Un algorithme pour la phase d'adaptation du raisonnement à partir de cas

Béatrice Fuchs
LISI,
IAE - Université Lyon 3,
6, cours Albert Thomas,
69 008 Lyon
fuchs@univ-lyon3.fr

Jean Lieber
LORIA, BP 239,
54 506 Vandœuvre-lès-Nancy,
lieber@loria.fr

Alain Mille
Université C. Bernard, Lyon 1,
LISI, bâtiment 710,
69 622 Villeurbanne,
amille@bat710.univ-lyon1.fr

Amedeo Napoli
LORIA, BP 239,
54 506 Vandœuvre-lès-Nancy,
napoli@loria.fr

Résumé *Cette étude est une contribution à la conception d'un algorithme général d'adaptation en raisonnement à partir de cas (*RÀPC*). Le processus d'adaptation constitue une phase importante et complexe du* RÀPC *qui est la plupart du temps conçue pour une application spécifique. Cet article présente un algorithme d'adaptation indépendant du domaine d'application. Les cas sont décrits par un ensemble de descripteurs dont les valeurs sont des nombres ou des contraintes locales sous la forme d'intervalles numériques. L'algorithme détermine, pour chacun de ces descripteurs un intervalle de valeurs admissibles en combinant une solution source, un appariement exprimé sous la forme d'intervalles de variations et des dépendances entre le problème source et sa solution. Cette approche générique de l'adaptation est opérationnelle et introduit des opérateurs d'adaptation généraux et indépendants du domaine d'application.*

1 Introduction

Le processus de raisonnement à partir de cas (RÀPC) s'appuie sur trois opérations principales : la remémoration, l'adaptation et la mémorisation [10, 12]. L'adaptation est au cœur du processus de RÀPC et y joue un rôle central. Elle est considérée habituellement comme une opération dépendant du domaine d'application, complexe et difficile à appréhender. Dans cet article, nous proposons une vision générique et opérationnelle de l'adaptation qui est conçue pour être (adaptée et) réutilisée dans le contexte de « problèmes numériques », c'est-à-dire de problèmes dont les caractéristiques peuvent être décrites par des attributs dont les valeurs sont partiellement ordonnées. Dans la suite, nous étudions :

(1) Une stratégie de l'adaptation fondée, d'une part, sur un appariement entre le cas

antérieur et le nouveau cas et, d'autre part, sur les dépendances entre descripteurs du cas antérieur ;

(2) Un algorithme général pour adapter un cas antérieur en un nouveau cas s'appuyant sur une détermination d'intervalles de variations pour les attributs du nouveau cas.

Nous illustrons cette vision de l'adaptation par un exemple significatif. Cette approche contribue à la conception d'approches de l'adaptation indépendantes du domaine d'application. En ce sens, l'adaptation, qui est généralement considérée comme une tâche cruciale et complexe pour le processus de RÀPC, peut être plus facilement prise en compte et implantée dans un système de RÀPC spécifique, en utilisant notre algorithme générique comme un guide d'implantation. Cette étude est principalement utile pour la conception d'un système de RÀPC travaillant sur des « problèmes numériques » et, plus généralement, pour une compréhension plus précise du mécanisme d'adaptation.

L'article est organisé comme suit. Dans un premier temps, la description de deux travaux antérieurs sur l'adaptation nous permet d'introduire les hypothèses et les principes généraux de la présente approche. Puis, nous proposons et détaillons un algorithme indépendant du domaine d'application pour l'adaptation d'un cas. Ce paragraphe est poursuivi par une discussion sur les qualités de notre approche et ses limitations, ainsi que par des perspectives de recherches.

Cet article est une version traduite, revue et complétée de [6].

2 Deux modèles généraux de l'adaptation

Le RÀPC utilise des cas résolus, appelés *cas sources*, mémorisés dans une base de cas, afin de résoudre un nouveau problème, appelé le *problème cible* et dénoté par $\mathtt{cible}$. Un cas décrit un problème $\mathtt{pb}$ et sa solution $\mathtt{Sol}(\mathtt{pb})$, et est dénoté par le couple $(\mathtt{pb}, \mathtt{Sol}(\mathtt{pb}))$. Un cas de la base de cas est dénoté par le couple $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ et $\mathtt{srce}$ est appelé le problème source. Les principales étapes du cycle de RÀPC sont : la *remémoration*, qui consiste à choisir dans la base de cas un problème $\mathtt{srce}$ similaire à $\mathtt{cible}$, l'*adaptation*, qui consiste à réutiliser la solution $\mathtt{Sol}(\mathtt{srce})$ de $\mathtt{srce}$ afin de construire $\mathtt{Sol}(\mathtt{cible})$ et ainsi de résoudre le problème $\mathtt{cible}$, et la *mémorisation*, qui consiste à mémoriser le cas nouvellement résolu $(\mathtt{cible}, \mathtt{Sol}(\mathtt{cible}))$ dans la base de cas.

Lors de travaux précédents, nous avons proposé deux modèles généraux du processus d'adaptation sur lesquels s'appuie notre algorithme. Un premier modèle général, décrit dans [5], propose de connecter les étapes de remémoration et d'adaptation de façon unifiée, comme l'illustre la figure 1.

