MEDIAN d'IA02 2015

Voici quelques commentaires et corrections à propos du médian d'IA02.

POUR LA BONNE CLAUSE

Soit la phrase suivante :

On connaît un homme qui aime toutes les femmes aimant au moins un végétarien.

1) Traduire cette phrase en une formule logique (ω) du calcul des prédicats du premier ordre. Prédicats : FEMME(x), HOMME(x), VEGETARIEN(x), AIME(x,y). Attention au parenthésage !

2) Et maintenant, clausons un peu : mettre ($\neg\ \omega$) sous forme clausale.

Question 1 : cet énoncé est facile à exprimer en L1, à condition toutefois d'agir avec méthode, en procédant étape par étape.

Niveau supérieur : *on connaît un homme qui vérifie la propriété P.*
$\rightarrow\ \exists x\ (\text{homme}(x) \wedge P(x))$
P(x) : *x aime toutes les femmes qui vérifient la propriété Q.*
$\rightarrow\ \forall y\ ((\text{fem}(y) \wedge Q(y)) \Rightarrow \text{aime}(x,y))$
Q(y) : *y aime u moins un végétarien.*
$\rightarrow\ \exists z\ (\text{vég}(z) \wedge \text{aime}(y,z))$
Donc finalement, on obtient :
$\exists x\ (\text{homme}(x) \wedge \forall y\ ([\text{fem}(y) \wedge \exists z\ (\text{vég}(z) \wedge \text{aime}(y,z))] \Rightarrow \text{aime}(x,y)))$

Question 2 : la mise sous forme clausale ne pose pas de problèmes particuliers non plus, dès l'instant où l'on suit scrupuleusement les étapes, avec rigueur. Attention, ici il s'agit de clausifier ($\neg\ \omega$) :

1 : $\neg\exists x\ (\text{homme}(x) \wedge \forall y\ (\neg[\text{fem}(y) \wedge \exists z\ (\text{vég}(z) \wedge \text{aime}(y,z))] \vee \text{aime}(x,y)))$

2 : $\forall x\ (\neg\text{homme}(x) \vee \exists y\ \neg(\neg[\text{fem}(y) \wedge \exists z\ (\text{vég}(z) \wedge \text{aime}(y,z))] \vee \text{aime}(x,y)))$
$\forall x\ (\neg\text{homme}(x) \vee \exists y\ ([\text{fem}(y) \wedge \exists z\ (\text{vég}(z) \wedge \text{aime}(y,z))] \wedge \neg\text{aime}(x,y)))$
$\forall x\ (\neg\text{homme}(x) \vee \exists y\ (\text{fem}(y) \wedge \exists z\ (\text{vég}(z) \wedge \text{aime}(y,z)) \wedge \neg\text{aime}(x,y)))$

4 : $\forall x\ (\neg\text{homme}(x) \vee (\text{fem}(f(x)) \wedge \text{vég}(g(x)) \wedge \text{aime}(f(x),g(x)) \wedge \neg\text{aime}(x,f(x))))$

5 : $\neg\text{homme}(x) \vee (\text{fem}(f(x)) \wedge \text{vég}(g(x)) \wedge \text{aime}(f(x),g(x)) \wedge \neg\text{aime}(x,f(x)))$

6 : $\{\neg\text{homme}(x) \vee \text{fem}(f(x))\} \wedge \{\neg\text{homme}(x) \vee \text{vég}(f(x))\} \wedge \{\neg\text{homme}(x) \vee \text{aime}(f(x),g(x))\}$
$\wedge \{\neg\text{homme}(x) \vee \neg\text{aime}(x,f(x))\}$

Soit finalement 4 clauses, écrites après renommages :

C1 $\{\neg\text{homme}(x) \vee \text{fem}(f(x))\}$
C2 $\{\neg\text{homme}(y) \vee \text{vég}(f(y))\}$
C3 $\{\neg\text{homme}(z) \vee \text{aime}(f(z),g(z))\}$
C4 $\{\neg\text{homme}(u) \vee \neg\text{aime}(u,f(u))\}$

MODELES

P et *Q* sont deux prédicats d'arité 2 de la Logique des Prédicats du 1er ordre. On considère les deux formules suivantes :

(ω_1) $\exists x\ \forall y\ (\ P(x,y) \Rightarrow Q(x,y)\ )$

(ω_2) $\forall x\ \exists y\ (\ P(x,y) \Rightarrow Q(x,y)\ )$

Le domaine d'interprétation *D* est ici l'ensemble des entiers naturels supérieurs ou égaux à 2. L'interprétation de *P* est la relation d'inégalité ≤, celle de *Q* est la relation divise (souvent notée |), au sens de la division euclidienne définie sur les entiers.

