SaMaRis: visualiser et manipuler interactivement le raisonnement

Jérôme Euzenat
Laboratoire ARTEMIS/IMAG, GRENOBLE

Les systèmes de maintien du raisonnement (SMR[1]) sont destinés à être couplés à un moteur d'inférence pour assurer la cohérence d'un raisonnement au cours de son déroulement. En général, ils considèrent le raisonnement produit sous la forme d'un graphe dans lequel ils propagent les conditions de validité des formules que représentent les nœuds. Si le comportement d'un tel système est aisé à spécifier, son fonctionnement se révèle ardu à expliquer. En particulier, le comportement non monotone de certains systèmes, ainsi que la présence possible de cycles, obligent à des traitements particuliers.

SaMaRis est une interface graphique dédiée aux SMR et fondée sur le principe de manipulation interactive des divers graphes qu'ils exploitent. Elle s'appuie sur une interface de commande générique des SMR mise au point lors de travaux antérieurs avec les systèmes à base de règles IROISE, TOKEN et d'objets SHIRKA et KOOL. Ceci permet d'utiliser la même interface (légèrement modulée en fonction du SMR) pour divers systèmes (TMS, ATMS, CP-TMS).

---

[1] Sont regroupés sous ce vocable les systèmes de maintenance de la vérité (truth maintenance systems): TMS, ATMS...

L'intérêt de SaMaRis est triple:

- Pédagogique: il permet d'enseigner les principes des SMR par l'exemple. Le travail de ces systèmes étant en effet très souvent souterrain, il est difficile de mettre en évidence leur action.
- Explicatif: l'interface peut se décomposer en modules incorporables dans une application. Elle permet alors de proposer les graphes de dépendance du SMR comme des arbres de preuve. Les manipulations des graphes, faites à l'aide d'une souris, seront répercutées sur le raisonnement.
- Démonstratif: il permet au chercheur de tester le comportement de son système en lui soumettant des tests de manière naturelle (en utilisant le symbolisme dans lequel il pense), mais aussi de démontrer à ses collègues les propriétés du SMR qu'il a construit.

SaMaRis a été conçu de manière à représenter naturellement un raisonnement qui peut être complexe, et à être suffisamment instinctif et robuste pour pouvoir être manipulé par des novices.

SaMaRis ne sera présenté qu'au travers d'une instanciation du système: SaMaRis/TMS utilisant un système de maintien du raisonnement de type TMS. D'autre part, l'accent sera mis délibérément sur l'utilisation de SaMaRis et permettra de montrer toute sa puissance dans la manipulation du SMR. C'est donc le point de vue de l'utilisateur et la réactivité du système qui priment. Le point de vue du chercheur développant son propre SMR ou du développeur interfaçant SMR, SaMaRis et application ne sera donc que peu évoqué ici et, par conséquent, l'aspect générique de SaMaRis sera peu détaillé.

Les systèmes de maintenance de la vérité sont brièvement exposés dans la section suivante avant d'illustrer la manipulation de tels systèmes au travers de SaMaRis. Le paramétrage de SaMaRis est ensuite évoqué avant d'aborder les aspects architecturaux du système.

1.Systèmes de maintien du raisonnement

Il existe deux principaux types de *systèmes de maintien du raisonnement.* Les premiers utilisent des inférences non monotones pour poser automatiquement des hypothèses (par exemple: en l'absence de l'information «Claude est une femme», poser que «Claude est un homme») et se chargent d'établir la validité absolue des items («Claude doit donc avoir fait son service militaire»). Les seconds ont connaissance des hypothèses posées par l'utilisateur et établissent, pour chaque item inféré, sous quelles hypothèses il doit être considéré comme présent dans la base (par exemple: sous l'hypothèse que «Claude est un homme», «Claude doit avoir fait son service militaire»).

Ceci permet de passer facilement d'un ensemble d'hypothèses à un autre: c'est le raisonnement multi-contextes. Les deux types de systèmes sont capables de supprimer les hypothèses si elles mènent à une contradiction (par exemple: «Claude est un homme» et «Claude porte une jupe au bureau»). Le premier fera alors en sorte que les hypothèses posées ne soient plus considérées (ainsi, il va considérer que «Claude est une femme» et donc supprimer de la base que «Claude doit avoir fait son service militaire»). Le second ne considérera plus les hypothèses incriminées («Claude est un homme» soutenant la contradiction sera alors évincé et, par conséquent, «Claude doit avoir fait son service militaire» ne sera plus présent sous aucune hypothèse). Le premier correspond au système de maintenance de la vérité à propagation qui, dans la suite, va permettre d'illustrer l'utilisation de SaMaRis. Le second correspond au système de maintenance de la vérité à contextes dont certaines caractéristiques seront évoquées plus loin (voir §4.1).

Le principe du *système de maintenance de la vérité à propagation*, ou TMS pour "truth maintenance system" [2], est d'enregistrer, pour chaque inférence, la formule inférée au sein d'un *nœud* et de lui associer une *justification* représentant l'inférence. Une justification est un couple d'ensembles de nœuds, le premier ensemble étant l'ensemble des nœuds dont la présence dans la base est nécessaire à l'inférence (IN-liste) tandis que le second ensemble est celui des nœuds dont l'absence dans la base est nécessaire à l'inférence (OUT-liste). Les justifications ont donc la structure suivante: $<\{Fait_1,...Fait_n\}\ \{NFait_1,...NFait_m\}>$. L'ensemble des nœuds considérés par le système, associés à leurs justifications, forme un graphe orienté nommé *graphe de dépendances* (voir figures 2 et 3).

Une justification est dite *valide* si tous les nœuds de sa IN-liste sont IN et tous ceux de sa OUT-liste sont OUT; un nœud, à son tour, est *IN* s'il a au moins une justification valide, dans le cas contraire, il est *OUT*. L'apparente circularité de ces deux définitions est stoppée par les nœuds sans justification qui sont forcément OUT et les justifications ayant une IN et OUT-liste vide qui sont forcément valides. Un étiquetage de chaque nœud du graphe comme IN ou OUT respectant ces contraintes est un étiquetage admissible; un étiquetage qui étiquette OUT tous les nœuds inconsistants est un étiquetage consistant. Le but de l'algorithme du TMS est de trouver un *étiquetage faiblement fondé*, c'est-à-dire un étiquetage admissible et consistant tel qu'un nœud ne puisse être étiqueté IN sur la base d'un argument circulaire.

Le système de maintenance de la vérité agit sur le graphe principalement quand une justification correspondant à une inférence lui est fournie. Il se charge alors de propager la validité nouvelle introduite par cette inférence au sein du graphe. Puis, si des faits contradictoires sont présents simultanément dans la base à la suite de cette

inférence, le système invoque un mécanisme de *rétrogression*[2] chargé de supprimer cette contradiction. La rétrogression choisit d'invalider l'une des hypothèses soutenant le nœud inconsistant. Une *hypothèse* est un nœud IN dont la justification supportante possède sa OUT-liste non vide, autrement dit, un nœud dont la présence est due à l'absence d'autres nœuds dans la base. Elle sera donc invalidée en ajoutant dans la base un des nœuds de cette seconde liste.