Adapter un cas consiste à construire une liste de descripteurs solution conduisant à la satisfaction d'un but, un but étant un des objectifs à atteindre afin de construire une solution complète. La remémoration débute par l'élaboration d'un index $\mathtt{idx}(\mathtt{cible})$ de $\mathtt{cible}$ (étape (a)) qui consiste à sélectionner les descripteurs pertinents de cible. À partir de cet index $\mathtt{idx}(\mathtt{cible})$, l'index $\mathtt{idx}(\mathtt{srce})$ d'un cas source $\mathtt{srce}$ est retrouvé (étape (b)), ce qui permet d'obtenir $\mathtt{srce}$ (étape (c)). L'étape (d) correspond au passage

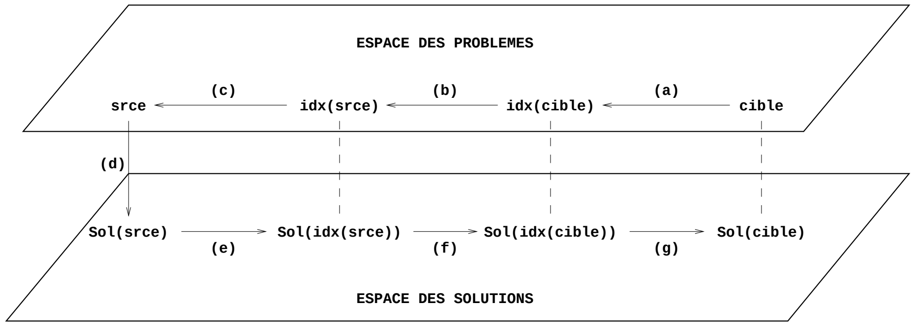


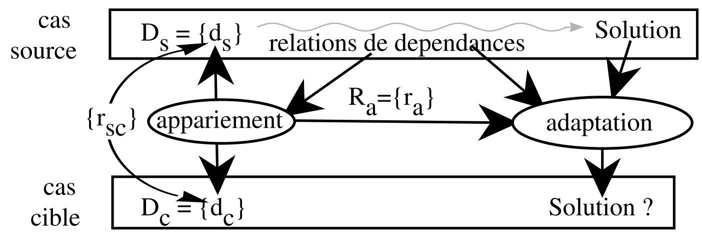
FIG. 1 – Remémoration dans l'espace des problèmes et adaptation dans l'espace des solutions.

de la remémoration, qui opère dans l'espace des problèmes, à l'adaptation, qui opère dans l'espace des solutions. Le processus d'adaptation commence par la généralisation de $\mathtt{Sol}(\mathtt{srce})$ en $\mathtt{Sol}(\mathtt{idx}(\mathtt{srce}))$ (étape (e)), en accord avec l'étape (c), c'est-à-dire en prenant en compte les relations établies entre $\mathtt{srce}$ et $\mathtt{idx}(\mathtt{srce})$. Puis, lors de l'étape (f), $\mathtt{Sol}(\mathtt{idx}(\mathtt{srce}))$ est transformé en $\mathtt{Sol}(\mathtt{idx}(\mathtt{cible}))$ par un processus de généralisation/spécialisation. Finalement, $\mathtt{Sol}(\mathtt{idx}(\mathtt{cible}))$ est spécialisé en $\mathtt{Sol}(\mathtt{cible})$ (étape (g) qui correspond à l'opération inverse de l'étape (a) répercutée sur la solution). Les lignes horizontales (a), (b) et (c) correspondent à un *chemin de similarité* entre problèmes dans l'espace des problèmes : `srce` et `cible` sont liés par un chemin de similarité composé d'une séquence de relations entre problèmes. Les lignes horizontales (e), (f) et (g) correspondent à l'application d'une séquence d'opérateurs d'adaptation dans l'espace des solutions. Les lignes verticales en pointillés représentent des relations de dépendances entre problèmes et solutions ; elles expriment le fait que les descripteurs d'un problème jouent un rôle dans la détermination de sa solution.

Un deuxième modèle général [7] a pour objectif de décrire le processus d'adaptation au niveau des connaissances. Le processus d'appariement est le point de départ du processus d'adaptation car il met en évidence des relations entre les problèmes source et cible. Ces relations réifient les similarités et dissimilarités entre les descripteurs source et cible qui vont guider le processus d'adaptation. Un appariement R_a est un ensemble de triplets $r_a = (d_s, d_c, r_{sc})$ où d_s est un descripteur du problème source, d_c est un descripteur du problème cible, et r_{sc} est une *explication* reliant d_s et d_c. Une telle explication est constituée d'une séquence de relations du graphe des connaissances exprimant l'identité, ou le cas échéant une similarité de rôle des descripteurs dans le contexte du cas.

Le processus d'adaptation modifie l'ensemble des descripteurs de la solution étant donné un appariement R_a et un ensemble de dépendances entre un problème source et sa solution.

Le schéma suivant résume ce modèle :



L'algorithme d'adaptation général que nous proposons décrit la transformation de $\texttt{Sol}(\texttt{srce})$ en $\texttt{Sol}(\texttt{cible})$ et correspond à l'étape d'adaptation (f) du premier modèle. Il s'appuie sur les relations de dépendances mises en évidence dans les deux modèles : une dépendance entre un problème et sa solution exprime l'influence de la variation d'un descripteur de problème sur les descripteurs de solution. La spécification de l'algorithme d'adaptation est inspirée du second modèle : il prend en entrée un appariement entre un problème source et un problème cible qui met en évidence les objectifs de l'adaptation, et un ensemble de dépendances exprimant quels descripteurs de la solution sont à modifier en conséquence.

L'algorithme est illustré par un exemple présenté au paragraphe suivant.

3 Hypothèses et principes de l'approche

Le processus d'adaptation peut être spécifié ainsi : étant donné un cas source $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$, un problème cible $\texttt{cible}$, un appariement entre $\texttt{srce}$ et $\texttt{cible}$, et un ensemble de dépendances entre $\texttt{srce}$ et $\texttt{Sol}(\texttt{srce})$, quelle est la solution $\texttt{Sol}(\texttt{cible})$ de $\texttt{cible}$?

Notre proposition se restreint à la catégorie particulière des « problèmes numériques », pour lesquels les valeurs des attributs des problèmes sont partiellement ordonnés. C'est pour cela que nous utilisons des nombres pour l'illustrer. Cependant, les considérations suivantes peuvent être appliquées pour n'importe quel type de problème dont les attributs ont des valeurs symboliques qui peuvent être partiellement ordonnées : une extension linéaire (c'est-à-dire, un ordre total) peut toujours être construite sur un ordre partiel [4].