1. Donner, selon cette interprétation, les valeurs de vérité respectives des formules (ω_1) et (ω_2). On ne vous demande pas de faire de calcul ; en revanche, des explications seront bienvenues.

2. En gardant les mêmes interprétations pour *P* et *Q*, modifier le domaine d'interprétation *D* pour que les deux formules soient vraies.

Question 1 : question a priori facile….

L'interprétation de (ω_1) nous dit qu'il existe un entier supérieur ou égal à 2 tel qu'il diviserait exactement tout entier qui lui serait supérieur. Ceci est évidemment faux. Supposons un instant que cet entier existe, soit a. Alors a+1 n'est pas divisible par a, donc contradiction. Par conséquent, pour cette interprétation, (ω_1) est fausse.

En revanche, (ω_2) est vraie pour cette interprétation : soit x un entier supérieur ou égal à 2. Alors, par exemple, x divise 2x, qui est bien supérieur ou égal à x et aussi à 2.

Question 2 : question a priori encore plus facile….

La solution est très simple : il suffit de prendre D = $\mathbb{N}^*$. 1 divisant tout entier, (ω_1)devient vraie pour l'interprétation et (ω_2) le reste. En revanche, ce serait une mauvaise idée que de prendre D = $\mathbb{N}$…..

TIENS-TIENS….

Soient les deux atomes suivants : *P(x,f(A,y),x)* et *P(f(u,u),z,z)*.

Si ceci vous rappelle quelque chose, vous n'avez pas tout à fait tort..... Il vous est simplement demandé d'appliquer l'algorithme de Robinson encore appelé *unifier2* à ces deux atomes et d'en tirer les conséquences.

Cet exemple est analogue à celui du cours, et on procédera donc de la même façon.

Déroulement de l'algorithme de Robinson : appelons L1 et L2 ces deux atômes.

$\sigma \leftarrow \{\ \}$

Itération 1 :

$\sigma(L1) \neq \sigma(L2)$
Recherche des premiers symboles discordants : x et f.
Détermination des termes respectifs : x d'une part et $f(u,u)$ d'autre part.
Liaison : $< x \,.\, f(u,u) >$.
$\sigma \leftarrow \{ < x \,.\, f(u,u) > \}$ o σ soit $\sigma \leftarrow \{ < x \,.\, f(u,u) > \}$

Itération 2 :

$\sigma(L1) = P(f(u,u),f(A,y),x)$
$\sigma(L2) = P(f(u,u),z),z)$
$\sigma(L1) \neq \sigma(L2)$
Recherche des premiers symboles discordants : f et z.
Détermination des termes respectifs : $f(A,y)$ d'une part et z d'autre part.
Liaison : $< z \,.\, f(A,y) >$.
$\sigma \leftarrow \{ < z \,.\, f(A,y) > \}$ o σ soit $\sigma \leftarrow \{ < x \,.\, f(u,u) > ; < z \,.\, f(A,y) > \}$

Itération 3 :

$\sigma(L1) = P(f(u,u),f(A,y),f(u,u))$ et $\sigma(L2) = P(f(u,u),f(A,y),f(A,y))$
$\sigma(L1) \neq \sigma(L2)$
Recherche des premiers symboles discordants : u et A.
Détermination des termes respectifs : u d'une part et A d'autre part.
Liaison : $< u \,.\, A >$
$\sigma \leftarrow \{ < u \,.\, A > \}$ o σ soit $\sigma \leftarrow \{ < u \,.\, A > ; < x \,.\, f(A,A) > ; < z \,.\, f(A,A) > \}$

Itération 4 :