Les systèmes de maintenance de la vérité à propagation permettent donc, à partir d'une base de données initiale et d'un ensemble d'inférences produites, éventuellement non monotones, de garantir la validité des faits dans la base vis-à-vis de ces inférences et données initiales. Ils permettent en outre, en cas de violation de ce qui représente une contrainte d'intégrité, de rétablir la cohérence de la base en la complétant.

La définition d'un système de maintien du raisonnement comme le TMS s'exprime donc en peu de mots et n'explique en rien:

- comment procède un TMS,
- ce qu'il fait dans des cas précis.

L'idée de SaMaRis est de permettre l'apprentissage du SMR par la découverte de son comportement en exploitant la représentation graphique des données manipulées. SaMaRis est exposé ici dans son instanciation aux TMS, les plus complexes parmi les systèmes de maintien du raisonnement.

2.SaMaRis et le fonctionnement du SMR

La conception de SaMaRis a été guidée par quelques idées simples:

1) Clarté: représenter de manière naturelle un raisonnement qui peut être complexe.
2) Robustesse: être suffisamment robuste pour pouvoir être manipulé par des novices.
3) Naturel: être suffisamment instinctif pour être rapidement compris.

L'interface, en général, s'inspire des principes qui ont fait leurs preuves avec le Macintosh. En particulier, la façon de manipuler les graphes est similaire à celle utilisée dans le logiciel MacDraw et largement répandue depuis.

---

[2] Aussi appelé «retour-arrière».

Le système est doté de diverses fonctionnalités regroupées en 5 zones sur l'écran. La partie principale d'un écran est un éditeur de graphe (a) qui permet d'exposer l'état des formules et leur dérivation. Une palette d'opérateurs sur le graphe (b) permet de disposer de divers outils de manipulation du graphe (déclaration d'inconsistance, d'hypothèse, d'inférence...). La barre de menus (c) permet de paramètrer, à la fois, l'interface de SaMaRis (mode verbeux, persistance des outils...) et le SMR (type de rétrogression, opérateurs de requête...). Il est possible, par des commutateurs (d), de passer à l'édition d'autres graphes manipulés par le SMR (treillis de contextes, super-graphe du graphe de dépendances...). Enfin, la zone de communication (e) permet d'afficher les attributs affectés aux nœuds ou d'interroger en Lisp le contenu de la base.

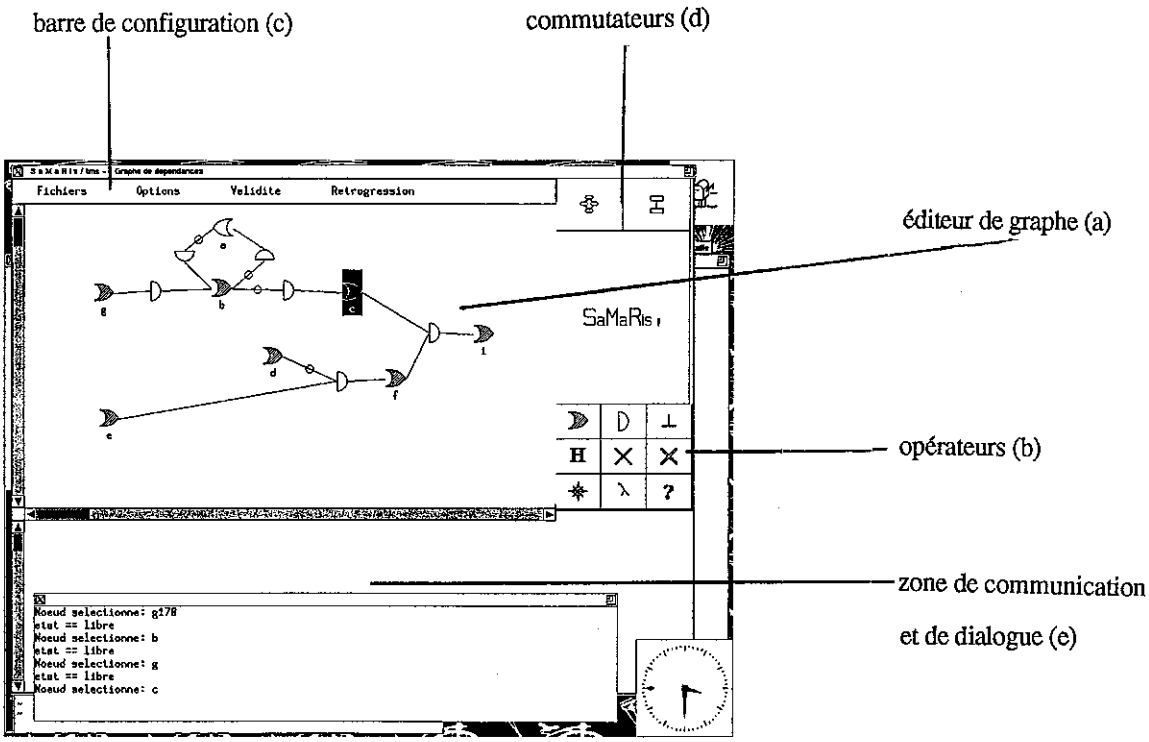

figure 1. Les 5 zones de SaMaRis

Dans cette section, l'utilisation du SMR va être exposée au travers de SaMaRis ce qui correspond à l'exploitation des zones (a) et (b). Puis les options (c) qui permettent de moduler cette utilisation seront détaillées (§3).

Le cœur de SaMaRis est la représentation graphique du raisonnement. La partie principale du système consiste donc en un plan d'édition de graphe qui représente le raisonnement à l'aide des formalismes habituellement utilisés par les concepteurs de SMR.

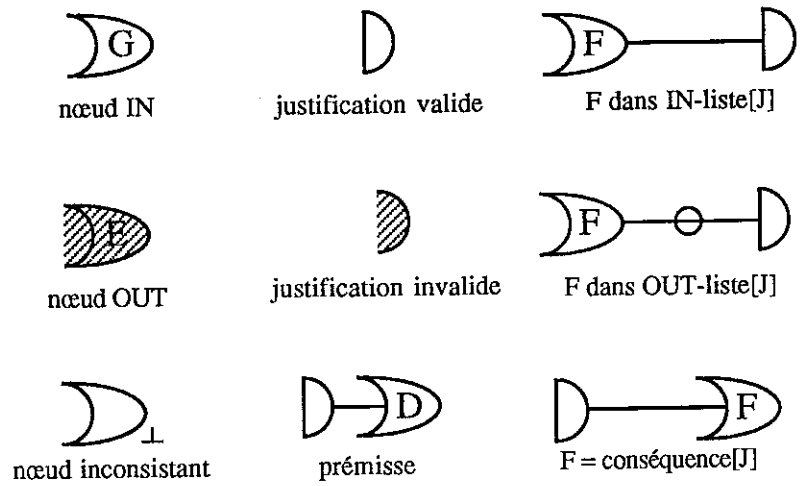

figure 2. Mosaïque des icônes utilisées par SaMaRis pour représenter les graphes de dépendances. Un graphe de dépendances est représenté comme un circuit logique dont les portes «ou» sont des nœuds et les portes «et» des justifications où les nœuds de la IN-liste arrivent directement, alors que les nœuds de la OUT-liste passent au travers d'un inverseur. Les nœuds dont une justification a ses deux listes vides représentent des formules toujours valides car elles n'ont pas besoin d'être inférées, ce sont les prémisses du raisonnement (D). Les nœuds et justifications hachurés sont considérés comme invalides et ceux en blanc comme valides.