L'exemple du paragraphe suivant s'appuie sur les problèmes de configurations de PC et s'inspire de [2]. L'utilisateur spécifie des besoins tels que les jeux (J), la musique (Mu), la programmation (Pgm), etc., avec des intervalles de valeurs numériques $[min, max]$ exprimant l'importance des besoins et tels que $0 \leqslant min \leqslant max \leqslant 10$. Une solution d'un problème de configuration de PC est un ensemble de composants satisfaisant les besoins de l'utilisateur : carte mère (CM), écran (Écr), etc. Par exemple :

$$\text{Mu} \in [2, 2] \text{ signifie que Mu} = 2.$$
$$\text{Pgm} \in [0, 10] \text{ signifie que Pgm est non contraint.}$$

Un problème source $\texttt{srce}$ est composé d'un ensemble de descripteurs contenant chacun une contrainte sur un attribut. Pour chaque attribut, il y a une seule contrainte à laquelle est attachée un intervalle de valeurs licites. Un descripteur de $\mathtt{srce}$ est dénoté par d_s ; d_s est un élément de l'ensemble de descripteurs $\mathtt{srce}$: $d_s \in \mathtt{srce}$.

La solution $\mathtt{Sol}(\mathtt{srce})$ est composée d'un ensemble de descripteurs. Chacun de ces descripteurs est dénoté par D_s et comprend un attribut et une valeur, par exemple, CM $= 8$. L'échelle des valeurs est supposé être l'intervalle $[0, 10]$ pour chaque attribut, et chaque attribut est supposé être normalisé selon cette échelle. CM $= 0$ correspond à la moins bonne carte mère du catalogue ; CM $= 10$ correspond à la meilleure. CS $= 0$ correspond à la moins bonne des cartes sons. On supposera que la moins bonne des cartes son est l'absence de carte son, donc le descripteur CS $= 0$ signifie qu'il n'y a pas de carte son dans le PC considéré.

L'exemple suivant de cas source va être utilisé pour illustrer l'algorithme :

$\mathtt{srce} =$

jeux	J	$\in$	[7, 9]
musique	Mu	$\in$	[0, 0]
traitement de texte	TT	$\in$	[6, 10]
programmation	Pgm	$\in$	[5, 8]
traitement d'image	TI	$\in$	[7, 10]
maniabilité	Man	$\in$	[8, 10]
prix	Pr	$\in$	[1800, 2200]

$\mathtt{Sol}(\mathtt{srce}) =$

carte mère	CM	=	8
processeur	CPU	=	8
dispositif de pointage	DP	=	6
CD-Rom	CD	=	4
couleur	Coul	=	3
écran	Écr	=	6
carte son	CS	=	0
imprimante	Imp	=	5

Le problème source est décrit par un ensemble de couples attributs-intervalles, tel que chaque attribut dénote une caractéristique d'un PC et l'intervalle correspondant dénote l'importance de cette caractéristique. Par exemple, le couple $(\text{J}, [7, 9])$ signifie que les jeux sont importants pour cette configuration de PC, alors que le couple $(\text{Mu}, [0, 0])$ signifie que la musique n'a aucune importance pour cette configuration. En prenant en compte les intervalles d'importance du traitement de texte et de la programmation, on peut interpréter ce problème comme étant la recherche d'une configuration de PC généraliste et, selon l'importance associé à l'attribut Man, que le PC doit être facile à déplacer.

Un *appariement* $\mathtt{app}(\mathtt{srce}, \mathtt{cible})$ entre un problème source $\mathtt{srce}$ et un problème cible $\mathtt{cible}$ est un ensemble de triplets $(d_s, d_c, \mathtt{ÉA})$ où :

- d_s est un descripteur du cas source ;
- d_c est le descripteur du problème cible ayant le même nom d'attribut que d_s et
- $\mathtt{ÉA}$ est une *étiquette d'appariement* exprimant les variations des contraintes sur les attributs :
  $\mathtt{ÉA} = (\Delta_{\mathtt{inf}}(d_s), \Delta_{\mathtt{sup}}(d_s))$ où $\Delta_{\mathtt{inf}}(d_s)$ (respectivement, $\Delta_{\mathtt{sup}}(d_s)$) est la variation de $\mathtt{srce}$ à $\mathtt{cible}$ pour la borne inférieure (respectivement, supérieure) de cet intervalle.

Par exemple, $(d_s, d_c, \mathtt{ÉA}) \in \mathtt{app}(\mathtt{srce}, \mathtt{cible})$ avec :

$$\begin{aligned} d_s &= (\text{Pgm} \in [5, 8]) \\ d_c &= (\text{Pgm} \in [6, 7]) \\ \mathtt{ÉA} &= (\Delta_{\mathtt{inf}}(\text{Pgm}), \Delta_{\mathtt{sup}}(\text{Pgm})) = (+1, -1) \end{aligned}$$

L'exemple suivant montre un appariement entre un problème source (partie gauche) et un problème cible (partie droite) :

J	$\in$	$[7, 9]$	$\xrightarrow{+1, +1}$	J	$\in$	$[8, 10]$
Mu	$\in$	$[0, 0]$	$\xrightarrow{+2, +4}$	Mu	$\in$	$[2, 4]$
TT	$\in$	$[6, 10]$	$\xrightarrow{\pm 0, \pm 0}$	TT	$\in$	$[6, 10]$
Pgm	$\in$	$[5, 8]$	$\xrightarrow{+1, -1}$	Pgm	$\in$	$[6, 7]$
TI	$\in$	$[7, 10]$	$\xrightarrow{-3, -2}$	TI	$\in$	$[4, 8]$
Man	$\in$	$[8, 10]$	$\xrightarrow{+2, \pm 0}$	Man	$\in$	$[10, 10]$
Pr	$\in$	$[1800, 2200]$	$\xrightarrow{+200, +100}$	Pr	$\in$	$[2000, 2300]$

Une *dépendance* est un triplet $(d_s, D_s, \texttt{ÉD})$ où $d_s \in \texttt{srce}$, $D_s \in \texttt{Sol}(\texttt{srce})$ et ÉD est une *étiquette de dépendance*. Une étiquette de dépendance ÉD est une valeur réelle qui peut être interprétée comme suit :

