon\_vert} \wedge \mathtt{bok\_choy} \wedge \mathtt{huile\_arachide}$$
$$\mathtt{B} = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{poireau} \wedge \mathtt{bok\_choy}$$

Cette substitution est appliquée à Sol(srce) pour construire la solution Sol(cible) suivante :

$$\mathtt{Sol(cible)} = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{poireau} \wedge \mathtt{bok\_choy} \wedge \textit{Rien d'autre}$$

Règle d'adaptation candidate. La définition d'une règle d'adaptation candidate est donnée par l'équation (4.2) : une règle d'adaptation $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ est candidate à la résolution d'un problème d'adaptation $(\mathtt{srce}, \mathtt{Sol(srce)}, \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{pb}}$ si

$$\begin{cases} \Delta(\mathtt{srce}, \mathtt{cible}) \models \mathtt{r} \\ \text{il existe } \mathtt{Sol(cible)} \in \mathcal{L}_{\mathtt{sol}} \text{ telle que } \Delta(\mathtt{Sol(srce)}, \mathtt{Sol(cible)}) \models \mathtt{R} \end{cases}$$

Or dans Taaable, la représentation en logique propositionnelle d'une recette correspond à une interprétation $\mathcal{I}$ (section 2.2 page 35). La variation $\Delta(\mathtt{Sol(srce)}, \mathtt{Sol(cible)})$ entre deux recettes Sol(srce) et Sol(cible) représente donc un couple d'interprétations. Par ailleurs, $\mathcal{L}_{\Delta\mathtt{sol}}$ a été défini dans la section 3.3 en utilisant trois relations $\mathtt{a}^{-}$, $\mathtt{a}^{+}$ et $\mathtt{a}^{=}$ dont la sémantique est donnée par :

$$\mathrm{Ext}(\mathtt{a}^{-}) = \{(x, y) \in \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{sol}} \mid x \models_{\mathcal{O}} \mathtt{a} \text{ and } y \not\models_{\mathcal{O}} \mathtt{a}\}$$
$$\mathrm{Ext}(\mathtt{a}^{+}) = \{(x, y) \in \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{sol}} \mid x \not\models_{\mathcal{O}} \mathtt{a} \text{ and } y \models_{\mathcal{O}} \mathtt{a}\}$$
$$\mathrm{Ext}(\mathtt{a}^{=}) = \{(x, y) \in \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{sol}} \mid x \models_{\mathcal{O}} \mathtt{a} \text{ and } y \models_{\mathcal{O}} \mathtt{a}\}$$

Une condition nécessaire et suffisante pour que la relation $\Delta(\mathtt{Sol(srce)}, \mathtt{Sol(cible)}) \models \mathtt{R}$ puisse être vérifiée pour au moins un élément Sol(cible) est que pour tout $\mathtt{a} \in \mathcal{V}$ tel que $\mathtt{R} \models \mathtt{a}^{-}$ ou $\mathtt{R} \models \mathtt{a}^{=}$ on ait $\mathtt{Sol(srce)} \models_{\mathcal{O}} \mathtt{a}$, et que pour tout $\mathtt{a} \in \mathcal{V}$ tel que $\mathtt{R} \models \mathtt{a}^{+}$ on ait $\mathtt{Sol(srce)} \not\models_{\mathcal{O}} \mathtt{a}$. Une règle d'adaptation $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ est donc candidate à la résolution d'un problème d'adaptation $(\mathtt{srce}, \mathtt{Sol(srce)}, \mathtt{cible}) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}} \times \mathcal{L}_{\mathtt{pb}}$ si et seulement si pour toute variable propositionnelle $\mathtt{a} \in \mathcal{V}$, on a :

$$\begin{cases} \langle \mathtt{srce}, \mathtt{cible} \rangle \models \mathtt{r} \\ \mathtt{Sol(srce)} \models_{\mathcal{O}} \mathtt{a} \text{ si } \mathtt{R} \models \mathtt{a}^{-} \text{ ou } \mathtt{R} \models \mathtt{a}^{=} \\ \mathtt{Sol(srce)} \not\models_{\mathcal{O}} \mathtt{a} \text{ si } \mathtt{R} \models \mathtt{a}^{+} \end{cases} \tag{4.3}$$

5

Extraction de connaissances d'adaptation à partir de la base de cas

Ce chapitre est consacré à la présentation d'un processus d'extraction de connaissances, appelé CabamakA, *qui permet d'apprendre des connaissances d'adaptation à partir de la base de cas. Ce processus s'appuie sur une méthode d'apprentissage par généralisation à partir d'une représentation des variations entre cas. Nous décrivons dans ce chapitre le processus d'extraction de connaissances ainsi que l'implémentation qui en a été faite. Enfin, nous montrons comment le processus peut être appliqué à* Taaable.

❄

[d'Aquin *et al.*, 2007] M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli et L. Szathmary. Case base mining for adaptation knowledge acquisition. Dans *Proc. of the International Conference on Artificial Intelligence, IJCAI'07*, p. 750–756. 2007.

[Badra et Lieber, 2007a] F. Badra et J. Lieber. Extraction de connaissances d'adaptation par l'analyse de la base de cas. Dans *Actes des septièmes journées Extraction et Gestion des Connaissances (EGC'07)*, Revue des Nouvelles Technologies de l'Information, p. 751–760. Cépaduès Éditions, 2007.

[d'Aquin *et al.*, 2006a] M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli et L. Szathmary. Adaptation Knowledge Discovery from a Case Base. Dans *Proc. of the European Conference on Artificial Intelligence*, p. 795–796. 2006.

5.1 L'extraction de connaissances

L'*extraction de connaissances à partir de données* désigne un ensemble de méthodes issues des statistiques, de l'intelligence artificielle et de la reconnaissance de formes dont le but est de valider des éléments de connaissances à partir du traitement des données. Ces méthodes sont appliquées sur les données suivant un processus qui se décompose classiquement en quatre étapes : (1) acquisition et stockage des données, (2) préparation des données, (3) fouille de données et (4) validation par un analyste. Les techniques d'apprentissage mises en œuvre ont pour but l'induction d'un modèle des données à partir de leur observation. Ce modèle est toujours constitué dans un but particulier, comme par exemple la synthèse d'une grande quantité de données, la décomposition d'un ensemble d'individus en sous-groupes homogènes ou encore la production de connaissances utiles pour un système de résolution de problèmes.

La construction d'un modèle de la réalité par induction à partir d'un ensemble de faits observés est utilisée pour modéliser des phénomènes qui ne se prêtent pas à une modélisation mathématique ou pour lesquels cette modélisation est inconnue. C'est le cas de nombreux phénomènes physiques, mais aussi de tous les phénomènes qui mettent en jeu l'expérience sociale, psychologique ou l'expertise d'un acteur humain. Dans ces situations, la relation « est un modèle de » entre un modèle et la partie de la réalité qu'il représente ne peut être définie que relativement à un observateur donné, et est donc intrinsèquement ternaire [Minsky, 1965]. L'extraction de connaissances est donc anthropocentrée et l'analyste humain est partie prenante du processus car il est le seul à même de juger de la qualité du modèle constitué à partir des données. Dans ce cadre, la tâche d'apprentissage peut être vue comme une tâche de formulation d'hypothèses visant à expliquer les faits observés. Ces hypothèses sont proposées à l'analyste humain pour validation qui les interprète à la lumière de son propre modèle de la partie de réalité étudiée [1]. Lorsqu'une partie du modèle construit à partir des données est jugé pertinente par l'analyste au regard de la tâche d'apprentissage, l'hypothèse correspondante est validée et acquiert le statut de connaissance extraite des données. L'apprentissage inductif peut donc être considéré comme un problème d'acquisition de connaissances. Il peut alors être exprimé comme un processus de modification itérative d'une représentation interne des connaissances, guidé par l'analyste, et réalisé par l'application d'*opérateurs*, suivant une certaine *stratégie de recherche*, de façon à améliorer une *fonction d'évaluation* de la qualité de la représentation [Munteanu, 1992].

5.2 Apprentissage par généralisation de règles d'adaptation

La méthode d'apprentissage par généralisation est présentée en suivant le cadre théorique introduit par Tom Mitchell dans [Mitchell, 1982]. La tâche d'apprentissage par généralisation y est présentée comme la tâche consistant, à partir :

1. d'un ensemble d'observations, ou *instances*, qui constituera l'ensemble d'apprentissage $\mathcal{E}$,
2. d'un langage de représentation de ces instances,
3. d'un langage de représentation de généralisations,
4. d'un mécanisme indiquant quelles instances sont couvertes par une généralisation donnée,

à apprendre un ensemble de généralisations cohérentes avec l'ensemble d'instances, c'est-à-dire qui couvrent chacune au moins une des instances de l'ensemble d'apprentissage.

[1] En toute rigueur, on devrait dire que l'analyste juge plutôt le modèle qu'il se fait du modèle constitué à partir des données — qui doit donc être intelligible— et non le modèle lui-même.

La méthode d'apprentissage a pour but d'extraire un ensemble de règles d'adaptation $(\mathtt{r},\mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ à partir de la base de cas. Comme l'a remarqué Kathleen Hanney dans [Hanney et Keane, 1996], certaines règles d'adaptation sont implicitement présentes dans la base de cas : une règle d'adaptation spécifique $(\Delta\mathtt{pb}_{k\ell}, \Delta\mathtt{sol}_{k\ell})$ peut être obtenue pour chaque couple de cas sources $(\mathtt{srce}_k, \mathtt{Sol}(\mathtt{srce}_k))$ et $(\mathtt{srce}_\ell, \mathtt{Sol}(\mathtt{srce}_\ell))$ en représentant la variation $\Delta\mathtt{pb}_{k\ell}$ de $\mathtt{srce}_k$ à $\mathtt{srce}_\ell$ et la variation $\Delta\mathtt{sol}_{k\ell}$ de $\mathtt{Sol}(\mathtt{srce}_k)$ à $\mathtt{Sol}(\mathtt{srce}_\ell)$. Mais une telle règle d'adaptation est très spécifique car elle ne permet de résoudre que le problème d'adaptation $(\mathtt{srce}_k, \mathtt{Sol}(\mathtt{srce}_k), \mathtt{srce}_\ell)$ en une solution $\mathtt{Sol}(\mathtt{srce}_\ell)$. L'idée de la méthode est alors de prendre comme ensemble d'apprentissage un ensemble de règles d'adaptation $(\Delta\mathtt{pb}_{k\ell}, \Delta\mathtt{sol}_{k\ell})$ et d'appliquer des techniques d'extraction de connaissances pour apprendre par généralisation un ensemble CA de règles d'adaptation plus générales (figure 5.1).

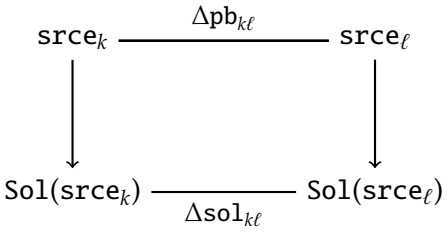


FIG. 5.1 – Apprentissage de règles d'adaptation à partir des représentations de variations entre couples de cas sources.

Si l'on applique le cadre théorique présenté ci-dessus pour décrire la tâche d'apprentissage par généralisation de régles d'adaptation, on obtient :

1. Une instance est un couple de cas sources $((\mathtt{srce}_k, \mathtt{Sol}(\mathtt{srce}_k)), (\mathtt{srce}_\ell, \mathtt{Sol}(\mathtt{srce}_\ell))$ distincts de la base de cas. L'ensemble d'apprentissage $\mathcal{E}$ est constitué d'un ensemble d'instances.
2. Le langage de représentation des instances est $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$. Une instance est représentée dans $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ par un couple $(\Delta\mathtt{pb}_{k\ell}, \Delta\mathtt{sol}_{k\ell})$, avec :

$$\begin{aligned} \Delta\mathtt{pb}_{k\ell} &= \Delta(\mathtt{srce}_k, \mathtt{srce}_\ell) \in \mathcal{L}_{\Delta\mathtt{pb}} \\ \Delta\mathtt{sol}_{k\ell} &= \Delta(\mathtt{Sol}(\mathtt{srce}_k), \mathtt{Sol}(\mathtt{srce}_\ell)) \in \mathcal{L}_{\Delta\mathtt{sol}} \end{aligned}$$

3. Le langage de représentation des généralisations est $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$.
4. La correspondance entre généralisations et instances découle de la sémantique du langage $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$: une règle d'adaptation $\mathrm{RA} = (\mathtt{r},\mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ couvre une règle $\mathrm{RA}_{k\ell} = (\Delta\mathtt{pb}_{k\ell}, \Delta\mathtt{sol}_{k\ell}) \in \mathcal{E}$ si $\mathrm{RA}_{k\ell} \vDash \mathrm{RA}$.

L'espace des hypothèses est défini comme étant l'ensemble des généralisations $(\mathtt{r},\mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ cohérentes avec l'ensemble d'apprentissage, c'est à dire pour lesquelles il existe au moins un couple de cas sources $((\mathtt{srce}_k, \mathtt{Sol}(\mathtt{srce}_k)), (\mathtt{srce}_\ell, \mathtt{Sol}(\mathtt{srce}_\ell))$ de la base de cas vérifiant $\Delta(\mathtt{srce}_k, \mathtt{srce}_\ell) \vDash \mathtt{r}$ et $\Delta(\mathtt{Sol}(\mathtt{srce}_k), \mathtt{Sol}(\mathtt{srce}_\ell)) \vDash \mathtt{R}$.

5.3 Description du processus d'extraction de connaissances

CabamakA est l'acronyme de *Case base mining for adaptation knowledge Acquisition*. CabamakA désigne à la fois un processus d'extraction de connaissances et son implémentation. Cette section décrit le processus d'extraction de connaissances. La section suivante l'implémentation qui en a été faite.

Chargement des données et formatage. Dans l'étape de chargement des données et de formatage, un ensemble de cas sources $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ est sélectionné dans la base de cas. Chacun de ces cas sources est représenté dans un formalisme $\mathcal{L}_{\texttt{pb}} \times \mathcal{L}_{\texttt{sol}}$.

Préparation des données. Dans l'étape de préparation des données est choisi un langage $\mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$ de représentation des variations entre cas. Un ensemble d'apprentissage $\mathcal{E} = \{(\Delta\texttt{pb}_{k\ell}, \Delta\texttt{sol}_{k\ell})\}_{kl}$ est formé en sélectionnant dans la base de cas un ensemble de couples de cas sources $(\texttt{srce}_k, \texttt{Sol}(\texttt{srce}_k))$ et $(\texttt{srce}_\ell, \texttt{Sol}(\texttt{srce}_\ell))$ et en représentant pour chaque couple de cas sources sélectionné la règle d'adaptation $\text{RA}_{k\ell} = (\Delta\texttt{pb}_{k\ell}, \Delta\texttt{sol}_{k\ell}) \in \mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$.

Fouille. Le but de la tâche d'apprentissage est de mettre en évidence des règles d'adaptation $(\texttt{r}, \texttt{R})$ qui sont plus générales qu'un « grand » nombre d'éléments de $\mathcal{E}$. Pour cela, on définit le *support* d'une règle d'adaptation RA par

$$\texttt{support}(\text{RA}) = \frac{\texttt{card}\,\{\text{RA}_{k\ell} \in \mathcal{E} \mid \text{RA}_{k\ell} \vDash \text{RA}\}}{\texttt{card}\,\mathcal{E}}$$

Apprendre des règles d'adaptation revient à trouver les règles d'adaptation $\text{RA} = (\texttt{r}, \texttt{R}) \in \mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$ telles que $\texttt{support}(\text{RA}) \geqslant \sigma_s$, où $\sigma_s \in [0; 1]$ est un paramètre d'apprentissage appelé seuil de support.

Remarquons que si $\text{RA}_1 \vDash \text{RA}_2$ alors $\texttt{support}(\text{RA}_1) \leqslant \texttt{support}(\text{RA}_2)$. Ordonner les règles d'adaptation apprises par support décroissant, en commençant par $(\top_{\Delta\texttt{pb}}, \top_{\Delta\texttt{sol}})$ dont le support est 1, revient à ordonner les règles par spécificité croissante. Le seuil de support a une influence sur le nombre de règles générées. Le nombre de règles générées augmente lorsque σ_s diminue. Donc, spécifier un haut seuil de support restreint la génération des règles d'adaptation aux règles les plus générales, ce qui peut limiter le nombre de règles générées et réduire le temps de calcul mais a pour effet d'écarter de l'ensemble des résultats les règles les plus spécifiques.

Sous l'hypothèse de symétrie, $\texttt{support}(\text{RA}^{-1}) = \texttt{support}(\text{RA})$ pour chaque règle d'adaptation RA. Deux règles RA et RA^{-1} pourront donc être présentées conjointement à l'expert pour validation (qui pourra éventuellement briser la symétrie en invalidant RA et en validant RA^{-1} ou le contraire.

Filtrage des résultats. Un filtre est appliqué sur les résultats pour sélectionner dans l'ensemble des résultats un ensemble de règles à présenter à l'expert pour validation.

Validation. Un modèle de la réalité construit par un processus d'extraction de connaissances n'est valide que relativement à un observateur humain donné. Pour cette raison, chaque règle d'adaptation apprise doit être validée par un observateur humain pour acquérir le statut de connaissance issue des données.

5.4 Implémentation

La première implémentation du système CabamakA a été réalisée en 2004 sur les principes décrits dans [d'Aquin *et al.*, 2004b]. Depuis, le système a été presque complètement réécrit pour tenir compte des avancées théoriques, améliorer ses performances et incorporer de nouvelles fonctionnalités. La version actuelle du système CabamakA contient plus de 30000 lignes de code Java comprenant une interface graphique Java/Swing, un langage de communication XML, une interface de communication par HTTP et des fonctionnalités d'export des résultats.

L'architecture du système CabamakA est modulaire. Les principaux modules réalisent le chargement des données, la préparation des données et la fouille. Cette architecture a été revue en 2006 afin de séparer la gestion des modules de l'interface utilisateur. L'intégration de nouveaux modules en est facilitée, la création de nouveaux types de modules (en particulier ceux qui font la communication avec d'autres applications) rendue possible et les modules sont maintenant chargés à la volée en fonction des besoins. De plus, les modules de préparation des données et de fouille on été réécrits en 2007 en collaboration avec Laszlo Szathmary pour réaliser les traitements directement sur les structures de données de la plateforme Coron [Szathmary et Napoli, 2005], ce qui a permis un gain considérable en performance.

Modules de chargement des données et formatage. Au cours du module de chargement des données et formatage, les cas sources sont convertis dans un formalisme attribut-contrainte $\mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}}$ comme celui défini à la section 3.2.

Dans le formalisme $\mathcal{L}_{\mathtt{pb}}$ choisi, une instance de problème est un élément d'un produit cartésien $\mathtt{Inst}_{\mathtt{pb}} = \mathtt{V}_1 \times \cdots \times \mathtt{V}_m$, où les ensembles de valeurs $\mathtt{V}_i$ sont choisis parmi l'ensemble des booléens $\mathbb{B}$, l'ensemble des entiers relatifs $\mathbb{Z}$ et l'ensemble des réels $\mathbb{R}$. Un problème $\mathtt{pb} \in \mathcal{L}_{\mathtt{pb}}$ est une expression de la forme $\mathtt{pb} = \mathtt{d}_1 \wedge \cdots \wedge \mathtt{d}_p$, où $\mathtt{d}_i$ est un descripteur de problèmes de la forme $\mathtt{d}_i = (\mathtt{a}_i, \mathtt{C})$. Selon l'attribut $\mathtt{a}_i$, une contrainte $\mathtt{C} \in \mathcal{L}_{\mathtt{a}_i}$ peut alors représenter :

- soit un singleton de booléens (Ext(C) = $\{x\}$ avec $x \in \mathbb{B}$),
- soit un singleton d'entier relatif (Ext(C) = $\{x\}$ avec $x \in \mathbb{Z}$),
- soit un intervalle d'entiers relatifs (Ext(C) = $[x, y]$ avec $x, y \in \mathbb{Z}$ et $x \leqslant y$),
- soit un singleton de réel (Ext(C) = $\{x\}$ avec $x \in \mathbb{R}$).

Les formalisme $\mathcal{L}_{\mathtt{sol}}$ est défini à l'aide de constructeurs similaires.

Trois modules de chargement des données ont été implantés selon le langage à partir duquel les cas sources sont convertis dans le formalisme $\mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}}$ choisi. Les langages pris en charge pour la conversion sont :

- le langage de représentation par objet utilisé pour représenter les référentiels de traitement du cancer dans les premières versions du système Kasimir [d'Aquin *et al.*, 2004a],
- le fragment du langage OWL utilisé pour représenter les référentiels de traitement du cancer dans [d'Aquin, 2005],
- le fragment de la logique propositionnelle utilisée pour représenter les recettes dans le système Taaable (voir section 2.2 page 35).

Module de préparation des données. Le module de préparation des données — ou module dit de *création des carrés* — réalise l'étape de préparation des données du processus d'extraction de connaissances. Dans cette étape, un langage $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ de représentation des variations entre cas est choisi et l'ensemble d'apprentissage $\mathcal{E} = \{(\Delta\mathtt{pb}_{k\ell}, \Delta\mathtt{sol}_{k\ell})\}_{kl}$ est formé en sélectionnant un ensemble de couples de cas sources et en représentant pour chacun d'eux la variation $\mathrm{RA}_{k\ell} = (\Delta\mathtt{pb}_{k\ell}, \Delta\mathtt{sol}_{k\ell}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$.

Le langage $\mathcal{L}_{\Delta\texttt{pb}}$ choisi pour représenter les variations entre problèmes est défini par les quatre constructeurs $\top_{\Delta\texttt{pb}}$, $\langle \texttt{srce}, \texttt{cible} \rangle$, $\wedge$ et a_i^δ, comme à la section 3.2. Le langage $\mathcal{L}_{\Delta\texttt{sol}}$ de représentation des variations entre solutions est défini de manière similaire. Les relations δ définies dans CabamakA pour le langage $\mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$ sont données par la table 5.1. Ces relations ont été choisies pour leur pertinence à représenter les variations entre cas dans les domaines d'application sur lesquels CabamakA a été appliqué durant cette thèse (Kasimir et Taaable). Les relations δ retenues pour représenter les variations entre singletons d'entiers (resp., de réels)

δ	$\{x\}\ \delta\ \{y\}$ si :
$\top_{\text{singletons d'entiers}}$	toujours vrai (pour tout $x, y \in \mathbb{Z}$)
$=$	$x = y$
$\leqslant$	$x \leqslant y$
$\neq$	$x \neq y$
$\geqslant$	$x \geqslant y$
$<$	$x < y$
$>$	$x > y$
$\texttt{ajouter}(\alpha)$	$x + \alpha = y$

Relations δ définies lorsque $\texttt{V}_i$ est l'ensemble des singletons de $\mathbb{Z}$.

δ	$\{x\}\ \delta\ \{y\}$ si :
$\top_{\text{singletons de réels}}$	toujours vrai (pour tout $x, y \in \mathbb{R}$)
$=$	$x = y$
$\leqslant$	$x \leqslant y$
$\neq$	$x \neq y$
$\geqslant$	$x \geqslant y$
$<$	$x < y$
$>$	$x > y$
$\texttt{ajouter}(\alpha)$	$x + \alpha = y$

Relations δ définies lorsque $\texttt{V}_i$ est l'ensemble des singletons de $\mathbb{R}$.

δ	$[x_1, y_1]\ \delta\ [x_2, y_2]$ si :
$\top_{\text{intervalles d'entiers}}$	toujours vrai (pour tout $x_1, x_2, y_1, y_2 \in \mathbb{Z}$ avec $x_1 \leqslant y_1$ et $x_2 \leqslant y_2$)
$=$	$x_1 = x_2$ et $y_1 = y_2$
$\neq$	$x_1 \neq x_2$ ou $y_1 \neq y_2$
b (*before*)	$y_1 < x_2$
a (*after*)	$y_2 < x_1$
quasi-m	$x_2 = y_1 + 1$
quasi-mi	$x_1 = y_2 + 1$

Relations δ définies lorsque $\texttt{V}_i$ est l'ensemble des intervalles de $\mathbb{Z}$.

δ	$\{x\}\ \delta\ \{y\}$ si :
$\top_{\text{singletons de booléens}}$	toujours vrai (pour tout $x, y \in \{\texttt{vrai}, \texttt{faux}\}$)
-	$x = \texttt{vrai}$ et $y = \texttt{faux}$
+	$x = \texttt{faux}$ et $y = \texttt{vrai}$
$=$	$x = \texttt{vrai}$ et $y = \texttt{vrai}$

Relations δ définies lorsque $\texttt{V}_i$ est l'ensemble des singletons de $\mathbb{B}$.

Tab. 5.1 – Relations δ définies dans CabamakA pour le langage $\mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$.

sont les relations $\top_{\text{singletons d'entiers}}$ (resp., $\top_{\text{singletons de réels}}$) , $=, \leqslant, \neq, \geqslant, <, >$ ainsi que la famille de relations $\texttt{ajouter}(\alpha)$ pour $\alpha \in \mathbb{Z}$ (resp., $\alpha \in \mathbb{R}$). Parmi les relations δ retenues pour représenter les variations entre intervalles d'entiers se trouvent les relations de Allen b (*before*) et a (*after*). Deux relations *quasi-m* (pour *quasi-meets*) et *quasi-mi* (pour *quasi-meets inverse*) sont également introduites et sont des adaptations des relations de Allen m (*meets*) et mi (*meets inverse*) : deux intervalles $[x_1, y_1]$ et $[x_2, y_2]$ sont liés par la relation *quasi-m* — $[x_1, y_1]$ *quasi-m* $[x_2, y_2]$ — si $x_2 = y_1 + 1$. Par exemple, les intervalles $[0, 15]$ et $[16, 45]$ vérifient $[0, 15]$ *quasi-m* $[16, 45]$ et $[0, 15]$ b $[16, 45]$. Enfin, les relations δ retenues pour représenter les variations entre singletons

de booléens sont les relations $\top_{\text{singletons de booléens}}$, -, + et =.

