Apprendre à assister l'utilisateur à partir de l'expérience tracée ?

Bruno Mascret[1]

LIRIS, UCBL, France,
bruno.mascret@liris.cnrs.fr,
Site web : http ://liris.cnrs.fr/bmascret/

Résumé L'acquisition de connaissances constitue un problème crucial dans les systèmes d'assistance à l'utilisateur. Dans le cadre de paradigmes de résolution de problèmes comme les raisonnements à partir de cas ou à partir de l'expérience tracée, elle permet aux systèmes assistants de se perfectionner, et éventuellement de s'adapter plus personnellement aux utilisateurs. Nous réalisons une étude en cherchant à définir en quoi l'expérience tracée peut faciliter cette opération d'apprentissage. Nous utilisons les travaux réalisés dans le cadre du RàPC comme base d'inspiration et proposons quelques améliorations, puis nous les adaptons au RàPET en mettant en évidence l'intérêt de cette approche. ...

Abstract. The acquisition of knowledge represents a crucial problem in the assistance systems to users. Within the framework of problem resolution paradigms, such as the case-based reasoning or the trace-based reasoning, it allows assistance systems to improve their possibilities, and perhaps also to adapt personally to users. In this study our issue is to try to define how the trace-based reasoning can facilitate these learning operations. We use the work realized in the CBR framework as an inspiration source. We propose some improvements and we adapt them to the TBR by showing the interest of this approach. ...

Arrivera-t-il jusqu'à la surface de ma claire conscience, ce souvenir, l'instant ancien que l'attraction d'un instant identique est venue de si loin solliciter, émouvoir, soulever tout au fond de moi ?

Marcel Proust, in *Du côté de chez Swann.*

Remerciements : Je tiens à remercier sincèrement mon équipe de recherche qui m'accueille depuis maintenant plus d'un an. Plus particulièrement, mes pensées vont à Alain Mille, qui a dirigé ce travail, Amélie Cordier, dont l'aide fut inestimable, Damien Cram pour ses remarques toujours pertinentes, et Aude Lancelle, dont le soutien – comme toujours – n'a jamais fait défaut.

1 Introduction

La problématique de l'acquisition de connaissances par les systèmes informatiques est peut-être ce qui explique le mieux les motivations qui ont animées les développements de l'intelligence artificielle, des systèmes experts, et aussi un certain optimisme dans la capacité des nouvelles technologies à pouvoir un jour ou l'autre faire quasiment n'importe quoi. Cette vision allègrement relayé par la litérature de science fiction – parfois même les média – n'est pas forcément partagée par les spécialistes mêmes dont on vante les prouesses.

En effet, malgré les inombrables travaux réalisés depuis l'apparition du premier ordinateur personnel, cette problématique est toujours présente et a fait déchanter les plus optimistes des informaticiens. Pendant ce temps-là, le besoin d'assistance et la démocratisation de l'informatique se sont développés de manière remarquable.

De nouveaux paradigmes de résolution de problèmes ont été inventés, comme le raisonnement à partir de cas, ou plus récemment le raisonnement à partir de l'expérience tracée, pour tenter d'améliorer l'assistance à l'utilisateur. Mais tous considèrent comme essentiel un mécanisme particulier de leur fonctionnement : l'acquisition de connaissances, qui est tout sauf triviale.

La question du "comment assister" se pose également : qui n'a pas fini par supprimer ce stupide tombone (ou du moins essayé), et ce malgré l'investissement conséquent en terme de relookage dont il a pu bénéficier lors du passage à la nouvelle version ?

Malgré ce constat, nous faisons partie de ces personnes optimistes qui pensent qu'à défaut de disposer d'un système intelligent, il est toujours possible d'essayer de le rendre moins bête. Nous nous proposons donc d'explorer une facette du problème de l'apprentissage de connaissances, en s'appuyant sur les travaux réalisés dans le cadre des raisonnements à partir de cas (RàPC) et à partir de l'expérience tracée (RàPET). Nous étudierons également les systèmes à bases de traces (SBT), et tenterons d'en démontrer toute l'utilité dans le mécanisme d'apprentissage.

La section 3 décrira une approche particulière d'acquisition de connaissances basée sur le RàPC, dont nous utiliserons les principes afin d'étendre sa portée dans le cadre du RàPC puis du RàPET. Nous poursuivrons en formulant quelques propositions sur les mécanismes d'apprentissage, et tenterons de démontrer en quoi le paradigme du RàPET, associé aux SBT, peut ouvrir des perspectives encourageantes en terme d'assistance personnalisée.

Nous terminerons cette contribution par une discussion résumant nos résultats actuels, et décrivant nos intentions pour l'avenir.

2 Mise en évidence du contexte et des problématiques

Cette partie introductive rend compte du contexte général dans lequel s'inscrit notre domaine de recherche. Nous y mettons en évidence les problématiques directement et indirectement liées à la notion d'assistance par réutilisation de

l'expérience et plus particulièrement l'assistance fondée sur le Raisonnement à Partir de Cas (RàPC) ou sur le Raisonnement à Partir de l'Expérience Tracée (RàPET). Enfin, nous y décrivons la question de recherche qui motivera la suite de nos travaux.

2.1 Contexte général

La complexité des applications informatiques n'a pas cessé de croître depuis l'apparition des premières interfaces graphiques et de la démocratisation de ces outils. Paradoxalement, les logiciels offrent de plus en plus de possibilités mais deviennent également de plus en plus difficiles à appréhender. Les environnements de travail s'enrichissent, la quantité d'information disponible augmente très rapidement, et l'environnement informatique s'étend de manière incommensurable par la connexion Internet.

Le cas extrême des personnes en situation de handicap illustre parfaitement cette nouvelle problématique : malgré les efforts réalisés en matière d'accessibilité du Web, les recommandations de la WAI ne permettent pour l'instant que de rendre palpable le contenu d'une page, et ne renseignent que très faiblement sur la manière possible ou convenable de l'utiliser et d'interagir avec elle. Pour les non-voyants par exemple, si le contenu est (dans le meilleur des cas) perceptible, il génère une nouvelle forme de handicap temporel[2][3], l'accès aux informations pertinentes nécessitant le parcours complet de l'interface graphique[1]. C'est donc l'activité qui est gênée considérablement, et seule l'expérience permet à l'utilisateur de s'approprier l'environnement et ses possibilités conformément à son activité et à ses capacités d'action et perception.

L'utilisateur doit la plupart du temps s'en remettre à lui–même, et espérer que lorsqu'il sera confronté à un nouveau problème de ce type, il sera capable de se souvenir de la soution qu'il avait trouvée. C'est en partie pour tenter de limiter ce risque que de nouveaux outils informatiques ont été imaginés. Nous en présentons les principes dans les sections suivantes.

2.2 Définition de l'assistance.

Il convient avant toute chose de définir correctement ce que nous entendons par assistance. Arnaud Stuber[12] propose ainsi un état de l'art sur la notion d'aide du point de vue des sciences cognitives puis des artefacts informatiques. Gapenne[15] définit l'aide comme *"une relation asymétrique et instrumentée, entre une personne ayant un projet d'action (souhaité ou suggéré voire imposé) dont les modalités de réalisation sont ignorées (ou oubliées) et une technologie censée rendre explicites ces modalités, d'une façon telle qu'elles soient appropriables par la personne sollicitant l'aide"*. Il insiste également sur l'importance de la réflexité de la relation et considère l'aide comme *"un aspect particulier de la dynamique d'apprentissage laquelle se trouve par nature [...] annonciatrice de transformations cognitives"*. Gapenne[15] en déduit quatre types de relations : la substitution, la suppléance, l'assistance et l'aide ; il distingue également la notion de *situation d'aide* de celle de *système d'aide*.

Stuber[12] cherche surtout à *"mettre en évidence la diversité des situations et des moyens*, et choisit de considérer ces deux notions comme *"situation et système d'assistance*. Nous avons choisi d'adopter cette même position en donnant un sens plus large à l'assistance que celui proposé par Gapenne.

2.3 Le RàPC et l'assistance en RàPC

Le Raisonnement à Partir de Cas est un paradigme de résolution de problème spécialisant le raisonnement par analogie. Il existe de très nombreux travaux à ce sujet dans la litérature. Notons parmi les plus importants ceux de Aamodt et Plazza [6], Leake[7], Kolodner[8] ou Althof[9].

Nous reprenons la description donné dans [5] que nous citons : *Il est commode de présenter le raisonnement à partir de cas selon le cycle des tâches à réaliser pour le mener à bien.*
Le schéma présenté en figure 1 est inspiré de celui présenté dans [6] et proposé dans une version revue par Cordier et Al[17].

Elaborer consiste à définir les éléments de description du problème en étant guidé par les nécessités de la recherche en mémoire et de l'adaptation en particulier. Le choix des indices est pour une part réalisé à l'aide des connaissances sur le cas et les inférences à réaliser. Un ou plusieurs cas similaires sont recherchés dans la base de cas. [...] Cette étape nécessite de disposer de critères de similarité pertinents. Les similarités portent une sémantique liée au problème et les décrire en les justifiant est un problème d'acquisition de connaissances. En effet, la similarité entre deux entités dépend complètement du point de vue considéré[...].

Nous insistons que cette dernière remarque qui nous paraît fondamentale lorsqu'on souhaite envisager l'assistance à l'utilisateur. Il est toujours complexe de modéliser un point de vue personnel, bien que de nombreux travaux aient été réalisés en la matière, comme par exemple dans le domaine de la documentation[26] ou du handicap[27].

A partir des cas retrouvés, un appariement plus précis permet de définir le ou les cas qui serviront de base à la construction d'une nouvelle solution. Les systèmes de classification ou de diagnostic s'arrêtent le plus souvent à ce niveau du cycle.
L'adaptation consiste à modifier les réponses du ou des cas retrouvés pour construire une nouvelle solution. L'adaptation est possible en utilisant les connaissances respectives des contextes du cas courant et du cas retrouvé. La connaissance générale permet de vérifier le résultat de l'adaptation. [...]La tâche d'adaptation est partagée avec l'utilisateur et, souvent, une étape de "révision" est nécessaire pour conclure à une solution définitive. Ce processus de révision consiste à évaluer la pertinence de la solution proposée par l'adaptation, et éventuellement de la corriger. C'est ce processus de révision qui prépare l'apprentissage. L'apprentissage consiste, dans les applications simples, à collecter le nouveau cas comme enrichissement de la base de cas, en utilisant le bilan de l'adaptation comme éléments d'indexation.

Nous aurons l'occasion de revenir plus en détail sur l'aspect assistance et la notion d'apprentissage dans la section 3.