– $\texttt{ÉD} > 0$ signifie que les valeurs des descripteurs d_s et D_s varient de la même façon ;

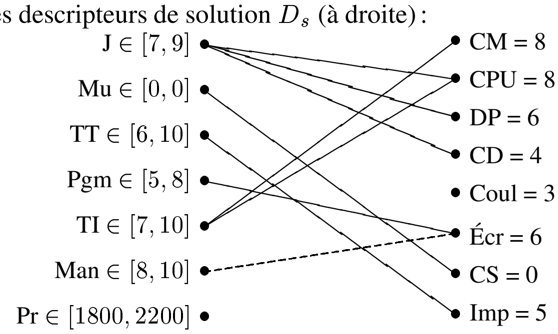
– $\texttt{ÉD} < 0$ signifie que les valeurs des descripteurs d_s et D_s varient en sens contraire et

– $\texttt{ÉD} = 0$ signifie que les variations des valeurs des descripteurs d_s et D_s sont indépendantes.

Plus précisément, $\texttt{ÉD} = \dfrac{\Delta D_s}{\Delta d_s}$, où Δd_s (respectivement, ΔD_s) est une petite variation de d_s (respectivement, de D_s), sous la condition que les autres descripteurs sont constants. En pratique, il est généralement facile de connaître le signe de ÉD, mais l'estimation de sa valeur est plus difficile. Si l'estimation de $\dfrac{\Delta D_s}{\Delta d_s}$ est 0, alors il n'y a pas de dépendance entre les deux descripteurs, et D_s ne varie pas quand d_s varie autour de sa valeur.

L'ensemble des dépendances est dénoté par $\texttt{Dep}^{\underline{\texttt{ces}}}(\texttt{srce})$. Il peut être défini par la personne qui enregistre le cas source dans la base de cas. Dans certains systèmes de RÀPC (voir, par exemple, [8]), $\texttt{Dep}^{\underline{\texttt{ces}}}(\texttt{srce})$ est estimé automatiquement. Notons que les dépendances telles que $\texttt{ÉD} = 0$ sont inutiles et n'ont pas besoin d'apparaître dans $\texttt{Dep}^{\underline{\texttt{ces}}}(\texttt{srce})$.

L'exemple suivant montre des dépendances entre des descripteurs de problème d_s (à gauche) et des descripteurs de solution D_s (à droite) :



Des liens sont associés avec les étiquettes de dépendance ÉD : les segments représentent les étiquettes de dépendances positives et les lignes en pointillés représentent les étiquettes de dépendance négatives. Pour simplifier, pour chaque ÉD positive, la valeur est supposée être $\texttt{ÉD} = +1$ et pour chaque ÉD négative, la valeur est supposée être $\texttt{ÉD} = -1$.

Dans l'exemple ci-dessus, il y a des segments entre J dans `srce` et CPU, PA et CD dans $\texttt{Sol}(\texttt{srce})$. Cela signifie que la valeur du descripteur J dans le problème source a une influence directe sur les valeurs des descripteurs CPU, PA et CD dans la solution de ce problème. Les dépendances sont connues à l'avance et sont mémorisées dans le cas source (elles peuvent être vues comme des *explications* qui lui sont associées, comme dans [7]).

Le résultat $\texttt{Sol}(\texttt{cible})$ de l'algorithme d'adaptation est un ensemble de descripteurs solution D_c définis comme suit. Chaque D_c est donné par un nom d'attribut a pris parmi les noms d'attributs de $\texttt{Sol}(\texttt{srce})$ (tels que CM ou CPU) et un couple $(\texttt{IR}, \texttt{IÉ})$ d'intervalles tels que :

- L'*intervalle restreint* $\texttt{IR} = [\mathtt{R}_{\mathtt{min}}, \mathtt{R}_{\mathtt{max}}]$, est l'intervalle dans lequel la valeur par défaut de l'attribut a peut être choisie et
- L'*intervalle étendu*, $\texttt{IÉ} = [\mathtt{E}_{\mathtt{min}}, \mathtt{E}_{\mathtt{max}}]$ est l'intervalle d'erreur maximum dans lequel la valeur de a peut être choisie.

Dans la suite, un descripteur D_c de $\texttt{Sol}(\texttt{cible})$ est dénoté par $D_c = (a = [\mathtt{E}_{\mathtt{min}}, [\mathtt{R}_{\mathtt{min}}, \mathtt{R}_{\mathtt{max}}], \mathtt{E}_{\mathtt{max}}])$. Les intervalles IR et IÉ calculés grâce à l'algorithme du paragraphe suivant vérifient l'inclusion ensembliste $\texttt{IR} \subseteq \texttt{IÉ}$.

Une *contrainte globale* exprime une propriété qui doit être satisfaite par la solution. Par exemple, la somme des prix des différents éléments de la configuration recherchée doit être dans l'intervalle de prix spécifié par le problème cible. Les contraintes globales ne sont pas formalisées ici ; néanmoins, nous montrons à la fin du paragraphe 4 comment des contraintes globales peuvent être prises en compte.

4 Un algorithme d'adaptation d'un cas

L'algorithme d'adaptation présenté dans ce paragraphe considère individuellement chaque descripteur $D_s \in \texttt{Sol}(\texttt{srce})$ et l'adapte en prenant en compte les dépendances et l'appariement entre `srce` et `cible`. La fonction **adapte_descripteur** effectue l'adaptation de D_s et retourne $D_c = (a = [\mathtt{E}_{\mathtt{min}}, [\mathtt{R}_{\mathtt{min}}, \mathtt{R}_{\mathtt{max}}], \mathtt{E}_{\mathtt{max}}])$, où a est un nom d'attribut et $[\mathtt{E}_{\mathtt{min}}, \mathtt{E}_{\mathtt{max}}]$ et $[\mathtt{R}_{\mathtt{min}}, \mathtt{R}_{\mathtt{max}}]$ dénotent les intervalles restreints et étendus de variation de la valeur de a. Avant de donner l'algorithme pour **adapte_descripteur**, trois exemples sont présentés afin de rendre explicites les étapes de cet algorithme.

