Modification de l'ATMS pour une prise en compte efficace de la justification d'hypothèses.

Sylvain Dekoker, Amal El Fallah-Seghrouchni, Patrick Taillibert

Modification de l'ATMS pour une prise en compte efficace de la justification d'hypothèses

Sylvain Dekoker[1,2], Amal El Fallah Seghrouchni[2], Patrick Taillibert[1]

[1] Thales Systèmes Aéroportés
2, Avenue Gay Lussac 78851 Elancourt Cédex
patrick.taillibert@fr.thalesgroup.com

[2] Laboratoire d'Informatique de Paris 6
4 place Jussieu 75005 Paris
sylvain.dekoker@lip6.fr et amal.elfallah@lip6.fr

Résumé : L'ATMS est un système de maintien de la cohérence qui calcule pour chaque fait les ensembles d'hypothèses cohérentes dont il dépend (son label). La justification d'hypothèses dans l'ATMS, bien que possible, pose un problème combinatoire. En effet, la taille du label d'une hypothèse croit linéairement en fonction du nombre d'hypothèse qui la justifie. Or l'ATMS est d'une complexité exponentielle en fonction de la taille des labels. Nous proposons une modification de la définition des labels en considérant des environnements clos pour l'implication. Ainsi le label des hypothèses contient un unique environnement qui croit linéairement en fonction du nombre d'hypothèses qui la justifie. Le calcul de la clôture d'un environnement est d'une complexité polynomiale. Nous montrerons une application de cette technique pour le raisonnement temporel.

Mots-clés : ATMS, maintenance de vérité, hypothèses, raisonnement hypothétique

1 Introduction

Les TMS (Truth Maintenance System) sont des systèmes créés pour faciliter la construction d'applications mettant en œuvre des connaissances pouvant s'avérer contradictoires.

Parmi les TMS on distinguera le Justification based Truth Maintenance System (JTMS) (Doyle, 1979) qui entretient un unique état consistant, et l'Assumption based Truth Maintenance System (ATMS) (de Kleer, 1986; de Kleer & Reiter, 1987; De Kleer, 1992) qui est multi-contexte, et entretient donc tous les états possibles en parallèle. C'est de ce dernier dont parle cet article.

Comme son nom l'indique l'ATMS utilise une approche à base d'hypothèses. Ce sont des sous-ensembles de faits, désignés par le concepteur, qui seront remis en cause lors de l'ajout de contradiction. Le rôle de l'ATMS est de calculer le label de chaque fait, c'est à dire les ensembles d'hypothèses dont il dépend.

Certaines applications nécessitent pour garantir la cohérence de faire se justifier les hypothèses entres elles (raisonnement temporel, diagnostic de cascade de pannes). L'ATMS le permet, mais cela a pour effet de faire croître le label des hypothèses ainsi justifiées. Or l'algorithme de calcul des labels est exponentiel en fonction de la taille des labels. La justification d'hypothèse pose donc rapidement un problème d'efficacité.

Après un rappel sur l'ATMS (section 1.1) nous illustrerons le problème sur un exemple motivant (section 1.2). Dans la partie 2 nous montrerons que la solution du papier d'origine (de Kleer, 1986) ne résout pas le problème (section 2.1). Nous présenterons ensuite un travail sur le raisonnement temporel qui laissait transparaitre une solution (section 2.2). Dans la partie 3 nous présenterons notre solution en commençant par la définition des environnements clos (section 3.1). Puis nous proposerons une modification de l'algorithme pour les calculer (section 3.2). Dans la dernière partie (section 4) nous monterons en quoi la modification des labels peut être utile dans le cadre du raisonnement temporel.

1.1 Assumption based Truth Maintenance System (ATMS)

L'application – aussi appelé résolveur de problème (problem solver) – utilisant un ATMS lui transmet incrémentalement les étapes de son raisonnement et en retour l'ATMS calcule les états cohérents du système.

L'application déclare à l'ATMS des *faits*, des *hypothèses* (qui sont des faits distingués) et les *inférences* qu'elle effectue.