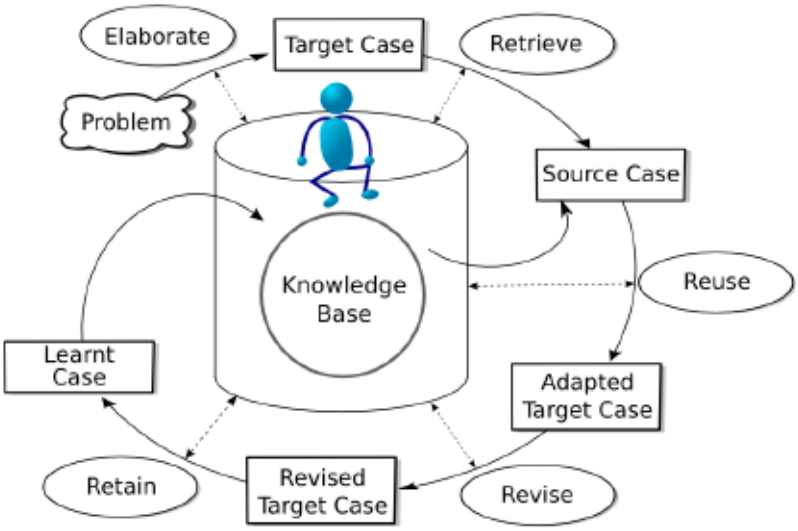

Fig. 1. Le cycle du RàPC, illustration proposée par Amélie Cordier[17]

2.4 Le RàPET

Les principes du RàPET sont décrits par Mille, dont nous reprenons la description proposée dans [10] :

Nous nous proposons d'exploiter les traces d'utilisation d'un dispositif informatique comme sources possibles d'inscription de la connaissance mobilisée par un utilisateur dans sa tâche médiée par le dispositif. Nous proposons donc une théorie permettant de définir précisément ce que nous appelons trace, comment elle peut être codée et l'objet de calculs permettant d'y retrouver des épisodes d'utilisation. Lorsque les traces sont exploitées sur la base de calculs de similarités autorisant un processus d'adaptation d'épisodes repérés comme similaires, alors nous proposons de parler de "Raisonnement à Partir de l'Expérience Tracée" vu comme une évolution généralisante des principes du "Raisonnement à Partir de Cas". Par analogie avec le cycle du RàPC, le cycle du RàPET se présenterait comme illustré [sur la figure 2].

Tout comme dans le raisonnement à partir de cas, nous considérons que la plupart des étapes du cycle de raisonnement peuvent être assurées par le

système informatique ou/et par l'utilisateur lui-même. [...] Le cycle RàPC manipule des cas stockés dans une base de cas en tant qu'épisodes prédéfinis :le cycle RàPET élabore dynamiquement des épisodes potentiellement utilisables dans les traces disponibles ; l'épisode cible accepté et révisé par l'utilisateur est simplement intégré dans la trace en cours. Ce sont les traces qui sont mémorisées en tant que conteneurs d'épisodes potentiels pour des tâches non complètement prévues à l'avance.

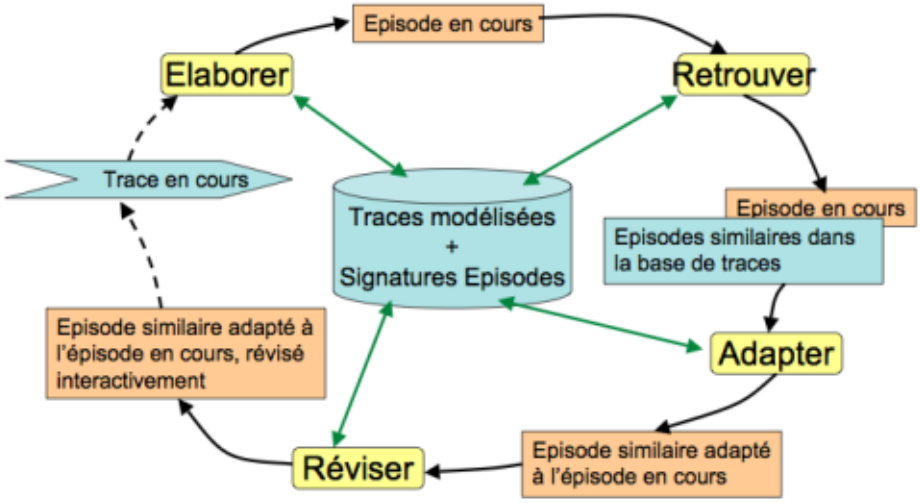

Fig. 2. Le cycle du RàPET, extrait de[10].

2.5 Le principe d'assistance dans le RàPET

Mille[10] pose les bases de la problématique d'assistance dans le RàPET. Nous reprenons son analyse dans ce début de section :

C'est une banalité de rappeler que les environnements informatiques sont largement utilisés pour des tâches de plus en plus nombreuses et variées comme outils d'organisation, de mémorisation, de communication, de recherche et de partage d'informations. Les environnements s'efforcent d'ailleurs de permettre à l'utilisateur de personnaliser au mieux les possibilités qui lui sont offertes pour se rapprocher de ses pratiques, de ses usages et d'une manière générale de ses besoins. Pour accompagner cette tendance forte, il convient de pouvoir rendre compte des tâches réalisées avec les environnements pour tracer les utilisations correspondantes en considérant au mieux des contextes d'utilisation. Le défi est donc d'être capable de tracer l'expérience sous une forme telle qu'elle pourra être remobilisée pour répondre à des questions non complètement définies à l'avance.

Le cycle de vie théorique d'un système basé sur le RàPET et capable d'assistance a été modélisé dans le modèle MUSETTE (Modéliser les USages et les Tâches pour Tracer l'Expérience)[11] comme l'illustre la figure 3.

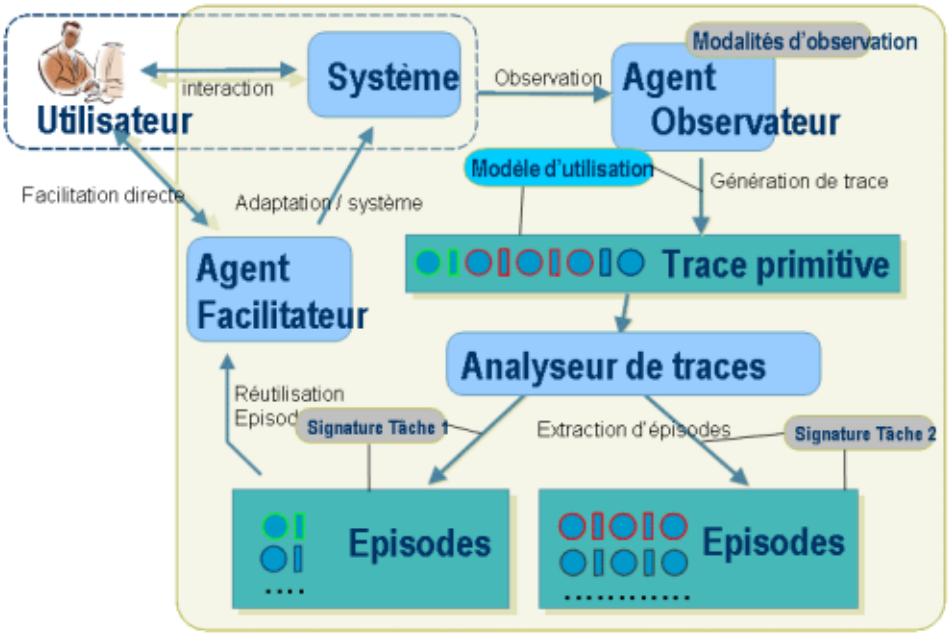

Fig. 3. Musette : une Architecture générale d'un système exploitant des traces d'utilisation à des fins d'assistance[10].

Stuber[12] propose dans sa thèse un système d'assistance basé sur le RàPET. Il expose la problématique de l'acquisition de connaissances mais son intention est de permettre le partage d'expérience entre utilisateurs, avec un système *d'alter-ego*. Il réalise donc un travail remarquable portant sur l'émergence de sens, sa contribution visant à découvrir et mettre en place des ontologies dans des architectures collaboratives, à l'aide des expériences des différents utilisateurs. Ses travaux ne concernent donc pas directement la découverte de connaissances "fonctionnelles".

2.6 Trace informatique et Systèmes à Base de Traces

Lors de l'élaboration de sa solution, l'utilisateur laisse des traces que le système est capable de mémoriser, à condition d'être instrumenté dans cet objectif. Une des principales ambitions des systèmes à base de traces (SBT) est

d'arriver à restituer pertinemment, quand un problème se présente, l'expérience potentiellement acquise lors de la résolution d'un problème similaire.

Pour ce faire, les SBT disposent de mécanismes de manipulation des traces destinés à faciliter leur utilisation par les modules directement impliqués dans les opérations d'assistance. Il ne faut donc pas confondre SBT et RàPET. De notre point de vue, un SBT peut être vu comme une API de communication réalisant des opérations génériques sur les traces qu'elle modélise[14] : collecte, transformation, stockage, manipulation, représentées sur la figure 4.

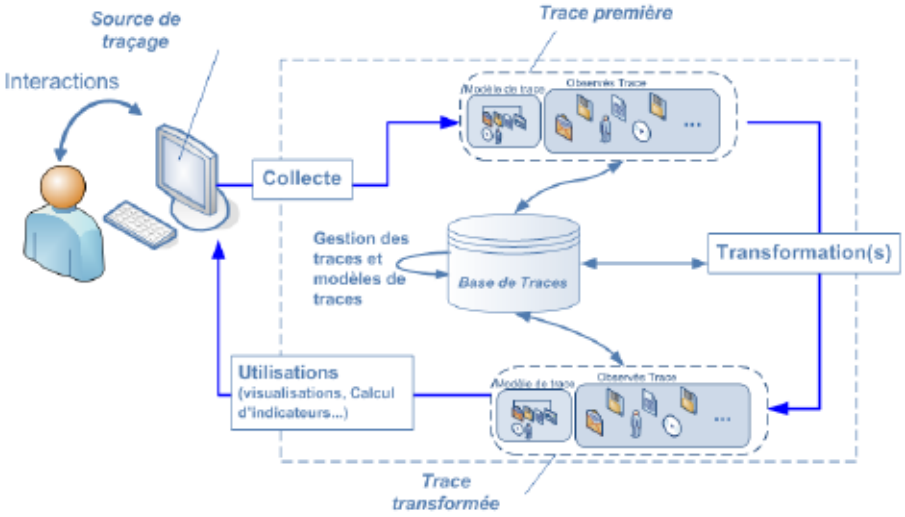

Fig. 4. SBT et trace modélisée : les opérations réalisées par le SBT, illustration de Settouti et Al.[14].

2.7 L'assistance générique : un problème d'apprentissage de connaissances

Nous cherchons maintenant à déterminer quelle orientation donner à nos travaux. Nous avons présenté les différents éléments constituant notre domaine de recherche, et cet état de l'art a montré la variété des approches basées sur le RàPC et le RàPET.

Pourtant, si de nombreux travaux ont déjà vu le jour ou sont en cours de réalisation, aucun d'entre eux n'implémente pour l'instant un système d'assistance générique : une des principales difficultés de ce genre de système provient de la complexité à amorcer un mécanisme d'assistance alors que le SBT et l'assistant ne disposent pas encore de connaissance sur le monde tracé, ni de connaissance d'adaptation. Ces solutions *ad-hoc* démontrent cependant l'intérêt

et la faisabilité de ce type d'approche, nous aurons l'occasion de revenir plus particulièrement sur l'une d'entre elles par la suite. Forts de ces résultats encourageants, il nous semble naturel d'envisager maintenant la réalisation d'un moteur d'assistance afin d'augmenter la portée de cette théorie.

Etudions plus finement les différents scénarii actuels, représentés sur la figure 5. Nous nous intéressons aux scénarii 2 et 3 qui échouent. La solution actuelle proposée pour résoudre le scénario 2 semble satisfaisante, sous réserve qu'une méthode de similarité soit disponible et efficace pour garantir un cas source adaptable. Cependant, il n'y a pas d'acquisition de connaissances avec ce type de stratégie : le problème risque donc de se répéter à nouveau et de reproduire un échec.