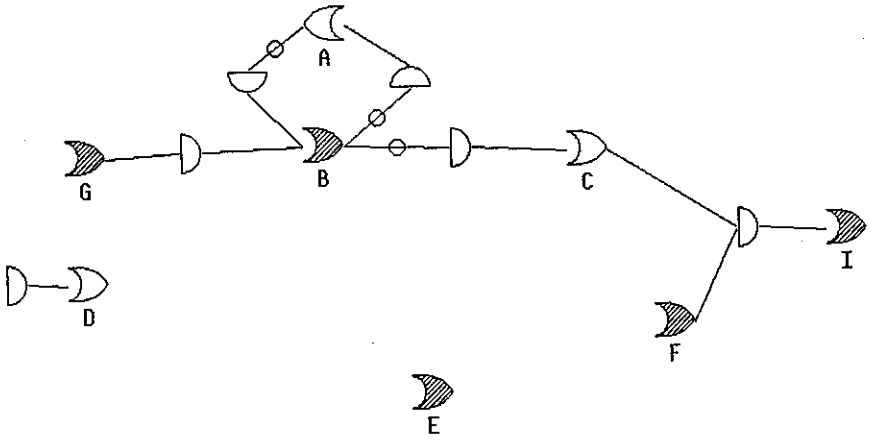

figure 3. Graphe de dépendances correspondant aux justifications <{G}{}>:B, <{}{B}>:A, <{}{A}>:B, <{}{B}>:C, <{}{}>:D, <{F,C}{}>:I. Bien sûr, l'affectation de la validité est faite selon les règles dictées par les composants du graphe. Les faits dans la base sont ainsi assurés d'avoir une justification valide — c'est-à-dire correspondant à une inférence valide.

Utiliser SaMaRis, c'est d'abord créer un graphe représentant les inférences. Ceci peut-être fait:

- Par l'utilisation d'un système d'inférence.
- Par le chargement d'un fichier contenant ce raisonnement (voir §3.1).
- Interactivement.

C'est la dernière option qui est détaillée ici. La principale façon d'interagir avec SaMaRis consiste à manipuler les nœuds du graphe. Ces manipulations consistent à créer des nœuds et des justifications mais aussi à leur appliquer un certain nombre d'opérations qui correspondent aux boutons du panneau (b).

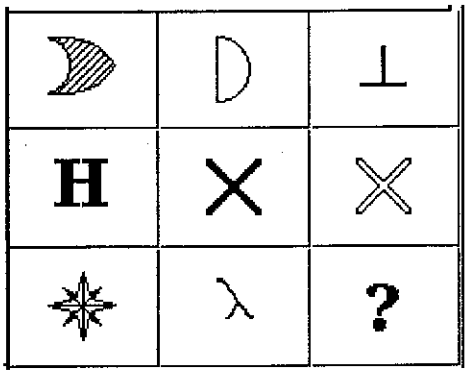

figure 4. Le panneau d'opérations sur les nœuds. La première rangée représente les opérateurs les plus importants du TMS: création de nœuds et de justifications et déclaration d'inconsistance. La seconde rangée contient les opérateurs de déclarations d'hypothèses et d'oubli. Enfin la troisième rangée contient les opérateurs d'orientation des nœuds et d'accès à leur structure.

2.1. Création de nœuds

Le placement automatique des nœuds d'un graphe quelconque de façon lisible étant impossible en un temps raisonnable (d'autant plus que le graphe du SMR n'a aucune raison d'être planaire), les nœuds sont donc placés par l'utilisateur sur la zone du graphe afin de conserver des performances acceptables.

La création d'un nœud se fait en cliquant sur l'opérateur puis en désignant à l'écran la position où va se trouver le nœud. Ceci est facilité par le pointeur de souris qui prend la forme de l'objet à placer sur le graphe. Une fois la position désignée, une boite de dialogue s'affiche permettant de saisir le nom du nœud à traiter.

Une fois cliqué sur «D'accord», la création du nœud est transmise au SMR afin qu'il entérine dans sa propre base de nœuds la présence du nouveau nœud. Le nouveau nœud du SMR est aussitôt relié au nœud graphique le représentant. La liaison est implémentée par un couplage faible de façon à ne pas contraindre l'interface générique des SMR (voir §4.1).

2.2. Création de justifications

Pour créer une justification, il est nécessaire de fournir le conséquent, la IN-liste, la OUT-liste et la position graphique de la justification. Ceci peut être fait grâce à la boite de dialogue de construction des justifications ou en pointant les différents éléments à la souris.

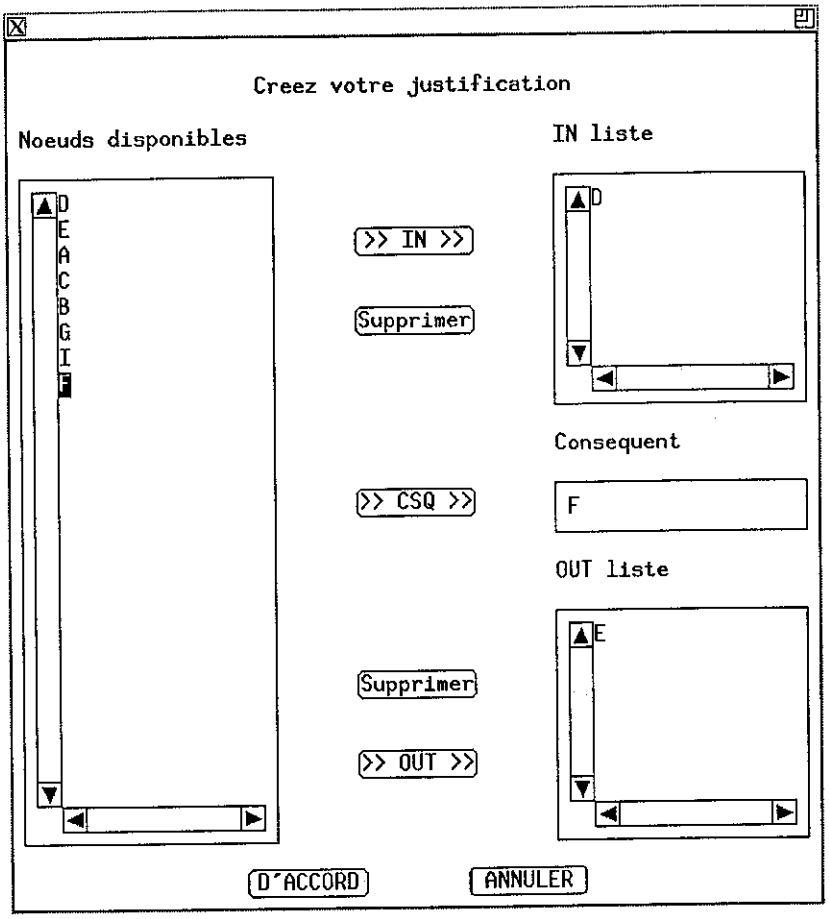

figure 5. Boite de dialogue permettant de construire une justification. La partie gauche permet la sélection de n'importe quel nœud déclaré à un moment précis. Les boutons du milieu permettent de verser les nœuds sélectionnés dans un des composants de la justification (IN-liste, OUT-liste ou conséquent) représentés dans la partie droite.