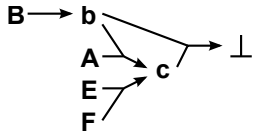
L'ATMS opère en logique des propositions (l'application peut être dans une logique différente), les faits sont des propositions, donc des termes constants (sans variables).

On rappelle que :

- Un ensemble d'hypothèses s'appelle un *environnement*.
- L'ensemble des environnements cohérent dont dépend un fait est son *label*.
- Les environnement minimaux incohérents sont appelés les *nogoods*.

Par exemple (les hypothèses sont notées avec une majuscule, et les faits "simples" avec une minuscule) :

$$\begin{array}{ccc} A \wedge b & \rightarrow & c \\ B & \rightarrow & b \\ E \wedge F & \rightarrow & c \\ b \wedge c & \rightarrow & \bot \end{array}$$



L'ATMS établira que :

- le label de b est $\{B\}$
- le label de c est $\{\{E, F\}\}$
- $\{A, B\}$ et $\{B, E, F\}$ sont incohérent.

Les ensembles de fait représentent des conjonctions, par exemple $\{A, B\}$ signifie $A \wedge B$.

Les inférences transmises à l'ATMS s'appellent des *justifications* et peuvent être de trois formes :

1. une clause de Horn : *liste de faits* $\rightarrow$ *fait*, exprime l'implication habituelle ;
2. une assertion : $\rightarrow$ *fait*, pour désigner un fait toujours vrai ;

3. une contradiction : *liste de faits* $\rightarrow\bot$, tous les faits de la liste ne peuvent être vrais simultanément.

Le rôle de l'ATMS est de calculer le label de chaque fait. Il garantit pour les labels les quatre propriétés qui suivent. Soit f un fait (hypothèse ou non), J l'ensemble des justifications, H l'ensemble des hypothèses. Un label est :

1. Fondé : le fait peut être déduit de chaque environnement du label à l'aide de l'ensemble des justifications. $\forall E \in Lab(f), J \vdash E \rightarrow f$
2. Cohérent : chacun de ses environnements est cohérent (n'est pas un sur-ensemble d'un nogood). $\forall E \in Lab(f), \neg(E, J \vdash \bot)$
3. Complet : tous les environnements susceptibles de produire ce fait sont sur-ensemble d'un environnement du label. $\forall E \subseteq H, \text{ tel que } J \vdash E \rightarrow f, \exists E' \in Lab(f), E' \subseteq E$
4. Minimal : aucun de ses environnements n'est sur-ensemble d'un autre. $\forall E \in Lab(f), \neg(\exists E' \in Lab(f), E' \subset E)$

1.2 Enoncé du problème

La justification d'hypothèse est possible. La cohérence globale est maintenue, et les labels calculés conservent les propriétés exposées. Toutefois, un problème de performance réside. Soit une cascade d'hypothèses :
$H_1 \rightarrow H_2 \rightarrow ... \rightarrow H_n$.

Les labels sont :
$Lab(H_1) = \{\{H_1\}\}$
$Lab(H_2) = \{\{H_1\}, \{H_2\}\}$
...
$Lab(H_n) = \{\{H_1\}, \{H_2\}...\{H_n\}\}$

Chaque nouvelle justification à gauche de la cascade, rajoute un environnement dans les labels des hypothèses de droite. La cardinalité du label d'une hypothèse croit linéairement en fonction du nombre d'hypothèses qui la justifient (y compris transitivement).

Or l'ATMS est exponentiel en fonction de la taille des labels.

2 Etat de l'art

2.1 Assumed node

Dans le papier de référence, De Kleer déconseille de justifier directement des hypothèses et propose les "assumed nodes". Cette technique permet de dissocier la donnée supposée, de l'action de supposer. Un couple (hypothèse, assumed-node) $A \rightarrow a$ représente une donnée rétractable. La donnée portée par a peut être retiré en contredisant A, puis rajoutée à nouveau par une hypothèse A' justifiant a. Cela permet d'éviter de faire se justifier les hypothèses. Sur notre exemple on aurait :
$H_1 \rightarrow h_1$, $H_2 \rightarrow h_2$, ..., $H_n \rightarrow h_n$,
$h_1 \rightarrow h_2 \rightarrow ... \rightarrow h_n$.