1^er^ exemple. Considérons $D_s = (\text{Coul} = 3) \in \texttt{Sol}(\texttt{srce})$. Ce descripteur ne dépend d'aucun descripteur de `srce`. Les variations de `srce` à `cible` ne provoquent donc aucune variation de ce descripteur. Par conséquent, on considère que la valeur 3 peut être conservée pour l'attribut Coul dans $\texttt{Sol}(\texttt{cible})$. Cependant, comme il n'y a pas de dépendance sur Coul, toutes les valeurs de $[0, 10]$ sont permises. Ainsi, $\texttt{IR} = [3, 3]$ et $\texttt{IÉ} = [0, 10]$, et donc Coul $= [0, [3, 3], 10]$ est le descripteur adapté.

2^ème^ exemple. $D_s = (\text{CM} = 8)$ est un exemple de descripteur de $\texttt{Sol}(\texttt{srce})$ qui ne dépend que d'un seul descripteur de `srce`, en l'occurrence $d_s = (\text{TI} \in [7, 10])$.

L'étiquette de dépendance est $\texttt{ÉD} = +1$. L'étiquette d'appariement associé à TI est $\texttt{ÉA} = (\Delta_{\texttt{inf}}(\text{TI}), \Delta_{\texttt{sup}}(\text{TI})) = (-3, -2)$ (voir ci-dessus la description du problème cible courant). L'étiquette de dépendance $\texttt{ÉD} = \dfrac{\Delta D_s}{\Delta d_s} = \dfrac{\Delta\text{Man}}{\Delta\text{TI}}$ entraîne deux variations pour CM, une pour $\Delta_{\texttt{inf}}(\text{TI})$ et l'autre pour $\Delta_{\texttt{sup}}(\text{TI})$:

$$\Delta_{\texttt{inf}}(\text{CM}) = \texttt{ÉD} \cdot \Delta_{\texttt{inf}}(\text{TI}) = (+1) \cdot (-3) = -3$$
$$\Delta_{\texttt{sup}}(\text{CM}) = \texttt{ÉD} \cdot \Delta_{\texttt{sup}}(\text{TI}) = (+1) \cdot (-2) = -2$$

Selon ce raisonnement, la valeur proposée pour CM dans $\texttt{Sol(cible)}$ est comprise entre $\text{CM} = 8 + \Delta_{\texttt{inf}}(\text{TI}) = 5$ et $\text{CM} = 8 + \Delta_{\texttt{sup}}(\text{TI}) = 6$. Cela donne un premier intervalle de valeurs pour CM dans $\texttt{Sol(cible)}$: $\texttt{IR} = [5, 6]$.

La détermination de $\texttt{IR}$ s'appuie principalement sur la valeur $\texttt{ÉD}$ qui peut être difficile à estimer précisément en pratique. Si, à présent, on suppose que le signe de $\texttt{ÉD}$ est connu de façon certaine, les signes de $\Delta_{\texttt{inf}}(\text{CM})$ et $\Delta_{\texttt{sup}}(\text{CM})$ sont aussi connus de façon certaine bien que leurs valeurs ne le soient pas. Dans l'exemple, ces deux signes sont négatifs, ce qui implique que la variation de CM de $\texttt{Sol(srce)}$ à $\texttt{Sol(cible)}$ est négative. Par conséquent, la valeur de CM dans $\texttt{Sol(cible)}$ est inférieure à la valeur de CM dans $\texttt{Sol(srce)}$. Donc, selon ce raisonnement, l'intervalle de valeurs de CM dans $\texttt{Sol(cible)}$ qui doit être choisi est $\texttt{IÉ} = [0, 8]$. En effet, 8 est la valeur associée à CM dans $\texttt{Sol(srce)}$ et 0 est la plus petite valeur possible. Finalement, $\text{CM} = [0, [5, 6], 8]$ est le descripteur adapté D_c.

3ème exemple. $D_s = (\text{Écr} = 6)$ est un exemple de descripteur de $\texttt{Sol(srce)}$ dépendant de deux descripteurs de $\texttt{srce}$, à savoir $(\text{Pgm} \in [5, 8])$ avec $\texttt{ÉD}_{\text{Pgm}} = +1$ et $(\text{Man} \in [8, 10])$ avec $\texttt{ÉD}_{\text{Man}} = -1$. En considérant la dépendance de D_s par rapport à $d_s = (\text{Pgm} \in [5, 8])$, la variation inférieure de Écr est

$$\Delta_{\texttt{inf},\text{Pgm}}(\text{Écr}) = \texttt{ÉD}_{\text{Pgm}} \cdot \Delta_{\texttt{inf}}(\text{Pgm}) = (+1) \cdot (+1) = +1$$

En considérant la dépendance de D_s par rapport à $d_s = (\text{Man} \in [8, 10])$, la variation inférieure de Écr est

$$\Delta_{\texttt{inf},\text{Man}}(\text{Écr}) = \texttt{ÉD}_{\text{Man}} \cdot \Delta_{\texttt{inf}}(\text{Man}) = (-1) \cdot (+2) = -2$$

La variation $\Delta_{\texttt{inf}}(\text{Écr})$ est donnée par la somme des deux variations ci-dessus :

$$\Delta_{\texttt{inf}}(\text{Écr}) = \Delta_{\texttt{inf},\text{Pgm}}(\text{Écr}) + \Delta_{\texttt{inf},\text{Man}}(\text{Écr}) = -1$$

La variation $\Delta_{\texttt{sup}}(\text{Écr})$ est calculée de façon similaire :

$$\Delta_{\texttt{sup}}(\text{Écr}) = \Delta_{\texttt{sup},\text{Pgm}}(\text{Écr}) + \Delta_{\texttt{sup},\text{Man}}(\text{Écr}) = (+1) \cdot (-1) + (-1) \cdot (\pm 0) = -1$$

Puisque $D_s = (\text{Écr} = 6)$ est le descripteur à adapter, les variations ci-dessus donnent l'intervalle $\texttt{IR} = [5, 5]$.