$\sigma(L1) = P(f(u,u),f(A,y),f(A,A))$ et $\sigma(L2) = P(f(u,u),f(A,y),f(A,y))$
$\sigma(L1) \neq \sigma(L2)$
Recherche des premiers symboles discordants : y et A.
Détermination des termes respectifs : y d'une part et A d'autre part.
Liaisons : $< y \,.\, A >$
$\sigma \leftarrow \{ < y \,.\, A > \}$ o σ soit $\sigma \leftarrow \{ < u \,.\, A > ; < \mathrm{y} \,.\, \mathrm{A} > ; < x \,.\, f(A,A) > ; < z \,.\, f(A,A) > \}$

Et dès lors : $\sigma(L1) = \sigma(L2)$. Sortie de la boucle *tant que* et fin du déroulement.

Conséquences :

- Les deux expressions sont unifiables
- Un unificateur le plus général est : $\sigma = \{ < u \,.\, A > ; < y.\, A > ; < x \,.\, f(A,A) > ; < z \,.\, f(A,A) > \}$
- L'instance commune la plus générale est : $P(f(A,A),f(A,A),f(A,A))$

<u>VOUS AVEZ DIT BIZARRE ?</u>

Les propos qui suivent ne sont que pure conjecture…. et toute ressemblance avec des personnages ou des événements réels ne serait que fortuite…..

1) Exprimer en calcul des prédicats L1 les énoncés ci-dessous :

   H_1 : *"<u>Seuls</u> les gens bizarres font des cours d'IA"*

   H_2 : *"Les gens bizarres et enfermés ne font pas de cours d'IA"*

   H_3 : *"Tout cours d'IA est fait par quelqu'un"*

   H_4 : *"On n'enferme que les gens bizarres"*

   H_5 : *"Il y a des cours d'IA"*

   Et enfin C : *"Il y a des gens bizarres en liberté"*

   On utilisera à cet effet les prédicats suivants (ou leurs abréviations soulignées) :

   <u>COU</u>RS-IA(x) : *x* est un cours d'IA

   <u>FAI</u>RE-COURS(x,y) : *x* fait un cours *y*

   <u>BIZ</u>ARRE(x) : *x* est bizarre

   <u>ENF</u>ERME(x) : *x* est enfermé

Cette question est vitale pour la suite puisqu'il s'agit en quelque sorte d'écrire les instructions du programme qui permettra de répondre à la question posée. Toute erreur à ce niveau compromettrait la suite de l'exercice. Il faut donc prendre son temps, procéder avec méthode, c'est-à-dire par étapes quand nécessaire, et bien disposer les quantifications et les parenthèses.

H1 : $\forall x\ \forall y\ ((\text{fai}(x,y) \wedge \text{cours}(y)) \Rightarrow \text{biz}(x))$
Cet énoncé exprime une nécessité, à savoir que « ceux qui font des cours d'IA sont nécessairement bizarres ». A cet égard, on rappelle que quand $p \Rightarrow q$, ceci signifie que q est une condition nécessaire de p. On retrouve cette structure logique dans l'expression proposée.

H2 : $\forall x\ ((\text{biz}(x) \wedge \text{enf}(x)) \Rightarrow \neg\exists y\ (\text{cours}(y) \wedge \text{fai}(x,y)))$
A priori, il n'y a pas de problème pour l'expression en L1 de cet énoncé. La version ci-dessus peut éventuellement être obtenue en deux étapes. Il en existe évidemment beaucoup d'autres, dont la forme suivante, apparemment plus simple, mais qui n'est pas forcément plus facile à trouver, à savoir : $\forall x\ \forall y\ ((\text{biz}(x) \wedge \text{enf}(x) \wedge \text{cours}(y)) \Rightarrow \neg\text{fai}(x,y))$

H3 : $\forall x\ \text{cours}(x)) \Rightarrow \exists y\ \text{fai}(y,x))$
Une forme extrêmement classique.

H4 : $\forall x\ (\text{enf}(x) \Rightarrow \text{biz}(x))$
Il fallait là encore bien percevoir la nécessité au cœur de cet énoncé. Autrement, dit, « les gens enfermés sont nécessairement bizarres".

H5 : $\exists x\ \text{cours}(x)$
Sans commentaires.