Le scénario 3 est également préoccupant : il n'existe aucun moyen de débloquer la situation. Il faudrait pouvoir acquérir une forme de connaissance permettant de se sortir de cette impasse. Nous proposons donc de rechercher des solutions capables de lever ce verrou. Nous voyons bien que la situation d'échec ne peut être résolue par le système qu'en acquérant de nouvelles connaissances sur le problème. Notre étude va donc s'intéresser à tout ce qui peut directement ou indirectement contribuer à rendre l'assistance plus performante, en essayant d'inventer de nouvelles manières d'acquérir de l'expérience ou en proposant des pistes de réflexion allant dans ce sens. Afin de rendre notre contribution le plus exploitable possible, nous nous efforcerons d'envisager l'acquisition de connaissances d'un point de vue générique.

Nous situerons notre recherche dans le cadre du RàPC et du RàPET. Nous chercherons également à mesurer les progrès que pourraient représenter l'association des systèmes à base de traces et du RàPET par rapport au RàPC.

3 Etude des principes d'un mécanisme d'assistance dans le RàPC : IAKA

IAKA (InterActive Knowledge Acquisition) a été imaginé par Amélie Cordier et constitue une partie de sa thèse [16]. Cette approche de l'acquisition interactive de connaissances utilise le RàPC comme paradigme de résolution de problème. Le principal intérêt de ces travaux dans notre problématique vient du principe fondamental de l'approche : *utiliser les corrections d'adaptation pour découvrir de nouvelles connaissances.*

Cordier propose un ensemble de définitions lui permettant d'expliquer les principes de IAKA et s'appuie pour illustrer ses propositions sur le monde de *Flatland* inspiré de la célèbre nouvelle éponyme de Edwin A.Abbott[18]. Elle définit également un formalisme pour exprimer ses propos.

Ce travail nous intéresse particulièrement puisqu'il se situe dans le cadre de l'acquisition des connaissances. Il représente pour nous une base solide sur laquelle développer notre problématique et une source d'inspiration pertinente.

Il nous a semblé utile de reprendre pour la suite de nos propres travaux les exemples et le formalisme utilisé, ce qui peut expliquer l'importance que nous accordons à ces aspects dans cette section.

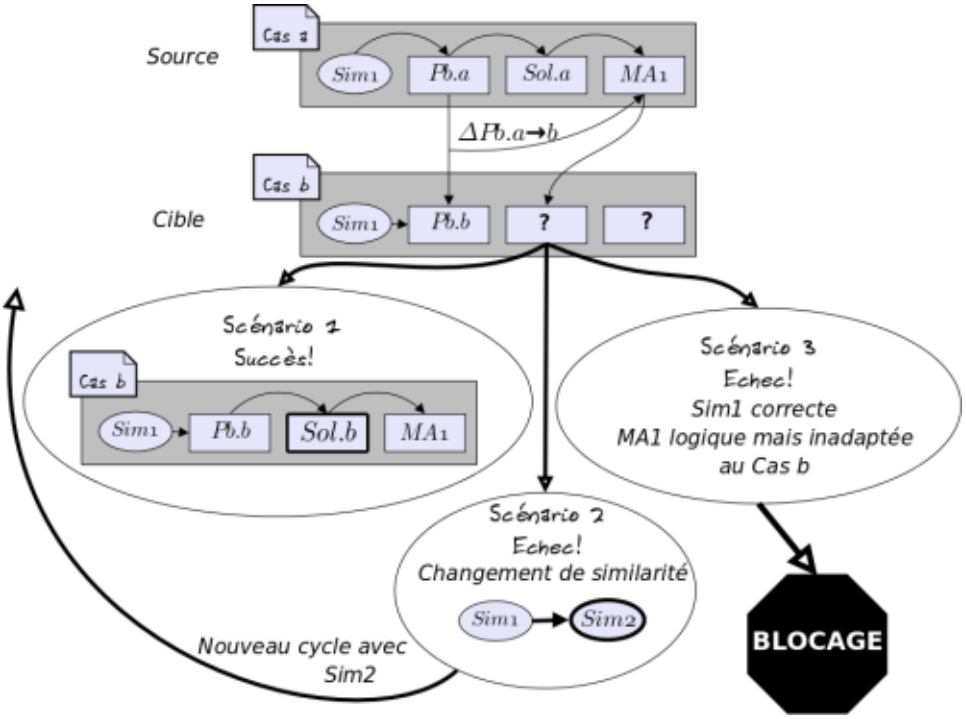

Fig. 5. Les différents scénarii possibles lors d'une adaptation en RàPC : succès (scénario 1), ce qui signifie que la méthode de similarité a proposé un cas source disposant d'une méthode d'adaptation efficace pour traiter le problème ; échec par défaut de similarité (scénario 2), la mesure de similarité ne propose pas de cas valables pour résoudre le problème ; une autre mesure de similarité peut cependant convenir et le cycle reprend ; échec par défaut de connaissance : le cas proposé est bien similaire du point de vue de l'utilisateur, mais la méthode d'adaptation ne convient pas pour résoudre ce type de problème, même si d'un point de vue logique elle n'a pas échoué.

3.1 Le monde de Flatland

La voir figure 6 représente le monde de Flatland. Un individu d'autant plus grand que son nombre de côtés est important. Un certains nombre de règles de composition régissent ce monde, **mais elles ne sont pas connues du système**. Dans IAKA, une seule règle est utilisée : la règle du mariage.

Règle du mariage : lors d'un mariage dans Flatland, deux individus donnent naissance à un nouvel individu dont les caractéristiques dépendent des deux parents. Le nouveau-né aura un côté de plus que le plus petit de ses parents et la couleur de sa mère (le deuxième individu dans un problème). Le cas illustré sur la figure 6 est un mariage.

Nous utiliserons deux autres types de règles dans nos propres travaux, nous les présentons ici bien qu'elles ne soient pas à l'origine décrites dans IAKA :

Règle du combat : lors d'un combat dans Flatland, un nouvel individu est créé en enlevant un côté au plus grand des deux individus impliqués.

Règle de l'alliance : une alliance entre deux individus produit un nouvel individu dont le nombre de côtés représente la somme des côtés des deux individus. Le nouvel individu aura la couleur de l'individu le plus grand.

La figure 7 propose une illustration de ces trois règles. Nous insistons sur le fait que **le système n'a pas connaissance de ces règles**.

3.2 Principes et définitions

Nous présentons dans cette section une partie des principes proposés par Cordier[16]. Nous en reprenons et commentons les définitions en utilisant le même formalisme.

Définition 1. ***Cas, problèmes, solutions*** *: Un cas $\mathcal{C}$ est composé d'une partie problème $\mathtt{pb}$ et d'une partie solution $\mathtt{Sol}(\mathtt{pb})$; la relation de dépendance $\mathcal{H}$ entre $\mathtt{pb}$ et $\mathtt{Sol}(\mathtt{pb})$ représente les liens existant entre les descripteurs de $\mathtt{pb}$ et ceux de $\mathtt{Sol}(\mathtt{pb})$.*

Définition 2. ***Problème source, problème cible, solution candidate*** *: Dans un processus d'adaptation, on appelle problème source $\mathtt{srce}$ la partie problème d'un cas connu et problème cible $\mathtt{tgt}$ (target) la partie problème du cas à adapter. La solution obtenue par adaptation de $\mathtt{Sol}(\mathtt{srce})$ est appelée solution canditate et notée $\widetilde{\mathtt{Sol}}(\mathtt{tgt})$.*

Si on note $\mathcal{L}_{\mathtt{pb}}$ l'espace des problèmes et $\mathcal{L}_{\mathtt{sol}}$ celui des solutions, un *cas source* est donc un doublet $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ tel que $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce})) \in \mathcal{L}_{\mathtt{pb}} \times \mathcal{L}_{\mathtt{sol}}$ ($\mathtt{srce}$ a pour solution $\mathtt{Sol}(\mathtt{srce})$).

Définition 3. ***Fonction d'adaptation*** *$\mathcal{A}_r$: $\mathcal{A}_r$ est une fonction d'adaptation :*

si $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{tgt}) \in \mathcal{L}_{pb} \times \mathcal{L}_{sol} \times \mathcal{L}_{pb}$ et $\mathtt{srce}\ \mathtt{r}\ \mathtt{tgt}$
alors $\mathcal{A}_r(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{tgt})$ est une solution candidate de $\mathtt{srce}$.

Soient deux problèmes $\mathtt{pb}_1$ et $\mathtt{pb}_2$ reliés par $\mathtt{r}$: la fonction d'adaptation $\mathcal{A}_r$ permet de fabriquer une solution candidate $\widetilde{\mathtt{Sol}}(\mathtt{pb}_2)$ par adaptation de $\mathtt{Sol}(\mathtt{pb}_1)$ en considérant $\mathtt{r}$.

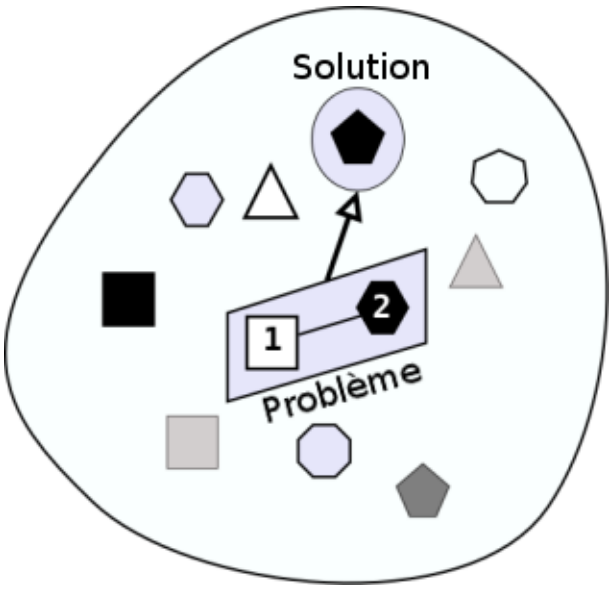

Fig. 6. Le monde de Flatland : les individus sont représentés par des surfaces caractérisées par un nombre de côté et une couleur. Un problème est une paire ordonnée de deux individus (deux surfaces) et une solution un individu seul. Le plus important est qu'on ne dispose pas des règles permettant de déterminer directement la solution d'un problème.

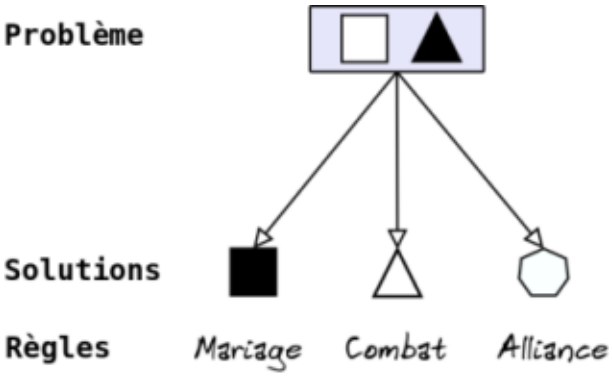

Fig. 7. Les différentes règles (non connues du système) régissant le monde de Flatland : le mariage, le combat et l'alliance. La figure montre les différentes solutions possibles pour un problème de type carré blanc + triangle noir : carré noir (mariage), triangle blanc (combat) ou heptagone noir (alliance).