$\texttt{IÉ}$ est calculé grâce aux signes des quatre variations $\Delta_{\texttt{inf},\text{Pgm}}(\text{Écr})$, $\Delta_{\texttt{inf},\text{Man}}(\text{Écr})$, $\Delta_{\texttt{sup},\text{Pgm}}(\text{Écr})$ et $\Delta_{\texttt{sup},\text{Man}}(\text{Écr})$. Ces signes sont respectivement $+$, $-$, $-$ et 0. Si tous ces signes avaient été positifs (respectivement négatifs), alors, une variation positive (respectivement, négative) de la valeur de Écr de $\mathtt{Sol}(\mathtt{srce})$ à $\mathtt{Sol}(\mathtt{cible})$ aurait donné l'intervalle $\mathtt{IÉ} = [6, 10]$ (respectivement, $\mathtt{IÉ} = [0, 6]$). Cela n'est pas le cas, donc le sens de variation de cette valeur est inconnu et donc $\mathtt{IÉ} = [0, 10]$. Finalement, Écr $= [0, [5, 5], 10]$ est le descripteur adapté.

La figure 2 présente un algorithme pour l'adaptation d'un descripteur de $\mathtt{Sol}(\mathtt{srce})$, généralisé à partir de ces trois exemples.

L'algorithme principal est décrit à la figure 3.

Après l'exécution de cet algorithme, les valeurs associées aux attributs de $\mathtt{Sol}(\mathtt{srce})$ sont donnés par les intervalles IR et IÉ. Pour l'exemple, le résultat est :

$$\mathtt{Sol}(\mathtt{cible}) = \boxed{\begin{array}{lcr} \text{CM} & = & [0, [5, 6], 8] \\ \text{CPU} & = & [0, [6, 7], 10] \\ \text{DP} & = & [6, [7, 7], 10] \\ \text{CD} & = & [4, [5, 5], 10] \\ \text{Coul} & = & [0, [3, 3], 10] \\ \text{Ecr} & = & [0, [5, 5], 10] \\ \text{CS} & = & [0, [2, 4], 10] \\ \text{Imp} & = & [5, [5, 5], 5] \end{array}}$$

Pour chaque attribut de $\mathtt{Sol}(\mathtt{cible})$, une valeur précise doit encore être choisie. Ce choix doit être fait en accord avec les contraintes globales et locales. Comme exemple de contrainte globale, le prix du PC doit satisfaire la relation Pr $\in [2000, 2300]$ spécifiée dans cible, le prix étant calculé comme la somme des prix de chaque composant. Les contraintes locales sont données par les intervalles IÉ. Par exemple, la valeur d'une carte mère doit être choisie dans $\mathtt{IÉ} = [0, 6]$. L'intervalle IR donne des suggestions plus précises pour le choix de la valeur : puisque $\mathtt{IR} = [5, 6]$, les valeurs entre 5 et 6 sont suggérées pour CM.

Une dernière remarque doit être faite au sujet des résultats de l'algorithme d'adaptation pour le domaine d'application considéré, le problème de configuration d'un PC. Les opérations d'adaptations décrites sont des substitutions de valeurs. Dans certaines situations, ces substitutions peuvent être interprétées comme des ajouts ou des retraits de composants du PC. Par exemple, (CS $= 0) \in \mathtt{Sol}(\mathtt{srce})$ peut être substitué par (CS $= 3) \in \mathtt{Sol}(\mathtt{cible})$. Puisque CS $= 0$ signifie « pas de carte son », la substitution de 0 par 3 est interprétée, dans ce contexte, comme l'ajout d'une carte son.

Remarques sur la remémoration. Une approche classique de la remémoration [10] consiste à choisir un cas source minimisant les différences entre les énoncés des problèmes source et cible, c'est-à-dire les $|\Delta_{\mathtt{inf}}(d_s)|$ et les $|\Delta_{\mathtt{sup}}(d_s)|$ dans l'approche décrite ici. À l'inverse, selon la philosophie de la remémoration guidée par l'adaptabilité (*adaptation-guided retrieval* [13]), les différences sont à minimiser entre la solution du problème source et la solution du problème cible (à construire), c'est-à-dire, les $|\Delta_{\mathtt{inf}}(D_s)|$ et les $|\Delta_{\mathtt{sup}}(D_s)|$. Les variations des descripteurs solution sont liés aux

adapte_descripteur ***(adaptation d'un descripteur)***

Entrées : • $D_s = (a = V) \in \mathtt{Sol}(\mathtt{srce})$ (a : attribut, V : valeur) ;
• $\mathtt{Dép}$, un ensemble de dépendances
reliant les descripteurs $d_s \in \mathtt{srce}$ au descripteur D_s ;
• $\mathtt{app}(\mathtt{srce}, \mathtt{cible})$, un appariement entre $\mathtt{srce}$ et $\mathtt{cible}$, c'est-à-dire,
un ensemble de triplets $(d_s, d_c, \mathtt{ÉA})$.
Sortie : un descripteur solution $D_c = (a = (\mathtt{IR}, \mathtt{IÉ}))$.