Après avoir sélectionné l'option justification (bouton), le curseur prend la forme et permet de désigner le conséquent parmi les nœuds du graphe. Il prend alors la forme d'une justification afin de placer l'image de la justification, après quoi le curseur devient ce qui permet de désigner les membres de la IN-liste. Cette phase de saisie s'achève par un nouveau clic sur le bouton de justification; le curseur devient alors et permet de sélectionner les membres de la OUT-liste. Cette dernière phase s'achève lors d'un nouveau clic sur le bouton de justification. La justification s'affiche alors à l'écran.

Cette justification affichée à l'écran est communiquée au SMR qui l'intègre à son propre réseau. Celui-ci assure la propagation de la validité au sein du graphe s'il y a lieu puis, il communique à SaMaRis l'ensemble des nœuds et justifications ayant changé d'état. SaMaRis peut alors réagir en mettant graphiquement à jour son propre graphe: l'allure des nœuds et des justifications est alors modifiée (voir figure 6).

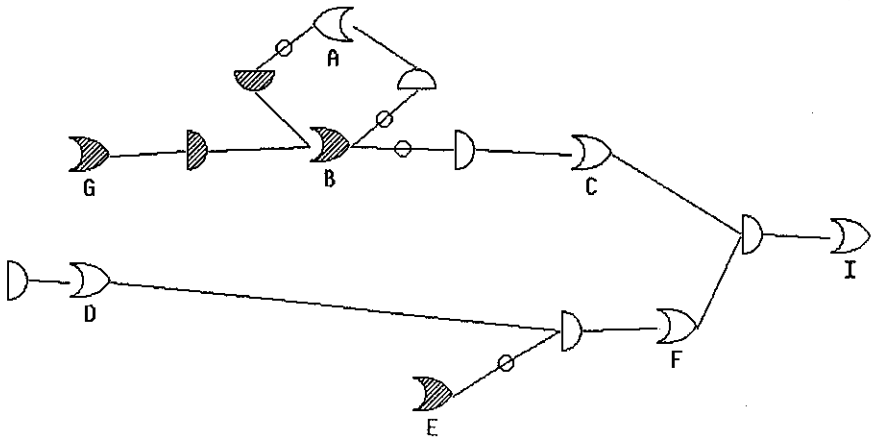

figure 6. Une propagation après l'ajout de la justification <{D}{E}>:F qui est valide car D est IN et E est OUT. Le nœud F devient alors IN ce qui rend la justification <{C, F}{}>:I valide et donc le nœud I IN à son tour. Comparez à la figure 3.

2.3.Déplacement des nœuds

Il peut être nécessaire, après leur placement, de modifier la position de nœuds et de justifications. Leur position peut être modifiée à tout moment en cliquant dessus avec le bouton de gauche et en déplaçant la souris. Il est aussi possible de modifier l'orientation des nœuds en utilisant l'opérateur qui permet de faire pivoter un nœud d'un quart de tour vers la droite.

2.4.Déclaration d'inconsistance

La déclaration d'une contrainte d'intégrité est une des opérations importantes sur un graphe de SMR. Celle-ci est représentée en SaMaRis comme la déclaration d'un nœud inconsistant. Ceci se fait à l'aide de l'opérateur $\perp$ et suit le protocole général d'utilisation des opérateurs: cliquer sur l'opérateur puis cliquer sur le nœud auquel il faut l'appliquer.

La désignation d'un nœud comme inconsistant conduit à l'ajout du symbole $\perp$ en dessous du nœud puis cette inconsistance est communiquée au SMR. Si le nœud inconsistant est IN, il faut alors procéder instantanément à une rétrogression, ce que fait le SMR. À l'issue de cette rétrogression, le SMR communique à SaMaRis les justifications ajoutées ainsi que les nœuds ayant changé d'état. Comme précédemment, SaMaRis met à jour le graphe affiché (voir figure 7).

Les possibilités de diriger la rétrogression grâce à SaMaRis sont évoquées plus loin (voir §3.3).

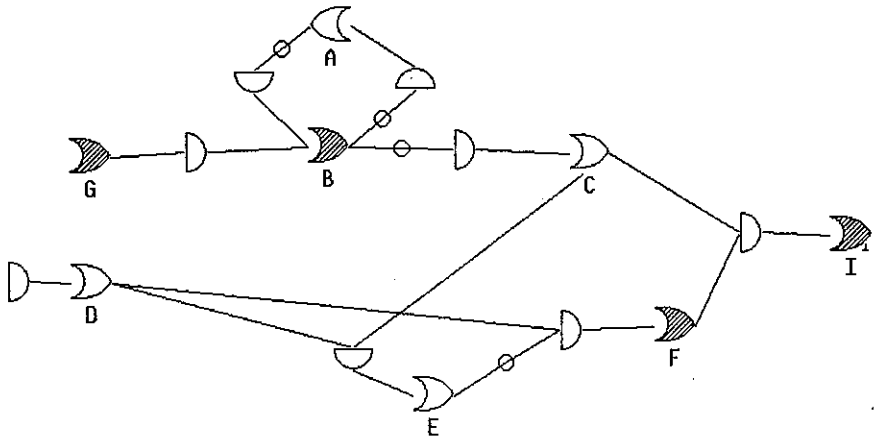

figure 7. Une rétrogression après déclaration d'inconsistance de I. I étant IN, l'étiquetage est inconsistant. Le système va alors déterminer les hypothèses soutenant I: il s'agit de {C, F}. Il y a donc deux choix possibles: invalider C ou F. Si F est choisi, il faut alors que E soit validé par une nouvelle justification. C'est cette nouvelle justification <{C,D}{}>:E qui est ajoutée à la base, rendant l'étiquetage consistant.

2.5. Oubli

Les opérateurs d'oubli permettent de supprimer (physiquement) du graphe un nœud initial (un nœud initial étant un nœud n'ayant aucun antécédent, c'est-à-dire, soit un nœud sans aucune justification, soit un nœud ayant pour unique justification une justification vide [5]). Il y a deux façons d'oublier un nœud:

- L'oublier ainsi que toutes ses conséquences, ce qui est appelé *invalidation*. Cela consiste à supprimer, avec le nœud, l'ensemble des justification dans lesquelles ce nœud apparaît dans la IN- ou OUT-liste. C'est utile quand l'état du nœud est erroné et qu'il ne doit plus être tenu compte de ce nœud.
- L'oublier en laissant tel quel l'ensemble de ses conséquences, ce qui est appelé *consolidation*. Cela consiste à supprimer le nœud ainsi que sa présence dans chacune des justifications dans lesquelles il apparaît. Cela est utile quand l'étiquette du nœud est considérée comme correcte mais que la trace du nœud ne doit plus être conservée.