Les labels sont :
$Lab(h_1) = \{\{H_1\}\}$
$Lab(h_2) = \{\{H_1\}, \{H_2\}\}$
...
$Lab(h_n) = \{\{H_1\}, \{H_2\}...\{H_n\}\}$

Le problème combinatoire n'est pas résolu puisque les labels sont les mêmes.

2.2 HEART

How to Enhance Assumption based Reasoning with Time (HEART) (Joubel & Raiman, 1990) est une généralisation de l'ATMS pour des données temporelles. Là où l'ATMS classique manipule des faits, HEART manipule des épisodes. Un épisode est un couple (dat, int) qui représente un intervalle de temps Int durant lequel la donnée dat est vraie.

Un environnement est un ensemble d'épisodes. HEART adapte la notion de minimalité en définissant une relation d'ordre permettant de comparer de tels environnements. Les environnements non-minimaux sont redondants et éliminés.

HEART maintient des environnements dont les épisodes sont maximaux en fusionnant les épisodes, portant sur la même donnée, qui se recouvrent, se superposent ou se touchent.

Les nogoods sont des environnements pour l'épisode faux (noté $\perp$), leurs épisodes sont donc maximaux.

Comme Joubel & Raiman (1990) n'évoquent pas la justification directe d'hypothèses, nous utiliserons des assumed nodes. Soit $H_1 = (d, [0, 3[)$ et $H_2 = (d, [1, 2[)$ deux hypothèses, h_1 et h_2 deux épisodes quelconque, avec $H_1 \rightarrow h_1$, $H_2 \rightarrow h_2$, et $h_1 \rightarrow h_2$. En utilisant un ATMS classique on aurait :
$Lab(h_1) = \{\{H_1\}\}$
$Lab(h_2) = \{\{H_1\}, \{H_2\}\}$

Mais dans HEART H_1 porte sur la même donnée que H_2 mais sur un épisode plus grand ($[1, 2[\subseteq [0, 3[$) ; $\{H_1\}$ est donc redondant car non-minimal. HEART simplifie le label de h_2 pour ne conserver que les environnements minimaux :
$Lab(h_2) = \{\{H_2\}\}$

Cette manière de faire résout le problème combinatoire puisque les labels des hypothèses justifiées n'augmentent plus (même si les tests de comparaison d'environnements sont plus coûteux que de simples tests d'inclusion d'ensembles).

Cependant, la notion de redondance qui permet de simplifier les labels, ne s'applique pas forcément à d'autres applications que le raisonnement temporel. De plus les nogoods calculés ne sont pas minimaux par rapport à ceux calculés par l'ATMS classique puisque HEART fusionne les épisodes des environnements de $\perp$ afin de les rendre maximaux. C'est pourquoi nous avons cherché une méthode plus générale pour résoudre notre problème.

3 Nouvelle définition des labels

3.1 Environnements clos pour l'implication

Prenons l'exemple minimal : $H_1 \rightarrow H_2$. Peut-on parler de H_1 indépendamment de H_2 ? L'idée de base est que H_2 est nécessairement vraie si H_1 est vrai. En effet, la seule assignation pour $\{H_1, H_2\}$ où H_1 est vraie est : $\{H_1 \leftarrow true, H_2 \leftarrow true\}$

Nous changeons la définition d'un label pour que chacun de ses environnements contiennent toutes les hypothèses nécessairement vraies en même temps que lui. C'est la clôture d'un environnement pour l'implication : toute hypothèse impliquée par un environnement appartient à cet environnement. Nous parlerons d'environnements clos pour désigner ces nouveaux environnements. Soit H l'ensemble des hypothèses, J l'ensemble des justifications et f un fait :

Définition 1
$\forall E \in Lab(f), E$ clos si et seulement si $: \forall h \in H$ tel que $E, J \vdash h$ alors $h \in E$

On dira qu'un label est clos si et seulement si tous ses environnements le sont.