---

début (algorithme)
  si $\mathtt{Dép} = \emptyset$ /* *pas de dépendance, comme dans le 1^er^ exemple* */
    alors retourner $(a = ([V, V], [0, 10]))$ /* $\mathtt{IR} = [V, V]$ *et* $\mathtt{IÉ} = [0, 10]$. */
  $\Delta_{\mathtt{inf}}(D_s) \leftarrow 0$
  $\Delta_{\mathtt{sup}}(D_s) \leftarrow 0$
  $\mathtt{signes} \leftarrow \emptyset$ /* *L'ensemble des signes est initialement vide* */
  pour tout $(d_s, D_s, \mathtt{ÉD}) \in \mathtt{Dép}$ **faire**
    Soit d_c et $\mathtt{ÉA} = (\Delta_{\mathtt{inf}}(d_s), \Delta_{\mathtt{sup}}(d_s))$ tel que $(d_s, d_c, \mathtt{ÉA}) \in \mathtt{app}(\mathtt{srce}, \mathtt{cible})$
    $\Delta_{\mathtt{inf}}(D_s) \leftarrow \Delta_{\mathtt{inf}}(D_s) + \mathtt{ÉD} \cdot \Delta_{\mathtt{inf}}(d_s)$
    $\Delta_{\mathtt{sup}}(D_s) \leftarrow \Delta_{\mathtt{sup}}(D_s) + \mathtt{ÉD} \cdot \Delta_{\mathtt{sup}}(d_s)$
    Calculer les signes de $\mathtt{ÉD} \cdot \Delta_{\mathtt{inf}}(d_s)$ et $\mathtt{ÉD} \cdot \Delta_{\mathtt{sup}}(d_s)$
      ($-$, $+$ ou 0 pour des valeurs négatives, positives ou nulles)
      et les ajouter à l'ensemble $\mathtt{signes}$.
  fin (pour tout)
  $ri_1 \leftarrow V + \Delta_{\mathtt{inf}}(D_s)$
  $ri_2 \leftarrow V + \Delta_{\mathtt{sup}}(D_s)$
  $\mathtt{IR} \leftarrow [\min\{ri_1, ri_2\}, \max\{ri_1, ri_2\}]$
  si $\mathtt{signes} = \{0\}$ /* *tous les signes sont nuls* */
    alors $\mathtt{IÉ} \leftarrow [V, V]$
  si $\mathtt{signes} = \{+\}$ **ou** $\mathtt{signes} = \{0, +\}$ /* *tous les signes sont positifs* */
    alors $\mathtt{IÉ} \leftarrow [V, 10]$
  si $\mathtt{signes} = \{-\}$ **ou** $\mathtt{signes} = \{-, 0\}$ /* *tous les signes sont négatifs* */
    alors $\mathtt{IÉ} \leftarrow [0, V]$
  si $\mathtt{signes} = \{-, +\}$ **ou** $\mathtt{signes} = \{-, 0, +\}$ /* *autres situations* */
    alors $\mathtt{IÉ} \leftarrow [0, 10]$
  retourner $(a = (\mathtt{IR}, \mathtt{IÉ}))$
fin (algorithme)

FIG. 2 – Algorithme d'adaptation d'un descripteur.

adaptation ***(d'un cas)***

Entrées : • $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$, un cas source ;
• $\mathtt{Dep}^{\underline{\mathtt{ces}}}(\mathtt{srce})$, l'ensemble des dépendances entre $\mathtt{srce}$ et $\mathtt{Sol}(\mathtt{srce})$, c'est-à-dire un ensemble de triplets $(d_s, D_s, \text{ÉD})$;
• $\mathtt{cible}$, un problème cible ;
• $\mathtt{app}(\mathtt{srce}, \mathtt{cible})$, un appariement entre $\mathtt{srce}$ et $\mathtt{cible}$.
Sortie : Une solution $\mathtt{Sol}(\mathtt{cible})$ de $\mathtt{cible}$.

---

début (algorithme)
  $\mathtt{Sol}(\mathtt{cible}) \leftarrow \emptyset$
  pour tout $D_s \in \mathtt{Sol}(\mathtt{srce})$ **faire**
    soit Dép, l'ensemble des dépendances entre descripteurs de $\mathtt{srce}$ et D_s :
      $\text{Dép} \leftarrow \{(d, D, \text{ÉD}) \in \mathtt{Dep}^{\underline{\mathtt{ces}}}(\mathtt{srce}) \mid D = D_s\}$
    $D_c \leftarrow$ **adapte_descripteur**$(D_s, \text{Dép}, \mathtt{app}(\mathtt{srce}, \mathtt{cible}))$
    $\mathtt{Sol}(\mathtt{cible}) \leftarrow \mathtt{Sol}(\mathtt{cible}) \cup \{D_c\}$
  fin (pour tout)
  retourner $\mathtt{Sol}(\mathtt{cible})$
fin (algorithme)

FIG. 3 – Algorithme d'adaptation d'un cas.

variations des descripteurs de problèmes par les dépendances. Par exemple, minimiser $|\Delta_{\mathtt{inf}}(\text{CM})|$ consiste à minimiser $|\text{ÉD} \cdot \Delta_{\mathtt{inf}}(\text{TI})|$.

Cela suggère que les étiquettes de dépendance soient utilisées pour choisir les poids des attributs problèmes dans la mesure de similarité : plus un descripteur de problème influence la solution, plus il est important. Cela peut être rapproché de l'utilisation d'une mesure de similarité fondée sur les traces (*foot-printing similarity metrics*, [14]). Une étude précise et détaillée de la remémoration guidée par la procédure d'adaptation décrite dans cet article doit encore être effectuée.

5 Discussion

L'adaptation a été étudiée selon différents points de vue. Parmi les points de vue opérationnels, on peut noter la remémoration guidée par l'adaptabilité [13], l'adaptation comme une configuration [15] et l'adaptation comme une planification [5]. Il y a d'autres travaux sur l'adaptation, en particulier sur les connaissances d'adaptation, mais les trois approches précédentes sont les sources d'inspiration principales de ce travail.

Dans tous ces travaux, l'objectif principal est de concevoir une approche de l'adaptation pratique et indépendante du domaine d'application. Les travaux mentionnés ci-dessus sont généraux et définis à un niveau abstrait. En revanche, dans le travail de recherche décrit dans cet article, nous essayons d'introduire une approche de l'adaptation qui soit à la fois pratique, opérationnelle et indépendante du domaine d'application. Les points importants qui peuvent être mis en évidence et qui sont également pour la plupart présents dans les autres approches sont : (1) l'appariement entre les descripteurs source et cible, (2) les dépendances entre le problème source et sa solution et (3) la transformation des descripteurs et des cas. Selon le signe des dépendances (positives, nulles ou négatives), la valeur d'un descripteur source est changée dans le cas cible suivant ce signe. De plus, les descripteurs sont manipulés un par un dans le cas source (ceci correspond à une décomposition de cas simple).