L'oubli est commandé par les opérateurs  et  , le premier traitant l'invalidation et le second la consolidation. Après désignation du nœud à oublier, l'application de l'opérateur correspondant est faite au niveau du SMR. Le SMR applique donc son propre opérateur qui consiste à supprimer le nœud et à aménager l'ensemble des justifications en fonction de l'opération puis à assurer une nouvelle propagation si la validité de certains nœuds venait à être modifiée. À l'issue de cette propagation, les nœuds et justifications ayant changé d'état sont communiqués à SaMaRis qui met de nouveau à jour son graphe (voir figure 8).

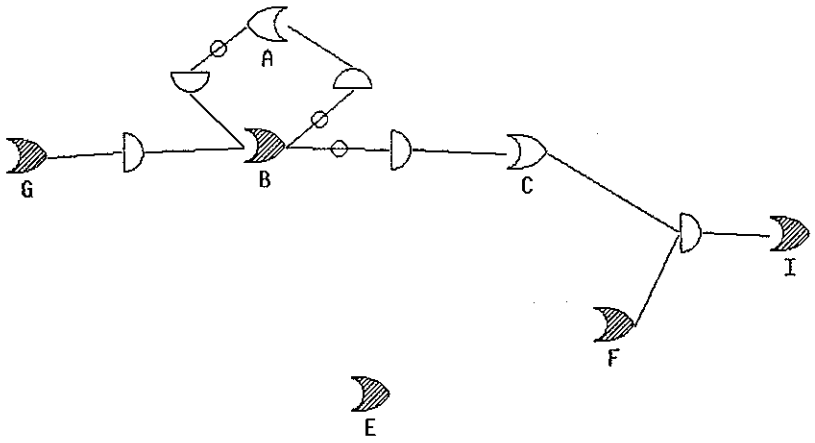

figure 8. Oubli du nœud D et invalidation. Une fois le nœud D oublié, la justification <{D}{E}>:F est supprimée, le nœud F devient OUT et la propagation fait le reste, à savoir invalider la justification <{C, F}{}>:I et le nœud I.

2.6. Interrogation des nœuds

Il est possible d'interroger la structure interne des nœuds du SMR. Cette structure est composée d'un ensemble d'attributs qui représentent des informations nécessaires au système de maintien du raisonnement. La sélection de cet opérateur permet d'afficher un menu dont chaque item correspond à un attribut de la structure interne du SMR. La visualisation de cette structure profonde s'obtient en cliquant sur l'opérateur  et en désignant l'objet à interroger à l'écran. Ceci fait apparaître un menu dont chaque item correspond à un attribut du nœud dans l'implémentation du SMR correspondant. Sélectionner un item permet d'afficher la valeur de cet attribut, soit dans une boite de dialogue, soit dans la zone de communication.

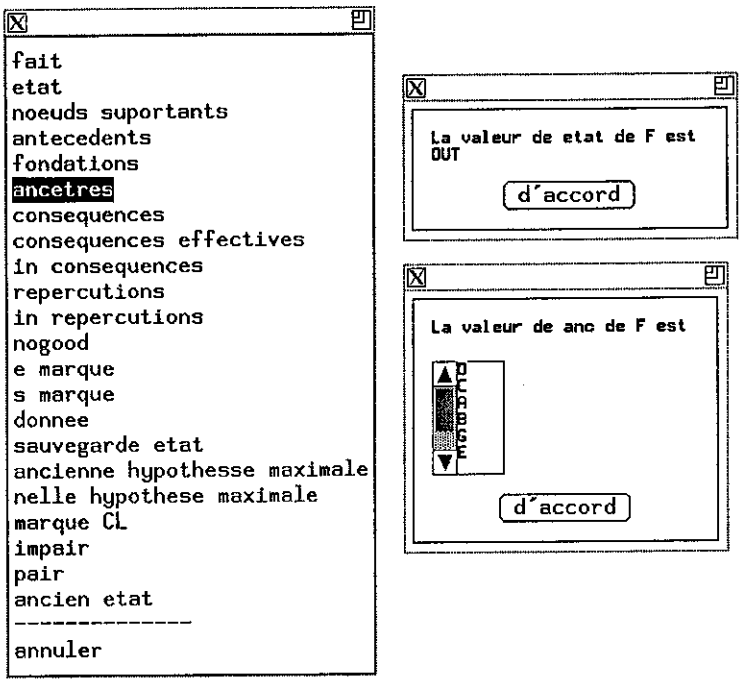

figure 9. Le menu de visualisation d'un nœud et la visualisation des attributs «état» et «ancêtres». Les 12 premiers items du menu correspondent aux attributs définis par Jon Doyle [2] qui sont stockés dans le nœud. Les autres sont des attributs booléens «de service» qui servent aux algorithmes de parcours de graphe procédant par marquage (propagation d'étiquette, décomposition en composantes fortement connexes...).

Il faut noter que ce menu est une fenêtre et peut donc être déplacé et rester disponible pendant que l'utilisateur procède à d'autres manipulations.

3.Paramétrage de SaMaRis

En dehors des manipulations sur le graphe de dépendances, SaMaRis offre diverses commodités accessibles par la barre de menus (c). Ces commodités concernent la communication avec le système de fichier, le paramétrage du protocole d'interaction et celui des SMR.

3.1.Gestion des fichiers

SaMaRis permet, non seulement de créer des graphes mais aussi de les sauvegarder et de les recharger. À cet effet, trois niveaux de sauvegarde/chargement sont disponibles. Ils sont le reflet de l'interface entre SaMaRis et SMR.

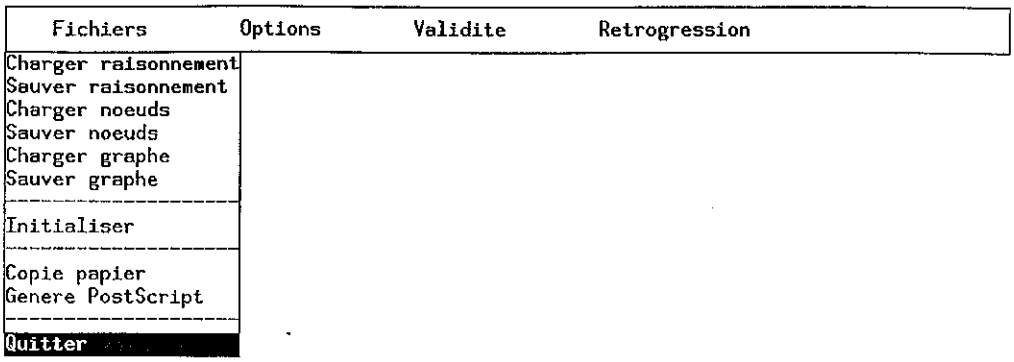

figure 10. Menu de sauvegardes/chargements. Les raisonnements peuvent être chargés et sauvegardés sous différents formats. Il est possible d'initialiser le raisonnement (supprimer tous les nœuds). Il est enfin possible d'engendrer des copies d'écran sous différents formats (bitmap ou PostScript).