Les relations δ sont organisées par généralité (figure 5.2). La généralité des relations δ

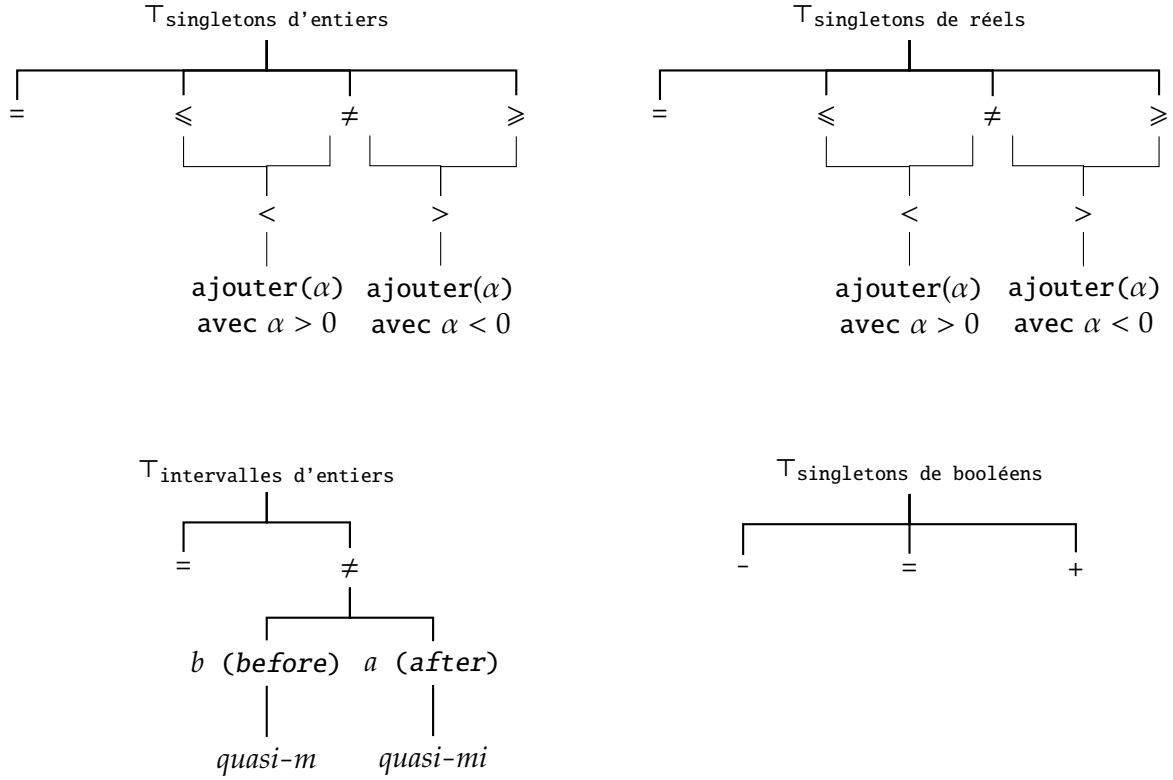


FIG. 5.2 – Organisation hiérarchique des relations δ définies dans CABAMAKA.

est définie sur leurs extensions : dans $\mathcal{L}_{\Delta a_i}$, une relation binaire δ est plus spécifique qu'une relation binaire δ' si $\text{Ext}(\delta) \subseteq \text{Ext}(\delta')$. Ainsi, toute règle d'adaptation $\text{RA} \in \mathcal{L}_{\Delta \text{pb}} \times \mathcal{L}_{\Delta \text{sol}}$ qui vérifie $\text{RA} \vDash \text{a}_i^{\delta}$ vérifie également $\text{RA} \vDash \text{a}_i^{\delta'}$ (pour $\delta, \delta' \in \mathcal{L}_{\Delta a_i}$ et δ plus spécifique que δ').

Module de fouille. Le module de fouille de CABAMAKA s'appuie sur l'algorithme CHARM [Zaki et Hsiao, 2002], un algorithme de fouille qui effectue de manière efficace l'extraction de *motifs fermés fréquents* (MFF). CHARM est implanté dans la plateforme CORON [Szathmary et Napoli, 2005] développée dans l'équipe par Laszlo Szathmary.

Etant donné un ensemble fini $\mathcal{P}$ (l'ensemble des *propriétés*), un *motif* m est un sous-ensemble de $\mathcal{P}$: $m \in 2^{\mathcal{P}}$. CHARM prend en entrée un ensemble de motifs appelés *transactions*. Le *support* d'un motif m est la proportion $\texttt{support}(m)$ des transactions T qui contiennent m ($T \supseteq m$). Un motif m est dit *fréquent* relativement à un seuil $\sigma_s \in [0;1]$ si $\texttt{support}(m) \geqslant \sigma_s$. m est dit *fermé* si l'ajout de toute propriété à ce motif change son support : $\texttt{support}(m) > \texttt{support}(m \cup \{x\})$ pour tout $x \in \mathcal{P}$ tel que $x \notin m$.

Un des atouts de CHARM réside dans son efficacité à traiter un grand nombre de transactions. Une difficulté néanmoins vient du fait que cet algorithme opère sur des données, et non sur des connaissances, ce qui fait que l'ensemble d'apprentissage $\mathcal{E}$ doit être traduit en un ensemble de transactions préalablement à l'étape de fouille. Comme l'ensemble d'apprentissage est constitué de règles d'adaptation spécifiques $\text{RA}_{k\ell}$, qui doivent être appréhendées modulo la relation

d'inférence $\vDash$, l'idée est de traduire $RA_{k\ell}$ en la transaction $\texttt{Clôture}(RA_{k\ell})$. Pour être cohérent avec cette définition, l'ensemble $\mathcal{P}$ considéré est l'ensemble $\bigcup_{RA_{k\ell} \in \mathcal{E}} \texttt{Clôture}(RA_{k\ell})$. Soit alors $m = \{RA_1, RA_2, RA_3\}$, avec RA_i, $i \in \{1,2,3\}$, trois règles d'adaptation. S'il existe exactement n règles d'adaptation $RA_{k\ell} \in \mathcal{E}$ telles que $RA_{k\ell} \vDash RA_1 \wedge RA_2 \wedge RA_3$ alors le motif m est fréquent ssi $n \geqslant \sigma_s \times \texttt{card}(\mathcal{E})$. Si, pour les *mêmes* n $RA_{k\ell} \in \mathcal{E}$, $RA_{k\ell} \vDash RA$ avec $RA \notin m$, alors m n'est pas fermé. En d'autres termes, si m n'est pas fermé, cela signifie que m est une sur-généralisation qui peut être spécialisée dans le langage de représentation des règles d'adaptation sans que la règle perde en couverture sur l'ensemble d'apprentissage.

Un problème pratique intervient lorsqu'un ensemble de relations $\texttt{Clôture}(RA_{k\ell})$ n'est pas fini : ceci engendre un ensemble $\mathcal{P}$ infini que CHARM ne peut pas traiter. L'idée est alors de choisir un ensemble $\mathcal{P}$ fini et de restreindre chaque clôture déductive à $\texttt{Clôture}(RA_{k\ell}) \cap \mathcal{P}$.

Dans CABAMAKA, l'ensemble $\mathcal{P}$ est défini comme étant l'ensemble des règles d'adaptation de la forme a_i^δ tel que a_i est un attribut du langage $\mathcal{L}_{\Delta\texttt{pb}}$ (resp., $\mathcal{L}_{\Delta\texttt{sol}}$) choisi et δ est une relation binaire du langage $\mathcal{L}_{\Delta a_i}$ associé. De plus, on ne conserve dans $\mathcal{P}$ que les règles d'adaptation a_i^δ partagées par au moins une règle d'adaptation $RA_{k\ell}$ de l'ensemble d'apprentissage (c'est-à-dire vérifiant $RA_{k\ell} \vDash a_i^\delta$ pour au moins une règle $RA_{k\ell} \in \mathcal{E}$). Un contexte formel est formé dans lequel chaque objet représente une règle d'adaptation $RA_{k\ell}$ de l'ensemble d'apprentissage $\mathcal{E}$ et chaque attribut représente une propriété a_i^δ de $\mathcal{P}$ (figure 5.4).

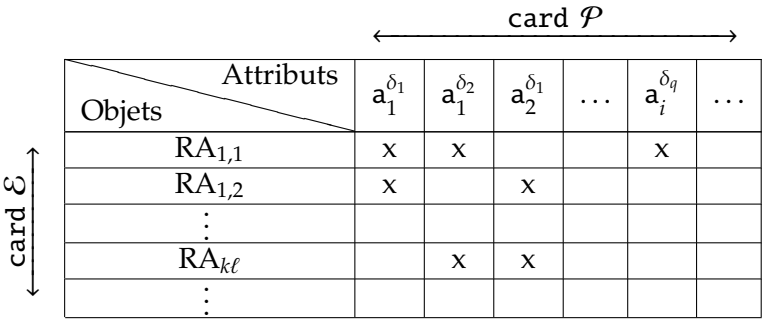


FIG. 5.3 – Le contexte formel utilisé dans CABAMAKA pour l'extraction de motifs fréquents.

5.5 Application à TAAABLE

Dans TAAABLE, la base de cas contient un ensemble de recettes. Comme nous l'avons vu au chapitre 2 (page 37), problèmes et solutions ne sont pas distingués de manière explicite dans la base de cas. Un problème `srce` n'est pas représenté tel quel mais est choisi dynamiquement lors de la remémoration parmi l'ensemble des problèmes que `Sol(srce)` résoud. Pour cette raison, les connaissances d'adaptation `CA` seront acquises à partir de la comparaison de deux ensembles de recettes. On ne cherchera à acquérir que des règles d'adaptation de la forme $(\top_{\Delta\texttt{pb}}, \Delta) \in \mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$, où $\Delta \in \mathcal{L}_{\Delta\texttt{sol}}$ représente une variation entre recettes.

Chargement des données et formatage. Dans TAAABLE l'étape de chargement des données et de formatage consiste à sélectionner un ensemble de recettes `Sol(srce)` de la base de recettes et à convertir chacune de ces recettes de la logique propositionnelle vers le formalisme attribut-contrainte $\mathcal{L}_{\texttt{sol}}$ considéré. Pour ce faire, un attribut booléen $\texttt{A}_i$ est introduit dans $\mathcal{L}_{\texttt{sol}}$ pour

chaque variable propositionnelle a de $\mathcal{V}$. Une recette est alors représentée par une conjonction de descripteurs de la forme $(\mathtt{A}_i, \mathtt{C})$, où $\mathtt{A}_i$ est l'attribut associé à la variable propositionnelle a et $\mathtt{C} = \{\mathtt{vrai}\}$ si $\mathtt{Sol(srce)} \models_{\mathcal{O}} \mathtt{a}$ et $\mathtt{C} = \{\mathtt{faux}\}$ si $\mathtt{Sol(srce)} \models_{\mathcal{O}} \neg\, \mathtt{a}$. Par exemple, la recette de la soupe Wonton représentée dans TAAABLE en logique propositionnelle par la formule :

$$\mathtt{Sol(srce)} = \mathtt{soupe} \wedge \mathtt{chinoise} \wedge \mathtt{oignon\_vert} \wedge \mathtt{bok\_choy} \wedge \mathtt{huile\_arachide} \wedge \textit{Rien d'autre}$$

où *Rien d'autre* dénote une conjonction de littéraux négatifs est traduite dans le formalisme attribut-contrainte $\mathcal{L}_{\mathtt{sol}}$ en :

$$\begin{aligned}\mathtt{Sol(srce)} = (\mathtt{soupe}, \{\mathtt{vrai}\}) \wedge (\mathtt{chinoise}, \{\mathtt{vrai}\}) \wedge (\mathtt{oignon\_vert}, \{\mathtt{vrai}\}) \\ \wedge\, (\mathtt{bok\_choy}, \{\mathtt{vrai}\}) \wedge (\mathtt{huile\_arachide}, \{\mathtt{vrai}\}) \wedge \textit{Rien d'autre}\end{aligned}$$

où *Rien d'autre* dénote une conjonction de descripteurs de la forme $(\mathtt{A}_i, \{\mathtt{faux}\})$.

La figure 5.4 présente une copie d'écran de l'onglet de chargement des données et formatage de l'interface graphique de CABAMAKA. Dans cette figure, un fichier de recettes a été chargé. Le panneau résultat donne la représentation dans le langage attribut-contrainte $\mathcal{L}_{\mathtt{sol}}$ de chacune des recettes $\mathtt{Sol(srce)}$ chargée depuis la base de recettes. Ici l'ensemble *Rien d'autre* des descripteurs de la forme $(\mathtt{A}_i, \{\mathtt{faux}\})$ n'est pas affiché pour plus de lisibilité.

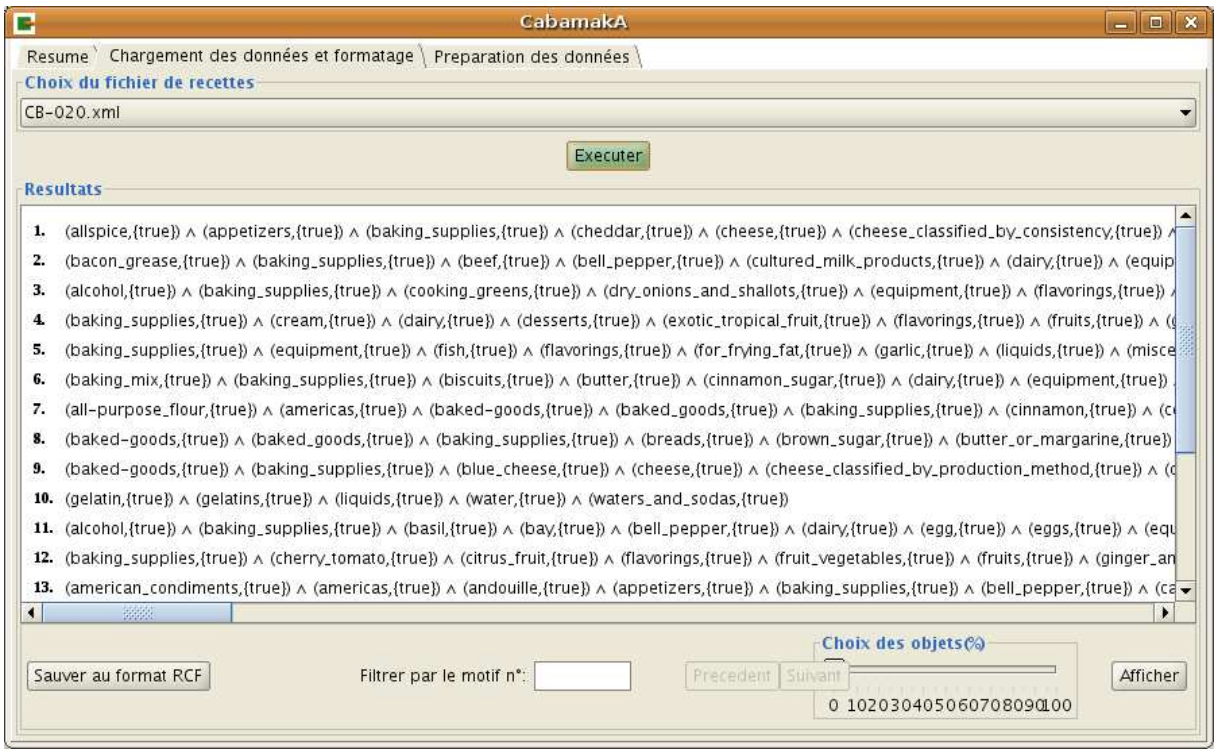


FIG. 5.4 – Copie d'écran de CABAMAKA (étape de chargement des données et formatage dans TAAABLE).

Préparation des données. Dans l'étape de préparation des données, un ensemble d'apprentissage $\mathcal{E} = \{(\top_{\Delta\mathtt{pb}}, \Delta\mathtt{sol}_{k\ell})\}_{kl}$ est constitué en sélectionnant dans la base de recettes un ensemble

de couples $(\texttt{Sol}(\texttt{srce}_k), \texttt{Sol}(\texttt{srce}_\ell))$ de recettes et en représentant la variation $\Delta\texttt{sol}_{k\ell} \in \mathcal{L}_{\Delta\texttt{sol}}$ entre les recettes $\texttt{Sol}(\texttt{srce}_k)$ et $\texttt{Sol}(\texttt{srce}_\ell)$. Comme tous les attributs sont booléens, les relations binaires δ utilisées sont les relations $\top_{\text{singletons de booleens}}$, $-$, $+$ et $=$ définies dans la figure 5.1. On notera que la sémantique des relations $-$, $+$ et $=$ utilisées dans le formalisme attribut-contrainte $\mathcal{L}_{\texttt{sol}}$ est la même que celle des relations $-$, $+$ et $=$ utilisées pour représenter les variations entre recettes en logique propositionnelle (équation 3.1 page 46). Ainsi, la variation $\Delta\texttt{sol}_{k\ell}$ entre deux recettes $\texttt{Sol}(\texttt{srce}_k)$ et $\texttt{Sol}(\texttt{srce}_\ell)$ représentées en logique propositionnelle peut être retrouvée à partir de la variation $\Delta\texttt{sol}_{k\ell}$ entre ces recettes dans le formalisme attribut-contrainte en substituant chaque attribut $\texttt{A}_i$ par la variable propositionnelle $\texttt{a}$ correspondante, comme le montre la table 5.2.

Sémantique des variations en attribut-contraintes	Sémantique des variations en logique propositionnelle
$\Delta\texttt{sol}_{k\ell} \vDash \texttt{A}_i^-$	$\Delta\texttt{sol}_{k\ell} \vDash_{\mathcal{O}} \texttt{a}^-$
$\Delta\texttt{sol}_{k\ell} \vDash \texttt{A}_i^+$	$\Delta\texttt{sol}_{k\ell} \vDash_{\mathcal{O}} \texttt{a}^+$
$\Delta\texttt{sol}_{k\ell} \vDash \texttt{A}_i^=$	$\Delta\texttt{sol}_{k\ell} \vDash_{\mathcal{O}} \texttt{a}^=$

TAB. 5.2 – Équivalences entre les sémantiques des variations dans le formalisme attribut-contrainte et en logique propositionnelle.

FIG. 5.5 – Copie d'écran de CABAMAKA (étape de préparation des données dans TAAABLE).

La figure 5.5 présente une copie d'écran de l'onglet de préparation des données de l'inter-

face graphique de CABAMAKA. Le panneau résultats donne la représentation dans le langage de représentation des variations $\mathcal{L}_{\Delta\text{sol}}$ de chaque règle d'adaptation $\text{RA}_{k\ell}$ de l'ensemble d'apprentissage $\mathcal{E}$.

Fouille. L'étape de fouille génère un ensemble de règles d'adaptation $(\top_{\Delta\text{pb}}, \Delta) \in \mathcal{L}_{\Delta\text{pb}} \times \mathcal{L}_{\Delta\text{sol}}$, dans lesquelles chaque variation Δ obtenue est interprétée comme une substitution $\sigma = \texttt{A} \rightsquigarrow \texttt{B}$, où `A` et `B` sont des conjonctions de littéraux. La figure 5.6 présente une copie d'écran de l'onglet de fouille de l'interface graphique de CABAMAKA. Le panneau résultats donne la représentation des variations $\Delta \in \mathcal{L}_{\Delta\text{sol}}$ extraites avec un support minimum de 10%, ainsi que le support relatif et absolu avec lequel elles ont été trouvées dans l'ensemble d'apprentissage.

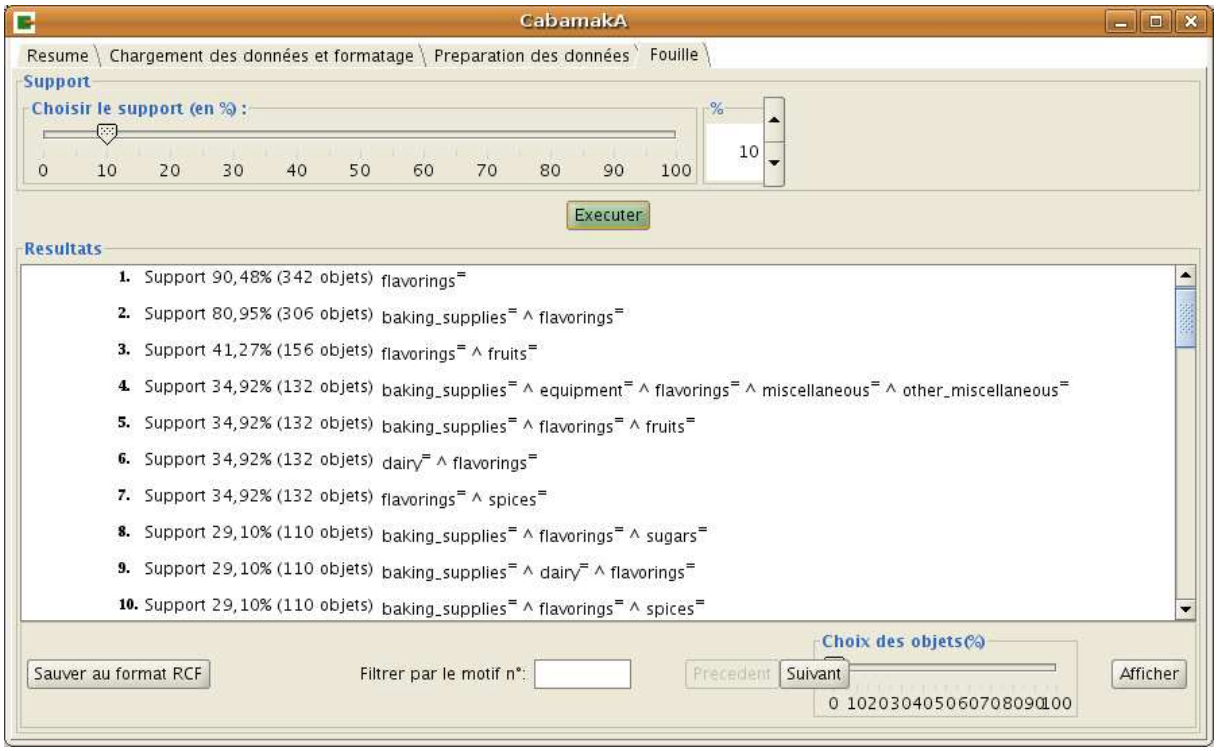


FIG. 5.6 – Copie d'écran de CABAMAKA (étape de fouille dans TAAABLE).

Règles d'adaptation candidates. D'après l'équation 4.3 établie au chapitre précédent, une règle d'adaptation $(\top_{\Delta\text{pb}}, \Delta) \in \mathcal{L}_{\Delta\text{pb}} \times \mathcal{L}_{\Delta\text{sol}}$ est candidate à la résolution d'un problème d'adaptation $(\texttt{srce}, \texttt{Sol(srce)}, \texttt{cible}) \in \mathcal{L}_{\text{pb}} \times \mathcal{L}_{\text{sol}} \times \mathcal{L}_{\text{pb}}$ si, pour toute variable propositionnelle $\texttt{a} \in \mathcal{V}$:

$$\begin{cases} \langle \texttt{srce}, \texttt{cible} \rangle \vDash \top_{\Delta\text{pb}} \\ \texttt{Sol(srce)} \vDash_{\mathcal{O}} \texttt{a} \text{ si } \Delta \vDash \texttt{a}^{-} \text{ ou } \Delta \vDash \texttt{a}^{=} \\ \texttt{Sol(srce)} \nvDash_{\mathcal{O}} \texttt{a} \text{ si } \Delta \vDash \texttt{a}^{+} \end{cases}$$

La condition $\langle \texttt{srce}, \texttt{cible} \rangle \vDash \top_{\Delta\text{pb}}$ est toujours vérifiée, pour tout couple (`srce`, `cible`). D'après l'équation 3.2 (page 46), une variation $\Delta \in \mathcal{L}_{\Delta\text{sol}}$ est interprétée comme une substitution de la

forme $\mathtt{A} \rightsquigarrow \mathtt{B}$, où $\mathtt{A}$ et $\mathtt{B}$ sont des conjonctions de littéraux tels que :

$$\begin{cases} \mathtt{A} \models_O \mathtt{a} \text{ et } \mathtt{B} \not\models_O \mathtt{a} \text{ si } \Delta \models \mathtt{a}^- \\ \mathtt{A} \not\models_O \mathtt{a} \text{ et } \mathtt{B} \models_O \mathtt{a} \text{ si } \Delta \models \mathtt{a}^+ \\ \mathtt{A} \models_O \mathtt{a} \text{ et } \mathtt{B} \models_O \mathtt{a} \text{ si } \Delta \models \mathtt{a}^= \end{cases}$$

On en déduit que si une substitution $\sigma = \mathtt{A} \rightsquigarrow \mathtt{B}$ associée à une variation Δ vérifie (1) $\mathtt{Sol(srce)} \models_O \mathtt{A}$, et (2) $\mathtt{Sol(srce)} \not\models_O \mathtt{a}$ pour toute variable propositionnelle $\mathtt{a} \in \mathcal{V}$ telle que $\mathtt{A} \not\models_O \mathtt{a}$ et $\mathtt{B} \models_O \mathtt{a}$, alors la substitution σ pourra être utilisée pour adapter la recette $\mathtt{Sol(srce)}$. Par exemple, la substitution $\sigma = \mathtt{oignon\_vert} \rightsquigarrow \mathtt{poireau}$ pourra être utilisée pour adapter une recette $\mathtt{Sol(srce)}$ si $\mathtt{Sol(srce)} \models_O \mathtt{oignon\_vert}$ et $\mathtt{Sol(srce)} \not\models_O \mathtt{poireau}$.

5.6 Application à la fouille de référentiels de décision en cancérologie

Le domaine applicatif choisi dans cette thèse est l'adaptation de recettes de cuisine. Néanmoins, pendant les deux premières années de la thèse, CabamakA a été appliqué à la fouille de référentiels de traitement du cancer dans le cadre du projet Kasimir [d'Aquin *et al.*, 2006a, Badra et Lieber, 2007a, d'Aquin *et al.*, 2007, Badra et Lieber, 2008].

Le projet Kasimir. Le projet Kasimir [Lieber *et al.*, 2002] est un projet pluridisciplinaire qui a pour but d'étudier et d'outiller l'aide à la décision et la gestion de connaissances décisionnelles en cancérologie en région Lorraine. Il regroupe depuis 1997 des informaticiens du laboratoire lorrain de recherche en informatique et ses applications (LORIA), des experts cancérologues du centre régional de lutte contre le cancer Alexis Vautrin (CAV) et du réseau de santé Oncolor, des professionnels de la santé du Groupement de Coopération Sanitaire Télésanté Lorraine et des ergonomes du Conservatoire National des Arts et Métiers (CNAM) de Paris. Ses principaux objectifs sont la capitalisation et le partage de connaissances médicales ainsi que la découverte de nouvelles connaissances par l'application de techniques d'extraction de connaissances.

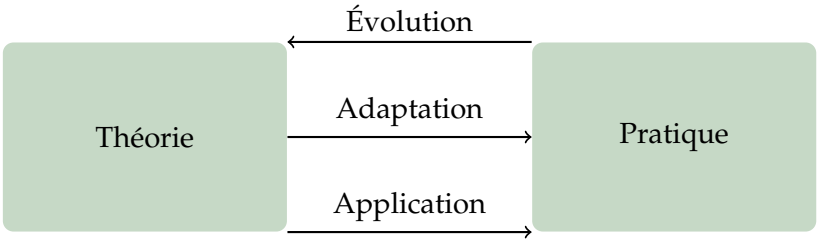


Fig. 5.7 – Les principales étapes du projet Kasimir.