Le principe d'adaptation exprimé par $\mathcal{A}_{\mathtt{r}}$ est que *les différences entre deux parties problèmes se traduisent par des différences sur leurs solutions.* Les connaissances ne résident pas dans la découverte des mécanismes permettant de calculer une solution à partir d'un problème, mais bien de *savoir adapter une solution existante à un nouveau problème.*

Définition 4. ***Opérateur d'adaptation (Adaptation operator)*** $\mathtt{AO}_r = (\mathtt{r}, \mathcal{A}_r)$ *Un* Opérateur d'adaptation $\mathtt{AO}_r$ *est un doublet* $(\mathtt{r}, \mathcal{A}_r)$ *où* $\mathtt{r}$ *est une relation binaire entre problèmes* ($\mathtt{r} \subseteq \mathcal{L}_{pb} \times \mathcal{L}_{pb}$) *et* $\mathcal{A}_r$ *une fonction d'adaptation.*

Un opérateur d'adaptation permet donc d'exprimer les modifications à apporter sur la solution source en fonction d'une différence constatée entre les problèmes ($\mathtt{r}$) grâce à l'utilisation de $\mathcal{A}_{\mathtt{r}}$. La figure 8 met en relation les notions introduites jusqu'à présent et la figure 9 illustre ces notions dans le monde de Flatland.

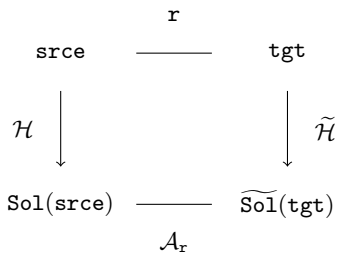

Fig. 8. Un opérateur d'adaptation dans IAKA. La relation $\mathtt{r}$ mesure la (di)similarité entre les deux problèmes, et la fonction d'adaptation $\mathcal{A}_{\mathtt{r}}$ s'applique sur $\mathtt{Sol}(\mathtt{srce})$ afin de déduire de $\mathtt{r}$ ce qu'il faut adapter pour $\widetilde{\mathtt{Sol}}(tgt)$. Illustration de [16]

Définition 5. ***Méthode d'adaptation (Adaptation method)*** $\mathtt{AM}_{srce}$ *: La méthode d'adaptation* $\mathtt{AM}_{srce}$ *associée au cas* $\mathcal{C}1 = (\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ *est un ensemble fini d'opérateurs d'adaptation* $\mathtt{AO}_r = (\mathtt{r}, \mathcal{A}_r)$.

Une méthode d'adaptation contient donc l'ensemble des opérateurs d'adaptation qui peuvent être utilisés pour adapter un cas. La méthode d'adaptation "marier" dans Flatland contiendrait les 6 opérateurs suivants :

1. $\mathtt{AO}_{\mathtt{rn1}}$: $\mathtt{r}_1$ indique que le plus petit individu de $\mathtt{srce}$ a un côté de moins que le plus petit individu de $\mathtt{tgt}$. Dans ce cas $\widetilde{\mathtt{Sol}}(tgt)$ aura un côté de moins que $\mathtt{Sol}(\mathtt{srce})$.
2. $\mathtt{AO}_{\mathtt{rn2}}$: $\mathtt{r}_2$ indique que le plus petit individu de $\mathtt{srce}$ a un côté de plus que le plus petit individu de $\mathtt{tgt}$. Dans ce cas $\widetilde{\mathtt{Sol}}(tgt)$ aura un côté de plus que $\mathtt{Sol}(\mathtt{srce})$.
3. $\mathtt{AO}_{\mathtt{rn3}}$: $\mathtt{r}_3$ indique une différence de couleur entre le deuxième individu de $\mathtt{srce}$ et le deuxième individu de $\mathtt{tgt}$. Dans ce cas $\widetilde{\mathtt{Sol}}(tgt)$ aura la même différence de couleur avec $\mathtt{Sol}(\mathtt{srce})$.

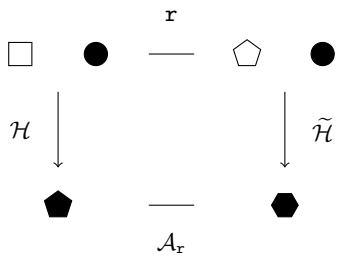

Fig. 9. Un opérateur d'adaptation dans Flatland : le problème cible (pentagone blanc et rond noir) diffère du problème source (carré blanc et rond noir) car la relation `r` nous indique que le premier individu du problème cible a un côté de plus que que celui du problème source. La fonction $\mathcal{A}_{\mathtt{r}}$ ajoute dans ce cas un côté à `Sol(srce)` pour obtenir $\widetilde{\texttt{Sol}}(\texttt{tgt})$. La connaissance exprimée par l'opérateur d'adaptation dans ce cas est "*lorsque le plus petit des individus du problème cible a un côté supplémentaire que le plus petit individu du problème source, alors la solution cible aura un côté de plus que la solution du cas source*".

4. $\overline{\texttt{AO}_{\texttt{rn4}}}$: $\mathtt{r}_4$ indique que le plus grand individu de `srce` a un côté de moins que le plus grand individu de `tgt`. Dans ce cas il n'y a pas d'incidence sur $\widetilde{\texttt{Sol}}(tgt)$.
5. $\overline{\texttt{AO}_{\texttt{rn5}}}$: $\mathtt{r}_5$ indique que le plus grand individu de `srce` a un côté de plus que le grand petit individu de `tgt`. Dans ce cas il n'y a pas d'incidence sur $\widetilde{\texttt{Sol}}(tgt)$.
6. $\overline{\texttt{AO}_{\texttt{rn6}}}$: $\mathtt{r}_6$ indique une différence de couleur entre le premier individu de `srce` et le premier individu de `tgt`. Dans ce cas il n'y a pas d'incidence sur $\widetilde{\texttt{Sol}}(tgt)$.

Donc $\texttt{AM}_{marier} = \{\texttt{AO}_{\texttt{rn1}}, \texttt{AO}_{\texttt{rn2}}, \texttt{AO}_{\texttt{rn3}}, \overline{\texttt{AO}_{\texttt{rn4}}}, \overline{\texttt{AO}_{\texttt{rn5}}}, \overline{\texttt{AO}_{\texttt{rn6}}}\}$. L'utilisation des conjugués sur les opérateurs est une commodité de notation que nous proposons afin de montrer que la relation $\mathtt{r}_i$ n'a pas d'influence sur la solution.

Définition 6. ***Chemin de similarité (Similarity path)*** *: Un chemin de simiarité allant d'un probème* `srce` *à un problème* `tgt` *est un ensemble q de triplet $(\texttt{pb}_{i-1}, \texttt{r}_i, \texttt{pb}_i)$ avec :*

- $\texttt{pb}_i$ *: i^{th} problème du chemin ;*
- $\texttt{pb}_0 = \texttt{srce}$*;*
- $\texttt{pb}_q = \texttt{tgt}$*;*
- $\texttt{pb}_{i-1}\ \texttt{r}_i\ \texttt{pb}_i$ *(pour $i \in \{1, \ldots, q\}$) ;*
- $\texttt{r}_i$ *telle que $(\texttt{r}_i, \mathcal{A}_{\texttt{r}_i})$ ets un opérateur d'adaptation valable.*

Si $\texttt{srce} = \texttt{pb}_0$ *et* $\texttt{tgt} = \texttt{pb}_q$*, alors le chemin de similarité peut être écrit :*

$$\texttt{pb}_0\ \texttt{r}_1\ \texttt{pb}_1\ \texttt{r}_2\ \texttt{pb}_2 \ldots \texttt{pb}_{q-1}\ \texttt{r}_q\ \texttt{pb}_q$$

Définition 7. ***Etape d'adaptation*** *: Chaque tripet $(\widetilde{Sol}(\texttt{pb}_{i-1}), \mathcal{A}_{\texttt{r}_i}, \widetilde{Sol}(\texttt{pb}_i))$ est appelé une étape d'adaptation.*

Définition 8. ***Chemin d'adaptation*** $\mathtt{AP}$ ***(Adaptation path)*** *: Le chemin d'adaptation* $\mathtt{AP}$ *associé au chemin de similarité* $\mathtt{SP}$ *est un ensemble* q *de triplets* $(\widetilde{Sol}(pb_{i-1}), \mathcal{A}_{r_i}, \widetilde{Sol}(pb_i))$ *avec :*

- $\widetilde{Sol}(pb_0) = Sol(srce)$;
- $\widetilde{Sol}(pb_i) = \mathcal{A}_{r_i}(pb_{i-1}, \widetilde{Sol}(pb_{i-1}), pb_i)$;
- $\widetilde{Sol}(pb_q) = \widetilde{Sol}(tgt)$.

La figure 10 représente un chemin d'adaptation dans IAKA dans lequel on retrouve les notions présentées plus haut. La figure 11 propose un chemin d'adaptation dans Flatland.

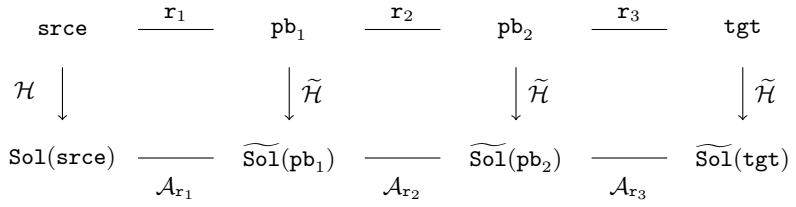

Fig. 10. Un chemin d'adaptation dans IAKA : la première ligne représente le chemin de similarité entre le problème `srce` et le problème `tgt`, sur lequel apparaissent différentes étapes et problèmes intermédiaires $\mathtt{pb}_i$. La deuxième ligne représente les solutions candidates obtenues par application d'un opérateur d'adaptation pour chaque étape du chemin. La dernière étape d'adaptation aboutit à la proposition d'une solution candidate $\widetilde{\mathtt{Sol}}(\mathtt{tgt})$ pour `tgt`.

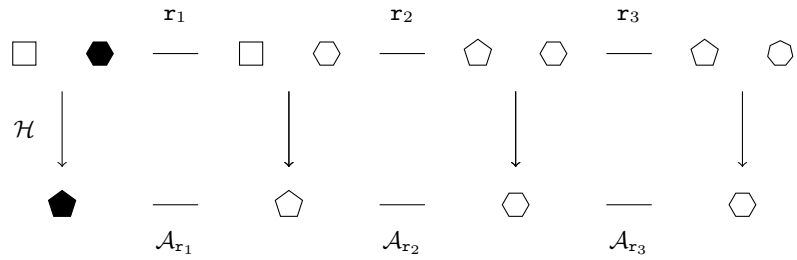

Fig. 11. Une adaptation en trois étapes dans Flatland. En reprenant les opérateurs définis plus haut, on part d'un problème (carré blanc, hexagone noir) de solution (pentagone noir) et on applique successivement $\mathtt{AO}_{\mathtt{rn3}}$, $\mathtt{AO}_{\mathtt{rn2}}$ et $\overline{\mathtt{AO}_{\mathtt{rn5}}}$ pour arriver au problème cible (pentagone blanc, heptagone blanc).

Cordier[17] propose encore d'autres principes complémentaires (erreurs d'adaptation, oracle et experts) que nous ne présenterons pas ici car nous ne les utiliserons pas par la suite.

Intérêts et limites de IAKA IAKA présente pour nous un intérêt certain dans notre problématique d'acquisition de connaissances : les techniques et principes utilisés s'appuyent sur le RàPC, nous pouvons peut-être nous en inspirer dans le cadre du RàPET.