Le format *raisonnement* ne sauvegarde que le raisonnement représenté par le graphe sous une forme définie par l'interface générique des SMR. Le raisonnement sauvegardé peut donc être rejoué par un SMR, indépendamment du fait qu'il soit connecté à SaMaRis ou non! Le format *nœud* est destiné à être rejoué par SaMaRis qui recrée alors tous les nœuds mais ne les place pas. Le format *graphe* enregistre non seulement les nœuds mais aussi leur position et leur orientation; il permet donc de restaurer des graphes complets placés dans l'éditeur.

Il faut noter que si le raisonnement est sauvegardé, le résultat du travail du SMR (l'étiquetage des nœuds) ne l'est pas. Ainsi, à chaque chargement, le SMR effectue de nouveau toutes les tâches de propagation.

Il est possible d'effectuer toutes sortes de copies d'écran, mais le graphe peut aussi être sauvegardé comme un fichier PostScript, ce qui permet d'inclure des graphes sous une forme compacte au sein de documents. C'est ainsi qu'ont été produits les graphes des figures 3, 6, 7 et 8.

3.2. Options de l'interface

SaMaRis doit être manipulable par des utilisateurs novices comme par des utilisateurs expérimentés. Ainsi, comme la plupart des logiciels interactifs, il autorise la modification de ses propres protocoles de manipulation. Le menu «options graphiques» permet de modifier facilement et dynamiquement le mode d'interaction avec le logiciel.

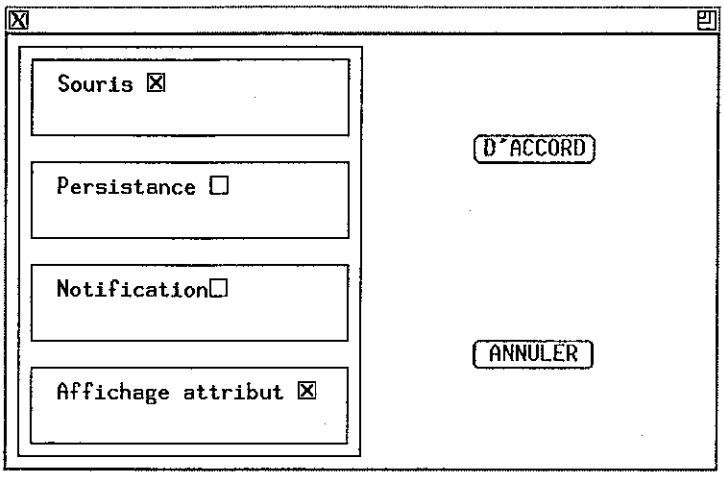

figure 11. Panneau des options graphiques. Il permet de configurer selon ses préférences le protocole de communication entre l'utilisateur et l'interface.

3.2.1. Souris

La manipulation des données à l'aide de la souris n'est pas acceptée par tout le monde. Aussi, SaMaRis offre-t-il la possibilité d'éviter l'utilisation systématique de la souris et de la remplacer par un pointage par nommage. Ainsi, l'activation de chacun des outils disponibles donne lieu à la présentation d'une boite de dialogue dans laquelle l'utilisateur peut nommer ou sélectionner dans un sélecteur d'objets les objets devant subir l'opération. Bien sûr, il reste nécessaire de désigner avec la souris pour placer initialement justifications et nœuds.

3.2.2. Persistance

L'option de persistance permet de faire persister les opérateurs sélectionnées parmi l'ensemble des outils. Ainsi, après l'application d'une opération à un objet, l'opérateur reste sélectionné, le pointeur de souris conserve le même aspect et l'opération se répètera sur l'objet ou la position désignée tant qu'un autre opérateur n'est pas sélectionné.

3.2.3. Affichage des résultats

L'affichage des résultats peut se faire dans le terminal situé au bas de l'écran de travail ou dans des boites de dialogue notifiant l'accomplissement des actions.

3.2.4. Affichage des caractéristiques

L'affichage des attributs associés aux nœuds peut se faire dans le terminal situé au bas de l'écran ou dans des boites de dialogue.

3.2.5. Visualisation des nœuds

Il existe différentes façons de visualiser les nœuds dans les SMR (notation de McDermott, notation de Goodwin). Dans l'implémentation actuelle de SaMaRis, la notation utilisée est celle introduite dans [4]. Il est cependant aisé d'utiliser une autre notation en redéfinissant les pictogrammes utilisés. Il suffit pour cela de redessiner ceux-ci à l'aide d'un éditeur de bitmaps.

3.3. Maîtrise de la rétrogression

Il existe différentes stratégies de rétrogression pour les SMR et en particulier pour le TMS. Dans l'instanciation actuelle de SaMaRis/TMS, deux algorithmes sont implémentés. Le premier, dû à Jon Doyle, cherche l'ensemble des hypothèses soutenant l'inconsistance afin de choisir l'hypothèse à invalider. Il cherche alors l'ensemble des nœuds invalides soutenant l'hypothèse désignée et choisit celui qui va être validé. Le second, dû à Charles Petrie [6], remonte le graphe en arrière et choisit, à chaque embranchement, la branche à examiner. Une telle stratégie, si elle est bien dirigée, permet de trouver plus rapidement une solution. SaMaRis/TMS permet de modifier dynamiquement l'algorithme utilisé en le sélectionnant simplement dans le menu «rétrogression».

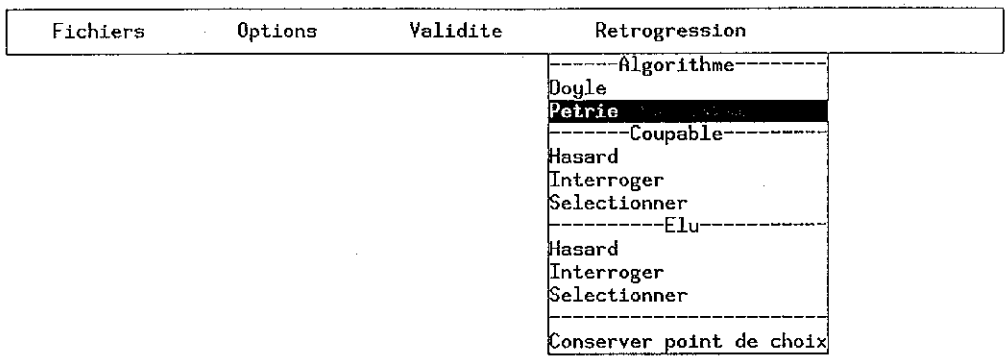

figure 12. Menu de rétrogression. Il permet de sélectionner l'algorithme de rétrogression à utiliser ainsi que les procédures de sélection des hypothèses à invalider et des nœuds à justifier pour cela.

Le menu rétrogression propose, par ailleurs, divers mécanismes de choix de l'hypothèse à invalider et du nœud à valider pour cela. Les trois stratégies, sélectionnables dynamiquement, consistent à:

- Choisir l'item au hasard,
- Demander à l'utilisateur, à l'aide d'une boite de dialogue, de choisir cet item (voir figure 13).