Définition 2
On dira que $\widetilde{E}$ est la clôture minimale de E si et seulement si $\widetilde{E}$ est clos et $E \subseteq \widetilde{E}$ et $\neg\exists E'$ clos tel que $E \subseteq E' \subset \widetilde{E}$.

La notation $\widetilde{}$ sera utilisée dans le reste de l'article pour désigner la clôture minimale d'un environnement.

Sur l'exemple minimal on avait :
$Lab(H_1) = \{\{H_1\}\}$
$Lab(H_2) = \{\{H_1\}, \{H_2\}\}$

Avec la nouvelle définition $\{H_1\}$ n'est plus un environnement clos et doit être remplacé par $\{H_1, H_2\}$:
$Lab(H_1) = \{\{H_1, H_2\}\}$
$Lab(H_2) = \{\{H_1, H_2\}, \{H_2\}\}$

Or $\{H_2\} \subseteq \{H_1, H_2\}$ donc pour respecter la propriété de minimalité des labels :
$Lab(H_1) = \{\{H_1, H_2\}\}$
$Lab(H_2) = \{\{H_2\}\}$

Le label d'un fait dont les environnements sont clos est complet au sens ou il contient tous les environnements clos susceptibles de produire ce fait. La troisième propriété est donc modifiée ainsi :

Définition 3
Un label est complet *si et seulement si tous les environnements clos susceptibles de produire ce fait sont sur-ensembles d'un environnement du label ;* $\forall \widetilde{E} \subseteq H$ tel que $J \vdash \widetilde{E} \rightarrow f, \exists E' \in Lab(f), E' \subseteq \widetilde{E}$.

C'est pourquoi dans l'exemple, $\{H_1\}$ est fondé, cohérent et minimal (il n'est pas un sur-ensemble d'un environnement du label) mais il n'est pas clos, $\{H_1, H_2\}$ est fondé,

cohérent et clos mais il n'est pas minimal. Le seul environnement fondé, cohérent, clos et minimal est $\{H_2\}$.

Cette nouvelle définition des labels respecte les quatre propriétés classiques (dont une est légèrement modifiée) auxquelles est ajoutée la propriété de clôture.

Définition 4
Un label est fondé, cohérent, clos, complet (au sens de la définition 3) et minimal.

Deux remarques :
- les labels des hypothèses contiennent au plus un seul environnement,
- les environnements des labels sont toujours les implicants du fait, par contre ils ne sont plus premiers (puisque les implicants premiers par définition sont minimaux).

En appliquant notre nouvelle définition à l'exemple du 1.2 les labels sont les suivants :
$Lab(H_1) = \{\{H_1, H_2, ...H_n\}\}$
$Lab(H_2) = \{\{H_2, ...H_n\}\}$
...
$Lab(H_n) = \{\{H_n\}\}$

Autrement dit lorsqu'une hypothèse en justifie une autre, au lieu d'augmenter la taille des labels on augmentera la taille de l'environnement qu'elle contient. L'ATMS étant d'une complexité exponentielle en fonction de la taille des labels et seulement logarithmique en fonction de la taille des environnements, on comprend bien l'intérêt d'une telle modification.

3.2 Mise en œuvre

Les exemples que nous avons vus jusqu'à présent font justifier une hypothèse que par une seule autre hypothèse. Ce cas simplifié laisse penser que le calcul de la clôture d'un label est immédiat, et ne nécessite qu'une petite modification de l'algorithme. Dans le cas général les hypothèses peuvent aussi être justifiées par des conjonctions. Et alors les environnements non clos peuvent apparaître bien après la prise en compte de la justification qui les rend non clos. Un des piliers de l'ATMS est son fonctionnement incrémental : on doit pouvoir prendre en compte une nouvelle justification sans avoir à analyser l'intégralité des justifications déjà présentes. Nous proposons d'utiliser une table des environnements non-clos qui n'apparaissent pas forcément dès la prise en compte de la justification d'une hypothèse. Prenons l'exemple suivant (figure 1), les H_i sont des hypothèses, les autres lettres des faits :