Nous souhaitons cependant compliquer un peu le monde proposé par Cordier [16] : dans les exemples de IAKA, il n'y a qu'un seul type de connaissances à acquérir : la connaissance du mariage. Or il nous semble intéressant d'augmenter ce nombre de possibilités afin de mettre en lumière certaines aspects non envisagés au départ dans le cas de systèmes plus complexes, tels que les scenarii 2 et 3 de la figure 5. La figure 12 présente ce que pourrait être alors une situation d'échec dans Flatland.

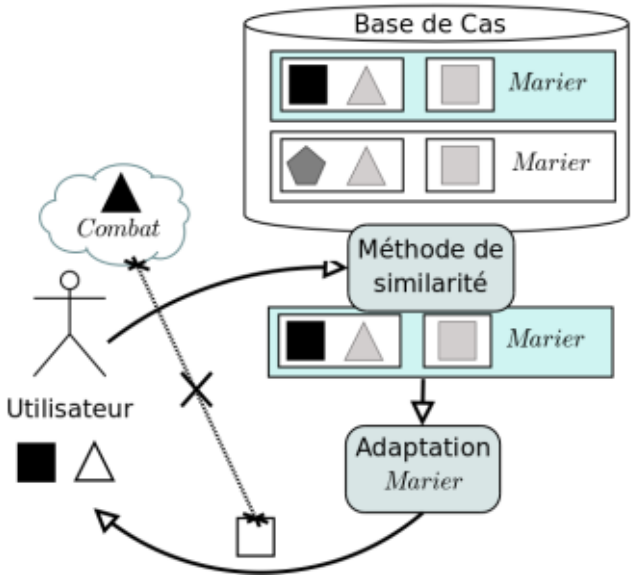

Fig. 12. Une situation d'échec du RàPC dans Flatland : l'épisode source (carré noir + triangle blanc) donne un résultat correct par similarité avec un cas connu (carré noir + triangle gris clair) ; cependant, l'intention de l'utilisateur est de réaliser un combat, alors que la connaissance d'adaptation correspondant au cas retenu est "marier". La solution proposée ne correspond donc pas aux attentes de l'utilisateur.

Nous commencerons par tenter de prolonger les principes de IAKA pour trouver une solution à l'apprentissage d'ambiguïtés dans le cadre du RàPC,

puis étudierons ce que le RàPET offre comme possibilités supplémentaires pour améliorer l'apprentissage des connaissances.

4 Propositions

4.1 Vers une approche interactive de l'apprentissage de connaissances d'assistance en RàPC

Améliorer l'analyse ou apprendre à adapter ? Lorsqu'on est dans la situation de considérer que deux problèmes sont *similaires*, c'est que l'on considère que la solution de l'un peut servir de base à la solution de l'autre (principe fondamental du RàPC) moyennant une *méthode d'adaptation* (liée au cas source).

Les *différences constatées entre les problèmes* permettent de caluler les *différences à appliquer à la solution source* pour obtenir la solution cible. La figure 5 présente les différents scénarii possible :

1. Le résultat de l'adaptation est directement satisfaisant (scénario 1) ;
2. Le résultat de l'adaptation n'est pas satisfaisant (en contradiction avec une connaissance connue de l'assistant et/ou de l'utilisateur) (scenarii 2 et 3).

L'origine du problème peut provenir de deux causes :

1. **la méthode de similarité utilisée n'est pas satisfaisante** pour distinguer un cas adaptable d'un cas non adaptable (scénario 1) ; on peut alors envisager une révision de la méthode de similarité et des méthodes d'adaptation associées ;
2. **la mesure de similarité semble correcte** (on retrouve bien des problèmes jugés similaires par l'utilisateur) **mais la méthode d'adaptation n'est pas la bonne** (scenarii 2 et 3).

Plusieurs pistes complémentaires nous permettraient d'améliorer le premier scénario :

1. **combiner des méthodes de similarité existantes**[22] : une méthode de similarité est d'autant plus performante qu'elle est spécifique ; ainsi, elle sera efficace dans certains cas et inutile dans d'autres. Disposer de nombreuses méthodes de similarité permettrait de discriminer plus facilement deux problèmes, mais serait également plus coûteux en terme de performance ;
2. **offrir un paramétrage dynamique des méthodes**[23] : une méthode de similarité peut échouer simplement parce qu'elle a été configurée trop largement pour un problème particulier ; dans ce cas, une modification du degré de finesse pourrait suffir pour amener la résolution de la situation ;
3. **adapter les méthodes de similarité à l'utilisateur** : une méthode de similarité peut plus ou moins convenir à l'utilisateur suivant que son activité se rapproche ou s'éloigne du cadre prévu par une méthode de similarité. Par exemple, les méthodes temporelles[19] peuvent s'avérer très performantes ou au contraire peu adéquates pour un déficient visuel, qui est capable de

réaliser très rapidement certains types d'actions (saisie clavier, navigation par raccourci clavier, etc.) mais peu performant en terme de temps sur d'autres (exploration de page web par tabulation, positionnement rapide dans une interface graphique, etc.). En prenant en compte ces différences de performances en fonction des activités, la méthode s'adapterait au profil d'utilisation.

Il existe dans la littérature de nombreuses contributions traitant déjà de genre de problématique[20]. Cependant, si l'amélioration du mécanisme de détection des cas reste un apport essentiel au RàPC et au RàPET, nous allons en montrer les limites en terme de performance et d'ingénierie. Tout d'abord, d'un point de vue pratique, il nous paraît illusoire d'imaginer qu'un concepteur soit en mesure de prévoir l'ensemble des cas d'utilisation d'un environnement informatique. En effet, chaque utilisateur a sa propre manière d'interagir avec cet environnement, une finalité difficilement prévisible, et une configuration matérielle et logicielle particulière. Il en résulte des disparités suffisamment conséquentes pour affaiblir considérablement l'exhaustivité d'une solution se basant uniquement sur la similarité. Il ne faut pas oublier que les méthodes de similarité sont par nature *prévues à l'avance* par un concepteur. Un cas d'école dans le monde de la déficience visuelle illustre parfaitement cette situation : un webmestre propose sur son site de vente de posters, de manière anodine, une horloge capable de lire l'heure ; dès lors une foule de non-voyants mémorisent l'adresse du site non pas pour les services qu'il propose, mais pour disposer d'une montre parlante ! Il est peu probable que le concepteur du site ait prévu ce genre d'utilisation.

Ensuite, en terme de performances, il semble plus naturel d'adapter un jeu réduit de cas une fois la similarité effectuée plutôt que de *sur-développer* la phase d'analyse. Même si les configurations matérielles tendent à respecter la loi de Moore quant aux vitesses des processeurs, il reste plus rapide de diminuer le nombre d'opérations, d'autant que dans de nombreux cas un mécanisme de similarité un peu évolué suffira largement à limiter le nombre de cas critiques. Quoiqu'il en soit, une solution basée uniquement sur la phase d'analyse conduira plus ou moins vite à un échec à un moment donné ; il nous apparaît plus pertinent en l'occurence d'adopter une position pessimiste, quitte à se réjouir des résultats proposés par un bon mécanisme d'analyse.

Cette étude rapide nous conforte donc dans notre intention de chercher à améliorer l'aspect assitance du RàPC, tout en mettant en évidence les avantages qu'offrirait un mécanisme d'analyse efficace dans la limitation des cas critiques. Il faut donc trouver une possibilité d'augmenter le contexte d'un problème si on veut que l'assistant puisse le résoudre.

Reformulation du problème Reprenons l'exemple d'échec du RàPC présenté sur la figure 12 : l'erreur commise pourrait-être imputable à la mesure de similarité, mais nous cherchons maintenant à considérer ce genre de situation directement dans l'assistant (scénarii 2). Le mécanisme de révision propre au RàPC ne doit pas être sollicité, car il ne s'agit pas de corriger un cas devenu obsolète : la méthode d'adaptation "marier" demeure satisfaisante pour le système.

Formalissons la situation :

Situation 1. Soient $\mathcal{C}_1$ et $\mathcal{C}_2$ deux cas sources différents dont les parties problème sont $\texttt{srce}_1$ et $\texttt{srce}_2$, ayant pour solutions respectives $\texttt{Sol}(\texttt{srce}_1)$ et $\texttt{Sol}(\texttt{srce}_2)$. Soient $\texttt{tgt}$ la partie cible d'un épisode à adapter et σ_1 une méthode de similarité déterministe.

Imaginons :

$$\texttt{srce}_1 = \texttt{srce}_2 \tag{1}$$

$$\texttt{Sol}(\texttt{srce}_1) \not\approx \texttt{Sol}(\texttt{srce}_2) \tag{2}$$

Trivialement, nous pouvons déduire de (1) et du caractère déterministe de σ_1 :

$$\sigma_1(\texttt{tgt}, \texttt{srce}_1) = \sigma_1(\texttt{tgt}, \texttt{srce}_2). \tag{3}$$

Donc, du point de vue du système d'assistance, $\mathcal{C}_1 = \mathcal{C}_2$ ce qui nous conduit à supposer que $\texttt{Sol}(\texttt{srce}_1) = \texttt{Sol}(\texttt{srce}_2)$, en contradiction avec la propriété (2). Observons sur la figure 13 les chemins d'adaptation simplifiés de $\texttt{tgt}$ $\texttt{AP}_1$ et $\texttt{AP}_2$ correspondant respectivement à $\texttt{srce}_1$ et $\texttt{srce}_2$:

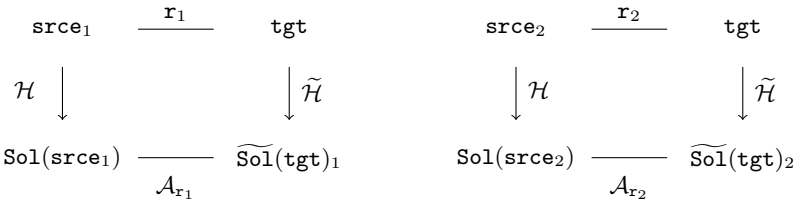

Fig. 13. Chemins d'adaptation (simplifiés à une seule relation) des problèmes $\texttt{srce}_1$ et $\texttt{srce}_2$ pour le problème–cible $\texttt{tgt}$ aboutissant à deux solutions candidates possibles $\widetilde{\texttt{Sol}}(\texttt{tgt})_1$ et $\widetilde{\texttt{Sol}}(\texttt{tgt})_2$

Si (2) est vrai, cela nous conduit à regarder dans le chemin quel composant génère les différences observées ; or, la propriété (1) nous permet de déduire que $\texttt{r}_1{=}\texttt{r}_2$. La relation $\mathcal{H}$ est une simple relation de dépendance qui signifie uniquement qu'il existe des liens entre les parties problème et solution d'un cas. On ne peut donc expliquer (2) qu'en admettant $\mathcal{A}_{\texttt{r}_1} \neq \mathcal{A}_{\texttt{r}_2}$. Nous pouvons donc proposer :

Proposition 1. *Si deux cas présentent dans leur signature un problème identique et une solution différente, alors leurs méthodes d'adaptation pour un même problème cible sont également différentes.*

Corollaire 1. *: Si deux cas source utilisent la même méthode d'adaptation sur un même problème cible et que les solutions candidates obtenues ne sont pas équivalentes, alors leur partie problème est différente.*