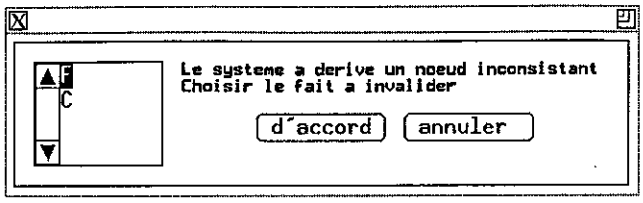

figure 13. Boite de dialogue de rétrogression. Elle permet de choisir l'hypothèse à invalider en cas d'inconsistance. Bien sûr, l'utilisateur n'est pas importuné quand un seul choix est possible.

4.Un pas plus loin

4.1.La généricité de SaMaRis

SaMaRis n'est pas seulement SaMaRis/TMS. Il est basé sur une conception générique de l'interface des SMR [3], qui a été utilisée plusieurs fois pour connecter plusieurs SMR et systèmes de raisonnement.

Cette interface permet la communication entre SMR et système de raisonnement. Pour cela, elle est basée sur un nombre minimum de primitives permettant au système de raisonnement de communiquer les informations nécessaires (nœuds, inférences, hypothèses, inconsistances...) au SMR et de consulter le résultat du travail de ce dernier: l'état des nœuds. À ces primitives s'ajoutent un certain nombre de modules du SMR qui sont paramétrables en fonction du système de raisonnement (paramétrage des choix lors de la rétrogression, de la messagerie d'erreur, de la notification des modifications dans l'étiquetage). Deux niveaux d'interfaçage sont considérés:

- Un couplage faible où le graphe du SMR est totalement indépendant de la base de faits du système de raisonnement. Ceci oblige le SMR à notifier chaque changement d'état au système de raisonnement.
- Un couplage fort où les nœuds du SMR sont intégrés aux faits du système de raisonnement. Ce dernier peut alors directement disposer de l'état des faits.

Ces deux niveaux ne modifient en rien le fonctionnement des SMR. Le couplage fort est simplement plus rapide à l'exécution mais demande plus de travail à l'implémentation (indépendamment du fait qu'il n'est pas toujours possible).

Pour ce qui est de l'interface SaMaRis/TMS, il s'agit d'un couplage faible: les structures de données du TMS ne sont pas intégrées à SaMaRis. À chaque création (nœud/justification) ou application d'un opérateur sur un nœud, l'ordre est transmis au SMR qui crée l'objet correspondant ou applique l'opérateur correspondant. Une fois ce travail fini, il notifie, à SaMaRis, les modifications que l'opération a entraînées sur les nœuds du graphe.

SaMaRis utilise donc les éléments paramétrables de l'interface générique ainsi que quelques modules additionnels (visualisation des nœuds, qui appartient aux possibilités d'un couplage fort, et multiplicité des algorithmes de rétrogression, qui n'est pas un paramètre traditionnel).

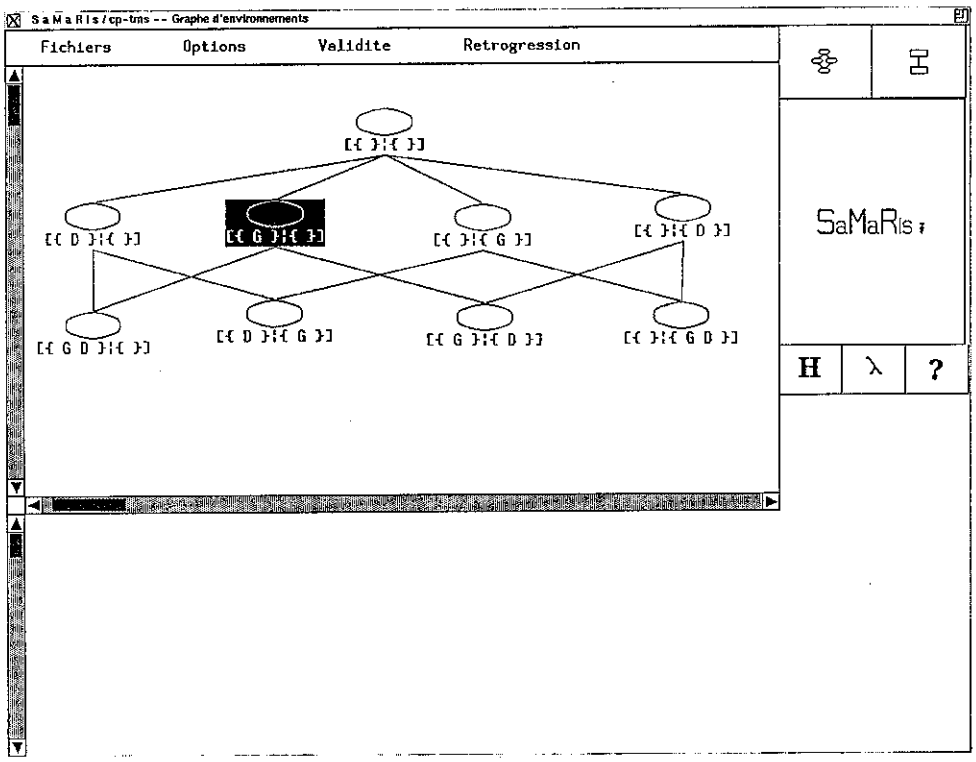

figure 14. L'écran des environnements. Il présente l'ensemble des environnements possibles à partir des hypothèses D et G.

La démonstration de la généricité de l'interface SaMaRis est faite au travers de son instanciation en SaMaRis/CP-TMS (le CP-TMS ayant été conçu suivant les directives de l'interface générique, mais avant SaMaRis). Le CP-TMS est un SMR qui, à l'instar de l'ATMS [1], permet de manipuler des hypothèses et permet donc de considérer le raisonnement sous plusieurs contextes (dénotés par des ensembles d'hypothèses). Ces contextes peuvent être représentés sous forme d'un graphe de complétion [4] (voir figure 14).

SaMaRis/CP-TMS permet la plupart des opérations autorisées par SaMaRis/TMS mais permet aussi de visualiser le graphe de complétions. L'accès au graphe de complétion se fait au travers des commutateurs (voir figure 15) qui permettent de passer à un autre écran de SaMaRis. Sur cet autre écran, seuls changent l'éditeur de graphe (a) et les opérateurs qui lui sont applicables (b) (voir figure 14).

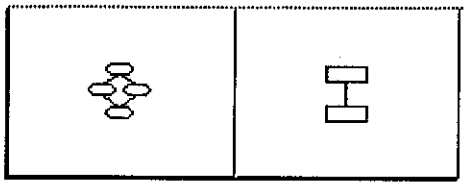

figure 15. Les commutateurs. Ici, ils permettent d'accéder au graphe des environnements et au graphe de composantes fortement connexes du graphe de dépendances.

SaMaRis/CP-TMS permet de sélectionner une complétion particulière et de connaître ses caractéristiques grâce à une boite d'inspection de contextes. Ainsi, la validité des nœuds sous un contexte précis peut être consultée (voir figure 16).