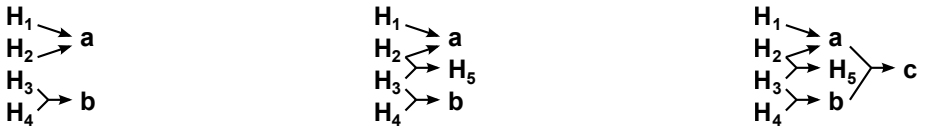


FIGURE 1 – Hypothèse justifiée par une conjonction

On a pour le 1(a) :
$Lab(a) = \{\{H_1\}, \{H_2\}\}$
$Lab(b) = \{\{H_3, H_4\}\}$

Puis (figure 1(b)) rajoutons-y la justification $H_2 \wedge H_3 \rightarrow H_5$, aucun label n'a besoin d'être modifié pour l'instant puisque l'environnement $\{H_2, H_3\}$ n'apparaît nulle part. Par contre il convient de noter que cet environnement n'est pas clos et que lorsqu'il apparaîtra il faudra le remplacer par $\{H_2, H_3, H_5\}$.

Enfin (figure 1(c)) rajoutons maintenant la justification $a \wedge b \rightarrow c$ le label de c devrait être $Lab(c) = \{\{H_1, H_3, H_4\}, \{H_2, H_3, H_4\}\}$, mais comme $\{H_2, H_3\}$ n'est pas clos on a $Lab(c) = \{\{H_1, H_3, H_4\}, \{H_2, H_3, H_4, H_5\}\}$.

3.2.1 Définition et calcul de la table de clôture

Pour pouvoir calculer la clôture de chaque environnement il faut noter les environnement non clos trouvés au fur et à mesure. Nous allons construire une table de clôture C composés de couples d'environnement $(E, \widetilde{E})$ où $\widetilde{E}$ est la clôture minimale de E. C sera complète si elle nous permet de calculer la clôture de tout environnement, et cohérente si elle ne contient aucun environnements incohérent.

Afin de stocker un minimum d'information, C ne contiendra que les éléments minimaux. Intuitivement un élément de C est plus petit qu'un autre s'il est inclu dans ce dernier et permet de calculer les mêmes clôtures. Par exemple : $(\{a\}, \{a, b\}) < (\{a, c\}, \{a, b, c\})$ car $\{a\} \subset \{a, c\}$ et $\{a, b\} \subset \{a, b, c\}$ et $\{a, b\} \cup \{a, c\} = \{a, b, c\}$. Soit la relation d'ordre suivante :

Définition 5
$<$ est une relation d'ordre partielle stricte sur les éléments de C définie par : $(E, \widetilde{E}) < (F, \widetilde{F})$ si et seulement si $E \subset F$ et $\widetilde{E} \cup F = \widetilde{F}$.

$<$ hérite de l'antisymétrie, et de l'anti réflexivité de $\subset$. Pour la transitivité si $(E, \widetilde{E}) < (F, \widetilde{F})$ et $(F, \widetilde{F}) < (G, \widetilde{G})$ alors $\widetilde{E} \cup F = \widetilde{F}$, $\widetilde{F} \cup G = \widetilde{G}$ d'où $\widetilde{E} \cup F \cup G = \widetilde{G}$ or $F \subset G$ donc $\widetilde{E} \cup G = \widetilde{G}$ d'où $(E, \widetilde{E}) < (G, \widetilde{G})$.

Définition 6
Soit C une table de clôture :

1. *C est fondée si : $\forall (E, \widetilde{E}) \in C, \widetilde{E}$ est la clôture minimale de E (définition 2) ;*
2. *C est cohérente si : $\forall (E, \widetilde{E}) \in C, \neg(\widetilde{E} \vdash \bot)$;*
3. *C est complète si :*
   $\forall E \subseteq H, \left[E \text{ est clos } \Leftrightarrow \left(\forall (F, \widetilde{F}) \in C, F \subseteq E \Rightarrow \widetilde{F} \subseteq E\right)\right]$;
4. *C est minimale si : $\forall c \in C, \neg(\exists c' \in C, c' < c)$.*