C : $\exists x\ (\text{biz}(x) \wedge \neg\text{enf}(x)S)$
Pas davantage de commentaire.

2) On veut montrer que (ω) : ($H_1 \wedge H_2 \wedge H_3 \wedge H_4 \wedge H_5) \Rightarrow C$, est une formule valide.

2.1 Rappeler le principe de la méthode, et les théorèmes qui la légitiment.

2.2 Mettre le problème sous forme clausale (détails d'obtention recommandés).

Question 2.1 : en quelques mots, la méthode consiste à réfuter l'ensemble des clauses issues des hypothèses H1, H2, H3, H4, H5 et de la négation du but (ou de la question) $\neg$C. Au-delà, réfuter, c'est obtenir un graphe de réfutation, c'est-à-dire un graphe qui se termine par la clause vide. Le développement de ce graphe est fait exclusivement par l'emploi de la règle de résolution pleine et entière. Rappelons que les théorèmes impliqués sont d'une part :
- le Théorème de Skolem qui établit qu'un ensemble de formules est insatisfiable ssi l'ensemble des formes de Skolem qui en résultent est insatisfiable,
- le Théorème de Complétude qui établit qu'un ensemble de clauses est insatisfiable ssi il en existe une réfutation par résolution.

Question 2.2 : a priori, compte-tenu de la structure logique particulièrement simple des formules considérées, il n'y a aucune difficulté qui mérite d'être mentionnée. De plus, chaque formule ici va donner une clause et une seule. Donc, avec quelques détails néanmoins, on obtient facilement les clauses suivantes, sans commentaires :

H1 : $\forall x \forall y\, ((fai(x,y) \wedge cours(y)) \Rightarrow biz(x))$
$\forall x \forall y\, (\neg(fai(x,y) \wedge cours(y)) \vee biz(x))$
$\forall x \forall y\, ((\neg fai(x,y) \vee \neg cours(y)) \vee biz(x))$
$C_1 : \{\neg fai(x,y) \vee \neg cours(y) \vee biz(x)\}$

H2 : $\forall x\, ((biz(x) \wedge enf(x)) \Rightarrow \neg\exists y\, (cours(y) \wedge fai(x,y)))$
$\forall x\, (\neg (biz(x) \wedge enf(x)) \vee \neg\exists y\, (cours(y) \wedge fai(x,y)))$
$\forall x\, (\neg biz(x) \vee \neg enf(x) \vee \forall y \neg (cours(y) \wedge fai(x,y)))$
$\forall x\, (\neg biz(x) \vee \neg enf(x) \vee \forall y \neg (cours(y) \wedge fai(x,y)))$
$\forall x\, (\neg biz(x) \vee \neg enf(x) \vee \forall y\, (\neg cours(y) \vee \neg fai(x,y)))$
$\neg biz(x) \vee \neg enf(x) \vee \neg cours(y) \vee \neg fai(x,y)$
$C_2 : \{\neg biz(u) \vee \neg enf(u) \vee \neg cours(v) \vee \neg fai(u,v)\}$

H3 : $\forall x\, (cours(x)) \Rightarrow \exists y\, fai(y,x))$
$\forall x\, (\neg cours(x) \vee \exists y\, fai(y,x))$
$\forall x\, (\neg cours(x) \vee fai(f(x),x))$
$\neg cours(x) \vee fai(f(x),x)$
$C_3 : \{\neg cours(z) \vee fai(f(z),z)\}$

H4 : $\forall x\, (enf(x) \Rightarrow biz(x))$
$\forall x\, (\neg enf(x) \vee biz(x))$
$C_4 : \{\neg enf(w) \vee biz(w)\}$