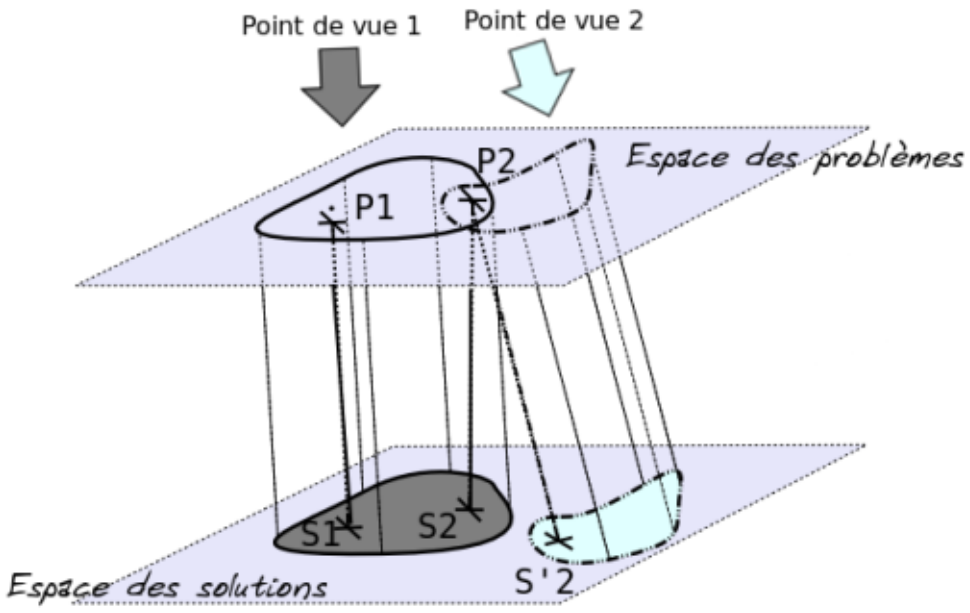

Fig. 14. Dans cettte illustration, l'espace des problèmes et celui des solutions sont représentés par deux plans parallèles, dont les axes représentent les différents descripteurs. Dans le premier plan $\mathcal{P}_{pb}$, plus les problèmes sont similaires, plus ils sont proches. Chaque classe de problème est donc représentée par une surface. Chaque point de vue ou intention est représenté par un vecteur $\vec{v_i}$. Un point de vue *intention* sur un ensemble de problème (surface) se traduit alors par une projection de surface de $\mathcal{P}_{pb}$ vers $\mathcal{P}_{sol}$ suivant $\vec{v}_{intention}$. On remarque que le problème $\texttt{pb}_1$ appartient à deux ensembles de problèmes (il est situé dans l'intersection des surfaces). Il a donc deux solutions distinctes dans $\mathcal{P}_{sol}$, une pour chaque point de vue (vecteur de projection)

Nous venons donc de démontrer qu'une partie du problème réside dans l'impossibilité du système actuel à déterminer l'intention de l'utilisateur, ou une partie du contexte. La représentation utilisée sur la figure 14 illustre moins formellement le genre de situation qui nous préoccupe dorénavant. Nous terminons cette section en proposant une première solution dans le cadre du RàPC. Nous aurons l'occasion de revenir sur cet aspect dans le RàPET en section 4.3.

Impliquer l'utilisateur dans le processus de décision. Notre idée est assez simple et n'occasionne que peu de modifications du système d'adaptation utilisé dans IaKa. Nous supposons que le système ne dispose pas d'instruments ou de connaissances lui permettant de choisir seul la méthode d'adaptation à utiliser dans une situation de blocage. Nous souhaitons également éviter que l'utilisateur, lorsque cela lui est permis, essaye de réviser un cas valable d'un certain point de vue, d'autant que les opérateurs d'adaptation disponibles ne permettent pas forcément d'exprimer son intention véritable. Bien que des dispositifs du RàPC permettent un certain suivi dans les révisions, le risque de perte d'informations pertinentes est bien réel : l'adaptation n'a pas "techniquement" échoué et reste utile.

Nous proposons donc, lorsque le cas se présente, de solliciter l'utilisateur afin qu'il nous indique lui-même son intention. Cela suppose qu'il soit en mesure de répondre, et qu'au moins un des choix proposés corresponde effectivement à son problème. Il s'agit donc de filtrer les cas sources en ajoutant une contrainte sur leur méthode d'adaptation. L'architecture du système s'en trouve légèrement modifiée, mais ne nécessite techniquement qu'une étape de filtrage supplémentaire. La figure 15 propose un cas d'utilisation général du système appliqué au monde de Flatland. Une telle solution peut néanmoins occasionner une gêne pour l'utilisateur si l'assistant le sollicite trop souvent. Stuber [12] insiste sur l'importance de favoriser le plus possible l'automatisation des assistants. Il n'y a pas à notre sens de stratégie d'adaptation optimale, il convient plus d'adapter l'assistant en fonction des attentes de l'utilisateur et de la complexité du domaine. Cette dernière remarque résonne particulièrement avec les travaux de Gapenne déjà cités[15] précédemment, et nous ramène à notre définition de l'assistance élargie.

Nous avons envisagé, sans chercher à être exhaustif, différentes stratégies pour utiliser efficacement ce moteur d'assistance dans des implémentations logicielles. Nous terminons cette section par quelques réflexions parmi celles que nous jugeons les plus intéressantes :

Stratégie d'assistance optimiste : Une stratégie optimiste considère que le problème source le plus similaire au problème cible propose la meilleure adaptation ; le bénéfice attendu dans ce cas est temporel, le cycle d'assistance ne sollicitant l'utilisateur qu'en cas d'ambiguïté (situation 1) ou en cas d'erreur. Les systèmes les plus enclins à admettre cette stratégie doivent garantir un faible taux d'échec et une bonne discrimination des problèmes.

Stratégie d'assistance pessimiste : Une stratégie pessimiste au contraire estime qu'un problème source similairement éloigné d'un problème cible dispose de chances suffisament importantes pour être retenu. Par exemple, si

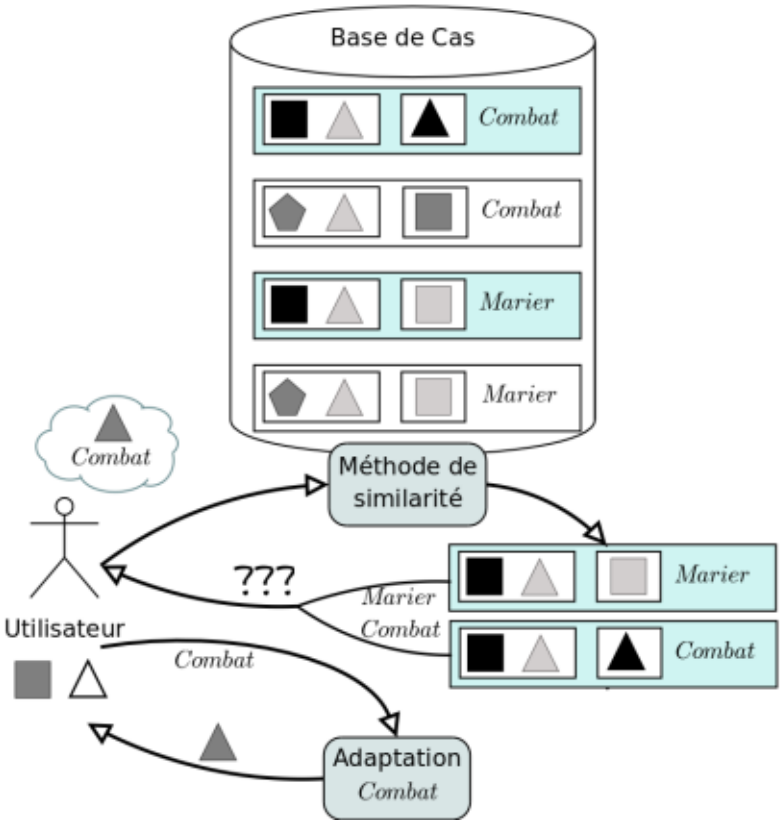

Fig. 15. Une situation à deux signatures valables dans Flatland : l'épisode source (carré gris clair + triangle blanc) donne un résultat correct par similarité avec deux cas connus (carré noir + triangle gris clair) ; ils proposent l'un une adaptation par $\mathcal{A}_{Marrier}$, l'autre par $\mathcal{A}_{Combat}$. L'utilisateur est alors sollicité par l'assistant et choisi la deuxième solution ($\mathcal{A}_{Combat}$) qui correspond à son problème actuel. Le cycle se poursuit et l'adaptation réussit.

on supprime sur la figure 15 le premier cas (carré noir, triangle gris clair, $\mathcal{A}_{Combat}$), le cas le plus intéressant pour adapter suivant $\mathcal{A}_{Combat}$ est plus éloigné que le cas (carré noir, triangle gris clair, $\mathcal{A}_{Marrier}$). Cette stratégie permet une adaptation plus sûre mais sollicitant davantage l'utilisateur.

Stratégie mixte L'idée est de combiner les deux stratégies précédentes, par exemple utiliser une stratégie optimiste lorsqu'un cas source a déjà été adapté un certain nombre de fois avec succès, sinon d'adopter une approche pessimiste.

Présentation catégorielle Les problèmes sources sont regroupés par leur méthode d'adaptation ; on peut également envisager un classement des différentes méthodes suivant leur généricité, fréquence d'utilisation, etc.

Ces quelques stratégies ne sauraient couvrir l'ensemble des possibilités de mode d'assistance. L'ergonomie, les sciences cognitives, la sociologie, les aides techniques ou la psychologie apportent à ce sujet de nombreuses contributions qu'il serait pertinent d'étudier plus en détail. Nous ne développerons donc pas plus cet aspect d'utilisation, qui nous écarte de notre problématique actuelle.

Nous avons proposé dans cette section une amélioration des performances du RàPC dans le cas de signatures de problème identiques, par sollicitation de l'utilisateur. Néanmoins, cette solution suppose qu'il existe une méthode d'adaptation pour répondre à notre problème. La section suivante présente un autre mécanisme offrant cette fois-ci la possibilité de créer de nouvelles méthodes d'adaptation par interaction avec l'utilisateur lorsqu'aucune solution n'est disponible.

4.2 Vers une acquisition interactive de connaissances d'adaptation

Lors de notre état de l'art, nous avons attiré l'attention du lecteur sur les difficultés tant en RàPC qu'en RàPET à amorcer ce type de systèmes. La plupart du temps, il est nécessaire d'avoir recours à une phase d'élaboration et de disposer de connaissances précises sur l'environnement assisté. Nous avons également soutenu l'idée que malgré tous ces efforts, il est quasiment impossible d'imaginer l'ensemble des adaptations que devra fournir l'assistant. En général, les méthodes d'adaptation sont prévues à l'avance et sont difficilement programmables par l'utilisateur.

Reprenons les exemples précédents : le deuxième scénario de la figure 5 peut encore aboutir à un échec, si aucune des méthodes d'adaptation disponible n'est satisfaisante. Dans Flatland, cela signifie qu'en cas d'échec d'adaptation par $\mathcal{A}_{Marrier}$ et $\mathcal{A}_{Combat}$, nous nous trouvons de nouveau dans une situation bloquante : l'intention de l'utilisateur n'est pas prévue par le système. De même, sur la figure 12, il n'y a pas de cas disposant de $\mathcal{A}_{Combat}$, donc pas d'adaptation possible.

Nous envisageons dans cette section un système offrant à l'utilisateur la possibilité de créer ses propres méthodes d'adaptations. Nous nous situons maintenant à la frontière entre le RàPC et le RàPET.

Création de méthodes à partir d'opérateurs existants Dans cette première partie, nous faisons l'hypothèse que l'utilisateur peut aider le système d'assistance à acquérir de nouvelles connaissances sur la manière d'adapter un problème en réutilisant des opérateurs d'adaptations déjà existants.