Dans le projet Kasimir, les connaissances médicales sont principalement appréhendées au travers des référentiels de décision. En cancérologie, les *référentiels de décision* sont des documents établis par les professionnels de santé qui consignent les bonnes pratiques en matière de prise de décision (concernant par exemple le traitement, le diagnostic ou la surveillance de patients atteints du cancer). Ce sont des « énoncés développés de manière systématique pour assister les praticiens et les patients dans leurs prises de décisions lorsque celles-ci visent à déterminer les soins médicaux appropriés dans des circonstances cliniques particulières » [Field et Lohr, 1990]. Ces référentiels sont principalement composés de diagrammes qui peuvent être assimilés à des

arbres de décision dans lesquels les nœuds internes correspondent à des conditions sur les caractéristiques du patient et les feuilles à des traitements recommandés. Un chemin dans un tel arbre correspond à une conjonction de conditions satisfaites par le patient. Ces référentiels constituent donc un modèle de l'ensemble des situations qui peuvent être rencontrées par les praticiens. On estime en général que les référentiels couvrent entre 60 et 70 % des situations rencontrées.

La première partie du projet KASIMIR a été consacrée à la représentation informatique des référentiels de décision et au développement d'outils permettant à un utilisateur de les interroger (partie « Application » de la figure 5.7). Ces travaux ont abouti à la création du système KASIMIR (figure 5.8) en 2002. Le système KASIMIR présente à l'utilisateur une interface graphique lui permettant de saisir certaines caractéristiques liées à son patient. Ces caractéristiques sont, par exemple dans le cas du cancer du sein, l'âge du patient, son sexe, la localisation de la tumeur au niveau du sein, la taille de la tumeur, l'existence d'une invasion cutanée, etc. KASIMIR interroge les référentiels informatisés pour associer à ces caractéristiques renseignées une proposition de traitement qu'il retourne à l'utilisateur.

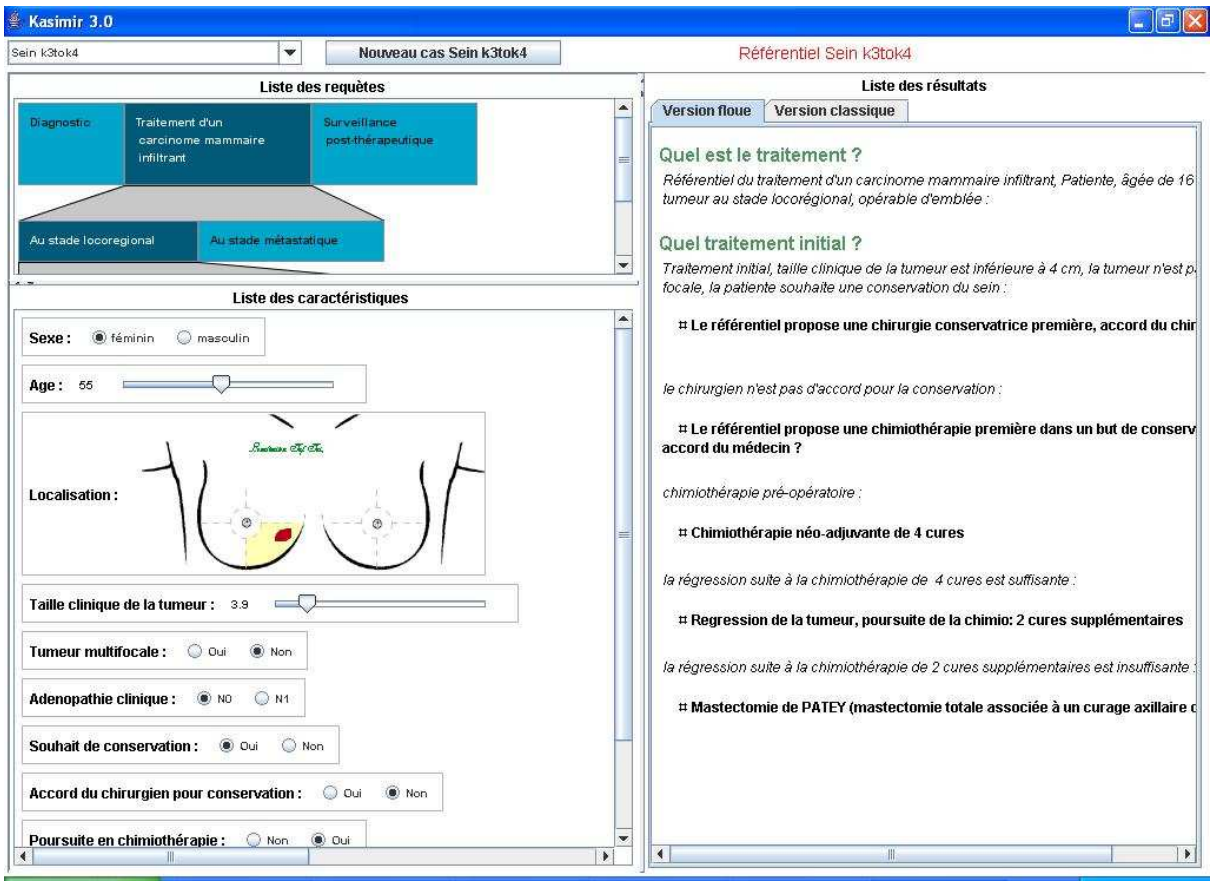


FIG. 5.8 – Copie d'écran du système KASIMIR (juin 2002).

Une deuxième partie du projet (étape « Adaptation » de la figure 5.7) consiste à étudier comment sont traités les cas médicaux lorsqu'ils présentent des caractéristiques qui ne sont pas prises en compte par les référentiels de décision. Des études ont montré que pour proposer un traitement aux patients hors référentiels, les experts se réfèrent aux cas protocolaires

correspondants et les adaptent [Sauvagnac, 2000, Mollo, 2004]. Ce type de raisonnement correspond en informatique à un raisonnement à partir de cas, dans lequel la base de cas est une représentation d'un référentiel de décision, un cas source correspond à un chemin dans un arbre de décision et un problème cible correspond à un patient hors référentiel. L'article [Lieber *et al.*, 2008] présente quelques exemples d'adaptation réalisées par des experts cancérologues et leur modélisation comme des processus de raisonnement à partir de cas. Plusieurs difficultés sont mises en évidence dans la modélisation de ces adaptations, comme la nécessité d'envisager plusieurs possibilités de traitement à proximité de certains seuils de décision, ou la nécessité de prendre en compte plusieurs points de vue dans le raisonnement pour déterminer le traitement à appliquer. L'article montre également que réaliser de telles adaptations implique que le système dispose de connaissances d'adaptation. Différentes méthodes permettant d'acquérir ces connaissances sont présentées, parmi lesquelles figure la méthode semi-automatique présentée dans cette thèse à l'aide de CabamakA. Les connaissances d'adaptation obtenues ont vocation à venir alimenter le portail sémantique de Kasimir [d'Aquin, 2005] pour être utilisées dans l'aide à la décision pour les cas non protocolaires lors d'une session de raisonnement à partir de cas.

Enfin, un objectif à long terme du projet est de proposer aux experts des évolutions des connaissances médicales (étape « Évolution » de la figure 5.7) par l'application de techniques d'extraction de connaissances sur une représentation des référentiels de décision ou sur des données réelles. Cette étape du projet n'a pour l'instant été que peu abordée et constitue une perspective de recherches.

Fouille de référentiels de décision en cancérologie. CabamakA a été appliqué dans le cadre du projet Kasimir pour fouiller les référentiels de décision (figure 5.9) dans une perspective d'acquisition de connaissances d'adaptation semi-automatique auprès d'experts. Nous décrivons dans ce qui suit une de ces expérience d'acquisition de connaissances d'adaptation semi-automatique auprès d'experts réalisée dans le domaine de la cancérologie. Les résultats expérimentaux sont tirés de [Badra et Lieber, 2008].

Lorsqu'il est appliqué à la fouille de référentiels de décision, CabamakA prend comme base de cas la représentation informatique d'un référentiel de décision. Un problème décrit une classe de patients et une solution un traitement. Dans cette expérience, la base de cas constitue une partie du référentiel de traitement du cancer du sein contenant 44 cas. Problèmes et solutions sont représentés dans le formalisme attribut-contrainte donné à la section 5.4. Un problème est représenté à l'aide de 22 attributs $\mathtt{a}_i$ et une solution est représentée à l'aide de 65 attributs $\mathtt{A}_i$. L'ensemble d'apprentissage $\mathcal{E}$ est traduit par le module de fouille en $44 \times (44 - 1) = 1\,892$ transactions, avec $\mathtt{card}\ \mathcal{P} = 300$. Pour cet ensemble d'apprentissage, Charm a extrait 342 994 motifs fréquents en 1 minute sur un PC actuel. Près de 84% des motifs extraits ont un support inférieur à 3%, c'est-à-dire généralisent moins de 56 transactions. Parmi les règles extraites figure la règle RA suivante :

$$\begin{aligned}\text{RA} = (&\hat{\mathtt{a}}\mathtt{ge}^{\neq} \wedge \mathtt{ctxt},\\ &\mathtt{nb\text{-}de\text{-}cycles\text{-}de\text{-}FEC}^{\neq} \wedge \mathtt{dose\text{-}de\text{-}FEC}^{\neq} \wedge \mathtt{Ctxt})\end{aligned}$$

où `ctxt` représente le contexte commun aux deux problèmes `srce` et `cible`. Plus précisément, `ctxt` est la conjonction des $\mathtt{a}_i^{=}$ (et donc, $\mathtt{ctxt}^{-1} \equiv \mathtt{ctxt}$), ce qui implique que $\mathtt{a}_i(\mathtt{cible}) = \mathtt{a}_i(\mathtt{srce})$ est une condition de la règle d'adaptation. De manière similaire, `Ctxt` représente le contexte commun aux deux solutions `Sol(srce)` et `Sol(cible)` et est une conjonction de $\mathtt{A}_j^{=}$, ce qui implique que $\mathtt{A}_j(\mathtt{Sol(cible)}) = \mathtt{A}_j(\mathtt{Sol(srce)})$. FEC est le nom d'un médicament en

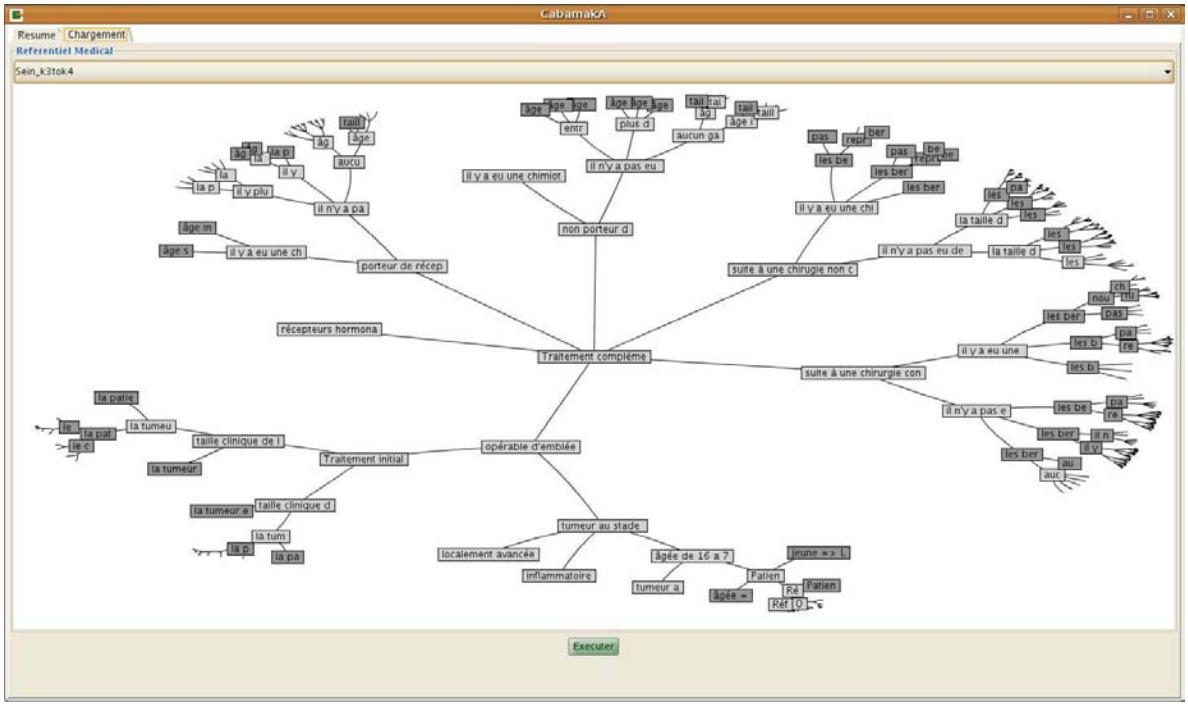


FIG. 5.9 – Copie d'écran de CABAMAKA (étape de chargement des données et formatage dans KASIMIR).

chimiothérapie qui est administré en plusieurs cycles (attribut `nb-de-cycles-de-FEC` à valeurs dans $\mathbb{Z}$) avec une dose fixe dans chaque cycle (attribut `dose-de-FEC` à valeurs dans $\mathbb{R}$). L'attribut `âge` représente l'âge du patient, et prend ses valeurs dans $\mathbb{N}$.

Cette règle exprime le fait que le choix du traitement par FEC dépend de l'âge du patient, mais n'explicite pas cette dépendance. En naviguant plus bas dans la hiérarchie des règles extraites, on trouve le couple de règles suivantes :

$$\begin{aligned} \mathrm{RA}_1 = (&\texttt{âge}^{b} \wedge \texttt{ctxt}', \\ &\texttt{nb-de-cycles-de-FEC}^{>} \wedge \texttt{dose-de-FEC}^{>} \wedge \texttt{Ctxt}') \\ \mathrm{RA}_2 = (&\texttt{âge}^{a} \wedge \texttt{ctxt}', \\ &\texttt{nb-de-cycles-de-FEC}^{<} \wedge \texttt{dose-de-FEC}^{<} \wedge \texttt{Ctxt}') \end{aligned}$$

où $\mathrm{RA}_2 \equiv {\mathrm{RA}_1}^{-1}$, $\texttt{ctxt}' \vDash \texttt{ctxt}$ et $\texttt{Ctxt}' \vDash \texttt{Ctxt}$. Chacune de ces règles établit que la dépendance exprimée par RA est décroissante : lorsque l'âge du patient augmente ($\texttt{âge}(\texttt{srce})$ b $\texttt{âge}(\texttt{cible})$), le nombre de cycles et la dose par cycle diminue.

L'expert explique cette règle par (1) le fait que la taille de la tumeur croît généralement plus vite chez les patients jeunes et donc requiert une dose supérieure de chimiothérapie et (2) le fait qu'il est nécessaire de faire un compromis entre la prise en compte de l'espérance de vie (qui décroît avec l'âge) et la qualité de vie (qui décroît avec la dose de FEC).

6

Discussion

Dans ce chapitre, nous mettons en évidence les principaux obstacles à une opérationnalisation du processus d'extraction de connaissances au sein d'un processus d'acquisition de connaissances d'adaptation. En particulier, nous montrons qu'il est nécessaire de choisir judicieusement l'ensemble d'apprentissage et de restreindre l'espace des hypothèses. Nous montrons en quoi l'étape d'aide à l'interprétation et de validation des résultats est également problématique. Enfin, nous discutons des intérêts et des limites d'un tel apprentissage lorsqu'il est réalisé hors ligne au sein d'un processus d'acquisition de connaissances semi-automatique auprès d'experts.

6.1 Apport des techniques d'extraction de connaissances

Lorsqu'elles sont utilisées pour acquérir des connaissances d'adaptation, les techniques d'extraction de connaissances permettent d'automatiser une partie du processus d'acquisition de connaissances. Dans les approches semi-automatiques, la validation des éléments de connaissances extraits se fait au travers d'interfaces homme-machine dédiées et selon un ensemble d'interactions prédéfinies. Le rôle de l'expert est réduit à la validation d'un ensemble de suggestions générées automatiquement par le système. De plus, les connaissances acquises le sont dans un format directement utilisable par le système de raisonnement à partir de cas auquel elles sont destinées. Aucune étape supplémentaire de formalisation des connaissances n'est nécessaire.

À cet égard, le processus d'extraction de connaissances peut être vu comme un processus de transfert de connaissances entre deux conteneurs de connaissances : la base de cas et les connaissances d'adaptation. L'approche proposée pour l'acquisition de connaissances d'adaptation peut être considérée comme une approche à moindre coût d'acquisition de connaissances (en anglais, *knowledge-light approach* — voir section 1.3.2 page 23), bien que l'approche ne prenne pas comme source de connaissances d'adaptation uniquement les conteneurs utilisés par le système de raisonnement à partir de cas puisque l'expert est également impliqué dans le processus pour valider les connaissances acquises.

Néanmoins, une des principales limitations des approches automatiques vient du fait que l'acquisition de connaissances d'adaptation, même si elle implique l'expert, reste limitée au vocabulaire utilisé pour décrire les cas dans la base de cas. Dépasser cette limitation suppose de coupler l'approche avec une approche d'acquisition de connaissances du domaine.

6.2 Problème du choix de l'ensemble d'apprentissage

L'opérationnalisation du processus d'extraction de connaissances nécessite un choix judicieux de l'ensemble d'apprentissage.

En effet, l'absence de restriction de l'ensemble d'apprentissage favorise l'acquisition de connaissances très générales au détriment des connaissances plus spécifiques. Or en pratique, les connaissances d'adaptation sont souvent très contextuelles. Par exemple, dans le domaine culinaire, un œuf peut parfois être remplacé par 100 grammes de tofu, mais cette règle d'adaptation n'est valable que pour certains types de plats, comme les cakes ou la mayonnaise, mais ne peut pas être appliquée pour adapter une recette de mousse ou d'omelette. L'obtention d'une telle règle avec un support élevé ne peut être réalisé que si l'on ne compare dans la base de recette que les recettes de cakes ou de mayonnaise entre elles. Si des recettes de cake sont comparées avec des recettes d'omelettes, cette règle pourra également être obtenue mais avec un support beaucoup plus faible. Ainsi, les règles d'adaptation les plus contextuelles seront « noyées » dans l'ensemble des résultats.

Par ailleurs, lorsque la taille de la base de cas excède quelques centaines de cas, l'étape de préparation des données du processus d'extraction de connaissances peut devenir très coûteux si l'ensemble d'apprentissage $\mathcal{E}$ n'est pas restreint. En l'absence de restriction, l'ensemble d'apprentissage est formé en comparant deux à deux tous les cas sources distincts de la base de cas. Si n est la taille de la base de cas, l'ensemble d'apprentissage obtenu est de taille $n\,(n-1)$. La figure 6.1 donne, lorsque l'ensemble d'apprentissage $\mathcal{E}$ n'est pas restreint, les temps de calcul (en secondes) puis l'occupation mémoire (en méga-octets) des modules de chargement et formatage des données, de préparation des données et de fouille (en prenant un support

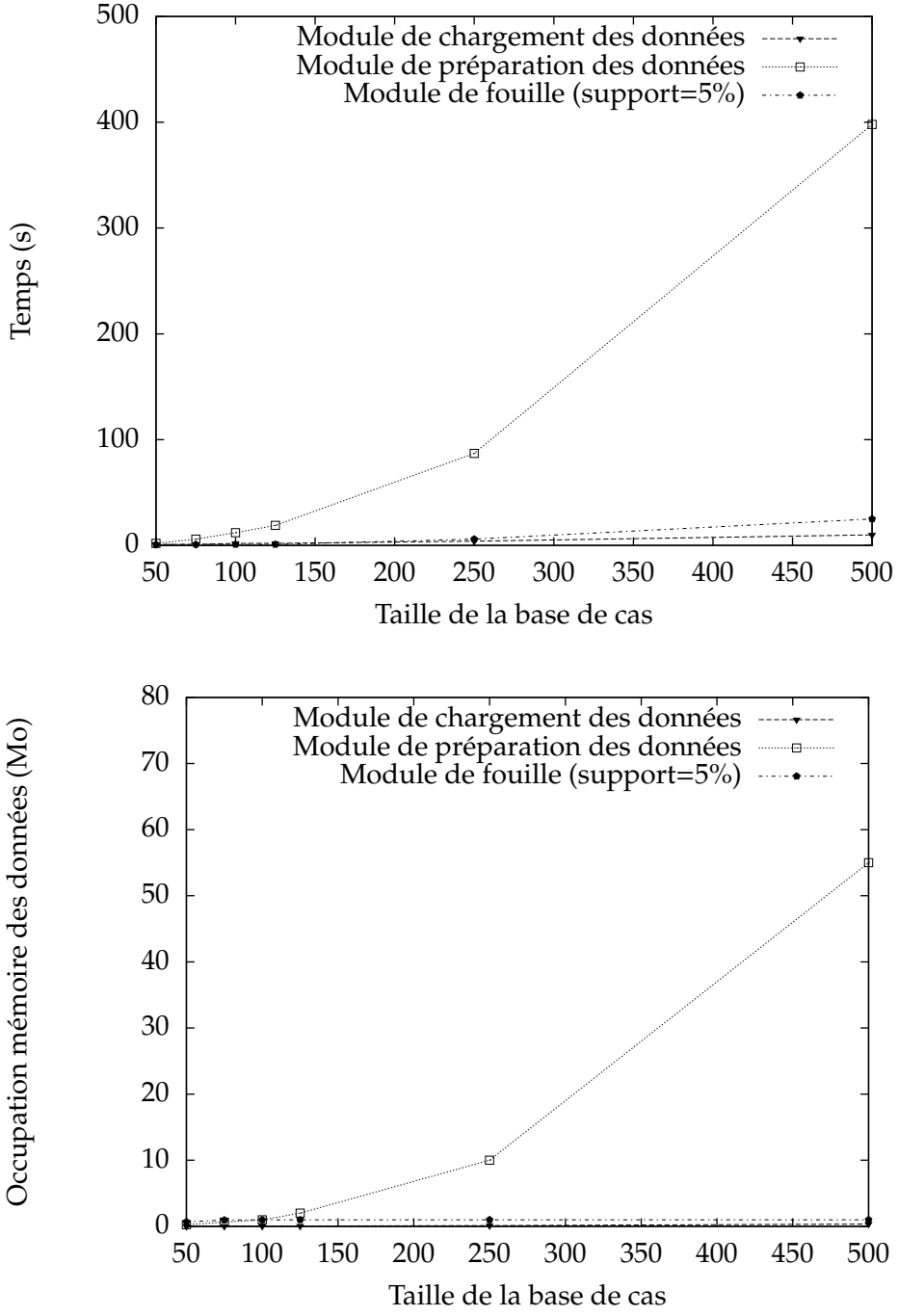


FIG. 6.1 – Temps de calcul et occupation mémoire des différents modules de CABAMAKA lorsque l'ensemble d'apprentissage n'est pas restreint, en fonction de la taille de la base de cas, pour des bases de cas dont la taille varie de 50 cas sources à 500 cas sources.

minimum de 5%) de CabamakA, en fonction de la taille de la base de cas. Le domaine applicatif des tests est Taaable. Les bases de cas utilisées, dont le nombre de cas varie de 50 à 500 cas sources, ont été constituées en sélectionnant de manière aléatoire un ensemble de recettes de la base de recettes du *Computer Cooking Contest*. Les tests ont été effectués sur une machine ayant un processeur cadencé à 3,2GHz et disposant de 2 Go de mémoire vive. On constate que c'est l'étape de préparation des données et de constitution de l'ensemble d'apprentissage qui demande de loin le plus de ressources. Par exemple, pour une base de cas de $n = 500$ cas sources, la représentation des variations prend 398 secondes (soit 6 minutes et 36 secondes) et le contexte formel obtenu occupe 55 Mo de mémoire vive, alors que l'étape de chargement et de formatage des données se fait en seulement 10 secondes (pour une occupation mémoire de 341 ko) et l'extraction de motifs en 25 secondes (pour une occupation mémoire de 1 Mo). Ceci est dû au fait qu'on ne restreint pas ici l'ensemble d'apprentissage. En conséquence, l'ensemble d'apprentissage formé pour une base de cas de $n = 500$ cas sources comprend $n\,(n-1) = 249500$ règles d'adaptation $\mathrm{RA}_{k\ell}$. Or la base de cas utilisée en 2008 pour le *Computer Cooking Contest* comprend $n = 865$ cas sources (recettes). Représenter dans l'étape de préparation des données les variations entre tous les couples de cas sources distincts de la base de cas supposerait de représenter $n\,(n-1) = 747360$ règles d'adaptation $\mathrm{RA}_{k\ell}$.

6.3 De la nécessité de restreindre l'espace des hypothèses

On constate également que lorsqu'on ne restreint pas l'espace des hypothèses, les règles d'adaptation obtenues sont trop nombreuses pour être toutes présentées à l'analyste pour validation, ce qui compromet grandement l'opérationnalisation du processus d'extraction de connaissances. Les figures 6.2 et 6.3 présentent l'évolution du nombre de motifs extraits en fonction du support minimum, pour des bases de cas BC comprenant un nombre de cas sources variant de 10 à 500 dans l'application Taaable. La figure 6.2 présente l'évolution du nombre de motifs extraits pour des valeurs du support relatif de 5% à 35% tandis que la figure 6.3 présente l'évolution du nombre de motifs extraits pour des valeurs du support relatif de 2% à 5%. Les tests ont été effectués sur une machine ayant un processeur cadencé à 3,2GHz et disposant de 2 Go de mémoire vive. Dans ces expériences, l'ensemble d'apprentissage n'est pas restreint : celui-ci est formé à partir de tous les couples de cas sources distincts de la base de cas. Si n est la taille de la base de cas, l'ensemble d'apprentissage obtenu est donc de taille $n\,(n-1)$. On constate que le nombre de motifs obtenus devient vite très conséquent lorsqu'on choisit de garder les motifs de faible support. En particulier, le nombre de motifs extraits augmente fortement lorsque le seuil de support choisi est inférieur à 10%.