H5 : $\exists x\, cours(x)$
$C_5 : \{cours(A)\}$

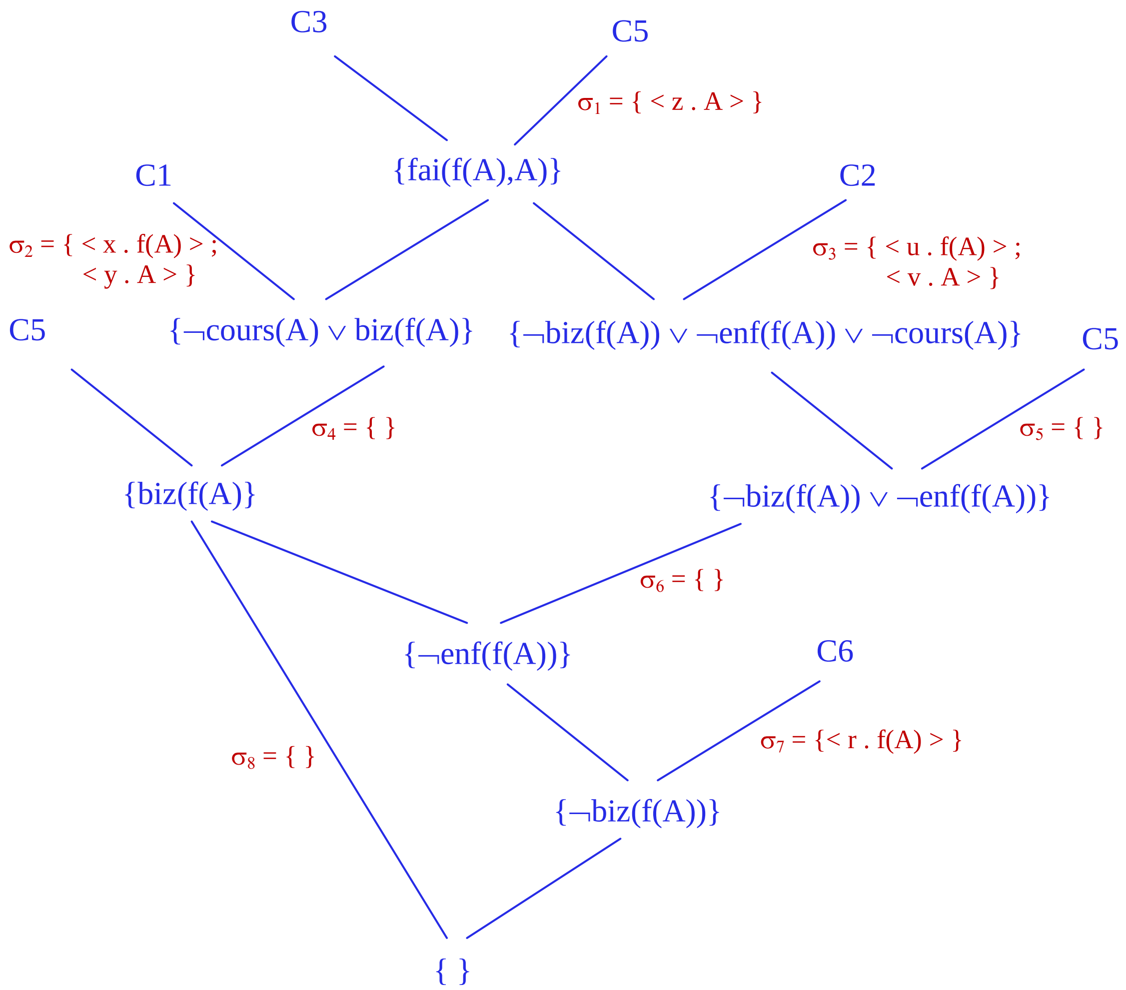
$\neg$C : $\neg\exists x\, (biz(x) \wedge \neg enf(x))$
$\forall x \neg(biz(x) \wedge \neg enf(x))$
$\forall x\, (\neg biz(x) \vee enf(x))$
$C_6 : \{\neg biz(r) \vee enf(r)\}$

Récapitulons :

$C_1 : \{\neg fai(x,y) \lor \neg cours(y) \lor biz(x)\}$
$C_2 : \{\neg biz(u) \lor \neg enf(u) \lor \neg cours(v) \lor \neg fai(u,v)\}$
$C_3 : \{\neg cours(z) \lor fai(f(z),z)\}$
$C_4 : \{\neg enf(w) \lor biz(w)\}$
$C_5 : \{cours(A)\}$
$C_6 : \{\neg biz(r) \lor enf(r)\}$

1) Déterminer un graphe de réfutation. Aucune stratégie de développement n'est imposée. Attention ! Il se peut qu'une clause ne soit pas utile….