Il faut garantir ces quartre propriétés. Pour cela, lorsqu'une justification conclut sur une hypothèse, $f_1 \wedge ... \wedge f_n \rightarrow H$ avec $Lab(H) = \{E\}$ (pour le cas où $Lab(H) = \emptyset$ voir section 3.2.3), il faut mettre à jour la table de clôture, afin de s'assurer qu'elle respecte toujours les quatre propriétés. Soit L l'ensemble des combinaisons des labels des f_i, et soit $C' = \{(F, E \cup F) | F \in L\}$ une table intermédiaire. Il faut :

- s'assurer de la propriété de fondement (c'est à dire vérifier que les éléments de C sont bien des couples de clôture) : $\forall (E, E') \in C$ il faut calculer $\widetilde{E}$ qui vérifie $E' \subseteq \widetilde{E}$ et $\forall (F, F') \in C', F \subseteq \widetilde{E} \Rightarrow F' \subseteq \widetilde{E}$. Si $E' \neq \widetilde{E}$ alors (E, E') est retiré de C et $(E, \widetilde{E})$ y est ajouté.
- verifier la complétude : $\forall (F, F') \in C', \exists (E, \widetilde{E}) \in C, E \subseteq F$, si ce n'est pas le cas alors (F, F') est ajouté dans C,
- vérifier la cohérence des environnements ajoutés, c'est-à-dire retirer les $(E, \widetilde{E})$ de C dont $\widetilde{E}$ est sur-ensemble d'un nogood,
- et retirer de C les éléments non minimaux : $\forall c \in C$ si $\exists c' \in C$ tel que $c' < c$ alors c est retiré de C.

La table de clôture est ainsi obtenue incrémentalement. Elle ne dépend pas de l'ordre d'application des justifications.

3.2.2 Calcul de la clôture d'un environnement

Avec C nous sommes maintenant en mesure de calculer la clôture de chaque environnement. Nous définissons un opérateur de clôture noté $clore$ (algorithme 1) qui calcule la clôture d'un environnement E pour une table de clôture C.

Algorithme 1: $clore(E, C)$ — E un environnement, C une table de clôture

si $\exists F$ *tel que* $(F, \widetilde{F}) \in C$ *et* $F \subseteq E$ **alors retourner** $clore(E \cup \widetilde{F}, C - \{(F, \widetilde{F})\})$
sinon retourner E

Cet opération est au pire cas de $n(n+1)/2$ où n est le cardinal de C. Dans la pratique ce pire cas n'est atteint que sur des instances pathologiques telle que $H_1 \wedge H_2 \rightarrow H_3, H_3 \wedge H_4 \rightarrow H_5, ...H_{n-2} \wedge H_{n-1} \rightarrow H_n$ pour le label de $\{H_1, H_2, H_4, H_6, ..., H_{n-1}\}$.

Cet opérateur est appelé lors du calcul du label des faits "simples" afin de calculer la clôture de chaque environnement de chaque label. Avant de rajouter un environnement E dans un label il faut d'abord le clore en calculant $clore(E, C)$. Si l'environnement trouvé n'est pas sur-ensemble d'un environnement du label, alors on l'y rajoute.

Lorsque C est modifiée (lorsque des éléments y sont rajoutés) il faut vérifier que les environnements des labels sont bien clos en appliquant l'opérateur $clore/2$ avec les éléments nouvellement ajoutés à C pour le deuxième argument.

La complexité rajoutée au calcul d'un fait "simple" est en $O(n^2)$ en fonction de la taille de C dans le pire cas. La taille de C est au maximum le nombre de justifications concluant sur une hypothèse (y compris transitivement).

3.2.3 Calcul des nogoods

La base des nogoods doit toujours contenir les environnements minimaux, par exemple pour utiliser les nogoods pour un calcul de candidats. Or les labels contiennent désormais des environnements clos. C'est pourquoi nous devons retrouver les environnements minimaux qui engendrent ces environnements clos avant de les rajouter aux

nogoods. Pour cela nous allons utiliser la table de clôture "à l'envers" et retirer toutes les hypothèses qui avaient été ajoutées par l'opération de clôture.