6.4 Influence des connaissances extraites sur l'adaptation

Les règles d'adaptation extraites par le processus d'extraction de connaissances doivent servir à faciliter l'adaptation en structurant la base de cas par des connaissances d'adaptation. En effet, chaque règle d'adaptation $(\mathtt{r}, \mathtt{R}) \in \mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ extraite couvre un ensemble de couples de cas sources de la base de cas. La relation d'inférence $\models$ définie sur le langage $\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}$ permet alors d'organiser les connaissances d'adaptation extraites par généralité ($\models$ fait de $(\mathcal{L}_{\Delta\mathtt{pb}} \times \mathcal{L}_{\Delta\mathtt{sol}}, \models)$ un préordre). Pourtant, les connaissances d'adaptation acquises n'ont pas toutes la même influence sur l'adaptation. Des règles d'adaptation $(\mathtt{r}, \mathtt{R}) \in \mathcal{R}$ très générales permettent de résoudre un grand nombre de problèmes d'adaptation. Mais dans ces règles, R représente une contrainte très lâche sur la solution $\mathtt{Sol}(\mathtt{cible})$. Par exemple, la règle d'adaptation $\mathrm{RA} = (\top_{\Delta\mathtt{pb}}, \top_{\Delta\mathtt{sol}})$ peut être

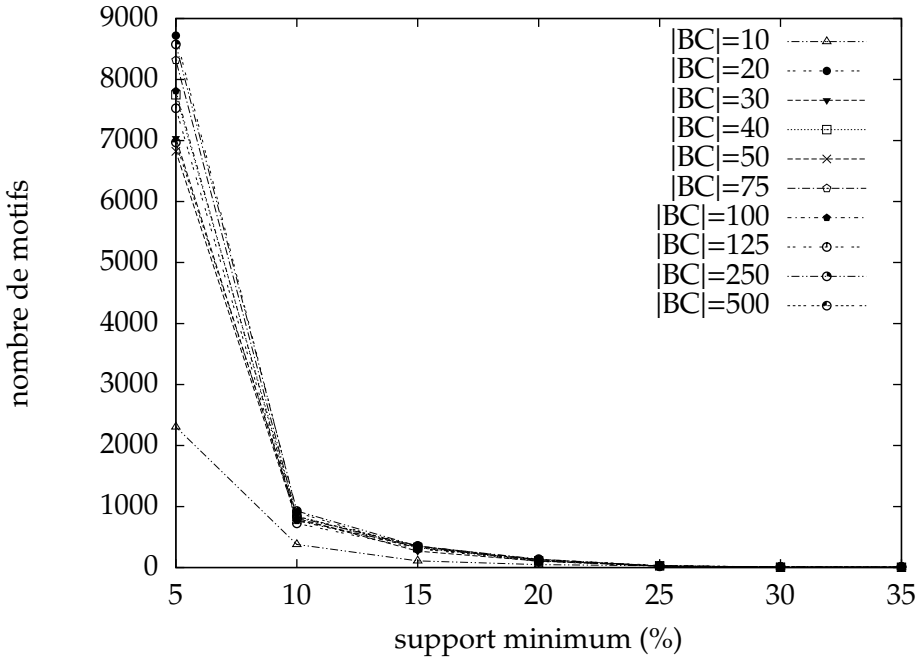


FIG. 6.2 – Nombre de motifs extraits en fonction du support minimum (pour un support minimum variant de 5% à 35%).

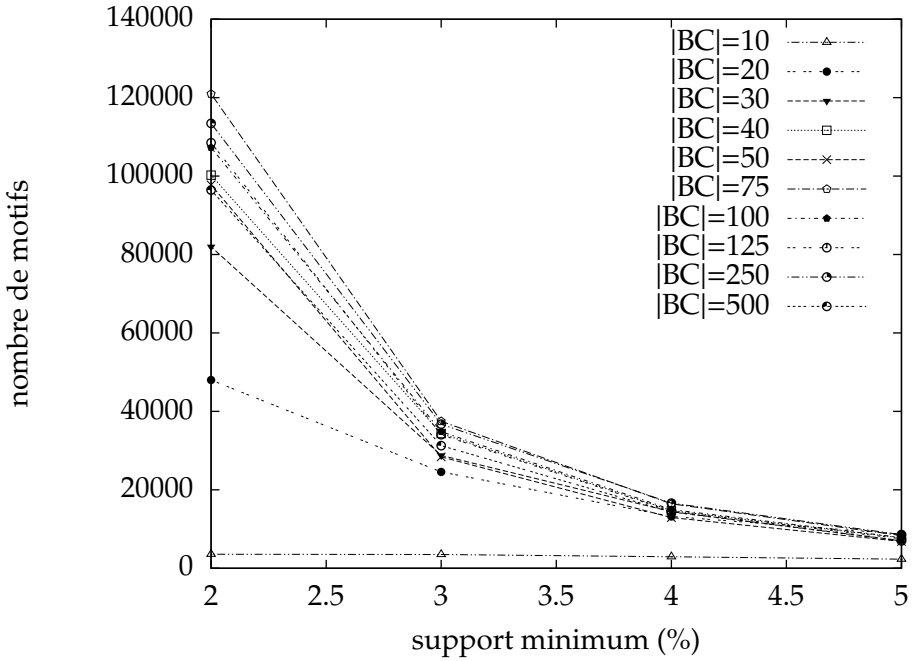


FIG. 6.3 – Nombre de motifs extraits en fonction du support minimum (pour un support minimum variant de 2% à 5%).

utilisée pour résoudre tout problème d'adaptation. Mais la donnée d'une telle règle ne réduit en rien l'effort d'adaptation du système, qui a besoin d'autres connaissances pour déterminer comment cette règle doit être spécialisée pour résoudre un problème donné. Au contraire, une règle très spécifique comme RA = $(\langle \texttt{srce}, \texttt{cible}\rangle, \langle \texttt{Sol(srce)}, \texttt{Sol(cible)}\rangle)$ dirigera la résolution vers une unique solution $\texttt{Sol(cible)}$ mais ne pourra être utilisée que pour résoudre un seul problème d'adaptation $(\texttt{srce}, \texttt{Sol(srce)}, \texttt{cible})$. En outre, une telle règle extraite des données n'a que peu d'intérêt pour l'adaptation car le problème `cible` qu'elle permet de résoudre peut être également résolu par une simple remémoration. Les règles apprises par CabamakA doivent donc être suffisamment spécifiques pour pouvoir être spécialisées lors de l'étape d'adaptation, mais ne doivent pas être trop spécifiques pour éviter un scenario de sur-apprentissage dans lequel les connaissances apprises « collent aux données » et sont alors peu ou pas informatives.

6.5 Des difficultés dans l'aide à l'interprétation

L'étape d'interprétation et de validation des résultats est également problématique. Une des difficultés vient du fait que les règles obtenues ne sont pas formulées dans un format intelligible par l'analyste : l'analyste n'est pas capable d'interpréter un motif. Pour devenir compréhensibles les règles obtenues doivent être exprimées en langue naturelle. De plus, lorsque l'espace des hypothèses n'est pas restreint, les règles obtenues sont trop nombreuses pour être toutes présentées à l'analyste pour validation. L'étape de validation nécessite alors l'intervention d'un ingénieur de la connaissance, dont le rôle est de sélectionner les motifs intéressants parmi les résultats, de les interpréter comme des règles d'adaptation puis de les présenter à l'analyste pour validation. Enfin, la validation des règles d'adaptation extraites devrait s'accompagner d'une spécification de leurs domaines d'application. Or celui-ci est généralement absent pour les règles les plus générales et n'est spécifié correctement que pour des règles très spécifiques, donc obtenues avec un support très faible.

6.6 Comment pallier à ces limitations ?

Plusieurs pistes ont été étudiées qui ont pour but de pallier à ces limitations.

Restreindre l'ensemble d'apprentissage. La première piste a consisté à se donner un moyen de restreindre l'ensemble d'apprentissage. Dans l'approche proposée, on ne peut pas *a priori* restreindre l'ensemble d'apprentissage $\mathcal{E}$ car pour un problème d'adaptation $\mathcal{A}_{\texttt{pb}} = (\texttt{srce}, \texttt{Sol(srce)}, \texttt{cible})$ donné, une règle d'adaptation qui résoud $\mathcal{A}_{\texttt{pb}}$ peut être extraite de tout couple $(\langle \texttt{srce}_k, \texttt{srce}_\ell\rangle, \langle \texttt{Sol(srce}_k\texttt{)}, \texttt{Sol(srce}_\ell\texttt{)}\rangle)$ de relations. En effet, la règle $(\top_{\Delta\texttt{pb}}, \top_{\Delta\texttt{sol}})$ est une généralisation de chacun de ces couples qui permet de résoudre tout problème d'adaptation. Elle est donc apprise quel que soit l'ensemble d'apprentissage avec un support quelconque.

Une façon de restreindre l'ensemble d'apprentissage est, comme dans [Hanney, 1997] et [Craw *et al.*, 2006], de ne sélectionner dans la base de cas que les couples de cas sources jugés similaires selon une mesure de similarité donnée. Les auteurs de ces travaux justifient ce choix par le fait que la taille de l'ensemble d'apprentissage doit être réduit pour minimiser le coût du processus de génération de règles d'adaptation et que seuls les cas sources similaires entre eux devraient être comparés. Néanmoins se donner une mesure de similarité entre cas comporterait forcément une part d'arbitraire. C'est la raison pour laquelle dans le processus d'extraction de

connaissances aucun filtre n'est appliqué sur l'ensemble d'apprentissage : $\mathcal{E}$ est formé à partir de tous les couples de cas sources distincts de la base de cas.

Une autre façon de restreindre l'ensemble d'apprentissage est de le faire en concertation avec l'analyste. Celui-ci peut spécifier quel sous-ensemble de la base de cas est pertinent pour apprendre des règles d'adaptation dans le but de résoudre un problème cible donné. Par exemple, dans le domaine culinaire, l'analyste pourra spécifier qu'une recette de tarte au fromage et à la banane ne peut être obtenue que par modification d'une recette de dessert, ce qui amènera à ne considérer dans la construction de l'ensemble d'apprentissage que les couples de cas sources qui représentent des recettes de desserts.

Définir des indices de qualité des règles d'adaptation. Un moyen de sélectionner les règles d'adaptation intéressantes parmi l'ensemble des résultats est de les classer selon un indice de qualité. Cet indice pourra être adapté des indices utilisés pour les règles d'association (voir [Lallich *et al.*, 2006] ou [Vaillant *et al.*, 2005] pour une présentation des principaux indices et de méthodes d'évaluation formelles et expérimentales de ces indices). Le support d'un motif mesure la fréquence d'apparition du motif (et donc de la règle d'adaptation associée) dans l'ensemble d'apprentissage. Une étude a été menée dans [Badra et Lieber, 2007a] pour adapter aux règles d'adaptation extraites par CabamakA l'indice de confiance communément utilisé pour mesurer la qualité les règles d'association. Si cette étude a montré qu'il est possible de définir un indice de confiance d'une règle d'adaptation, aucune implémentation n'a été réalisée à ce jour et la définition d'indices de qualité des règles d'adaptation reste une piste de recherche à explorer.

Permettre la navigation parmi les règles extraites. L'utilisation d'un langage expressif pour représenter les variations entre cas permet de proposer à l'analyste des moyens de naviguer parmi les règles extraites. Une stratégie de navigation a été proposée dans [Badra et Lieber, 2008] qui consiste à présenter d'abord à l'analyste les règles de support élevé puis à lui donner les moyens de naviguer vers des règles plus spécifiques. Les règles de support élevé constituent un bon point de départ car elles ont l'avantage d'être peu nombreuses et en général faciles à interpréter. Ensuite, des filtres ont été implantés dans CabamakA qui s'appuyent sur la sémantique du langage pour sélectionner dans l'ensemble de résultats toutes les règles RA′ qui sont plus spécifiques (resp., plus générales) qu'une règle RA donnée. Un tel filtre sélectionne dans l'ensemble des résultats les règles RA′ qui vérifient RA $\models$ RA′ (resp., RA′ $\models$ RA).

Guider la fouille. Une autre piste de recherche consisterait à guider la fouille en contraignant la forme des règles d'adaptation extraites. Un tel guidage permettrait de restreindre l'espace des hypothèses pour limiter la forme et la taille des motifs extraits ou pour privilégier les règles applicables pour résoudre une classe donnée de problèmes d'adaptation. Pour cela, on pourrait par exemple envisager d'attribuer un score aux motifs extraits, à la manière de ce qui a été réalisé dans le domaine de la fouille de graphes en chimie organique dans [Pennerath *et al.*, 2008]. Cette piste de recherche n'a pour le moment pas été explorée.

6.7 Intérêts et limites d'un apprentissage hors ligne

Le processus d'extraction de connaissances a d'abord été utilisé hors ligne, comme dans [d'Aquin *et al.*, 2006a], [d'Aquin *et al.*, 2007] et [Badra et Lieber, 2007a] pour effectuer une acquisition de connaissances semi-automatique auprès d'experts.

Le processus d'extraction de connaissances est alors utilisé pour générer à partir de la base de cas un ensemble de règles d'adaptation (figure 6.4). Ces règles d'adaptation sont présentées à l'expert du domaine pour validation.

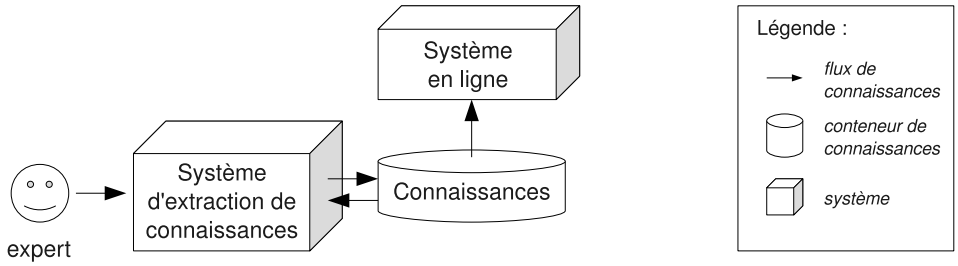


FIG. 6.4 – Acquisition semi-automatique de connaissances d'adaptation auprès d'experts.

Le principal avantage de ce type d'approche réside dans le mode de validation des connaissances. Celles-ci sont sélectionnées parmi les éléments de connaissance extraits puis présentées à un expert du domaine qui les interprète et les valide. Comme elles sont validées par un expert du domaine, les connaissances apprises peuvent être jugées comme fiables et sont généralement cohérentes entre elles et avec les connaissances existantes. De plus, aucune limite de temps n'est imposée au déroulement du processus d'extraction de connaissances.

Néanmoins, l'acquisition de connaissances d'adaptation hors ligne accroît considérablement les coûts de conception d'un système de raisonnement à partir de cas. En effet, le processus d'acquisition de connaissances nécessite la présence d'un expert du domaine et d'un ingénieur de la connaissance. De plus, l'acquisition hors ligne force le concepteur d'un système de raisonnement à partir de cas à anticiper les besoins en connaissances d'adaptation et à acquérir ces connaissances à l'avance, ce qui est très difficile, sinon impossible. On perd alors un des attraits majeurs du raisonnement à partir de cas, qui a été de limiter, par la réutilisation d'expériences passées l'effort d'acquisition initiale de connaissances du système [Schank et Slade, 1991]. Plus généralement, l'acquisition initiale de connaissances d'adaptation interdit au système d'apprendre avec l'expérience. Un système de raisonnement à partir de cas dont les connaissances d'adaptation sont fixées n'a d'autre moyen d'accroître ses capacités de résolution de problèmes que de mémoriser dans la base de cas une nouvelle expérience chaque fois qu'un problème a été résolu, comme c'est traditionnellement le cas dans les systèmes de raisonnement à partir de cas [de Mántaras et Plaza, 1997]. Pour être efficace, l'extraction de connaissances devrait au contraire être guidée par un problème à résoudre. C'est pourquoi nous étudions dans le chapitre suivant comment le processus d'extraction de connaissances peut être déclenché en ligne et être paramétré par un contexte particulier de résolution de problèmes.

7

Découverte opportuniste de connaissances d'adaptation

Dans ce chapitre, nous étudions comment le processus d'extraction de connaissances peut être déclenché en ligne au cours d'une session de résolution de problèmes. Pour cela, il est intégré à un processus interactif et opportuniste d'acquisition de connaissances d'adaptation. Celui-ci est déclenché suite à un échec d'adaptation et a pour but d'acquérir les connaissances manquantes au système. Le processus d'extraction de connaissances intervient alors pour rechercher dans la base de cas les informations nécessaires à l'instanciation d'une stratégie de réparation. Pour cela, des connaissances d'adaptation sont générées « à la volée » à partir de la base de cas, qui devient une source additionnelle de connaissances d'adaptation pour le système. Nous montrons que dans une telle approche le processus d'extraction de connaissances peut être paramétré pour ne découvrir que les connaissances nécessaires à la mise en œuvre de la stratégie de réparation choisie. Nous montrons aussi que cette approche permet de résoudre les problèmes de restriction de l'ensemble d'apprentissage et de l'espace des hypothèses rencontrés au chapitre précédent.

❄

[Badra *et al.*, 2009c] F. Badra, A. Cordier et J. Lieber. Opportunistic adaptation knowledge discovery. Dans L. McGinty et D. C.Wilson, rédacteurs, *Case-Based Reasoning Research and Development - ICCBR 2009*, p. 60–74. 2009.

[Badra *et al.*, 2009b] F. Badra, A. Cordier et J. Lieber. Découverte opportuniste de connaissances d'adaptation. Dans B. Fuchs et A. Napoli, rédacteurs, *17ème atelier sur le raisonnement à partir de cas - RàPC-09*, p. 23–34. Paris, France, 2009.

Les approches semi-automatiques d'acquisition de connaissances d'adaptation s'appuient sur des algorithmes efficaces d'apprentissage pour générer des connaissances d'adaptation à partir de la base de cas. Ces connaissances sont directement utilisables par le système et l'expert n'est sollicité que pour valider un ensemble d'unités de connaissances présélectionnés. Néanmoins nous avons vu que lorsque l'apprentissage est effectué hors ligne, les connaissances sont produites en trop grand nombre pour permettre leur validation auprès d'un expert du domaine.

D'un autre côté, les approches interactives et opportunistes d'acquisition de connaissances sont bien plus coûteuses en temps d'expert mais bénéficient du fait que l'expert est « en contexte » lorsqu'il valide les éléments de connaissances à acquérir.

L'idée de la méthode est donc de coupler les deux approches : le processus d'extraction de connaissances est déclenché au cours d'une session particulière de résolution de problèmes et est paramétré de manière à n'extraire de la base de cas que des connaissances utiles à la résolution d'un problème d'adaptation donné. Ainsi, le nombre de règles d'adaptation candidates qui sont présentées à l'expert pour validation est considérablement réduit et les connaissances acquises sont directement utilisables dans le raisonnement.

7.1 Une acquisition interactive et opportuniste de connaissances

Les connaissances d'adaptation sont acquises de manière interactive et opportuniste au cours d'une session particulière de résolution de problèmes. L'approche proposée suit les principes de l'approche FIKA pour l'acquisition de connaissances en raisonnement à partir de cas [Cordier, 2008]. FIKA désigne une approche générale d'acquisition de connaissances dans laquelle les connaissances sont acquises :

- **En ligne**, c'est-à-dire dans un contexte particulier de résolution de problèmes.
- **De manière interactive**, car la construction des connaissances à acquérir se fait à travers l'interaction du système avec son utilisateur.
- **De manière opportuniste**, car elle est déclenchée à la suite d'un échec de raisonnement, lorsque l'utilisateur n'est pas satisfait par la solution proposée par le système. L'acquisition de connaissances a alors pour but d'acquérir les connaissances manquantes au système.

7.2 Découverte opportuniste de connaissances d'adaptation

L'extraction de connaissances constitue alors une des étapes d'un processus d'acquisition interactive et opportuniste de connaissances d'adaptation. Son rôle est double : faire de la base de cas une source additionnelle de connaissances d'adaptation pour le système et fournir une assistance à l'utilisateur dans la formulation de nouvelles connaissances.

Faire de la base de cas une source de connaissances d'adaptation. Des connaissances d'adaptation sont extraites de la base de cas et transférées vers la base de connaissances d'adaptation pour être utilisées dans le raisonnement (figure 7.1). Le processus d'extraction de connaissances doit donc être paramétré pour ne générer que les connaissances nécessaires à la résolution d'un problème d'adaptation donné. La validation des connaissances extraites est réalisée en ligne par l'utilisateur du système.

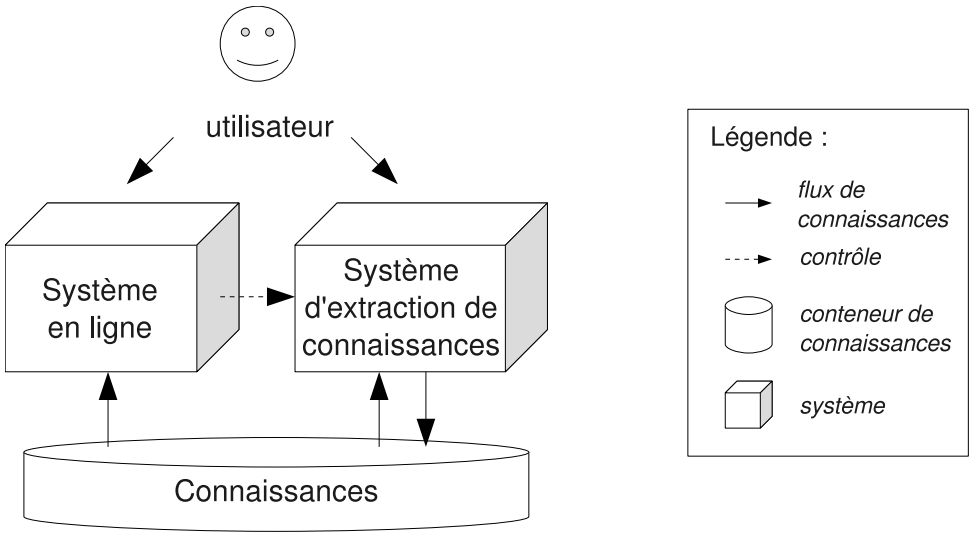


FIG. 7.1 – Pilotage du processus d'extraction de connaissances par le système en ligne.

Fournir une assistance à l'utilisateur. Le processus d'extraction de connaissances est utilisé pour fournir une assistance à l'utilisateur dans la formulation de nouvelles connaissances. Les connaissances extraites de la base de cas le sont dans un format directement utilisable par le système. L'utilisation de ces connaissances par le système lui permet de proposer une nouvelle solution pour le problème `cible`. Chaque élément de connaissance extrait peut ainsi être présenté à l'utilisateur pour validation conjointement à la nouvelle solution proposée par le système.

7.3 Description du processus d'acquisition de connaissances

Dans cette section est décrit le processus d'acquisition interactive et opportuniste de connaissances d'adaptation dans lequel s'intègre l'étape d'extraction de connaissances. Ce processus est déclenché lors de la phase de test de la solution proposée par le système, lorsque celle-ci ne satisfait pas l'utilisateur. Le but de l'étape d'extraction de connaissances est de rechercher dans la base de cas les connaissances nécessaires à l'instanciation d'une stratégie de réparation.

Réparation = explication de l'échec + choix d'une stratégie de réparation. L'approche proposée pour la réparation de l'échec d'adaptation est inspirée de la méthode d'adaptation par critiques présentée dans le système CHEF [Hammond, 1986]. Dans CHEF, l'adaptation est simulée et un modèle causal est utilisé pour détecter et expliquer les problèmes susceptibles d'être rencontrés dans la solution proposée. Pour résoudre ces problèmes, CHEF utilise un ensemble de critiques. Une critique identifie un ensemble de problèmes susceptibles d'être rencontés dans la solution proposée et y associe des stratégies de réparation. Comme nous l'avons vu, CHEF est capable d'apprendre de nouvelles critiques lorsqu'il réussit à réparer une adaptation. Pour réparer l'adaptation, l'approche suivie consistera donc à exploiter l'interaction avec l'utilisateur pour déterminer une explication de l'échec d'adaptation. En fonction de l'explication obtenue, un ensemble de stratégies de réparation est proposé à l'utilisateur.

Un processus d'acquisition de connaissances itératif. Le processus d'acquisition de connaissances est itératif : chaque fois que des connaissances d'adaptation sont acquises, celles-ci sont utilisées pour relancer l'adaptation et proposer une nouvelle solution à l'utilisateur. Lorsque la solution proposée satisfait l'utilisateur, les connaissances acquises sont mémorisées dans la base de connaissances d'adaptation `CA` du système de façon à pouvoir être utilisées pour résoudre de nouveaux problèmes.

Les différentes étapes du processus d'acquisition de connaissances sont résumées dans la figure 7.2.

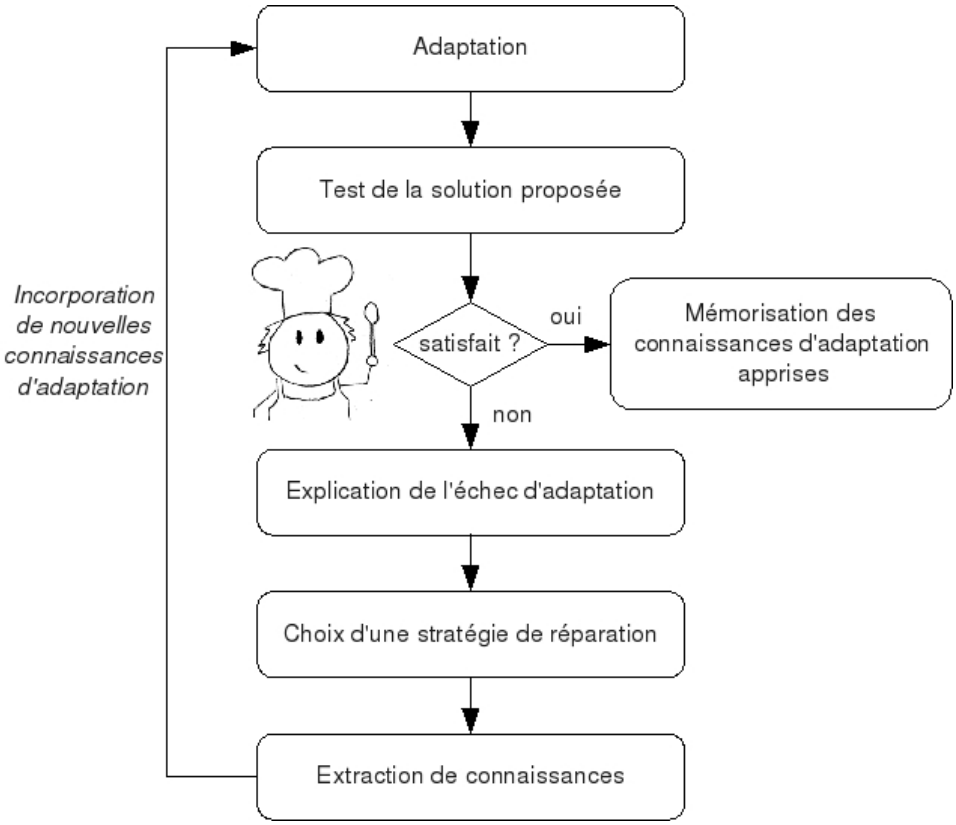


FIG. 7.2 – Les différentes étapes du processus d'acquisition opportuniste de connaissances.

7.4 Déclenchement opportuniste du processus d'extraction de connaissances

Le processus d'extraction de connaissances est déclenché lorsqu'une stratégie de réparation a été choisie. Son but est alors de rechercher dans la base de cas les informations nécessaires à l'instanciation d'une stratégie abstraite de réparation.