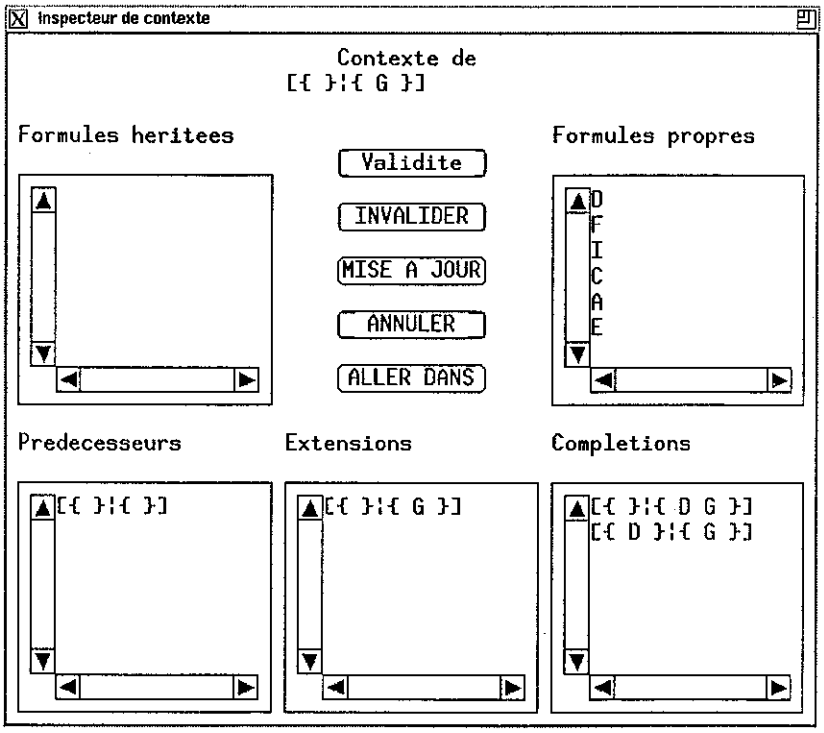

figure 16. Inspection de contexte. L'inspecteur de contextes permet d'atteindre les formules présentes dans un contexte parce qu'elles sont présentes sous un contexte antécédent ou parce qu'elles ont été inférées dans le contexte inspecté. Il est possible, à partir de cet inspecteur, d'aller inspecter un contexte connexe dans le graphe.

4.2. Etat des développements

SaMaRis a été développé sur Sun 3 et fonctionne sur écran couleur ou monochrome. Il utilise les couches logicielles suivantes: Unix (SunOS 4.0), X-Windows (X11R3), Lelisp (15.22) et Aïda (1.4) ainsi que quelques modules écrits en C. Il a été testé sur Sun4 (avec Lelisp 15.23) et devrait être portable sur toutes les plate-formes sur lesquelles ces couches logicielles sont disponibles.

Actuellement, deux SMR sont interfacés à SaMaRis: le TMS et le CP-TMS. Un manuel d'utilisation et d'interfaçage devrait voir le jour prochainement.

À partir de SaMaRis, tel qu'il se présente aujourd'hui, de nombreuses améliorations sont possibles:

- Inclure un ATMS dans la bibliothèque de SMR disponibles.
- Inclure des aspects particuliers des SMR dans l'interface. En particulier, la visualisation des composantes fortement connexes du graphe de dépendances. Cela suppose le dépassement de l'interface générique mais reste envisageable.
- La trace graphique (signalant chaque nœud examiné par le SMR lors du parcours de graphe par le SMR). Cela permettrait, en particulier, de piloter le SMR en mode pas à pas. Là encore, l'interface générique n'est pas suffisante.
- Dans le même ordre d'idées, la direction de la rétrogression pourrait se faire à la souris en désignant les nœuds à valider et invalider (ceci est aisé à réaliser sans modifier l'interface).

5.Conclusion

SaMaRis est donc un outil précieux, non seulement pour l'enseignement du fonctionnement des SMR mais aussi pour leur développement. En ce qui concerne son exploitation, les premiers essais avec des étudiants doivent commencer cette année. Sur le plan recherche, il donne déjà satisfaction sur plusieurs points. Tout d'abord, l'ensemble des figures de la section 1 ont été éditées avec SaMaRis et sauvegardées en PostScript. D'autre part, SaMaRis sert de support à une extension du TMS: l'oubli [5]. Enfin, l'implémentation même de SaMaRis a permis de confronter une fois de plus l'interface générique de commande des SMR avec une utilisation et d'en confirmer le bien fondé. Pour ce qui est de l'inclusion de SaMaRis, en tant que module explicatif, dans une application, l'expérience n'a pas été tentée. Toutefois, si l'implémentation actuelle, naïve, de SaMaRis convient pour une utilisation en démonstration, elle risque d'être trop lourde dans une utilisation opérationnelle où le nombre de nœuds peut être très grand.

En ce qui concerne l'avenir, deux projets importants pourraient donner à SaMaRis une autre dimension. Tout d'abord, SaMaRis devrait tirer parti d'un

gestionnaire d'hypertextes de façon à inclure la documentation très riche concernant les SMR déjà disponible sur support électronique. Un tel travail sera profitable pour les trois types d'acteurs susceptibles d'utiliser SaMaRis:

- L'apprenant pourra débrayer momentanément SaMaRis et naviguer dans l'hypertexte de façon à avoir des précisions sur certaines fonctionnalités. Mieux, c'est par le système hypertexte que l'apprenant entrera en contact avec les SMR et ce système lui permettra d'accéder à SaMaRis dans le cadre des travaux pratiques.
- Le développeur pourra, non seulement, fournir des explications graphiques grâce à SaMaRis et mais aussi inclure des explications hypertextuelles.
- Le chercheur rédigeant un article pourra le construire sur hypertexte et passer à SaMaRis pour créer des exemples. Il pourra aussi diffuser un «hyperarticle exécutable».

Le projet SaMaRis, quant à lui, dépasse le simple cadre du logiciel présenté ici. Il consiste à fédérer, autour de la représentation d'un raisonnement sous forme du graphe de dépendance, l'ensemble des services qui utilisent ce graphe: maintien du raisonnement, explication, oubli, propagation d'incertitude, de contextes, d'intervalles temporels... Ainsi, la charge que constitue, pour le système de raisonnement, l'enregistrement des dépendances peut être compensée par la mise à disposition du graphe à l'ensemble des services. Cela nécessite la généralisation des structures du graphe de façon à pouvoir y connecter facilement de nouveaux services. Le travail entamé avec la généricité des systèmes de maintien du raisonnement doit donc se poursuivre en «généricisant» le graphe de dépendances et son interface avec les services.

6.Références

[1] Johan De Kleer, **An assumption-based TMS**, *Artificial intelligence* 28-II, pp127-162, 1986

[2] Jon Doyle, **A truth maintenance system**, *Artificial intelligence* 12-III, pp231-272, 1979

[3] Jérôme Euzenat, **Un module TMS, version C0**, Rapport interne, Cognitech, Paris (FR), 1988

[4] Jérôme Euzenat, **Un système de maintenance de la vérité à propagation de contextes**, Thèse de l'université Joseph-Fourier, Grenoble (FR), 1990

[5] Jérôme Euzenat, Libero Maesano, **An architecture for selective forgetting**, Rapport interne, Laboratoire ARTEMIS, Grenoble (FR), 1990

[6] Charles Petrie, **Revised dependency directed backtracking for default reasoning**, Actes 6ième AAAI, pp167-172, Seatle (WA US), 1987