Lorsque l'algorithme classique trouve un nogood $\widetilde{N}$ on calculera :

$$N = \widetilde{N} - \bigcup_{(E,\widetilde{E})\in C, \widetilde{E}\subseteq\widetilde{N}} (\widetilde{E} - E) \quad (1)$$

C'est N qui sera ajouté aux nogoods. De plus pour tous les $(E, \widetilde{E}) \in C$ si $N \subseteq \widetilde{E}$ alors le couple est retiré de C et E est ajouté aux nogoods.

3.3 Expérimentation

Nous avons repris l'exemple initial (section 1.2) en construisant une cascade d'hypothèse. Les justifications sont transmises dans l'ordre : $H_1 \rightarrow H_2$, $H_2 \rightarrow H_3$, ..., $H_{n-1} \rightarrow H_n$. Après implémentation, nous obtenons les résultats suivants, la première ligne est le nombre d'hypothèse de la cascade, les deux suivantes donnent les temps d'exécution en ms pour déclarer les hypothèses, transmettre les justifications, et faire calculer à l'ATMS les labels de chaque hypothèse :

n	200	400	600	800	1000	1200	1400
ATMS classique	461	3575	12549	30945	62639	111732	183616
ATMS "clos"	240	390	881	1563	2663	3756	5066

On voit bien l'explosion combinatoire dans le cas classique, tandis que notre proposition évolue bien plus raisonnablement.

4 Application : les variables estampillées

Dans le cadre d'un travail sur un modèle de plan pour des agents cognitifs, nous avons appliqué cette technique à un type de donnée à évolution discrète dans le temps appelé *variable estampillée*. Le changement de valeur est instantané et ponctuel. L'information est potentiellement incomplète, parvient ponctuellement et éventuellement en retard. Et nous autorisons d'y accéder à tout instant.

Il y a de nombreuses sources d'informations qu'on ne peut pas connaître en temps réel : celles qui doivent transiter par des intermédiaires, qui arrivent en différé ou à des fréquences variables. L'exploitation de ces sources est difficile car une information sur leur changement peut nous parvenir a posteriori (disons à t_4) concernent le passé (à t_2) entre l'instant t_1 de la dernière valeur connue et l'instant t_3 où l'on a utilisé cette donnée ($t_1 < t_2 < t_3 < t_4$), voir la figure 2

Nous utilisons des hypothèses de persistance signifiant l'absence de changement de valeur. Sur des intervalles de temps passé elles permettent de se prémunir contre l'arrivée d'information tardive. Dans l'exemple l'utilisation de l'information à t_3 se fait sous l'hypothèse que la donnée reste constante entre t_1 et t_3. L'arrivée tardive d'une nouvelle valeur concernant t_2 engendrera une contradiction entre les deux valeurs et l'hypothèse de persistance. Si les informations ne nous parviennent pas de la même source alors soit

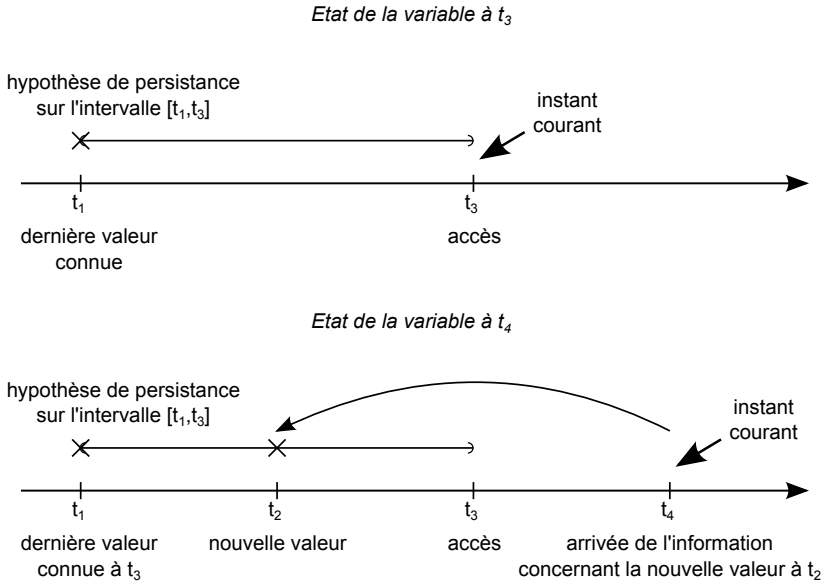


FIGURE 2 – Accès à une variable estampillée

l'une des deux est fausse, soit elles sont toutes deux justes et alors la valeur a effectivement changé. Dans ce dernier cas l'hypothèse de persistance deviendra un nogood, et tout ce qui avait était déduit à partir d'elle sera injustifié (l'agent cessera d'y croire).