L'approche suivie ici est proche de l'approche *adaptation = tranformation + exploration de la mémoire* utilisée dans le système DIAL [Leake *et al.*, 1996]. Dans DIAL, l'adaptation à partir de règles se fait par la sélection d'une règle abstraite puis par la recherche en mémoire des informations nécessaires à son instanciation. Dans notre approche, une stratégie de réparation définit un ensemble de contraintes sur la règle d'adaptation à appliquer. Ces contraintes servent ensuite

à restreindre l'espace des hypothèses du processus d'extraction de connaissances. CabamakA extrait à partir de la base de cas un ensemble de règles d'adaptation qui satisfont ces contraintes.

7.5 Paramétrage du processus d'extraction de connaissances

Le paramétrage du processus d'extraction de connaissances s'effectue lors de l'étape de préparation des données, lors de l'étape de fouille et lors de l'étape de filtrage des résultats.

Filtrage lors de l'étape de préparation des données. Un filtre est appliqué sur l'ensemble d'apprentissage lors de l'étape de préparation des données. Ce filtre a pour but de restreindre l'ensemble d'apprentissage de façon à ne comparer que deux ensembles de cas sources. Pour cela, une règle d'adaptation $\text{RA}_{\mathcal{E}}$ est formée à l'issue de l'étape de choix d'une stratégie de réparation du processus d'acquisition de connaissances. L'ensemble d'apprentissage $\mathcal{E}$ est alors formé par toutes les représentations $\text{RA}_{k\ell}$ de variations entre cas sources qui sont plus spécifiques que la variation $\text{RA}_{\mathcal{E}}$, c'est-à-dire qui vérifient $\text{RA}_{k\ell} \vDash \text{RA}_{\mathcal{E}}$.

Paramétrage de l'étape de fouille. L'étape de fouille est paramétrée en agissant sur la valeur du support minimum σ_s.

Filtrage des résultats de la fouille. Deux filtres sont également appliqués sur l'ensemble des règles extraites :

1. Un premier filtre exploite le fait que la variation $\text{RA}_{\mathcal{E}}$ exprime un ensemble de contraintes sur la règle d'adaptation à appliquer : les règles d'adaptation extraites sont filtrées pour ne conserver que les règles RA qui sont plus spécifiques que $\text{RA}_{\mathcal{E}}$, c'est-à-dire les règles qui vérifient $\text{RA} \vDash \text{RA}_{\mathcal{E}}$. Les règles d'adaptation extraites respectent donc les contraintes exprimées par $\text{RA}_{\mathcal{E}}$ et peuvent être utilisées pour instancier la stratégie de réparation choisie.
2. Un filtre est appliqué de façon à ne garder que les règles d'adaptation candidates à la résolution du problème d'adaptation $\mathcal{A}_{\texttt{pb}} = (\texttt{srce}, \texttt{Sol(srce)}, \texttt{cible}) \in \mathcal{L}_{\texttt{pb}} \times \mathcal{L}_{\texttt{sol}} \times \mathcal{L}_{\texttt{pb}}$ considéré. Ce deuxième filtrage effectué sur les résultats de la fouille a pour but de ne conserver que les règles d'adaptation extraites qui peuvent être utilisées pour résoudre le problème d'adaptation considéré. Les règles d'adaptation obtenues peuvent donc être utilisées pour relancer l'adaptation et proposer une nouvelle solution à l'utilisateur.

7.6 Application à Taaable

Cette section décrit le processus de découverte opportuniste de connaissances d'adaptation lorsqu'il est appliqué à l'application Taaable. Le système WikiTaaable [Badra *et al.*, 2009a], présenté pour la deuxième édition du *Computer Cooking Contest* qui s'est tenu à Seattle en juillet 2009 [Delany, 2009], a été développé sur les principes du système Taaable présenté au chapitre 2. Ce système inclut une implémentation du processus de découverte opportuniste de connaissances d'adaptation. Les différentes étapes du processus sont illustrées sur un exemple. Les copies d'écran présentées dans cette section sont celles de son implémentation dans le système WikiTaaable.

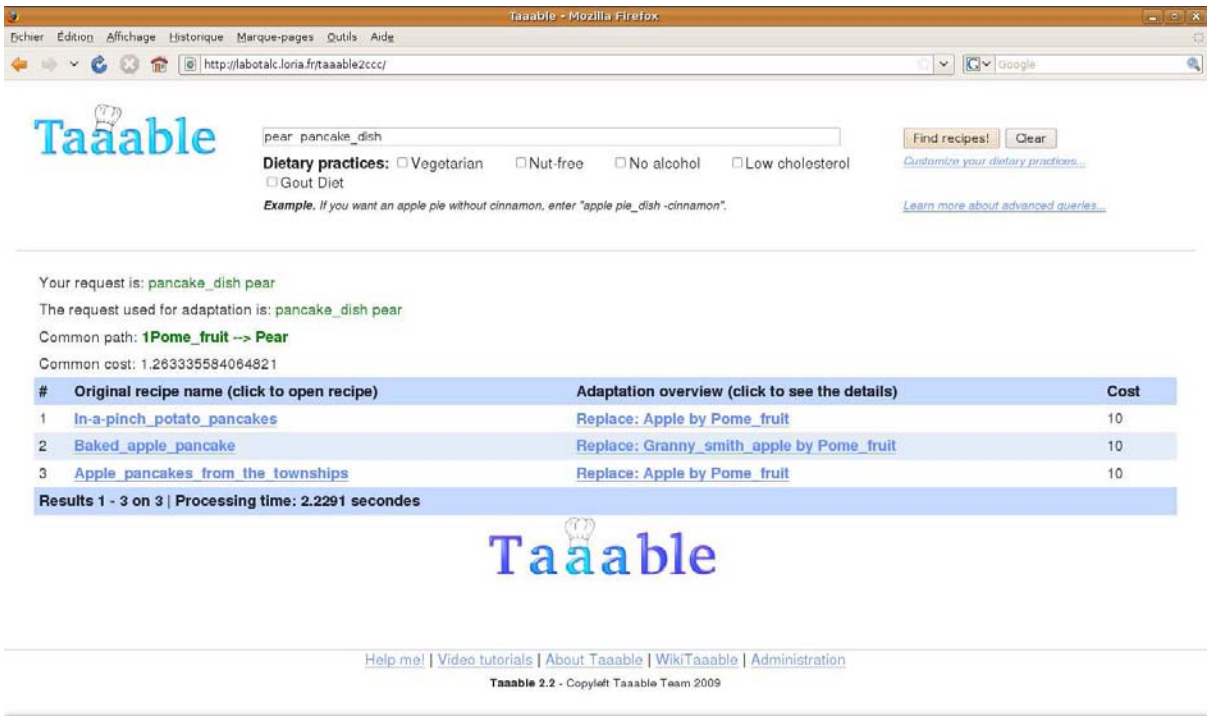


FIG. 7.3 – Copie d'écran de l'interface de WIKITAAABLE à l'issue de l'étape de remémoration. Trois recettes ont été remémorées, parmi lesquelles figure la recette *Apple pancakes from the townships*.

Présentation de l'exemple. L'exemple choisi consiste à résoudre la requête test « Je veux une recette de pancake aux poires. » qui a été proposée par les organisateurs du deuxième *Computer Cooking Contest*. Dans cet exemple, le problème cible est :

$$\texttt{cible} = \texttt{pancake} \wedge \texttt{poire}$$

Remémoration. Dans TAAABLE, l'étape de remémoration produit un chemin de similarité. Celui-ci prend la forme d'une composition de substitutions $\texttt{ChSim} = \gamma_q \circ \gamma_{q-1} \circ \cdots \circ \gamma_1$ telle qu'il existe au moins une recette $\texttt{Sol(srce)}$ qui satisfasse la requête modifiée $\texttt{srce} = \gamma_q(\gamma_{q-1}(\cdots \gamma_1(\texttt{cible})\cdots))$, c'est-à-dire qui vérifie $\texttt{Sol(srce)} \models_O \texttt{srce}$. Un chemin de similarité ChSim peut s'écrire :

$$\texttt{Sol(srce)} \models_O \texttt{srce} \xleftarrow{\gamma_q} \xleftarrow{\gamma_{q-1}} \cdots \xleftarrow{\gamma_1} \texttt{cible}$$

Dans l'exemple, le chemin de similarité produit est $\texttt{ChSim} = \gamma$, où $\gamma = \texttt{poire} \rightsquigarrow \texttt{piridion}$ est une substitution par généralisation qui est générée automatiquement à partir de l'axiome $\texttt{poire} \Rightarrow \texttt{piridion}$ présent dans l'ontologie $\mathcal{O}$. La substitution γ exprime le fait que les poires sont des piridions. La recette de pancake aux pommes *Apple pancakes from the townships* est remémorée (figure 7.3) ; sa représentation $\texttt{Sol(srce)}$ résoud la requête srce ($\texttt{Sol(srce)} \models_O \texttt{srce}$) car $\texttt{Sol(srce)}$ contient l'ingrédient pomme et $\texttt{pomme} \models_O \texttt{piridion}$ (les pommes sont des piridions).

Adaptation. L'étape d'adaptation produit alors un chemin d'adaptation ChAd qui prend la forme d'une composition de substitutions $\texttt{ChAd} = \sigma_1 \circ \sigma_2 \circ \cdots \circ \sigma_q$ telle que la recette modifiée

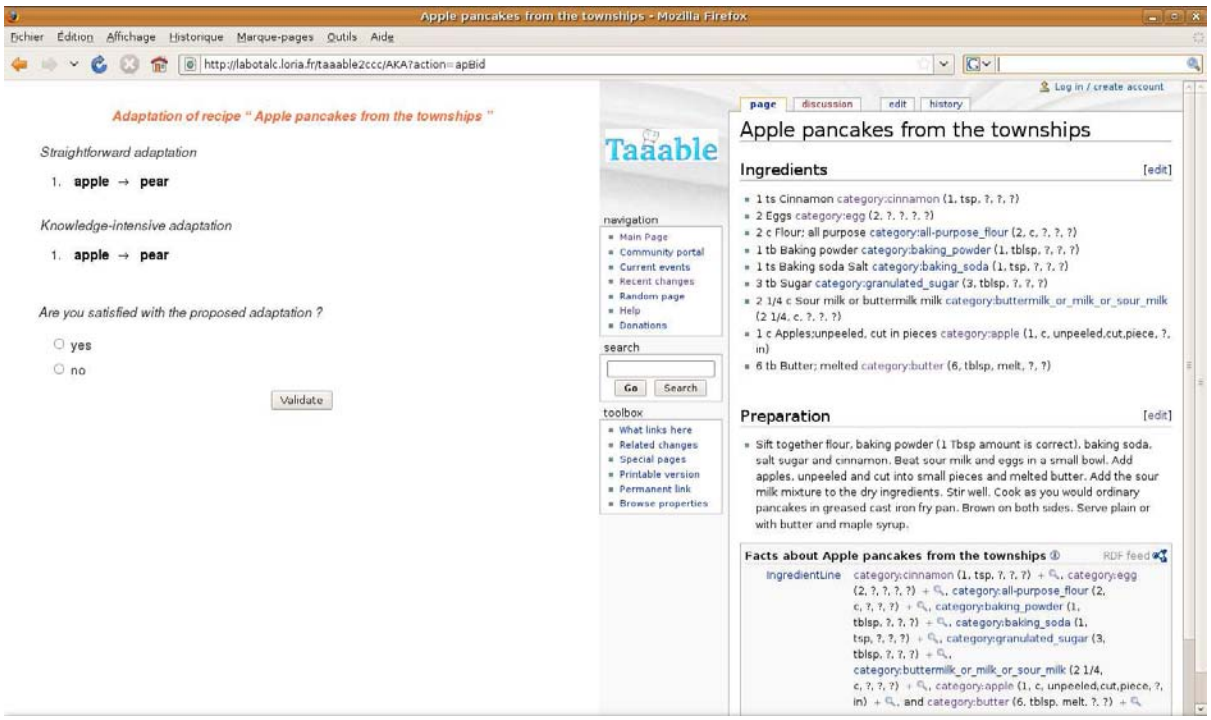


FIG. 7.4 – Copie d'écran de l'interface de l'interface d'acquisition de connaissances d'adaptation du système WIKITAAABLE lors de l'étape de test de la solution proposée. La partie droite de la figure reproduit la page du wiki décrivant la recette *Apple pancakes from the townships*, tandis que la partie gauche décrit l'adaptation de cette recette proposée par le système. Le chemin d'adaptation est donné sous la forme d'un ensemble de substitutions d'ingrédients. Deux chemins d'adaptation sont donnés ici : le chemin d'adaptation généré uniquement à partir du chemin de similarité (*Straightforward adaptation*), c'est-à-dire sans les connaissances d'adaptation $\texttt{CA}$ acquises au cours du processus de découverte opportuniste de connaissances d'adaptation, et le chemin d'adaptation généré en utilisant les connaissances d'adaptation $\texttt{CA}$ (*Knowledge-intensive adaptation*). À ce stade, $\texttt{CA} = \emptyset$, donc les deux chemins d'adaptation sont égaux.

$\widetilde{\texttt{Sol}}(\texttt{cible}) = \sigma_1(\sigma_2(\cdots\sigma_q(\texttt{Sol}(\texttt{srce}))\cdots))$ satisfait la requête $\texttt{cible}$ initiale, c'est-à-dire vérifie $\widetilde{\texttt{Sol}}(\texttt{cible}) \models_O \texttt{cible}$. Un chemin d'adaptation peut s'écrire :

$$\texttt{Sol}(\texttt{srce}) \xrightarrow{\sigma_q} \xrightarrow{\sigma_{q-1}} \cdots \xrightarrow{\sigma_1} \widetilde{\texttt{Sol}}(\texttt{cible}) \models_O \texttt{cible}$$

Dans l'exemple, un chemin adaptation $\texttt{ChAd} = \sigma$ est proposé dans lequel la substitution $\sigma = \texttt{pomme} \rightsquigarrow \texttt{poire}$ est générée automatiquement à partir de la substitution γ.

Test de la solution proposée. Lors de la phase de test de la solution proposée, la recette remémorée $\texttt{Sol}(\texttt{srce})$ est présentée à l'utilisateur ainsi que le chemin d'adaptation $\texttt{ChAd}$ qui doit être appliqué à $\texttt{Sol}(\texttt{srce})$ pour produire la solution $\widetilde{\texttt{Sol}}(\texttt{cible})$. Une copie d'écran de l'interface d'acquisition de connaissances d'adaptation du système WIKITAAABLE lors de la phase de test de la solution proposée est donnée par la figure 7.4. L'adaptation proposée par le système est résumée par la figure 7.5.

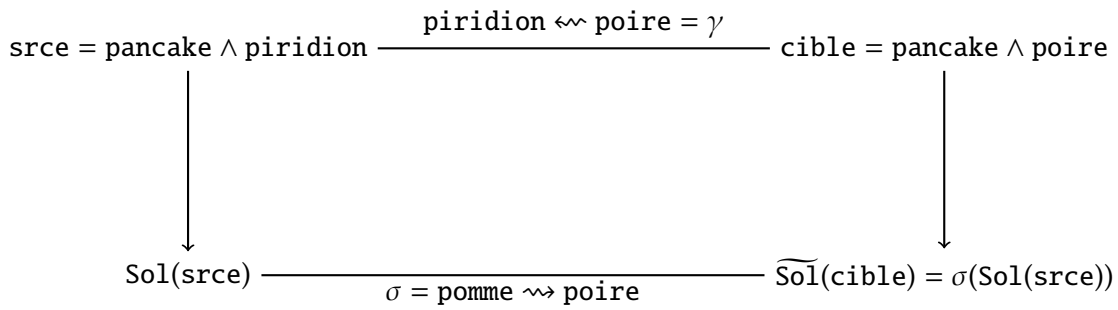
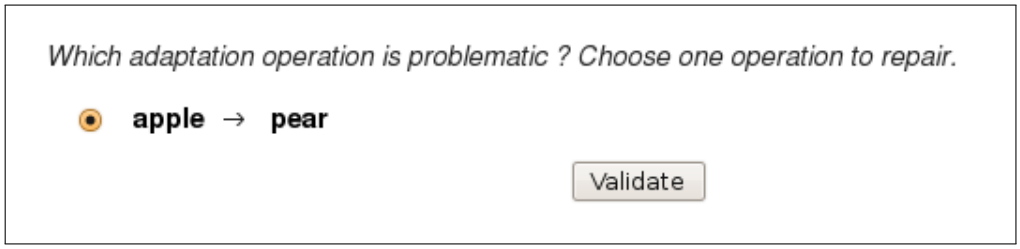


FIG. 7.5 – Une première adaptation pour résoudre la requête « Je veux une recette de pancake aux poires. ». La solution $\widetilde{\texttt{Sol}}(\texttt{cible})$ est construite à partir de la représentation Sol(srce) de la recette remémorée *Apple pancakes from the townships*.

Explication de l'échec d'adaptation. Lorsque l'utilisateur n'est pas satisfait par l'adaptation proposée, il est encouragé à formuler une explication de l'échec d'adaptation. L'explication de l'échec est formulé en trois étapes :

1. **Sélection d'une substitution**. L'utilisateur sélectionne une substitution $\sigma = \texttt{A} \rightsquigarrow \texttt{B}$ de ChAd qui pose problème, où A et B sont des conjonctions de littéraux. Dans l'exemple, l'utilisateur sélectionne la substitution $\sigma = \texttt{pomme} \rightsquigarrow \texttt{poire}$, donc A = pomme et B = poire.

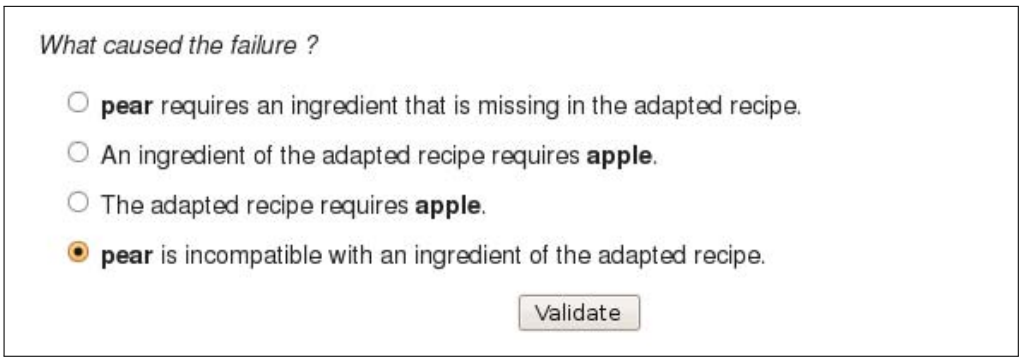


2. **Explication de l'échec d'adaptation**. Le but de l'étape d'explication de l'échec d'adaptation est d'instancier, à travers l'interaction avec l'utilisateur, un patron abstrait d'explication. Chaque patron d'explication ne s'instancie que pour expliquer l'échec d'une seule étape d'un chemin d'adaptation, dans laquelle une solution Sol(pb) d'un problème pb est transformée en une solution $\sigma(\texttt{Sol}(\texttt{pb}_))$ par l'application d'une substitution σ. Les trois patrons abstraits d'explication qui ont été considérés pour l'application WIKITAAABLE sont présentés dans la première colonne de la table 7.1.

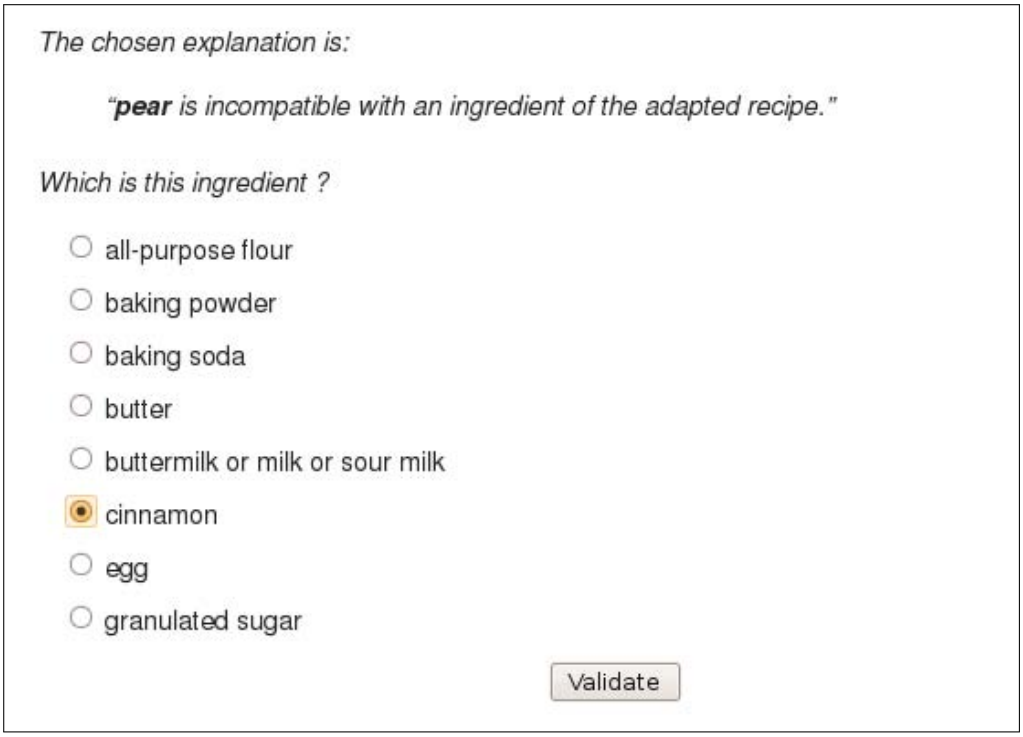
Patron d'explication	Type de dépendance	Cause de l'échec
Un ingrédient x de `B` nécessite un autre ingrédient y qui est absent de σ(`Sol(pb)`).	« x nécessite y »	σ(`Sol(pb)`) $\models_O x$ σ(`Sol(pb)`) $\not\models_O y$
Un ingrédient x de σ(`Sol(pb)`) nécessite un ingrédient y de `A` qui vient d'être supprimé.		
Un ingrédient x de `B` est incompatible avec un ingrédient y de σ(`Sol(pb)`).	« x et y incompatibles »	σ(`Sol(pb)`) $\models_O x$ σ(`Sol(pb)`) $\models_O y$

TAB. 7.1 – Résumé des patrons d'explication envisagés dans WIKITAAABLE. Un patron d'explication décrit un échec d'adaptation résultant de l'application d'une substitution σ = `A` $\rightsquigarrow$ `B` à la solution `Sol(pb)` d'un problème `pb`.

Chacun de ces patrons d'explication exprime une dépendance entre ingrédients qui n'est pas respectée par la recette σ(`Sol(pb)`) obtenue. Par exemple, le troisième patron d'explication de la table 7.1 exprime le fait qu'un ingrédient x ajouté à la recette suite à l'application de la substitution σ est incompatible avec l'un des ingrédients de la recette obtenue σ(`Sol(pb)`). Dans l'exemple, c'est ce patron qui est sélectionné par l'utilisateur parmi un ensemble de suggestions, avec x = `poire`.



3. **Identification des ingrédients mis en jeu**. L'utilisateur sélectionne les ingrédients x et y dans une liste de propositions. Dans l'exemple, l'utilisateur sélectionne les ingrédients x = `poire` et y = `cannelle`. Il exprime ainsi le fait que l'ajout de poires dans la recette remémorée est incompatible avec la conservation de la cannelle déjà présente dans la recette.



Choix d'une stratégie de réparation. À chaque patron d'explication est associé un ensemble de stratégies de réparation (table 7.2). L'étape de choix d'une stratégie de réparation consiste à

Explications possibles	Stratégies de réparation
Un ingrédient x de `B` nécessite un autre ingrédient y qui est absent de σ(`Sol(pb)`).	– si `cible` $\nvDash_O x$, supprimer x – si `cible` $\nvDash_O x$, trouver un substitut à x – si `cible` $\nvDash_O \neg y$, ajouter y – si `cible` $\vDash_O \neg y$, ajouter un substitut à y
Un ingrédient x de σ(`Sol(pb)`) nécessite un ingrédient y de `A` qui vient d'être supprimé.	
Un ingrédient x de `B` est incompatible avec un ingrédient y de σ(`Sol(pb)`).	– si `cible` $\nvDash_O x$, supprimer x – si `cible` $\nvDash_O x$, trouver un substitut à x – si `cible` $\nvDash_O y$, supprimer y – si `cible` $\nvDash_O y$, trouver un substitut à y

Tab. 7.2 – Résumé des différentes stratégies de réparation envisagées dans WikiTaaable.

sélectionner parmi ces stratégies celles qui sont applicables puis à les soumettre au choix de l'utilisateur[2]. Dans l'exemple, quatre stratégies de réparation correspondent au patron d'explication

[2]Remarquons qu'ici au moins une stratégie de réparation s'applique quel que soit le patron d'explication qui a été sélectionné. En effet, si le patron d'explication correspond à une dépendance de la forme « x nécessite y », on a forcément `cible` $\vDash_O \neg y$ ou `cible` $\nvDash_O \neg y$. Si le patron d'explication correspond à une dépendance de la forme « x et y incompatibles », et que l'on a à la fois `cible` $\vDash_O x$ et `cible` $\vDash_O y$, alors l'ensemble de contraintes énoncées par

sélectionné mais toutes ne sont pas applicables. En effet, comme la requête cible impose que la recette adaptée contienne des poires ($\texttt{cible} \vDash_O x$ avec $x = \texttt{poire}$), seules les deux dernières peuvent être appliquées (table 7.3).

Type de dépendance	Cause de l'échec	Stratégies de réparation
« poire et cannelle incompatibles »	$\sigma(\texttt{Sol(pb)}) \vDash_O \texttt{poire}$ $\sigma(\texttt{Sol(pb)}) \vDash_O \texttt{cannelle}$	– supprimer cannelle – trouver un substitut à cannelle

TAB. 7.3 – Stratégies de réparation envisagées dans l'exemple de la recette de pancake aux poires (application de la table 7.2 à l'exemple).

Dans TAAABLE, deux types de stratégies de réparation ont été considérés : les stratégies d'ajout ou de suppression d'un ingrédient et les stratégies de substitution d'ingrédients. Dans ce qui suit, ces deux types de stratégies sont décrits et illustrés sur deux scénarii : le scénario dans lequel l'utilisateur choisit la stratégie de réparation « supprimer cannelle », qui correspond à une stratégie de suppression d'un ingrédient, et le scenario dans lequel l'utilisateur choisit la stratégie de réparation « trouver un substitut à la cannelle », qui correspond à une stratégie de substitution d'ingrédients.