Le tableau 16 représente les connaissances d'adaptation déjà connues grâce aux deux méthodes d'adaptation $\mathcal{A}_{marier}$ et $\mathcal{A}_{combat}$. Par commodité et souci de simplicité, on suppose qu'il est possible de ne par tenir compte d'une relation $\mathtt{r}_i$ à condition qu'elle puisse s'exprimer autrement (c'est le cas entre $\mathtt{r}_3$ + $\mathtt{r}_6$ et $\mathtt{r}_7$).On s'aperçoit que la méthode d'adaptation $\mathcal{A}_{alliance}$ si elle était définie utiliserait des opérateurs d'adaptation déjà connus par l'assistant.

Relation	description	$\mathcal{A}_{marier}$	$\mathcal{A}_{combat}$	$\mathcal{A}_{alliance}$
$\mathtt{r}_1$	côté - 1 sur petits	$\mathtt{AO}_{\mathtt{rn1}}$	$\overline{\mathtt{AO}_{\mathtt{rn1}}}$	$\mathtt{AO}_{\mathtt{rn1}}$
$\mathtt{r}_2$	côté + 1 sur petits	$\mathtt{AO}_{\mathtt{rn2}}$	$\overline{\mathtt{AO}_{\mathtt{rn2}}}$	$\mathtt{AO}_{\mathtt{rn2}}$
$\mathtt{r}_3$	couleur $\neq$ sur 2ème	$\mathtt{AO}_{\mathtt{rn3}}$		
$\mathtt{r}_4$	côté - 1 sur grands	$\overline{\mathtt{AO}_{\mathtt{rn4}}}$	$\mathtt{AO}_{\mathtt{rn4}}$	$\mathtt{AO}_{\mathtt{rn4}}$
$\mathtt{r}_5$	côté + 1 sur grands	$\overline{\mathtt{AO}_{\mathtt{rn5}}}$	$\mathtt{AO}_{\mathtt{rn5}}$	$\mathtt{AO}_{\mathtt{rn5}}$
$\mathtt{r}_6$	couleur $\neq$ sur 1er	$\overline{\mathtt{AO}_{\mathtt{rn6}}}$		
$\mathtt{r}_7$	couleur $\neq$ sur grands		$\mathtt{AO}_{\mathtt{rn7}}$	$\mathtt{AO}_{\mathtt{rn7}}$

Fig. 16. Tableau présentant les différents opérateurs d'adaptation utilisés par les méthodes $\mathcal{A}_{marier}$, $\mathcal{A}_{combat}$ et $\mathcal{A}_{alliance}$ dans Flatland. Le méthode $\mathcal{A}_{alliance}$ utilise des opérateurs déjà présents dans $\mathcal{A}_{marier}$ ($\mathtt{AO}_{\mathtt{rn1}}$ et $\mathtt{AO}_{\mathtt{rn2}}$) et $\mathcal{A}_{combat}$ ($\mathtt{AO}_{\mathtt{rn4}}$,$\mathtt{AO}_{\mathtt{rn5}}$ et $\mathtt{AO}_{\mathtt{rn7}}$).

Il est donc possible qu'un utilisateur ait la possibilité de fabriquer une nouvelle méthode d'adaptation par composition d'opérateurs existants, lorsqu'aucune méthode proposée n'est satisfaisante. Cette idée nous semble potentiellement intéressante, dans la mesure où elle permet à l'utilisateur seul d'éviter une situation de blocage. On pourrait imaginer également dans le cadre de systèmes d'assistances communiquants que cette nouvelle méthode soit ensuite diffusée aux autres utilisateurs, afin de partager cette nouvelle connaissance d'adaptation.

Nous allons approfondir cette piste en imaginant ce que pourrait apporter un système proposant une grande variétés d'opérateurs à ses utilisateurs.

La conception d'opérateurs génériques et les avantages des SBT dans ce cadre. Dans un monde aussi simple que Flatland, il est certain que la plupart des opérateurs sont rapidement identifiables. Les limites imposées par le cadre du RàPC quant aux types de problèmes possibles sont rapidement atteintes. Aussi, il semble envisageable d'obtenir du concepteur tous les opérateurs permettant à l'utilisateur d'élaborer les méthodes dont il a besoin.

Néanmoins, plus le monde se complexifie, plus il semble difficile de prévoir l'ensemble des opérateurs dont on pourrait avoir besoin. Nous retrouvons le

même genre de problématique que nous évoquions avec les mesures de similarité : il est certes possible de les améliorer, mais rien ne nous permet d'acquérir la certitude que toutes les éventualités seront exprimables en terme d'opérateurs. De plus, il semble difficile de demander à un utilisateur de manipuler des opérateurs dans des structures complexes, ce travail recquérant assez rapidement des capacités de logique et de programmation.

La fabrication d'opérateurs risque également de n'être utilisable que pour un environnement particulier : si deux logiciels différents utilisent des assistants, il serait agréable de pouvoir bénéficier des mêmes opérateurs, à condition bien entendu qu'ils soient compatibles. Cette compatibilité repose avant tout sur la structure des cas. Or rien ne permet en RàPC de garantir que deux cas issus de deux systèmes d'assistance différents soient compatibles.

Sur cet aspect, les systèmes à base de traces pourraient apporter une certaine souplesse. Imaginons en effet qu'un environnement de travail soit suffisement équipé pour permettre un traçage de ses différents composants logiciels. Une trace provenant du logiciel A a la même structure logique qu'une trace provenant d'un logiciel B ; de plus, le SBT peut être sollicité pour transformer ces traces afin de les rendre compatibles, de les fusionner, ou de les adapter suivant les demandes formulées par les assistants.

Si l'assistant du logiciel A souhaite apprendre de celui du logiciel B, il faut bien que ces deux systèmes soient capables d'échanger et de communiquer. L'utilisation d'un SBT générique pourrait jouer ce rôle d'interface de communication.

Nous n'irons pas plus loin sur cette piste, que nous ne pourrions développer exhaustivement dans le cadre de ce rapport. Nous souhaitons néanmoins présenter quelques points qu'il nous semble indispensable d'envisager pour traiter correctement cette problématique :

1. **l'importance de la classification des opérateurs (et des méthodes d'adaptation)** : trop de possibilités peuvent décourager l'utilisateur ; il importe d'imaginer un système de classification permettant de regrouper les opérateurs par thèmes ou domaines d'application, et de définir une stratégie de présentation de ces opérateurs.
2. **création de nouveaux opérateurs par composition** : un opérateur générique peut certes s'appliquer plus facilement, mais son intérêt est souvent limité. Il nous apparaît particulièrement utile d'arriver à concevoir de nouvelles fonctions d'adaptation par composition de fonctions existantes. Une approche mathématique du problème, utilisant par exemple la logique combinatoire[24] ou la logique temporelle d'Allen[25] pourrait être une clef d'entrée.
3. **diffusion et partage de nouvelles connaissances** : cet aspect est abordé par Stuber et al.[13] et constitue un élément-clef de sa thèse[12].

Une première manière d'envisager l'amorce de RàPET Nous avons évoqué dans notre partie introductive la difficulté d'amorcer des systèmes d'assistance en RàPC et en RàPET, lorsqu'aucune connaissance d'adaptation n'est

disponible. Une des principales raisons à notre avis de cet état de fait provient de la manière même de concervoir ces systèmes :

1. Soit le système est conçu en vue d'assistance à l'origine même ; l'étape d'élaboration est donc prévue par les concepteurs, et le système dispose donc lors de sa mise en production d'un nombre suffisant de connaissances d'adaptation pour fonctionner ; l'ontologie des cas est déjà existante ;

2. Soit le système n'est pas conçu en vue d'assistance au départ mais peut potentiellement offrir des informations capables d'utiliser le RàPC ou le RàPET. Dans ce cas, la plupart des implémentations passent par une réinstrumentation du système et/ou une mise à disposition de la logique de conception et de l'ontologie des cas. L'amorce est réalisée par des experts lors d'une phase d'élaboration *adhoc*.

Nous nous intéressons maintenant à un troisième cas envisageable : nous nous situons dans le cadre de la deuxième possibilité, sans que la phase d'élaboration ait été réalisée. Le RàPC montre ses limites à s'adapter à des environnements dont la logique est à découvrir entièrement. Sans connaissance, il devient difficile de décrire ce qu'est un cas !

En revanche, l'utilisation des traces avec le RàPET prend toute son importance : le matériau de base est présent (la trace), il reste à le manipuler afin de le rendre adaptable. Une partie des opérations nécessaires est déjà réalisée par le SBT, notamment la collecte. Les cas ne sont plus des morceaux isolés d'évènements, mais sont tous contenus quelque part dans la trace. Un cas peut donc prendre des formes très variées suivant l'analyse effectuée.

La structure de base étant connue, reste à l'utiliser dans un mécanisme d'assistance. Notre proposition est d'utiliser des opérateurs au minimum génériques, et dans l'idéal plus spécifiques, afin de fabriquer les méthodes d'adaptation.

Nous pensons que le plus important au départ n'est pas nécessairement de disposer de cas, mais plutôt d'opérateurs et de traces "intéressantes", c'est-à-dire suffisemment évoluées pour que l'assistance soit facilitée. Une trace première peut s'avérer complexe à utiliser, le SBT a donc un rôle fondamental à jouer dans la préparation des traces.

Nous envisageons donc le travail des experts sous un autre angle d'approche : connaissant le système et son fonctionnement, leur rôle consisterait plus à spécifier des opérateurs pertinents afin de rendre possible le plus grand nombre d'adaptations, autrement dit de "faire confiance" à l'utilisateur en lui donnant les outils nécessaires lui permettant d'injecter ses connaissances dans l'assistant.

Nous terminons cette proposition théorique en rappelant l'importance du partage des connaissances et de l'expérience, qui constitue à notre avis une approche novatrice des mécanismes d'assistance, en adéquation avec les nouveaux comportements des utilisateurs face aux réseaux et à Internet. Nous renvoyons le lecteur aux travaux de Stuber à ce sujet[13].

4.3 Utilisation interactive des traces pour améliorer l'apprentissage

Nous étudions dans cette section un dernier scénario possible, pour lequel les propositions précédentes n'ont pas permis de résoudre la situation de blocage du scénario 3. Dans cette situation, il n'a pas été possible d'acquérir de nouvelles connaissances ni en utilisant une méthode d'adaptation existante, ni en réalisant de nouvelles méthodes d'adaptation pour traiter le problème.

Les limites du RàPC sont cette fois-ci définitivement atteintes : il est impossible de débloquer l'assistant ; il ne reste qu'à revoir le système dans son ensemble, et à proposer une nouvelle mesure de similarité et/ou ajouter de nouveaux descripteurs aux problèmes ou les élargir. Ces solutions nécesistent une refonte du système dans son ensemble, dont une phase de ré-apprentissage des cas.