Lors d'accès successifs à une variable estampillée on retombe dans le cas d'hypothèses en cascade. Prenons par exemple la figure 3. Soit l'attribut défini par (a, v, t_0), et trois accès à t_1, t_2 et t_3. Il y aura trois hypothèses de persistance $pers(a, [t_0, t_1])$ notée p_1, $pers(a, [t_0, t_2])$ notée p_2 et $pers(a, [t_0, t_3])$ notée p_3. Il y aura les justifications $p_3 \rightarrow p_2, p_3 \rightarrow p_1$ et $p_2 \rightarrow p_1$. Avoir trois hypothèses de persistance différentes permet de d'obtenir des environnements différents pour chacun des accès et tout ce qui en sera déduit. L'utilisation d'environnements clos permet de maîtriser la combinatoire engendrée.

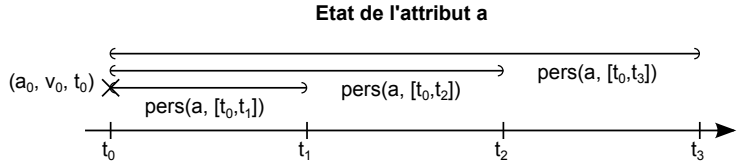


FIGURE 3 – Accès successifs à une variable estampillée

L'utilisation de HEART sur cet exemple ne permet pas d'utiliser les nogoods pour un calcul de candidats puisque les épisodes des nogoods sont fusionnés afin d'être

maximaux. Par ailleurs, notre approche s'applique à des raisonnements symboliques en dehors du raisonnement temporel.

5 Conclusion

Dans cet article nous avons proposé une modification de la définition des labels de l'ATMS. Dans cette nouvelle définition les environnements des labels sont clos pour l'implication, c'est à dire qu'ils contiennent toutes les hypothèses qu'ils impliquent. Cette modification permet de maîtriser la croissance des labels des hypothèses si des inférences concluent à des hypothèses. Leurs labels ne contiennent plus qu'un unique environnement, alors qu'il augmentait linéairement en fonction du nombre d'hypothèses qui la justifiait avec l'ATMS classique – qui est exponentiel en fonction de la taille des labels. Le calcul de la clôture des environnements est polynomial (dans le pire cas) en fonction du nombre de justifications concluant sur une hypothèse (y compris transitivement). Le gain en complexité, pour les instances de problème ayant besoin de justifier les hypothèses entre-elles, est considérable.

Nous avons appliqué l'utilisation de cette modification sur une structure de donnée appliquée au raisonnement temporel.

6 Remerciement

Nous tenons à remercier Philippe Dague pour son avis et ses remarques pendant l'élaboration de l'idée qui a conduit à cet article.

Références

DE KLEER J. (1986). An assumption-based TMS. *Artificial intelligence*, **28**(2), 127–162.

DE KLEER J. (1992). An improved incremental algorithm for generating prime implicates. In *Proceedings of the National Conference on Artificial Intelligence*, p. 780–780 : JOHN WILEY & SONS LTD.

DE KLEER J. & REITER R. (1987). Foundations for assumption-based truth maintenance systems : Preliminary report. In *Proc. American Assoc. for Artificial Intelligence Nat. Conf*, p. 183–188.

DOYLE J. (1979). A truth maintenance system. *Artificial Intelligence*, **12**(3), 231–272.

JOUBEL C. & RAIMAN O. (1990). How time changes assumptions. In *Proc. ECAI-90 : Ninth European Conference on Artificial Intelligence*, p. 378–383.