Stratégies d'ajout ou de suppression d'un ingrédient

Les stratégies d'ajout ou de suppression d'un ingrédient peuvent être instanciées directement : le système génère une nouvelle substitution σ' corrigée à partir de la substitution $\sigma = \texttt{A} \rightsquigarrow \texttt{B}$ à réparer. Si $\sigma = \texttt{pomme} \rightsquigarrow \texttt{poire}$ et que la stratégie de réparation choisie consiste à supprimer l'ingrédient $y = \texttt{cannelle}$, une nouvelle substitution

$$\sigma' = \texttt{pomme} \wedge \texttt{cannelle} \rightsquigarrow \texttt{poire}$$

est générée pour réparer σ. La substitution σ' est alors utilisée pour relancer l'adaptation et générer une nouvelle solution $\widetilde{\texttt{Sol}}(\texttt{cible})$. L'adaptation réparée est résumée par la figure 7.6.

Test de la solution proposée. Lors de la phase de test de la solution proposée, la recette remémorée $\texttt{Sol}(\texttt{srce})$ est présentée à l'utilisateur ainsi que le nouveau chemin d'adaptation $\texttt{ChAd}' = \sigma'$ qui doit être appliqué à $\texttt{Sol}(\texttt{srce})$ pour produire la solution $\widetilde{\texttt{Sol}}(\texttt{cible})$. La figure 7.7 présente une copie d'écran de l'interface d'acquisition de connaissances d'adaptation du système WIKITAAABLE lors de la phase de test de la solution corrigée.

l'utilisateur à travers la formulation de la requête cible seraient jugées incompatibles par ce même utilisateur.

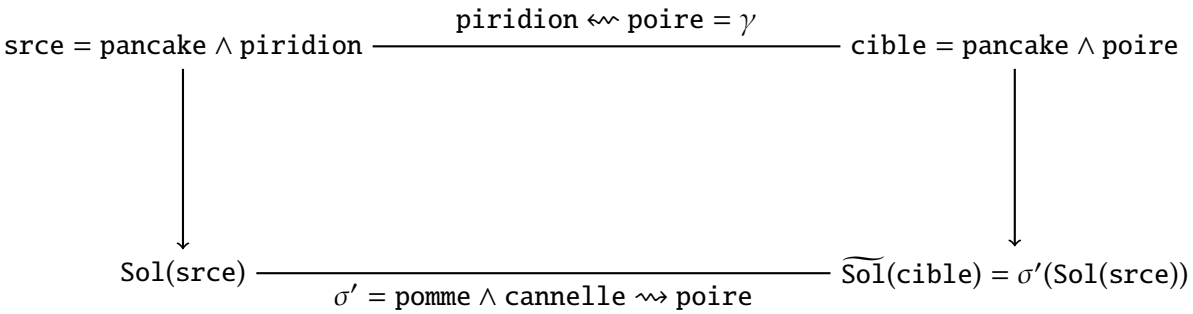


FIG. 7.6 – Une adaptation réparée qui résoud la requête « Je veux une recette de pancake aux poires. » Dans cette nouvelle adaptation, les pommes et la cannelle sont substituées dans la recette remémorée par des poires.

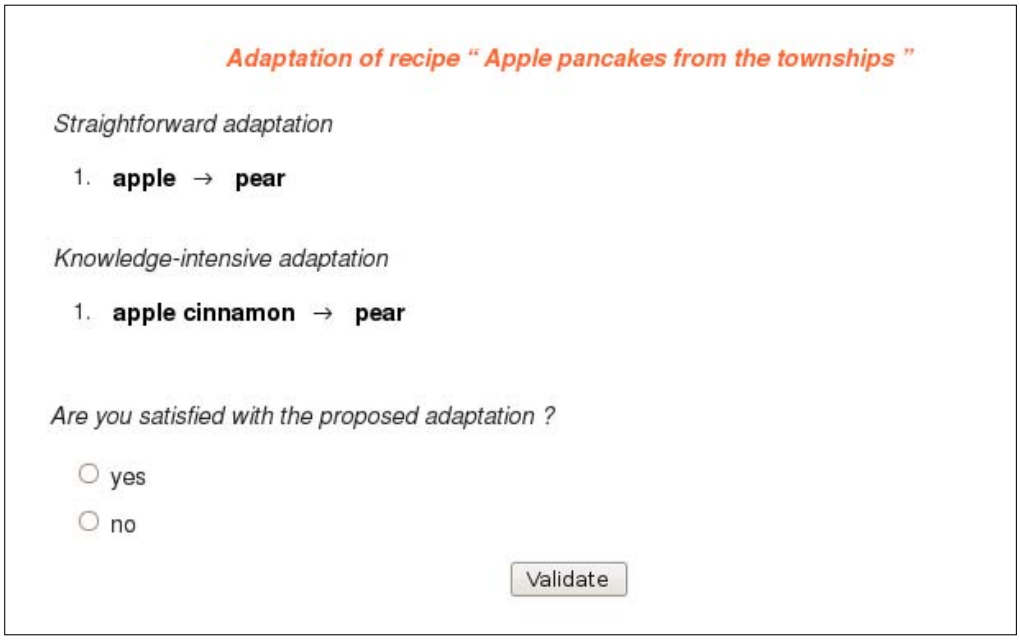
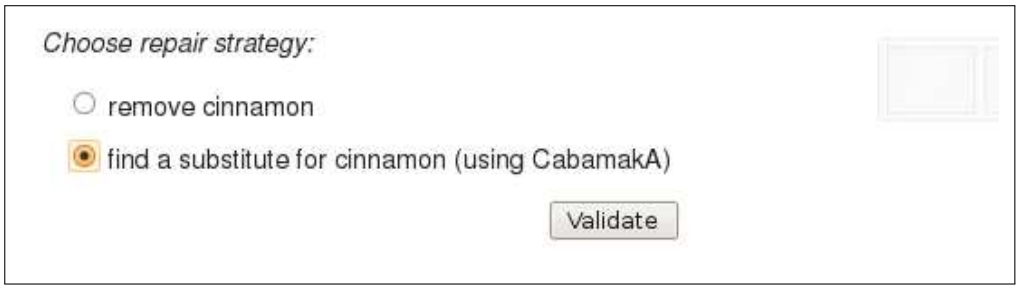


FIG. 7.7 – Proposition d'une nouvelle adaptation à l'utilisateur dans laquelle la cannelle est supprimée. L'adaptation a été relancée en utilisant un ensemble CA de connaissances d'adaptation acquises qui n'est pas vide : CA = $\{\sigma'\}$ avec σ' = pomme ∧ cannelle ⇝ poire.

Mémorisation des connaissances acquises. Lorsque l'utilisateur est satisfait par la solution proposée, les connaissances d'adaptation sont mémorisées dans la base de connaissances d'adaptation CA. Dans l'exemple, c'est la substitution σ' qui est mémorisée dans CA.

Stratégies de substitution d'ingrédients



CABAMAKA est déclenché de manière opportuniste pour instancier une stratégie de réparation par substitution d'ingrédient. Le processus d'extraction de connaissances a pour but d'apprendre un opérateur de substitution d'ingrédients σ' à partir de la comparaison de deux ensembles de recettes. Pour cela, une variation $\Delta_{\mathcal{E}}$ est générée à partir de σ et des ingrédients x et y qui définit un ensemble de contraintes que doit satisfaire la substitution σ'. Par exemple, si $\sigma = \texttt{pomme} \rightsquigarrow \texttt{poire}$ et que la stratégie de réparation choisie consiste à substituer l'ingrédient $y = \texttt{cannelle}$ de la recette remémorée par un autre ingrédient, la variation $\Delta_{\mathcal{E}}$ générée sera $\Delta_{\mathcal{E}} = \texttt{pomme}^- \wedge \texttt{cannelle}^- \wedge \texttt{poire}^+$. Cette variation correspond à des substitutions σ' cherchées de la forme :

$$\sigma' = \texttt{pomme} \wedge \texttt{cannelle} \wedge \textit{quelque chose}_1 \rightsquigarrow \texttt{poire} \wedge \textit{quelque chose}_2$$

où $\textit{quelque chose}_1$ et $\textit{quelque chose}_2$ sont des conjonctions de littéraux positifs à déterminer.

Paramétrage du processus d'extraction de connaissances. Dans TAAABLE, un ensemble d'apprentissage $\mathcal{E} = \{(\top_{\Delta\texttt{pb}}, \Delta\texttt{sol}_{k\ell})\}_{kl}$ est constitué lors de la phase de préparation des données en sélectionnant dans la base de recettes un ensemble de couples $(\texttt{Sol}(\texttt{srce}_k), \texttt{Sol}(\texttt{srce}_\ell))$ de recettes et en représentant la variation $\Delta\texttt{sol}_{k\ell} \in \mathcal{L}_{\Delta\texttt{sol}}$ entre les recettes $\texttt{Sol}(\texttt{srce}_k)$ et $\texttt{Sol}(\texttt{srce}_\ell)$. Puis l'étape de fouille génère un ensemble de règles d'adaptation $(\top_{\Delta\texttt{pb}}, \Delta) \in \mathcal{L}_{\Delta\texttt{pb}} \times \mathcal{L}_{\Delta\texttt{sol}}$, dans lesquelles chaque variation Δ obtenue est interprétée comme une substitution $\sigma' = \texttt{A}' \rightsquigarrow \texttt{B}'$, où $\texttt{A}'$ et $\texttt{B}'$ sont des conjonctions de littéraux.

Le paramétrage du processus d'extraction de connaissances s'effectue :

- **lors de l'étape de préparation des données** : $\Delta_{\mathcal{E}}$ est utilisée pour restreindre l'ensemble d'apprentissage $\mathcal{E}$. Celui-ci est alors formé par toutes les règles d'adaptation $(\top_{\Delta\texttt{pb}}, \Delta\texttt{sol}_{k\ell})$ pour lesquelles la représentation $\Delta\texttt{sol}_{k\ell}$ de variations entre les recettes $\texttt{Sol}(\texttt{srce}_k)$ et $\texttt{Sol}(\texttt{srce}_\ell)$ est plus spécifique que la variation $\Delta_{\mathcal{E}}$, c'est-à-dire vérifie $\Delta\texttt{sol}_{k\ell} \vDash \Delta_{\mathcal{E}}$.
- **lors de l'étape de fouille**, en agissant sur la valeur du support minimum σ_s,
- **lors de l'étape de filtrage des résultats**, par un filtrage de l'ensemble des variations $\Delta \in \mathcal{L}_{\Delta\texttt{sol}}$ extraites pour ne garder :
  1. que les variations $\Delta \in \mathcal{L}_{\Delta\texttt{sol}}$ satisfaisant les contraintes exprimées par la variation $\Delta_{\mathcal{E}}$: les variations extraites sont filtrées pour ne conserver que les variations Δ qui sont plus spécifiques que $\Delta_{\mathcal{E}}$, c'est-à-dire qui vérifient $\Delta \vDash \Delta_{\mathcal{E}}$.
  2. que les variations $\Delta \in \mathcal{L}_{\Delta\texttt{sol}}$ correspondant à des substitutions $\sigma' = \texttt{A}' \rightsquigarrow \texttt{B}'$ qui peuvent être utilisées pour modifier la recette remémorée $\texttt{Sol}(\texttt{srce})$, c'est à dire qui vérifient (1) $\texttt{Sol}(\texttt{srce}) \vDash_{\mathcal{O}} \texttt{A}'$, et (2) pour toute variable propositionnelle $\texttt{a} \in \mathcal{V}$, $\texttt{Sol}(\texttt{srce}) \nvDash_{\mathcal{O}} \texttt{a}$ si $\texttt{A}' \nvDash_{\mathcal{O}} \texttt{a}$ et $\texttt{B}' \vDash_{\mathcal{O}} \texttt{a}$.

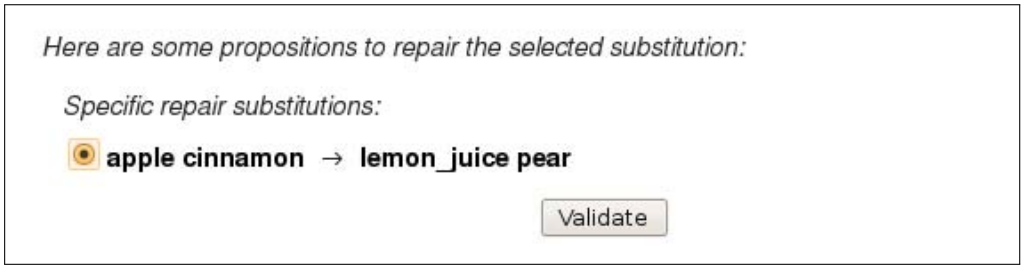
Dans l'exemple, l'utilisateur sélectionne la substitution

$$\sigma' = \texttt{pomme} \wedge \texttt{cannelle} \rightsquigarrow \texttt{poire} \wedge \texttt{citron}$$

qui correspond à la variation extraite $\Delta = \texttt{pomme}^- \wedge \texttt{cannelle}^- \wedge \texttt{poire}^+ \wedge \texttt{citron}^+$.

Spécialisation de la variation Δ extraite. La variation Δ vérifie les contraintes exprimées par la stratégie de réparation choisie ($\Delta \vDash \Delta_{\mathcal{E}}$) et est applicable pour modifier la recette remémorée `Sol(srce)`. Néanmoins, cette variation correspond à une règle d'adaptation assez générale, dans la mesure où le concept `citron` possède plusieurs sous-concepts dans l'ontologie, comme `zest-de-citron` ou `jus-de-citron`. C'est pourquoi il est proposé à l'utilisateur de spécialiser la variation Δ en remplaçant dans Δ le concept `citron` par un de ses sous-concepts. Pour cela, un algorithme d'extraction de motifs fréquents est appliqué sur l'ensemble des recettes pour déterminer quels sous-concepts du concept `citron` co-occure le plus souvent avec des poires dans la base de recettes. Le système identifie le jus de citron comme étant l'ingrédient le plus souvent présent avec des poires, et propose donc à l'utilisateur de spécialiser la variation Δ en la variation $\Delta_{\text{spécialisée}} = \texttt{pomme}^- \wedge \texttt{cannelle}^- \wedge \texttt{poire}^+ \wedge \texttt{jus-de-citron}^+$. La variation $\Delta_{\text{spécialisée}}$ est interprétée comme la substitution

$$\sigma' = \texttt{pomme} \wedge \texttt{cannelle} \rightsquigarrow \texttt{poire} \wedge \texttt{jus-de-citron}$$



La substitution σ' est alors utilisée pour relancer l'adaptation et générer une nouvelle solution $\widetilde{\texttt{Sol}}(\texttt{cible})$. L'adaptation réparée est résumée par la figure 7.8.

Test de la solution proposée. Lors de la phase de test de la solution proposée, la recette remémorée `Sol(srce)` est présentée à l'utilisateur ainsi que le nouveau chemin d'adaptation $\texttt{ChAd}' = \sigma'$ qui doit être appliqué à `Sol(srce)` pour produire la solution $\widetilde{\texttt{Sol}}(\texttt{cible})$. La figure 7.9 présente une copie d'écran de l'interface d'acquisition de connaissances d'adaptation du système WikiTaaable lors de la phase de test de la solution corrigée.

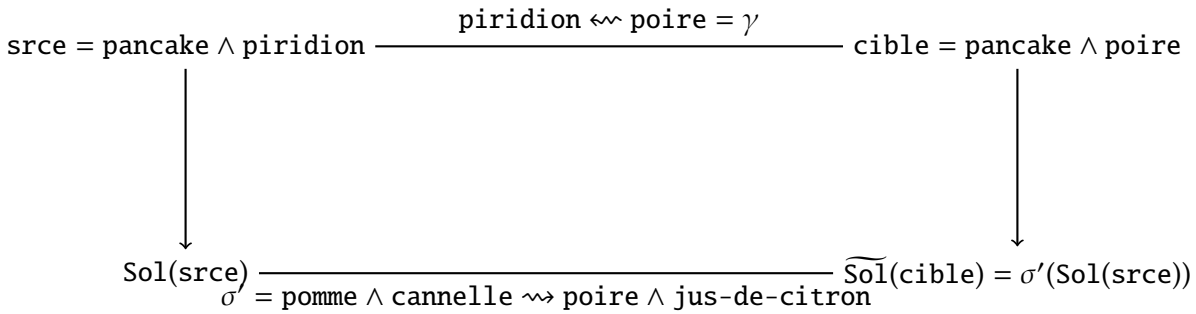


FIG. 7.8 – Une adaptation réparée qui résoud la requête « Je veux une recette de pancake aux poires ». Dans cette nouvelle adaptation, les pommes et la cannelle sont substituées dans la recette remémorée par des poires et du jus de citron.

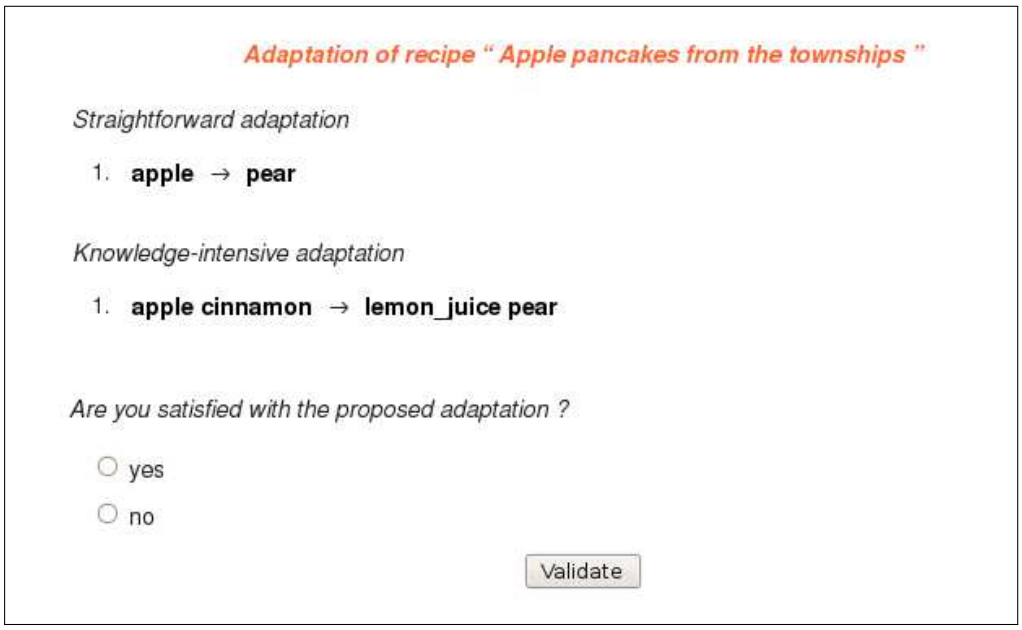


FIG. 7.9 – Proposition d'une nouvelle adaptation à l'utilisateur dans laquelle la cannelle est remplacée par du jus de citron. L'adaptation a été relancée en utilisant un ensemble CA de connaissances d'adaptation acquises qui n'est pas vide : CA = $\{\sigma'\}$ avec σ' = pomme ∧ cannelle ⇝ poire ∧ jus-de-citron.

Mémorisation des connaissances acquises. Lorsque l'utilisateur est satisfait par la solution proposée, les connaissances d'adaptation sont mémorisées dans la base de connaissances d'adaptation CA. Dans l'exemple, c'est la substitution σ' qui est mémorisée dans CA.

Conclusion et perspectives

En raisonnement à partir de cas, l'étape d'adaptation est à la fois cruciale et difficile à mettre en œuvre car elle nécessite des connaissances spécifiques au domaine qui doivent être acquises. Nous avons étudié dans cette thése comment des techniques d'extraction de connaissances peuvent être utilisées pour acquérir des connaissances d'adaptation. Un processus d'extraction de connaissances, appelé CabamakA, a été mis au point. Ce processus permet d'apprendre des connaissances d'adaptation à partir de la base de cas par généralisation à partir d'une représentation des variations entre cas. Nous avons discuté des conditions d'opérationnalisation de CabamakA au sein d'un processus d'acquisition de connaissances. L'étude aboutit à la proposition d'un nouveau type d'approche pour l'acquisition de connaissances d'adaptation dans lequel le processus d'extraction de connaissances est déclenché de manière opportuniste et interactive au cours d'une session particulière de résolution de problèmes. Dans cette approche, l'apprentissage a pour but d'acquérir les connaissances nécessaires à la résolution d'un problème donné. Le processus d'extraction de connaissances a été implémenté et appliqué dans deux domaines : la modification de recettes de cuisine et la gestion de connaissances décisionnelles en cancérologie.

Contributions

Les contributions de la thèse au raisonnement à partir de cas portent à la fois sur la théorie de l'adaptation et sur l'acquisition de connaissances d'adaptation.

Une nouvelle modélisation de l'adaptation est proposée qui s'appuie sur la définition d'un langage de représentation des variations entre cas. Un langage générique de représentation des variations entre cas est introduit dans lequel une variation est la représentation d'une relation binaire entre cas. Ce langage a en particulier été appliqué pour représenter les variations entre cas lorsque le langage de représentation des cas est de type attribut-contrainte. Nous avons montré que représenter les variations entre cas permet à la fois de structurer la base de cas par les connaissances d'adaptation et de représenter dans un langage unifié les similarités et dissimilarités entre cas et les connaissances d'adaptation. L'évaluation des similarités et différences entre cas est centrale en raisonnement à partir de cas. Elle intervient pour guider la remémoration car c'est elle qui permet de rapprocher le problème à résoudre des différents problèmes déjà résolus contenus dans la base de cas. Elle intervient aussi dans l'étape d'adaptation, car de l'existence de différences entre le problème à résoudre et les différents problèmes déjà résolus naît la nécessité d'effectuer une adaptation, qui a alors pour but de « résoudre ces différences ».

Une méthode d'acquisition de connaissances d'adaptation est également proposée dans laquelle les connaissances sont acquises en ligne et à partir de la base de cas par des techniques d'extraction de connaissances. Cette méthode combine deux types d'approches pour l'acquisition de connaissances d'adaptation : les approches semi-automatiques et les approches

interactives et opportunistes. Combiner ces deux types d'approches permet de tirer parti des avantages de chacune d'elles. Dans les approches interactives et opportunistes, l'expert est « en contexte » lorsqu'il valide les éléments de connaissances à acquérir et l'acquisition ne porte que sur des connaissances manquantes au système pour résoudre un problème donné. Dans l'approche proposée, les connaissances d'adaptation sont générées automatiquement à partir de la base de cas par application d'un processus d'extraction de connaissances. Celui-ci est alors vu comme un module additionnel au système en ligne, déclenché de manière opportuniste, et qui est destiné à fournir des suggestions à l'utilisateur dans la formulation de nouvelles connaissances. Des connaissances d'adaptation sont ainsi générées « à la volée » à partir de la base de cas, qui devient une source additionnelle de connaissances d'adaptation pour le système. Les connaissances extraites sont utilisées dans l'étape de réparation de la solution proposée par le système pour instancier une stratégie de réparation qui a été choisie en interaction avec l'utilisateur. Ainsi, le système bénéficie d'algorithmes efficaces d'apprentissage pour générer des connaissances d'adaptation et la charge dévolue à l'expert au sein du processus d'acquisition de connaissances est réduite du fait que celui-ci n'est sollicité que pour valider un ensemble d'éléments de connaissances présélectionnés.

Du point de vue de l'extraction de connaissances, les contributions se situent tant au niveau du processus d'extraction de connaissances proposé que de son opérationnalisation au cours d'une session de résolution de problèmes.

CabamakA est un processus original d'extraction de connaissances qui permet d'apprendre des connaissances d'adaptation à partir de la base de cas. Ce processus s'appuie sur un algorithme d'apprentissage par généralisation à partir d'une représentation des variations entre cas. CabamakA a été conçu de manière générique, c'est-à-dire indépendamment du langage utilisé pour représenter les cas et du langage choisi pour représenter les variations entre cas. Si l'implémentation du processus d'extraction de connaissances se limite actuellement au cas où les cas sont représentés dans un formalisme attribut-contrainte, ses principes théoriques lui permettent d'être appliqué pour fouiller les variations entre des cas représentés dans d'autres formalismes. CabamakA a d'ores et déjà permis de mettre à jour des co-variations complexes entre descripteurs problèmes et descripteurs solutions, tant dans le domaine de la cancérologie que dans le domaine culinaire.

L'approche est également originale par le rôle que joue l'extraction de connaissances dans le processus d'acquisition de connaissances. Comme les connaissances extraites par CabamakA sont destinées à être utilisées par un système de raisonnement à partir de cas pour résoudre des problèmes, le processus d'extraction de connaissances a été couplé au système en ligne et est piloté par ce dernier. L'étape de validation des résultats est alors pris en charge par l'utilisateur du système. Les principaux avantages d'un tel couplage sont que les connaissances extraites sont directement réutilisables par le système en ligne et que cela permet au système de raisonnement à partir de cas d'apprendre avec l'expérience. Dans cette perspective, nous avons montré comment le processus d'extraction de connaissances peut être paramétré par un contexte particulier de résolution de problèmes de façon à n'extraire de la base de cas que les connaissances manquantes au système pour résoudre un problème donné.

Perspectives

Les perspectives de recherche à donner à ce travail sont nombreuses. À court terme, les efforts devront être concentrés sur une validation expérimentale de l'approche proposée de découverte opportuniste de connaissances d'adaptation. Pour cela, des sessions d'acquisition

de connaissances devront être organisées auprès d'utilisateurs, ce qui nécessite de résoudre un certain nombre de problèmes d'interface homme-machine car l'interface d'acquisition de connaissances n'a pour l'instant été utilisée que par nos soins et à des fins de démonstration. Elle devra donc gagner en stabilité et en ergonomie, et devra permettre à l'utilisateur de naviguer parmi les règles extraites et de spécifier le contexte d'application d'une règle apprise. De plus, il sera nécessaire de se donner des moyens efficaces de gérer les connaissances acquises par le système, en établissant par exemple des priorités dans l'application des règles lorsque plusieurs règles s'appliquent. Pour l'application WikiTaaable, ces priorités sont gérées à l'aide d'une fonction de coût associée aux règles d'adaptation. Lorsque suffisamment de connaissances d'adaptation auront été acquises au cours de ces sessions d'acquisition de connaissances, plusieurs options sont envisageables pour valider l'approche expérimentalement. L'une d'elle consisterait à chercher à mesurer quantitativement l'augmentation des performances du système de raisonnement à partir de cas induite par l'ensemble de règles d'adaptation acquises. On pourra pour cela chercher à définir une mesure de similarité entre solutions, comme cela est fait dans [Craw *et al.*, 2006], et s'en servir pour évaluer la similarité entre la solution donnée par le système et une solution présente dans un jeu de test. Une autre forme de validation pourra consister à réaliser une enquête auprès d'utilisateurs du système WikiTaaable.