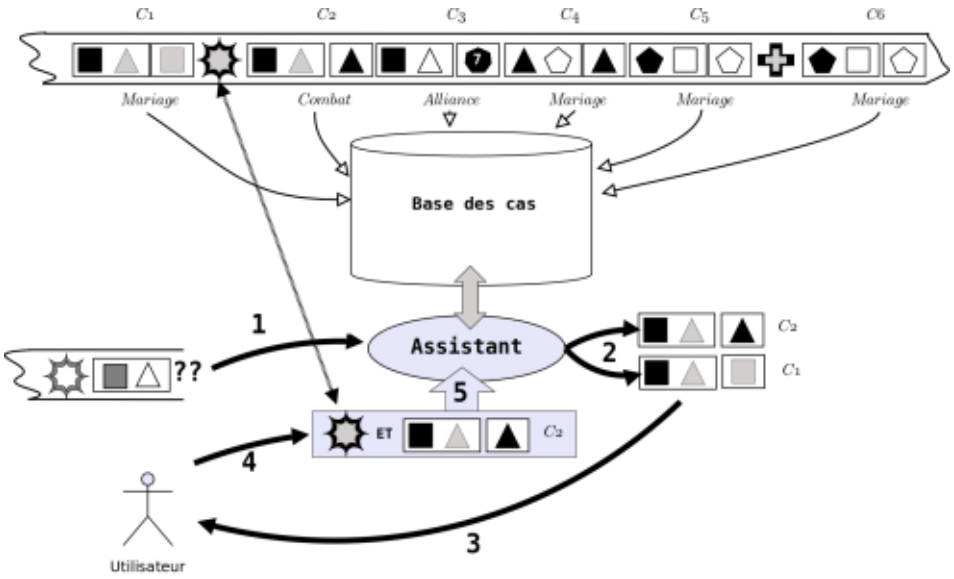

Fig. 17. Un déblocage par augmentation des connaissances sur le contexte du problème, exploitant les traces du RàPET et sollicitant l'utilisateur. Au départ, deux problèmes ayant deux méthodes d'adaptation différentes sont proposés (combat et mariage) ; l'utilisateur explique à l'assistant pourquoi il faut choisir combat (à cause de la déclaration de guerre contenue dans la trace) ; la nouvelle connaissance est apprise par l'assistant qui saura lorsqu'un cas similaire se présentera discrimer seul les deux problèmes.

En revanche, dans le cadre du RàPET, il nous reste la possibilité d'interroger l'utilisateur afin qu'il nous indique si dans son activité, ou dans la trace qui a fournit le cas source, un observé permettrait de lever l'ambiguïté. Nous pouvons donc envisager le cas d'utilisation présenté sur la figure 17. Dans cet exemple,

la trace présente une information importante : la déclaration de guerre. Une déclaration de guerre est réalisée entre deux couleurs, et tant qu'un armistice n'est pas conclu, il n'y a plus de mariage ni d'alliance possible entre deux individus dont les couleurs sont belligérantes. De plus, dans ce contexte particulier, la probabilité de réaliser des alliances avantageuses (c'est-à-dire dont la solution aboutit à la création d'un individu dont la couleur est actuellement en conflit) augmente pour les problèmes impliquant au moins un belligérant.

Deux formes de connaissances sont récupérables : des connaissances certaines (probabilité de 1 comme pour combat) et des connaissances probabilistes (comme pour l'alliance). Remarquons que ces connaissances peuvent également être universelles ou limitées à un contexte donné (la guerre s'achève lors de l'armistice).

Il convient maintenant d'arriver à représenter ces informations convenablement dans le système. Le SBT nous apporte une première solution à ce niveau : il dispose de capacités de reformulation de trace par transformation, et peut par exemple ajouter une propriété à un observé, insérer un observé "guerre" en début de problème, ou encore paramétrer l'analyseur pour qu'il tienne compte du contexte "guerre" dans ses propositions. Ainsi, en cas de guerre, un observé peut recevoir une propriété "ennemis" contenant les couleurs de ses ennemis actuels. Cette information doit ensuite être portée à la connaissance de l'assistant afin qu'il puisse l'utiliser à bon escient.

Nous n'avons pas appronfondi cette piste qui nous semble particulièrement prometteuse. Nous pensons cependant avoir présenté suffisamment d'éléments pour convaincre le lecteur de la faisabilité de cette approche, sans pouvoir encore en démontrer formellement la portée.

5 Discussion

Nous avons présenté une étude exploratoire s'interessant à la problématique de l'acquisition de connaissances dans les mécanismes d'assistance en RàPC et RàPET. Nous avons formulé plusieures propositions permettant de traiter certains cas dans le RàPC, et mis en évidence les limites de ce paradigme en matière de convivialité pour l'utilisateur et/ou son incapacité à augmenter la description des problèmes pour certains types de situations.

Le RàPET et les SBT en revanche semblent plus à même de gérer ces situations, quoique notre étude n'ait pu aboutir à une preuve formelle de cette hypothèse. Nous avons cependant suggéré quelques pistes de réflexion qui nous ont parues assez prometteuses.

Afin de tester nos propositions, nous avons commencé à développer un prototype reprenant les différents points exposés dans cette contribution. Les sources du logiciel sont téléchargeables sur le dépôt svn du projet[1].

Le prototype a permis de vérifier certains aspects exposés ici, comme la levée d'ambiguïté par l'utilisateur, mais n'implémente pas encore toutes les fonctionnalités destinées à vérifier l'ensemble des propositions.

[1] https ://svn.liris.cnrs.fr/bmascret/flatland.

Nous pensons également que si le monde de Flatland est un bon terrain d'expérimentation pour confronter la théorie au cas concrêt d'une implémentation, il serait judicieux d'envisager un système proposant des situations plus réelles. Nous avons commencé une collaboration avec la société Urbilog, spécialisée en accessibilité Web, afin d'exploiter les traces d'utilisation d'internet.

Nous avons plusieurs fois utilisé le monde du handicap dans nos exemples : plus qu'un cas particulier d'interaction, nous considérons les technologies d'assistance au handicap comme un formidable terrain d'expérimentation, où les différences d'usage sont immédiatement palpables. Par exemple, le lecteur se sera peut-être intérrogé sur la longueur des légendes de cet article ; elles ont été particulièrement soignées afin de les rendre accessibles à un non-voyant. Mais néanmoins, nous pensons qu'elles offrent un éclairage utile à tout lecteur !

Nous envisageons dans un premier temps de poursuivre cette étude dans le cadre de l'assistance au handicap pour l'accessibilité du Web. Il serait ensuite interessant de proposer un système plus générique d'assistance, situé au niveau de la couche session du modèle OSI, afin d'étendre nos travaux à un environnement complet de travail. Nos premiers contacts avec le Gnome Accessibility Project ont rencontré un vif intérêt pour notre approche, nous espérons pouvoir concrétiser dans l'avenir ce type de collaboration.

Références

[1] Sperandio, J.-C., Uzan, G., Oltra, R. : L'informatique comme barrière d'exclusion ou comme aide technique à l'intégration. Performances Humaines et Techniques, hors série : "Situation de Handicap", p. 34–40 (1999)

[2] Sperandio, J.-C., Uzan, G. : Ergonomie des aides techniques informatiques pour personnes handicapées. Handicap, revue de sciences humaines et sociales (2002).

[3] Uzan, G. : Temps technologiques, temps individuels, temps sociaux : l'articulation des contraintes temporelles dans l'utilisation de l'informatique par des aveugles. Colloque Fisaf Paris (2003).

[4] Cordier, A., Fuchs, B., Mille, A. : Le raisonnement à partir de cas : un paradigme de réutilisation de l'expérience. Réutilisation de l'expérience (2008) (à paraître)

[5] Mille, A., Fuchs, B.,Chiron,B. : Raisonnement fondé sur l'expérience : Un nouveau paradigme en supervision industrielle ? Revue d'intelligence artificielle, 13 :97–128 (1999)

[6] Aamodt, A.,Plaza,E. : Case Based Reasoning : Foundational issues, Methodological Variations, and System Approaches. AICOM, vol. 1, p. 39–59 (1994)

[7] Leake, D. B. : Case-Based Reasoning : Experiences, Lessons, and Future Directions. Menlo Park, CA : AAAI Press/MIT Press, Menlo Park, CA. (1996)

[8] Kolodner, J.L. : Case-Based Learning. Kluwer Academic Publishers, Dordrecht, Netherlands (1993)

[9] Althoff, K.-D. : Evaluating Case-Based Reasoning Systems : The INRECA Case Study. Habilitationsschrift (Postdoctoral Thesis), Department of Computer Science, University of Kaiserslautern, Germany (1997)

[10] Mille, A. : Raisonner à partir de l'expérience tracée (RàPET). Définition, illustration et résonnances avec le "story telling" (2005)

[11] Laflaquière, J., Prié, Y. : MUSETTE. Réutilisation de l'expérience (2008) (à paraître)

[12] Stuber, A. : Co-construction de sens par négociation pour la réutilisation en situation de l'expérience tracée. Thèse n.305.2007, Chap.4.2 p28–32 (2007)

[13] Stuber, A., Hassas, S., Mille, A. : Combiner le paradigme multi-agents et le raisonnement à partir d'expérience pour assister la réalisation collective de tâches. 12ème Atelier de Raisonnement à Partir de Cas - LIPN, Université Paris13, Villetaneuse - p. 99-103 (2004)

[14] Settouti, L.S., Prié, Y., Marty, J.C., Mille, A. : Vers des Systèmes à Base de Traces modélisées pour les EIAH Rapport de recherche RR-LIRIS-2007-016 (2007)

[15] Gapenne, O. : Introduction : Relation d'aide et transformation cognitive. Cité par [12] Intellectica, vol. 44, n.2, p7–16 (2006)

[16] Cordier, A. : Apprentissage interactif de connaissances en raisonnement à partir de cas. Thèse de doctorat, UCBL, Chapitre 4(IAKA) (2008) (à paraître)

[17] Cordier, A., Fuchs, B., and Mille, A. : Engineering and Learning of Adaptation Knowledge in Case-Based Reasoning. Staab, S. and Svatek, V., editors, 15th International Conference on Knowledge Engineering and Knowledge Management - EKAW 2006, pages 303–217. Springer-Verlag Berlin, LNAI 4248 (2006)

[18] Abbott,E. A. : Flatland, A Romance of Many Dimensions. Second, revised edition, Abbott (1884)

[19] Ben Saad, M. : Découverte de connaissances dans les traces d'interaction : une approche par similarité des séquences temporelles Mémoire de master rescherche, UCBL, LIRIS (2008)

[20] Bisson, G. : La similarité : une notion symbolique/numérique. Apprentissage symbolique-numérique (tome 2), chapitre XX,Eds Moulet, Brito. Editions CEPADUES (2000)

[21] Rifqi, M. : Mesures de comparaison, typicalité et classification d'objets flous : théorie et pratique. Thèse de doctorat (1996)

[22] Bouchon-Meunier, B., Rifqi, M. : A framework to unify and generate measures of comparison. Tatra Mountains, vol. 12, p. 89-97 (1997)

[23] Rifqi, M., Berger, V., Bouchon-Meunier, B. : Discrimination power of measures of comparison. Fuzzy Sets and Systems 110, p. 189–196 (2000)

[24] Curry, H.,Hindley,J. R., Seldin, J. P. : Combinatory Logic II. North-Holland (1972)

[25] Allen, J. : Maintaining knowledge about temporal intervals. Communications of the ACM 26(11) :832-843 (1983)

[26] Benel,A. : Consultation assistée par ordinateur de la documentation en sciences humaines : considérations épistémologiques, solutions opératoires et applications à l'archéologie". Thèse de doctorat, UCBL (2003)

[27] Moreau, C., Mascret, B. : Lexique LSF, A web academy for French Sign Language. Language Resources and Evaluation Conference (LREC 2008, 6th edition), 3rd workshop on the Representation and Processing of Sign Languages :Construction and Exploitation of Sign Language Corpora. Palais des Congrès Mansour Eddahbi, Marrakech, Maroc (2008)