En comparant avec les approches de l'adaptation mentionnées ci-dessus et celles qui traitent de connaissances d'adaptation (voir par exemple, [3], [12], et [10]), nous « implantons » dans notre proposition la plupart des opérateurs d'adaptation mentionnés le plus couramment : la substitution, la transformation (en particulier, la copie), la composition/décomposition et la généralisation/spécialisation, même si ces opérateurs apparaissent sous une forme simplifiée. De cette façon, nous proposons un algorithme indépendant du domaine d'application, contribuant à l'idée selon laquelle l'adaptation peut être automatisée dans beaucoup de situations.

Cependant, bien qu'opérationnel, simple et facile à adapter, notre approche souffre de limitations essentiellement dues au fait qu'elle ne traite que des descripteurs numériques et que, selon les termes de [15], nous n'effectuons qu'une « adaptation simple », c'est-à-dire une adaptation sur des attributs simples à valeurs partiellement ordonnées. Une étude plus précise doit être menée afin de prendre en compte des opérateurs d'adaptation plus complexes tels que la composition/décomposition ou la spécialisation/généralisation, dans un cadre plus large. Une façon d'améliorer notre approche est d'unifier le travail présenté dans [5] à propos de l'adaptation comme une planification avec le travail décrit dans cet article en prenant en compte les progrès récents sur l'adaptation (comme par exemple celui présenté dans [15]). L'extension de cet algorithme peut s'appuyer sur la conception de règles d'adaptation (contrôlant l'application d'opérateurs d'adaptation) et d'une stratégie générale (un algorithme) d'application de ces règles qui s'appuie sur les notions d'adaptation comme planification et de chemins de similarité. La dernière perspective, mais non la moindre, est l'implantation et le test de cet algorithme dans un cadre applicatif réel pour qu'il soit confronté avec des problèmes réalistes.

6 Conclusion

Dans ce papier, nous avons présenté une approche de l'adaptation dans le cadre du RÀPC qui est à la fois opérationnelle et indépendante du domaine d'application. Cette approche s'appuie principalement sur l'appariement entre des problèmes source et cible et sur les dépendances à l'intérieur d'un cas source. Elle peut être utilisée pour les problèmes dont les caractéristiques peuvent être décrites par des attributs ayant des valeurs numériques ou partiellement ordonnées. Elle a l'avantage d'être générale et facile à comprendre et à réutiliser. En particulier, cette approche correspond bien aux applications décrites dans [9] et [11], qui résolvent des problèmes du « monde réel », développées dans des contextes industriels. Cependant, il reste encore du travail à faire pour étendre cette approche à une plus large catégorie de problèmes et à l'exploitation de connaissances d'adaptation complexes.

Références

[1] K.-D. Althoff, R. Bergmann et L. K. Branting, éditeurs. *Case-Based Reasoning Reasearch and Development — Third International Conference on Case-Based Reasoning (ICCBR-99).* Lecture Notes in Artificial Intelligence 1650. Springer, Berlin, 1999.

[2] R. Bergmann et W. Wilke. Towards a New Formal Model of Transformational Adaptation in Case-Based Reasoning. In H. Prade, éditeur, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98), Brighton, United Kingdom*, pages 53–57, 1998.

[3] J. G. Carbonell. Learning by analogy: Formulating and generalizing plans from past experience. In R. S. Michalski and J. G. Carbonell and T. M. Mitchell, éditeur, *Machine Learning, An Artificial Intelligence Approach*, chapitre 5, pages 137–161. Tioga Press, 1983.

[4] B. A. Davey et H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, Cambridge, UK, 1990.

[5] B. Fuchs, J. Lieber, A. Mille et A. Napoli. Towards a Unified Theory of Adaptation in Case-Based Reasoning. In Althoff et al. [1], pages 104–117.

[6] B. Fuchs, J. Lieber, A. Mille et A. Napoli. An Algorithm for Adaptation in Case-Based Reasoning. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-2000), Berlin, Germany*, pages 45–49, 2000.

[7] B. Fuchs et A. Mille. A Knowledge-Level Task Model of Adaptation in Case-Based Reasoning. In Althoff et al. [1], pages 118–131.

[8] B. Fuchs, A. Mille et B. Chiron. Operator Decision aiding by Adaptation of Supervision Strategies. In M. Veloso et A. Aamodt, éditeurs, *First International Conference on Case-Based Reasoning - ICCBR-95*, Lecture Notes in Artificial Intelligence 1010, pages 23–32. Springer, Berlin, 1995.

[9] O. Herbeaux et A. Mille. ACCELERE: a Case-Based Design Assistant for Closed Cell Rubber Industry. *Knowledge-Based Systems*, 12:231—238, 1999.

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

[11] A. Mille, J.-L. Di-Martino et A. Michel. Adaptation: the key-point in Case Based Reasoning. A case study : Digester Programming Helping, 1995. présenté au *Workshop on practical developments strategies for industrial strength*, Case Based Reasoning applications, 16th International Conference on Artificial Intelligence, IJCAI'95, Montreal, Canada».

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

[13] B. Smyth et M. T. Keane. Retrieving Adaptable Cases. In S. Wess, K.-D. Althoff et M. M. Richter, éditeurs, *Topics in Case-Based Reasoning – First European Workshop (EWCBR'93), Kaiserslautern*, Lecture Notes in Artificial Intelligence 837, pages 209–220. Springer, Berlin, 1994.

[14] M. M. Veloso. *Planning and Learning by Analogical Reasoning*. Lecture Notes in Artificial Intelligence 886. Springer, Berlin, 1994.

[15] W. Wilke, B. Smyth et P. Cunningham. Using Configuration Techniques for Adaptation. In M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard et S. Wess, éditeurs, *Case-Based Reasoning Technologies. From Foundations to Applications*, Lecture Notes in Artificial Intelligence 1400, chapitre 6, pages 139–168. Springer, 1998.