À plus long terme, les recherches théoriques sur la modélisation de l'adaptation en raisonnement à partir de cas doivent être poursuivies. On cherchera en particulier à faire le lien entre la modélisation proposée et d'autres théories de l'adaptation comme la théorie de l'adaptation par reformulations, la théorie de l'adaptation différentielle ou la théorie de l'adaptation conservatrice. Une première étape, qui permettra de faire le lien avec la théorie des reformulations, sera d'étudier comment doter le langage de representation des variations d'une loi de composition. La définition d'une telle loi aura pour but de définir une base théorique permettant de calculer la composée de deux variations. On pourra s'appuyer sur une étude préliminaire qui a déjà été menée dans ce sens dans [Tixier *et al.*, 2008]. Un objectif à plus long terme serait d'aller vers une théorie unifiée de l'adaptation.

Par ailleurs, l'application de notre méthode de découverte opportuniste de connaissances d'adaptation a permis de faire du système WikiTaaable un système évolutif qui apprend à la fois à partir de ses interactions avec ses utilisateurs et à partir de la base de cas (cf. chapitre 7).

Dans le système WikiTaaable, le processus d'extraction de connaissances CabamakA a été utilisé pour faire de la base de cas une source additionnelle de connaissances d'adaptation pour le système. Une autre direction de recherche consiste à étudier comment prendre en compte d'autres sources de connaissances d'adaptation que la base de cas. Des premiers pas ont déjà été effectués dans ce sens, puisque dans le système qui a été présenté en 2009 au *Computer Cooking Contest*, CabamakA utilisait comme source de connaissances d'adaptation une base de données de 87 000 recettes de cuisine indexées provenant du site `http://www.recipesource.com`. Une autre source de connaissances d'adaptation quasiment intarissable, mais encore peu étudiée, est le Web. Seuls quelques travaux traitent d'acquisition de connaissances d'adaptation sur le Web (voir par exemple [Leake et Powell, 2007], [Leake et Powell, 2008] ou [Ihle *et al.*, 2009]). La principale difficulté rencontrée pour exploiter le Web comme source de connaissances d'adaptation est de parvenir à sélectionner un ensemble de ressources pertinentes sur le Web et à définir précisément la sémantique de ces ressources. C'est pourquoi nous pensons que l'apport des technologies du Web Sémantique sera déterminante pour y arriver. En particulier, les systèmes de raisonnement à partir de cas devront être capable d'exploiter l'énorme quantité de données qui a déjà été publiée sur le Web en RDF (voir par exemple [Bizer *et al.*, 2009] pour un état de l'art des données actuellement disponibles sur le Web et des travaux menés pour les connecter entre elles à travers l'initiative *Linked Data*).

Plus généralement, les résultats obtenus encouragent à poursuivre la recherche vers la création de systèmes collaboratifs qui s'adaptent à leurs utilisateurs. Ces systèmes ont vocation à être plus flexibles dans la mesure où les connaissances qu'ils utilisent ne sont pas fixées à l'avance mais évoluent avec l'expérience. Pour cela, les systèmes de raisonnement à partir de cas pourront exploiter un ensemble de traces de leurs interactions avec leurs utilisateurs, comme suggéré dans [Cordier *et al.*, 2009], ou raisonner sur un ensemble d'expériences formé à la demande à partir d'expériences glanées sur le Web (sur des blogs, des forums, des Wikis par exemple), comme suggéré dans [Smyth et Champin, 2009]. Étudier comment les systèmes de raisonnement à partir de cas peuvent tirer parti des ressources disponibles sur le Web semble une direction de recherche très prometteuse, et a déjà donné lieu à un atelier lors de la conférence internationale sur le raisonnement à partir de cas en 2009 [Delany, 2009].

A

EdHibou : a Customizable Interface for Decision Support in a Semantic Portal

Ce chapitre présente les travaux effectués au cours de la thèse dans le domaine du Web Sémantique, qui pour une large part ont été consacrés à la réécriture de deux des composants essentiels du portail sémantique de Kasimir *: le serveur de connaissances et l'interface graphique* EdHibou. *Nous reproduisons l'article [Badra* et al.*, 2008b] qui décrit* EdHibou *et son intégration dans le portail sémantique de* Kasimir. *Cet article a été présenté sous forme de poster/démonstration lors de la conférence internationale sur le Web Sémantique (ISWC) en octobre* 2008.

❄

[Badra *et al.*, 2008b] F. Badra, M. d'Aquin, J. Lieber et T. Meilender. EdHibou : a customizable interface for decision support in a semantic portal. Dans *Proc. of the Poster and Demonstration Session at the 7th International Semantic Web Conference (ISWC2008)*, Karlsruhe, Germany, October 28. 2008.

Abstract

The Semantic Web is becoming more and more a reality, as the required technologies have reached an appropriate level of maturity. However, at this stage, it is important to provide tools facilitating the use and deployment of these technologies by end-users. In this paper, we describe EdHibou, an automatically generated, ontology-based graphical user interface that integrates in a semantic portal. The particularity of EdHibou is that it makes use of OWL reasoning capabilities to provide intelligent features, such as decision support, upon the underlying ontology. We present an application of EdHibou to medical decision support based on a formalization of clinical guidelines in OWL and show how it can be customized thanks to an ontology of graphical components.

A.1 Introduction

The Kasimir project is a multidisciplinary project which aims at providing oncology practitioners of the Lorraine region of France with decision support and knowledge management tools. The Kasimir system is a clinical decision support system which relies on the formalization of a set of clinical guidelines issued by the regional health network. It uses decision knowledge contained in an OWL ontology to provide decision support to clinicians. In such an ontology $\mathcal{O}$, a class `Patient` denotes the class of all patients, a class `Treatment` denotes the class of all treatments and a property `recommendation` links a class of patients to a class of recommended treatments. Then to a class `P` of patients is associated a treatment `T` by an axiom

$$\texttt{P} \sqsubseteq \exists \texttt{recommendation.T} \quad \text{where} \quad \begin{cases} \texttt{P} \sqsubseteq \texttt{Patient} \\ \texttt{T} \sqsubseteq \texttt{Treatment} \end{cases} \tag{A.1}$$

A medical situation is represented by an instance `a` of the class `Patient` in the ontology $\mathcal{O}$. The system then exploits axioms of the form (A.1) to associate a set of recommended treatments to the patient represented by `a`. Deciding which treatments to recommend to the patient represented by `a` amounts to finding the most specific atomic concepts `T` in $\mathcal{O}$ such that $\models_{\mathcal{O}} (\exists \texttt{recommendation.T})(\texttt{a})$ holds.

The original motivation when developing EdHibou was to provide a user interface for the Kasimir system that lets the user describe a medical situation for which a decision has to be taken. Such a graphical user interface should let the user complete the description of an OWL instance `a` and trigger some reasoning tasks on the underlying OWL representation of clinical guidelines to propose a set of recommended treatments. We built EdHibou as a generic framework, allowing application developers to generate customizable interfaces to ontologies and ontology reasoning. The Kasimir system takes advantage of this framework as an application of EdHibou. The key idea in EdHibou is to allow the end-user to edit an OWL instance without having to manipulate the OWL syntax, by simply filling in values in a form. When developing this application, the main requirements were to make it generic — so that it can be easily reused in other applications, and easy to deploy. It also had to be customizable.

A.2 System Architecture

Our goal in developing EdHibou was to build a lightweight knowledge edition tool with (1) a very flexible knowledge model, and (2) highly configurable knowledge acquisition forms.

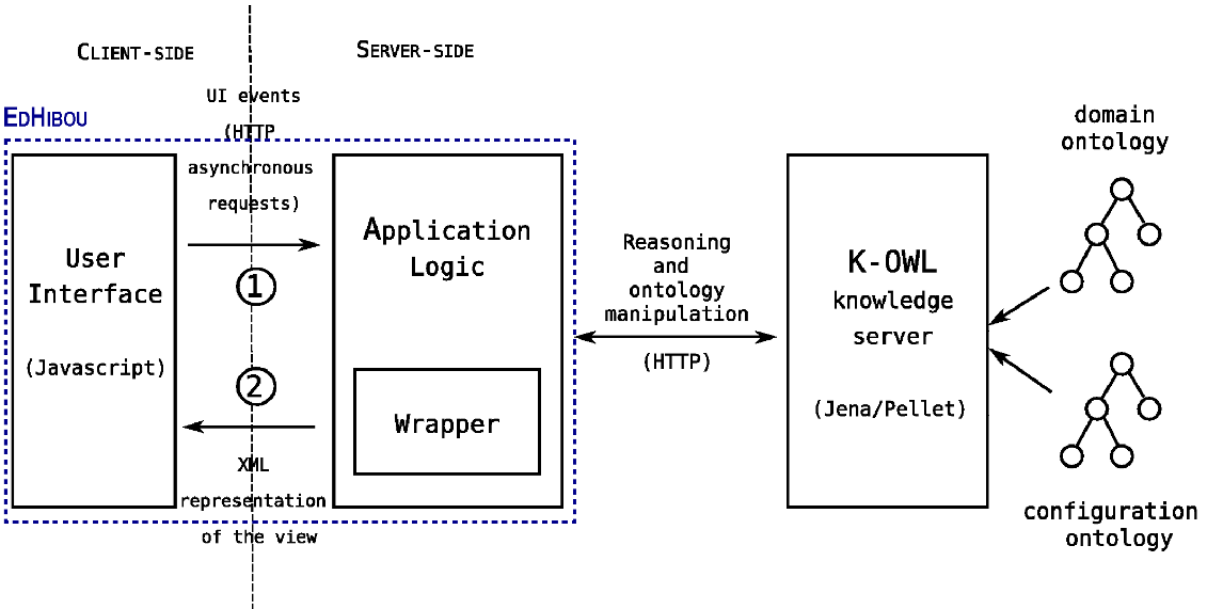


FIG. A.1 – EDHIBOU's software architecture.

Apart from the dynamic user interface update mechanism, anything had to be configurable, including the choice of the components to display and how they are displayed. Requirement (1) has been met by externalizing the knowledge model to a distant knowledge server. The role of this knowledge server is to manage a knowledge base and perform all reasoning tasks over OWL ontologies. Requirement (2) was fulfilled by pushing application configuration into an ontology. The generation of the user interface is then handled by a simple wrapper that takes as input an automatically generated XML representation of the content of an ontology together with a set of graphical component implementations.

EDHIBOU implements a Model-View-Controller architecture pattern (see figure A.2) and was developed using the Google Web Toolkit Java AJAX programming framework. K-OWL, the knowledge server, is a standalone component that plays the role of the model. Though it manages knowledge, and not persistent data, K-OWL has been designed in quite the same spirit as standard database management systems. It stores a set of Java models of OWL ontologies that are created with the Jena Java API coupled to the OWL DL reasoner Pellet. These ontologies are queried upon over HTTP using the SPARQL-DL query language [Sirin et Parsia, 2007]. Though K-OWL could be used remotely, it has been recently integrated to EDHIBOU's application logic for better performances.

A.3 An Ontology-Driven Graphical User Interface Generation

In EDHIBOU, all configuration information, that is information used to render the application knowledge model onto the user interface, is placed in a separate ontology. This ontology contains an exhaustive description of all graphical components used to build the user interface —including their Java implementation classes, as well as some decision knowledge used to determine which graphical component to associate to each property of the domain ontology. The application thus manages in a knowledge base an implementation-independant model of

the user interface, and generating the user interface amounts to wrapping this model onto the application view. Such a wrapper has been implemented. Its role is to produce a suitable XML representation of the graphical components to display and transmit it to the application view for rebuild. Explicitly representing in OWL the configuration knowledge has many advantages. One of these advantages, compared to hard-coding decision procedures in Java classes, is to allow easy customization by a simple ontology extension mechanism. A default ontology $O_{gui-default}$ is provided that contains some default graphical components as well as basic decision knowledge. EdHibou can then be customized for particular domain ontology O_{domain} by replacing this default ontology by a customized ontology $O_{gui-domain}$ that extends $O_{gui-default}$. This new ontology may add new components, specify which components to use for a particular property or even add new decision knowledge to change the behavior of the application. This customization can be done by a knowledge engineer by the means of some ontology editor (or ultimately, by EdHibou, as it is itself a knowledge acquisition tool).

The set of graphical components to display is determined at runtime according to the description of the currently edited individual. EdHibou's default behavior is to select the set of graphical components to be displayed by testing the description of the currently edited instance against the definition domains of the different properties present in the ontology. To determine which components to display, EdHibou keeps only the properties `p` of the ontology for which `a` is an instance of the domain of `p` (according to the reasoner).

A.4 Related Work

A number of systems have been developed with the aim of generating web interfaces on the basis of ontologies and RDF data. These systems generally consider the broad task of creating *semantic portals*, that are websites relying on semantic data. For example, OntoViews [Mäkelä *et al.*, 2004] is a tool to create such a semantic portal, presenting information contained in RDFS ontologies and providing navigation and search mechanisms within these ontologies. Another example is ODESeW [Corcho *et al.*, 2003], which generates complete *knowledge portal* dedicated to the publication and management of information in an organization. Compared to EdHibou, these tools are generally focused on the use of ontologies for the presentation of data in a website. ODESeW also includes a feature that generates forms to create and edit instances as a way to populate the portal. However, this functionality does not make use of the reasoning capability associated with OWL to guide the instance editing process or to infer new information from the elements entered by the user. For this reason, it could not be use as a way to provide advanced features exploiting the knowledge contained in the ontologies, like it is done in the Kasimir project with clinical decision support, thanks to EdHibou.

A.5 Conclusion

EdHibou is a programmatic framework that enables to edit an OWL instance by the means of some user-friendly forms. It implements an ontology-driven graphical user interface generation approach and enables to exploit the standard reasoning on the underlying ontologies to provide intelligent behavior. An application of EdHibou is presented in which it is integrated in a semantic portal as a user interface for a decision support system in oncology. A first demo is currently available online at the URI `http://labotalc.loria.fr/Kasimir`.

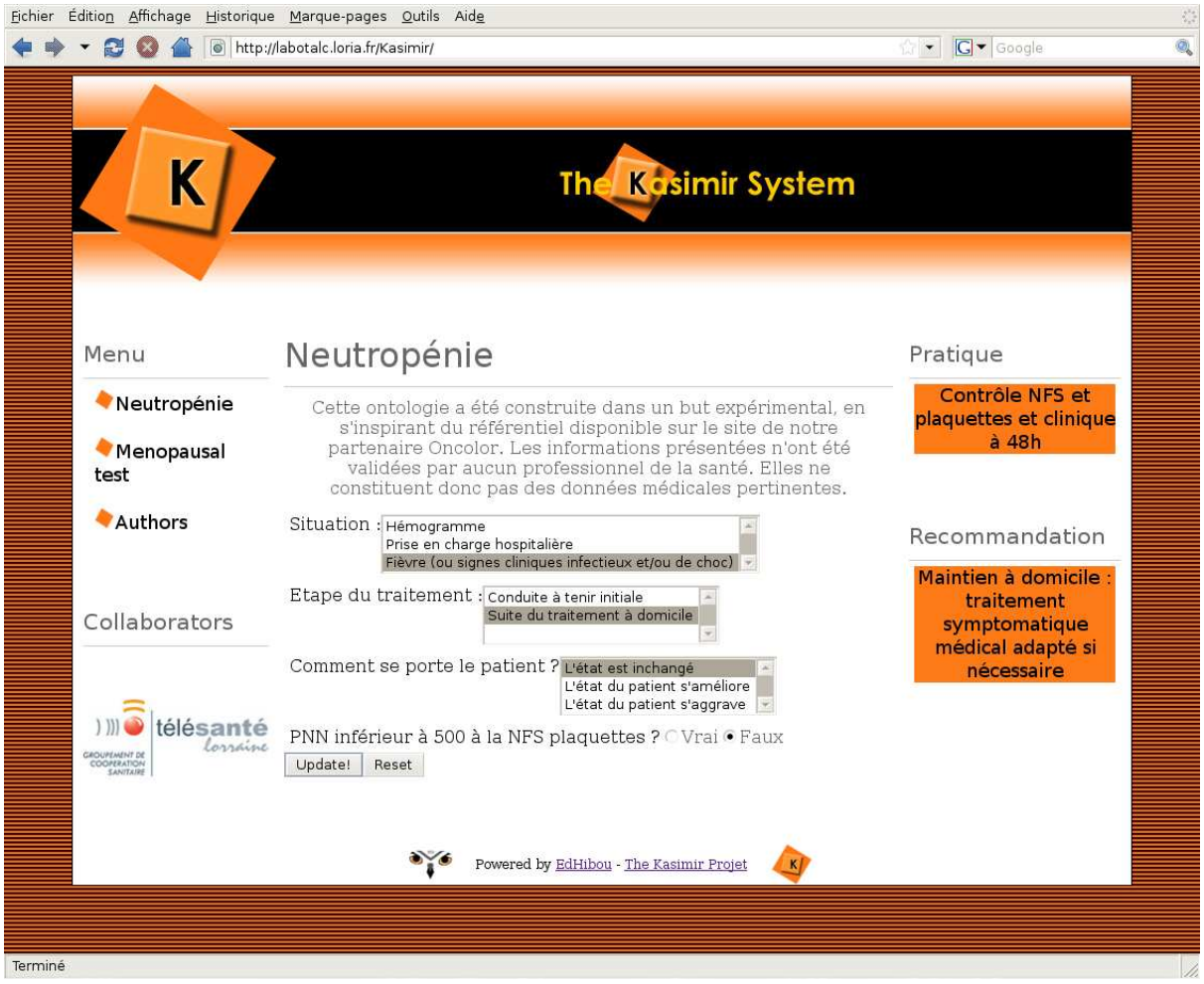


FIG. A.2 – EDHIBOU as a graphical interface for the KASIMIR system. A live demonstration of this software is available at the URI `http://labotalc.loria.fr/Kasimir`.

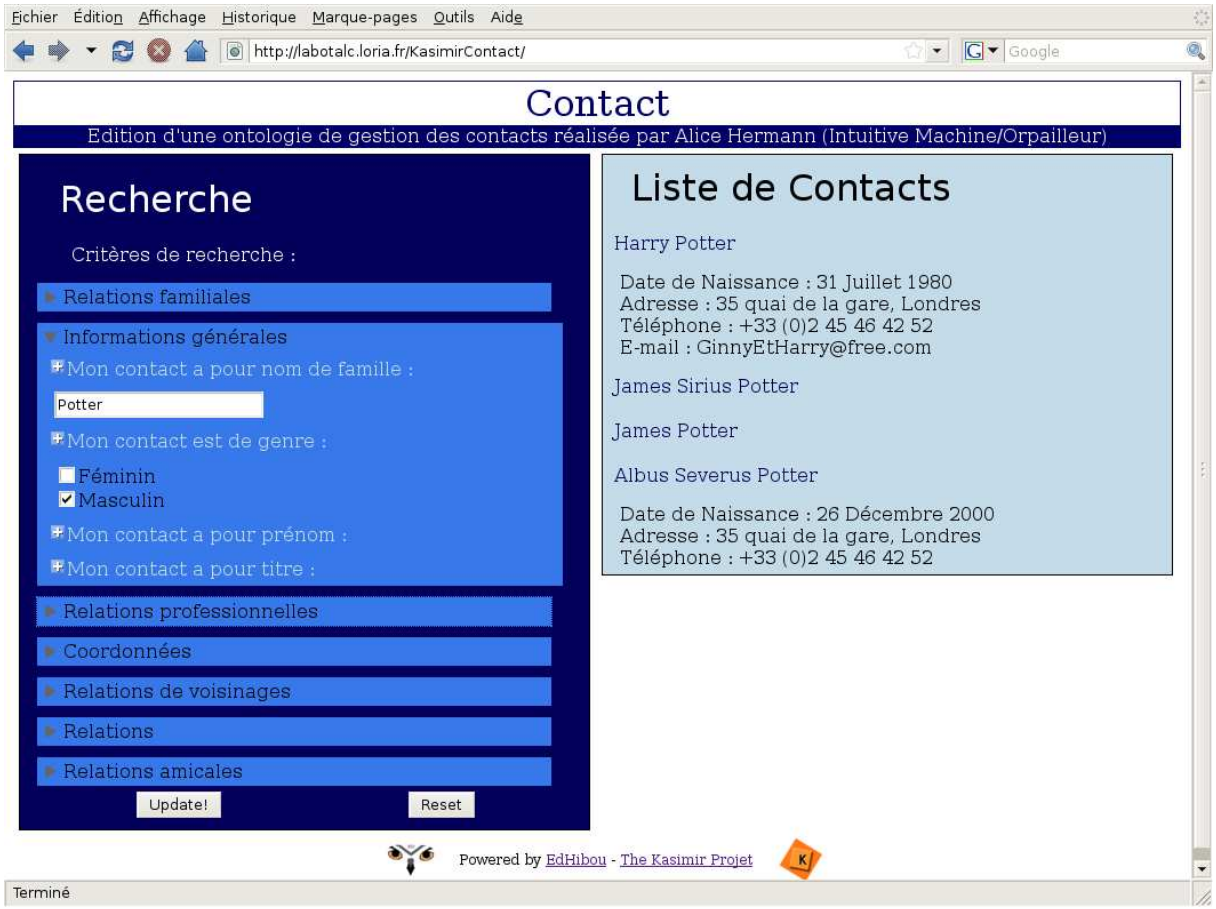


Fig. A.3 – EdHibou as a graphical interface for a contact management system. A live demonstration of this software is available at the URI `http://labotalc.loria.fr/KasimirContact`.

Bibliographie

[Aamodt, 2004] A. Aamodt. Knowledge-Intensive Case-Based Reasoning in CREEK. Dans P. F. et P. A. Gonzàlez-Calero, rédacteur, *Proceedings of the European Conference on Case-Based Reasoning, ECCBR'04*, tome 3155 de *Lecture Notes in Artificial Intelligence*, p. 1–15. Springer, 2004.

[Aha, 1998] D. W. Aha. The omnipresence of case-based reasoning in science and application. *Knowledge-Based Systems*, 11(5-6) :261–273, 1998.

[Aha et Muñoz-Avila, 2001] D. W. Aha et H. Muñoz-Avila. Introduction : Interactive case-based reasoning. *Applied Intelligence*, 14(1) :7–8, 2001.

[Aha *et al.*, 2001] D. W. Aha, L. A. Breslow et T. Maney. Conversational case-based reasoning. *Applied Intelligence*, 14 :9–32, 2001.

[Allen, 1983] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11) :832–843, 1983. ISSN 0001-0782. doi :http://doi.acm.org/10.1145/182.358434.

[Alterman, 1986] R. Alterman. An adaptive planner. Dans *AAAI*, p. 65–69. 1986.

[Baccigalupo et Plaza, 2006] C. Baccigalupo et E. Plaza. Case-based sequential ordering of songs for playlist recommendation. Dans T. Roth-Berghofer, M. H. Göker et H. A. Güvenir, rédacteurs, *Advances in Case-Based Reasoning, 8th European Conference, (ECCBR'06)*, tome 4106 de *Lecture Notes in Computer Science*, p. 286–300. Springer, 2006. ISBN 3-540-36843-4.

[Badra et Lieber, 2007a] F. Badra et J. Lieber. Extraction de connaissances d'adaptation par l'analyse de la base de cas. Dans *Extraction et gestion des connaissances (EGC'2007), Actes des septièmes journées Extraction et Gestion des Connaissances, Namur, Belgique, 23-26 janvier 2007, 2 Volumes*, Revue des Nouvelles Technologies de l'Information, p. 751–760. Cépaduès-Éditions, 2007a.

[Badra et Lieber, 2007b] F. Badra et J. Lieber. Une approche pour représenter les variations entre cas — Vers une application à l'extraction de connaissances d'adaptation. Dans Amélie Cordier, rédacteur, *15ème atelier sur le raisonnement à partir de cas - RàPC-07*, p. 47–56. Grenoble France, 2007b.

[Badra et Lieber, 2008] F. Badra et J. Lieber. Representing Case Variations for Learning General and Specific Adaptation Rules. Dans Amedeo Cesta et Nikos Fakotakis, rédacteurs, *Fourth Starting AI Researcher's Symposium (STAIRS 2008)*, p. 1–11. IOS Press, Patras Grèce, 2008.

[Badra *et al.*, 2008a] F. Badra, R. Bendaoud, R. Bentebibel, P.-A. Champin, J. Cojan, A. Cordier, S. Després, S. Jean-Daubias, J. Lieber, T. Meilender, A. Mille, E. Nauer, A. Napoli et Y. Toussaint. Taaable : Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. Dans Martin Schaaf, rédacteur, *9th European Conference on Case-Based Reasoning - ECCBR 2008*, p. 219–228. Trier Allemagne, 2008a.

[Badra *et al.*, 2008b] F. Badra, M. d'Aquin, J. Lieber et T. Meilender. Edhibou : a customizable interface for decision support in a semantic portal. Dans *Proceedings of the Poster and Demonstration Session at the 7th International Semantic Web Conference (ISWC2008), Karlsruhe, Germany, October 28*. 2008b.

[Badra *et al.*, 2009a] F. Badra, J. Cojan, A. Cordier, J. Lieber, T. Meilender, A. Mille, P. Molli, E. Nauer, A. Napoli, H. Skaf-Molli et Y. Toussaint. Knowledge acquisition and discovery for the textual case-based cooking system WikiTaaable. Dans L. McGinty et D. C. Wilson, rédacteurs, *8th International Conference on Case-Based Reasoning - ICCBR 2009, Workshop Proceedings*. 2009a.

[Badra *et al.*, 2009b] F. Badra, A. Cordier et J. Lieber. Découverte opportuniste de connaissances d'adaptation. Dans B. Fuchs et A. Napoli, rédacteurs, *17ème atelier sur le raisonnement à partir de cas - RàPC-09*, p. 23–34. Paris France, 2009b.

[Badra *et al.*, 2009c] F. Badra, A. Cordier et J. Lieber. Opportunistic adaptation knowledge discovery. Dans L. McGinty et D. C. Wilson, rédacteurs, *Case-Based Reasoning Research and Development / ICCBR 2009*, p. 60–74. 2009c.

[Bain, 1896] W. Bain. *Toward a model of subjective interpretation.* Thèse de doctorat, Université de Yale, département d'informatique, 1896.

[Bergmann et Wilke, 1998] R. Bergmann et W. Wilke. Towards a new formal model of transformational adaptation in case-based reasoning. Dans *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI'98)*, p. 53–57. 1998.

[Bergmann *et al.*, 2005] R. Bergmann, J. Kolodner et E. Plaza. Representation in case-based reasoning. *Knowledge Engineering Review*, 20(3) :209–213, 2005. ISSN 0269-8889. doi : http://dx.doi.org/10.1017/S0269888906000555.

[Bessière *et al.*, 2007] C. Bessière, E. Hebrard, B. Hnich et T. Walsh. The complexity of reasoning with global constraints. *Constraints*, 12(2) :239–259, 2007.

[Bizer *et al.*, 2009] C. Bizer, T. Heath et T. Berners-Lee. Linked data – the story so far. *Special Issue on Linked Data, International Journal on Semantic Web and Information Systems (IJSWIS)*, 2009. À paraître, disponible à l'URL http://tomheath.com/papers/bizer-heath-berners-lee-ijswis-linked-data.pdf.

[Carbonell, 1983a] J. G. Carbonell. Derivational analogy and its role in problem solving. Dans *AAAI*, p. 64–69. 1983a.

[Carbonell, 1983b] J. G. Carbonell. Learning by analogy : Formulating and generalizing plans from past experience. Dans R. S. Michalski, J. G. Carbonell et T. M. Mitchell, rédacteurs, *Machine Learning, An Artificial Intelligence Approach*, p. 137–162. Tioga, Palo Alto, CA, 1983b.

[Chen et Wu, 2003] H. Chen et Z. Wu. On case-based knowledge sharing in semantic web. *Tools with Artificial Intelligence, IEEE International Conference on*, 0 :200, 2003. ISSN 1082-3409. doi :http://doi.ieeecomputersociety.org/10.1109/TAI.2003.1250191.

[Cojan et Lieber, 2008] J. Cojan et J. Lieber. Conservative adaptation in metric spaces. Dans *Proceedings of the 9th European Conference on Case-Based Reasoning (ECCBR'08)*. 2008.

[Corcho *et al.*, 2003] O. Corcho, A. Gómez-Pérez, A. López-Cima, V. López-García et M. Suárez-Figueroa. ODESeW. Automatic Generation of Knowledge Portals for Intranets and Extranets. Dans *Lecture Notes in Computer Science Vol 2870. The Semantic Web - ISWC 2003*, p. 802–817. Springer-Verlag, 2003.

[Cordier, 2008] A. Cordier. *Interactive Knowledge Acquisition in Case Based Reasoning*. Thèse de doctorat, Université Claude Bernard Lyon 1, 2008.

[Cordier *et al.*, 2007] A. Cordier, B. Fuchs, J. Lieber et A. Mille. Failure analysis for domain knowledge acquisition in a knowledge-intensive cbr system. Dans *Case-Based Reasoning Research and Development, Proceedings of the 7th International Conference on Case-Based Reasoning, (ICCBR'07)*, p. 463–477. 2007.

[Cordier *et al.*, 2008] A. Cordier, B. Fuchs, L. L. de Carvalho, J. Lieber et A. Mille. Opportunistic acquisition of adaptation knowledge and cases - the IAKA approach. Dans *Advances in Case-Based Reasoning, 9th European Conference, ECCBR 2008, Trier, Germany, September 1-4, 2008. Proceedings*, p. 150–164. 2008.

[Cordier *et al.*, 2009] A. Cordier, B. Mascret et A. Mille. Extending case-based reasoning with traces. Dans *Grand Challenges for reasoning from experiences, Workshop at IJCAI'09*. 2009.

[Craw *et al.*, 2001] S. Craw, J. Jarmulak et R. Rowe. Learning and applying case-based adaptation knowledge. Dans *Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR'01)*, p. 131–145. Springer-Verlag, London, UK, 2001. ISBN 3-540-42358-3.

[Craw *et al.*, 2006] S. Craw, N. Wiratunga et R. Rowe. Learning adaptation knowledge to improve case-based reasoning. *Artificial Intelligence*, 170(16-17) :1175–1192, 2006.

[d'Aquin, 2005] M. d'Aquin. *Un portail sémantique pour la gestion des connaissances en cancérologie*. Thèse de doctorat, Université Henri Poincaré - Nancy 1, 2005.

[d'Aquin *et al.*, 2004a] M. d'Aquin, S. Brachais, J. Lieber et A. Napoli. Decision Support and Knowledge Management in Oncology using Hierarchical Classification. Dans *Proc. of the Symposium on Computerized Guidelines and Protocols, CGP-2004*, tome 101 de *Studies in Health Technology and Informatics*, p. 16–30. IOS Press, 2004a.

[d'Aquin *et al.*, 2004b] M. d'Aquin, S. Brachais, J. Lieber et A. Napoli. Vers une acquisition automatique de connaissances d'adaptation par examen de la base de cas — une approche fondée sur des techniques d'extraction de connaissances dans des bases de données. Dans *Actes du 12ème Atelier de Raisonnement à Partir de Cas - RàPC'04*, p. 41–52. 2004b.

[d'Aquin *et al.*, 2005] M. d'Aquin, J. Lieber et A. Napoli. Decentralized Case-Based Reasoning for the Semantic Web. Dans *Proc. of International Semantic Web Conference, ISWC 2005*, tome 3729 de *Lecture Notes in Computer Science*, p. 142–155. Springer, 2005.

[d'Aquin *et al.*, 2006a] M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli et L. Szathmary. Adaptation Knowledge Discovery from a Case Base. Dans *Proc. of the European Conference on Artificial Intelligence*, p. 795–796. 2006a.

[d'Aquin *et al.*, 2006b] M. d'Aquin, J. Lieber et A. Napoli. Adaptation knowledge acquisition : A case study for case-based decision support in oncology. *Computational Intelligence*, 22 :161–176(16), 2006b.

[d'Aquin *et al.*, 2007] M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli et L. Szathmary. Case base mining for adaptation knowledge acquisition. Dans *Proceedings of the International Conference on Artificial Intelligence, IJCAI'07*, p. 750–756. 2007.

[de Mántaras et Plaza, 1997] R. L. de Mántaras et E. Plaza. Case-based reasoning : An overview. *AI Communications*, 10(1) :21–29, 1997.

[Delany, 2009] S. J. Delany, rédacteur. *Case-Based Reasoning Research and Development, 8th International Conference on Case-Based Reasoning, ICCBR 2009, Seattle, WA, USA, July 20-23, 2009, Workshop Proceedings*. 2009.

[Díaz-Agudo et González-Calero, 2002] B. Díaz-Agudo et P. A. González-Calero. Cbronto : A task/method ontology for cbr. Dans *Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS Conference), May 14-16, 2002, Pensacola Beach, Florida, USA*, p. 101–105. 2002.

[Domingos, 1996] P. Domingos. Unifying instance-based and rule-based induction. *Machine Learning*, 24(2) :141–168, 1996.

[Fidjeland, 2006] M. K. Fidjeland. *Distributed Knowledge in Case-Based Reasoning - Knowledge Sharing and Reuse within the Semantic Web*. Thèse de maître, Norwegian University of Science and Technology, Department of Computer and Information Science, 2006.

[Field et Lohr, 1990] M. J. Field et K. M. Lohr, rédacteurs. *Clinical practice guidelines : directions for a new program*. Washington, DC, USA : National Academy Press, 1990.

[Fuchs *et al.*, 2000] B. Fuchs, J. Lieber, A. Mille et A. Napoli. An algorithm for adaptation in case-based reasoning. Dans *ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, August 20-25, 2000*, p. 45–49. 2000.

[Fuchs *et al.*, 2006] B. Fuchs, J. Lieber, A. Mille et A. Napoli. A general strategy for adaptation in case-based reasoning. Rapport technique RR-LIRIS-2006-016, LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon, 2006.

[Hammond, 1986] K. J. Hammond. CHEF : A model of case-based planning. Dans *AAAI*, p. 267–271. 1986.

[Hammond, 1989] K. J. Hammond. *Case-Based Planning : Viewing Planning as a Memory Task*. Academic Press, San Diego, 1989.

[Hanney, 1997] K. Hanney. *Learning Adaptation Rules from Cases*. Thèse de maître, Trinity College, Dublin, 1997.

[Hanney et Keane, 1996] K. Hanney et M. T. Keane. Learning Adaptation Rules from Cases. Dans I. Smith et B. Falting, rédacteurs, *Proceedings of the 3rd European Workshop on Case-Based Reasoning (EWCBR'96)*, tome 1168 de *LNAI*. Springer, 1996.

[Hanney *et al.*, 1995] K. Hanney, M. T. Keane, B. Smyth et P. Cunningham. Systems, tasks and adaptation knowledge : Revealing some revealing dependencies. Dans *Case-Based Reasoning Research and Development, First International Conference, (ICCBR'95)*, p. 461–470. 1995.

[Hastings *et al.*, 2002] J. D. Hastings, K. Branting et J. A. Lockwood. Carma : A case-based rangeland management adviser. *AI Magazine*, 23(2) :49–62, 2002.

[Hinrichs, 1992] T. Hinrichs. *Problem Solving in Open Worlds : A Case Study in Design*. Lawrence Erlbaum, Hillsdale, NJ, 1992.

[Ihle *et al.*, 2009] N. Ihle, A. Hanft et K.-D. Althoff. Extraction of adaptation knowledge from internet communities. Dans S. J. Delany, rédacteur, *Case-Based Reasoning Research and Development, 8th International Conference on Case-Based Reasoning, ICCBR 2009, Seattle, WA, USA, July 20-23, 2009, Workshop Proceedings*, p. 35–44. 2009.

[Jarmulak *et al.*, 2001] J. Jarmulak, S. Craw et R. Rowe. Using Case-Base Data to Learn Adaptation Knowledge for Design. Dans *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'01)*, p. 1011–1020. 2001.

[Kinley, 2001] A. Kinley. *Learning to improve case adaptation*. Thèse de doctorat, Computer Science Department - Indiana University, 2001.

[Kolodner, 1993] J. Kolodner. *Case-Based Reasoning*. Morgan Kaufmann Publishers, Inc, 1993.

[Koton, 1988] P. Koton. Reasoning about evidence in causal explanations. Dans *AAAI*, p. 256–263. 1988.

[Lallich *et al.*, 2006] S. Lallich, O. Teytaud et E. Prudhomme. *Association rules interestingness : measure and validation*. Quality Measures in Data Mining. Springer, Heidelberg, Germany, 2006.

[Lamontagne et Lapalme, 2002] L. Lamontagne et G. Lapalme. Raisonnement à base de cas textuels - état de l'art et perspectives. *Revue d'Intelligence Artificielle*, 16 :339–366, 2002.

[Leake, 1996a] D. B. Leake, rédacteur. *Case-Based Reasoning : Experiences, Lessons, and Future Directions*, chapitre 9 : Design à la Déjà Vu. The MIT Press, Cambridge, Massachusetts, 1996a.

[Leake, 1996b] D. B. Leake, rédacteur. *Case-Based Reasoning : Experiences, Lessons, and Future Directions*, chapitre 11 : Learning to Improve Case Adaptation by Introspective Reasoning and CBR. The MIT Press, Cambridge, Massachusetts, 1996b.

[Leake et Powell, 2007] D. B. Leake et J. H. Powell. Mining large-scale knowledge sources for case adaptation knowledge. Dans *Case-Based Reasoning Research and Development, 7th International Conference on Case-Based Reasoning, (ICCBR'07)*, p. 209–223. 2007.

[Leake et Powell, 2008] D. B. Leake et J. H. Powell. Knowledge planning and learned personalization for web-based case adaptation. Dans *Advances in Case-Based Reasoning, 9th European Conference, ECCBR 2008, Trier, Germany, September 1-4, 2008. Proceedings*, p. 284–298. 2008.

[Leake *et al.*, 1996] D. B. Leake, A. Kinley et D. Wilson. Acquiring Case Adaptation Knowledge : A Hybrid Approach. Dans *AAAI/IAAI*, tome 1, p. 684–689. 1996.

[Leake *et al.*, 1997] D. B. Leake, A. Kinley et D. C. Wilson. A Case Study of Case-Based CBR. Dans *Proc. of the Second International Conference on Case-Based Reasoning Research and Development, ICCBR 97*, p. 371–382. Springer, 1997.

[Lieber, 1999] J. Lieber. Reformulations and Adaptation Decomposition. Dans A. M. e. A. N. J. Lieber, E. Melis, rédacteur, *Formalisation of Adaptation in Case-Based Reasoning*. Third International Conference on Case-Based Reasoning Workshop, ICCBR-99 Workshop number 3, S. Schmitt et I. Vollrath (volume editor), LSA, University of Kaiserslautern, 1999.

[Lieber, 2002a] J. Lieber. Recopier c'est déjà adapter : six types d'adaptation par copie. Dans *Actes du* 10^{me} *atelier de raisonnement à partir de cas*. 2002a.

[Lieber, 2002b] J. Lieber. Strong, fuzzy and smooth hierarchical classification for case-based problem solving. Dans *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, (ECAI'02)*, p. 81–85. 2002b.

[Lieber, 2007] J. Lieber. Application of the revision theory to adaptation in case-based reasoning : The conservative adaptation. Dans R. Weber et M. M. Richter, rédacteurs, *ICCBR*, tome 4626 de *Lecture Notes in Computer Science*, p. 239–253. Springer, 2007. ISBN 978-3-540-74138-1.

[Lieber et Napoli, 1996] J. Lieber et A. Napoli. Using classification in case-based planning. Dans *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96)*, p. 132–136. John Wiley & Sons, Ltd, 1996.

[Lieber et Napoli, 1998] J. Lieber et A. Napoli. Correct and Complete Retrieval for Case-Based Problem-Solving. Dans H. Prade, rédacteur, *Proceedings of the European Conference on Artificial Intelligence (ECAI'98)*, p. 68–72. John Wiley & Sons Ltd, Chichester, 1998.

[Lieber *et al.*, 2001] J. Lieber, P. Bey, F. Boisson, B. Bresson, P. Falzon, A. Lesur, A. Napoli, M. Rios et C. Sauvagnac. Acquisition et modélisation de connaissances d'adaptation, une étude pour le traitement du cancer du sein. Dans Jean Charlet, rédacteur, *journées ingénierie des connaissances - IC'2001*, p. 409–426. Presses Universitaires de Grenoble, Grenoble, France, 2001. Colloque avec actes et comité de lecture. nationale.

[Lieber *et al.*, 2002] J. Lieber, M. d'Aquin, P. Bey, B. Bresson, O. Croissant, P. Falzon, A. Lesur, J. Lévêque, V. Mollo, A. Napoli, M. Rios et C. Sauvagnac. The Kasimir Project : Knowledge Management in Cancerology. Dans *Proceedings of the 4th International Workshop on Enterprise Networking and Computing in Health Care Industry, HealthCom 2002*. 2002.

[Lieber *et al.*, 2003] J. Lieber, M. d'Aquin, P. Bey, A. Napoli, M. Rios et C. Sauvagnac. Acquisition of adaptation knowledge for breast cancer treatment decision support. Dans *AIME*, p. 304–313. 2003.

[Lieber *et al.*, 2004] J. Lieber, M. d'Aquin, S. Brachais et A. Napoli. Une étude comparative de quelques travaux sur l'acquisition de connaissances d'adaptation en raisonnement à partir de cas. Dans *Actes du* 12^{me} *atelier de raisonnement à partir de cas (RàPC'04)*. 2004.

[Lieber *et al.*, 2008] J. Lieber, M. d'Aquin, F. Badra et A. Napoli. Modeling adaptation of breast cancer treatment decision protocols in the kasimir project. *Applied Intelligence*, 28(3) :261–274, 2008.

[Mäkelä *et al.*, 2004] E. Mäkelä, E. Hyvönen, S. Saarela et K. Viljanen. OntoViews - A Tool for Creating Semantic Web Portals. Dans S. A. McIlraith, D. Plexousakis et F. van Harmelen, rédacteurs, *International Semantic Web Conference, ISWC 2004*, tome 3298 de *Lecture Notes in Computer Science*, p. 797–811. Springer, 2004.

[McSherry, 1998] D. McSherry. An adaptation heuristic for case-based estimation. Dans *EWCBR '98 : Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning*, p. 184–195. Springer-Verlag, London, UK, 1998. ISBN 3-540-64990-5.

[McSherry, 1999] D. McSherry. Demand-driven discovery of adaptation knowledge. Dans *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages*, p. 222–227. 1999.

[McSherry, 2006] D. McSherry. Completeness criteria for retrieval in recommender systems. Dans *Advances in Case-Based Reasoning, 8th European Conference (ECCBR'06)*, p. 9–29. 2006.

[Melis *et al.*, 1998] E. Melis, J. Lieber et A. Napoli. Reformulation in case-based reasoning. Dans B. S. et P. Cunningham, rédacteur, *Proc. of the Fourth European Workshop on Case-Based Reasoning, (EWCBR'98)*, Lecture Notes in Artificial Intelligence 1488, p. 172–183. Springer, 1998.

[Michalski, 1983] R. S. Michalski. A theory and methodology of inductive learning. *Artificial Intelligence*, 20(2) :111–161, 1983.

[Minsky, 1965] M. Minsky. Matter, mind and models. Rapport technique AIM-77, MIT Artificial Intelligence Laboratory, 1965.

[Mitchell, 1982] T. M. Mitchell. Generalization as search. *Artificial Intelligence*, 18(2) :203–226, 1982.

[Mollo, 2004] V. Mollo. *Usage des ressources, adaptation des savoirs et gestion de l'autonomie dans la décision thérapeutique*. Thèse d'université en ergonomie, Conservatoire national des arts et métiers, 2004.

[Munteanu, 1992] P. Munteanu. *Extraction de Connaissances dans les Bases de Données Parole : Apport de l'Apprentissage Symbolique*. Thèse de doctorat, Institut de la Communication Parlée de Grenoble, 1992.

[Pennerath *et al.*, 2008] F. Pennerath, G. Polaillon et A. Napoli. A method for classifying vertices of labeled graphs applied to knowledge discovery from molecules. Dans *ECAI 2008 - 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25, 2008, Proceedings*, p. 147–151. 2008.

[Perrier-Robert, 2004] A. Perrier-Robert. *Confiture—l'écume des pots*. Editions S.A.E.P. 68040 Ingersheim - Colmar, 2004.

[Purvis et Pu, 1998] L. Purvis et P. Pu. Composer : A case-based reasoning system for engineering design. *Robotica*, 16(3) :285–295, 1998. ISSN 0263-5747. doi :http://dx.doi.org/10.1017/S0263574798000368.

[Quinlan, 1993] J. R. Quinlan. *C4.5 : Programs for Machine Learning*. Morgan Kaufmann, 1993. ISBN 1-55860-238-0.

[Redmond, 1990] M. Redmond. Distributed cases for case-based reasoning : Facilitating use of multiple cases. Dans *AAAI*, p. 304–309. 1990.

[Richter, 1998] M. Richter. Introduction. Dans M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard et S. Wess, rédacteurs, *CBR Technology : From Foundations to Applications*, chapitre 1, p. 1–15. Springer, Berlin, 1998.

[Riesbeck et Schank, 1989] C. K. Riesbeck et R. C. Schank. *Inside Case-Based Reasoning*. Lawrence Erbaum Associates, Inc., Hillsdale, New Jersey, 1989.

[Sauvagnac, 2000] C. Sauvagnac. *La construction de connaissances par l'utilisation et la conception de procédures. Contribution au cadre théorique des activités métafonctionnelles*. Thèse d'université en ergonomie, Conservatoire national des arts et métiers, 2000.

[Schaaf, 2008] M. Schaaf, rédacteur. *ECCBR 2008, The 9th European Conference on Case-Based Reasoning, Trier, Germany, September 1-4, 2008, Workshop Proceedings*. 2008.

[Schank et Slade, 1991] R. C. Schank et S. Slade. The future of artificial intelligence : learning from experience. *Applied Artificial Intelligence*, 5(1) :97–107, 1991.

[Schmidt et Vorobieva, 2005] R. Schmidt et O. Vorobieva. Adaptation and medical case-based reasoning focusing on endocrine therapy support. Dans *Artificial Intelligence in Medicine, 10th Conference on Artificial Intelligence in Medicine (AIME'05)*, p. 300–309. 2005.

[Schmidt *et al.*, 2001] R. Schmidt, D. Steffen et L. Gierl. Evaluation of a case-based antibiotics therapy adviser. Dans *Artificial Intelligence Medicine, 8th Conference on AI in Medicine in Europe, (AIME'01)*, p. 462–466. 2001.

[Shiu *et al.*, 2001] S. C. K. Shiu, D. S. Yeung, C. H. Sun et X. Z. Wang. Transferring case knowledge to adaptation knowledge : An approach for case-base maintenance. *Computational Intelligence*, 17(2) :295–314, 2001.

[Sirin et Parsia, 2007] E. Sirin et B. Parsia. Sparql-dl : Sparql query for owl-dl. Dans *OWL : Experiences and Directions Third International Workshop (OWLED 2007)*. 2007.

[Smith *et al.*, 1995] I. F. C. Smith, C. Lottaz et B. Faltings. Spatial composition using cases : Idiom. Dans *Case-Based Reasoning Research and Development, First International Conference, ICCBR-95, Sesimbra, Portugal, October 23-26, 1995, Proceedings*, p. 88–97. 1995.

[Smyth, 1996] B. Smyth. *Case-Based Design*. Thèse de doctorat, Computer Science Department, Trinity College, University of Dublin, 1996.

[Smyth et Champin, 2009] B. Smyth et P.-A. Champin. The Experience Web : A Case-Based Reasoning Perspective. Dans *Grand Challenges for reasoning from experiences, Workshop at IJCAI'09*. 2009.

[Smyth et Cunningham, 1992] B. Smyth et P. Cunningham. Déjà vu : A hierarchical case-based reasoning system for software design. Dans *Proceedings of the 10th European Conference on Artificial Intelligence, (ECAI'92)*, p. 587–589. 1992.

[Smyth et Keane, 1993] B. Smyth et M. T. Keane. Retrieving adaptable cases : The role of adaptation knowledge in case retrieval. Dans S. Wess, K.-D. Althoff et M. M. Richter, rédacteurs, *EWCBR*, tome 837 de *Lecture Notes in Computer Science*, p. 209–220. Springer, 1993. ISBN 3-540-58330-0.

[Smyth *et al.*, 2001] B. Smyth, M. T. Keane et P. Cunningham. Hierarchical case-based reasoning integrating case-based and decompositional problem-solving techniques for plant-control software design. *IEEE Transactions on Knowledge and Data Engineering*, 13(5) :793–812, 2001.

[Stahl et Bergmann, 2000] A. Stahl et R. Bergmann. Applying recursive cbr for the custumization of structured products in an electronic shop. Dans *Advances in Case-Based Reasoning, 5th European Workshop, EWCBR 2000, Trento, Italy, September 6-9, 2000, Proceedings*, p. 297–308. 2000.

[Szathmary et Napoli, 2005] L. Szathmary et A. Napoli. CORON : A Framework for Levelwise Itemset Mining Algorithms. Dans *Proc. of the Third International Conference on Formal Concept Analysis, ICFCA '05*, p. 110–113. 2005.

[Tixier *et al.*, 2008] M. Tixier, F. Badra et J. Lieber. Familles génératrices de règles d'adaptation pour assister leur acquisition semi-automatique. Dans *Actes d'IC 2008 : Ingénierie des connaissances 2008 (Proceedings of the 19th French Knowledge Engineering Conference), Nancy, France, June 18-20, 2008*, p. 225–236. 2008.

[Vaillant *et al.*, 2005] B. Vaillant, P. Meyer, E. Prudhomme, S. Lallich, P. Lenca et S. Bigaret. Mesurer l'intérêt des règles d'association. Dans *Atelier Qualité des Données et des Connaissances (DQK 05), EGC 05, Paris*, p. 69–78. 2005.

[Wilke *et al.*, 1997] W. Wilke, I. Vollrath, K.-D. Althoff et R. Bergmann. A framework for learning adaptation knowedge based on knowledge light approaches. Dans *Proceedings of the Fifth German Workshop on Case-Based Reasoning*, p. 235–242. 1997.

[Wiratunga *et al.*, 2002] N. Wiratunga, S. Craw et R. Rowe. Learning to adapt for case-based design. Dans S. C. et Alun D. Preece, rédacteur, *Advances in Case-Based Reasoning, 6th European Conference, ECCBR 2002 Aberdeen, Scotland, UK, September 4-7, 2002, Proceedings*, tome 2416 de *Lecture Notes in Computer Science*, p. 421–435. Springer, 2002. ISBN 3-540-44109-3.

[Zaki et Hsiao, 2002] M. J. Zaki et C.-J. Hsiao. CHARM : An efficient algorithm for closed itemset mining. Dans R. L. Grossman, J. Han, V. Kumar, H. Mannila et R. Motwani, rédacteurs, *Proceedings of the Second SIAM International Conference on Data Mining, Arlington, VA, USA, April 11-13, 2002*. SIAM, 2002. ISBN 0-89871-517-2.

Résumé

Cette thèse se situe à l'intersection de trois domaines de recherche : le raisonnement à partir de cas, l'extraction de connaissances et la représentation des connaissances. Raisonner à partir de cas consiste à résoudre un nouveau problème en utilisant un ensemble de problèmes déjà résolus, appelés cas. Dans cette thèse, un langage de représentation des variations entre cas est introduit. Nous montrons comment ce langage peut être utilisé pour représenter les connaissances d'adaptation et pour modéliser la phase d'adaptation en raisonnement à partir de cas. Ce langage est ensuite appliqué à la tâche d'apprentissage de connaissances d'adaptation. Un processus d'extraction de connaissances, appelé CabamakA, est mis au point. Ce processus permet d'apprendre des connaissances d'adaptation par généralisation à partir d'une représentation des variations entre cas. Une discussion est ensuite menée sur les conditions d'opérationnalisation de CabamakA au sein d'un processus d'acquisition de connaissances. L'étude aboutit à la proposition d'un nouveau type d'approche pour l'acquisition de connaissances d'adaptation dans lequel le processus d'extraction de connaissances est déclenché de manière opportuniste au cours d'une session particulière de résolution de problèmes. Les différents concepts introduits dans la thèse sont illustrés dans le domaine culinaire à travers leur application au système de raisonnement à partir de cas Taaable, qui constitue le contexte applicatif de l'étude.

Mots-clés: Raisonnement à partir de cas, acquisition de connaissances d'adaptation, extraction de connaissances.

Abstract

This thesis presents some contributions in three research domains : case-based reasoning, knowledge discovery and knowledge representation. Case-based reasoning consists in solving new problems by reusing a set of previous problem-solving experiences, called cases. In this thesis, a language is introduced to represent variations between cases. We first show how this language can be used to represent adaptation knowledge and to model the adaptation phase in case-based reasoning. This language is then applied to the task of adaptation knowledge learning. A knowledge discovery process, called CabamakA, is proposed, that learns adaptation knowledge by generalization from a representation of variations between cases. A discussion follows on how to make this knowledge discovery process operational in a knowledge acquisition process. The discussion leads to the proposition of a new approach for adaptation knowledge acquisition, in which the knowledge discovery process is triggered in an opportunistic manner at problem-solving time. The concepts introduced in the thesis are illustrated in the cooking domain through their application in the case-based reasoning system Taaable, that constitutes the application domain of the study.

Keywords: Case-Based Reasoning, Adaptation Knowledge Acquisition, Knowledge Discovery, Knowledge Representation.