La recherche de ce graphe ne présentait pas de problème particulier, et était semblable aux exemples de TD. Parmi les différentes solutions possibles, nous en donnerons une qui privilégie les constantes, afin d'éviter autant que possible les fastidieux renommages. A signaler une faute qu'il ne fallait surtout pas commettre et ce d'autant plus qu'elle avait été évoquée avec insistance en TD : en résumé, de $P \lor \neg Q$ et de $\neg P \lor Q$, on ne peut inférer la clause vide !!!! Soit on élimine les P, soit les Q, mais surtout pas les deux en même temps !



En conclusion, ce graphe de réfutation a été trouvé en 8 étapes ! A noter qu'il était possible de trouver un graphe de réfutation en moins d'étapes, à condition toutefois d'utiliser la résolution pleine et entière.

Conséquences :

- Théorique : C est une conséquence logique de {H1 ; H2 ; H3 ; H4 ; H5} et même de {H1 ; H2 ; H3 ; H5} puisqu'il s'avère en définitive que l'hypothèse H4 est superflue pour établir C.
- Pratique : par composition des substitutions, on obtient : < r . f(A) >, ce qui était hautement prévisible…..

MYSTERE

On considère les implications suivantes :

(R1) $\forall e\ \forall x\ \forall s\quad \{ MYS(x,s) \Rightarrow MYS(cons(e,x), cons(e,s)) \}$

(R2) $\forall e\ \forall x\ \forall s\quad \{ MYS(x,s) \Rightarrow MYS(x, cons(e,s)) \}$

(R3) $MYS(nil, nil)$

où *x,s* et *e* sont des symboles de variable, *cons* un symbole de fonction, *nil* un symbole de constante, et enfin MYS est l'abréviation du prédicat MYSTERE.

a) Appliquer la stratégie Prolog à la question : $\exists x\ MYS(x,cons(1,cons(2,nil)))$. On développera le graphe de réfutation jusqu'à l'obtention de deux solutions.

   On rappelle que la stratégie Prolog, dès qu'une solution a été trouvée, revient en arrière pas à pas (backtracking chronologique) pour rechercher d'éventuelles autres solutions.

b) Quelles sont, à votre avis, les autres solutions ?

c) **Bonus :** expliquez simplement ce que fait le prédicat MYSTERE. Une traduction explicite des règles serait appréciée !

   Indication : précisez ce que calcule ce prédicat quand le premier argument de la question est une variable *x* et que le second argument est une liste d'un nombre quelconque d'éléments représentée par $(cons(e_1,cons(e_2, \ldots cons(e_n, NIL) \ldots))$.

Cet exercice n'était pas évident, mais tellement proche de celui qui avait été traité en TD qu'il devait permettre de faire le plein !

Question a : commençons pas des remarques préalables. Tout d'abord, il est bien spécifié à deux reprises qu'il s'agit d'un problème Prolog pour lequel il faut adopter une stratégie Prolog. Par ailleurs, R_1 passe avant R_2, ce qui on le sait est essentiel. De plus, Prolog adopte le backtracking chronologique, qui fait qu'en cas d'impasse ou de recherche d'une autre solution on retourne en arrière mais pas à pas, et donc pas nécessairement au début. Enfin, on a vu en

TD (mais pas en cours) que les obligations de renommages étaient réajustées en Prolog. A cet égard, il a été dit que :

- quand on créait un résolvant il n'était plus indispensable de renommer ses variables,
- en contrepartie il était nécessaire avant chaque réutilisation d'une même clause de renommer au préalable ses variables à elle.

Dernier point : une fois qu'un graphe de réfutation est obtenu, la moindre des choses est de donner les liaisons portant sur les variables appartenant à la question, car, en définitive, c'est essentiellement ce qu'on cherche à obtenir en programmation logique.

Première action à entreprendre, bien sûr : il faut mettre le problème sous forme clausale, ce qui, on l'a vu, dans le cas d'un problème Prolog est une tâche enfantine puisque, précisément, les règles Prolog et la question Prolog sont déjà dès leur formulation des clauses de Horn. En fait, la seule action à mener est de renommer !

C_1 : { ¬MYS(x,s)) ∨ MYS(cons(e,x),cons(e,s)) }

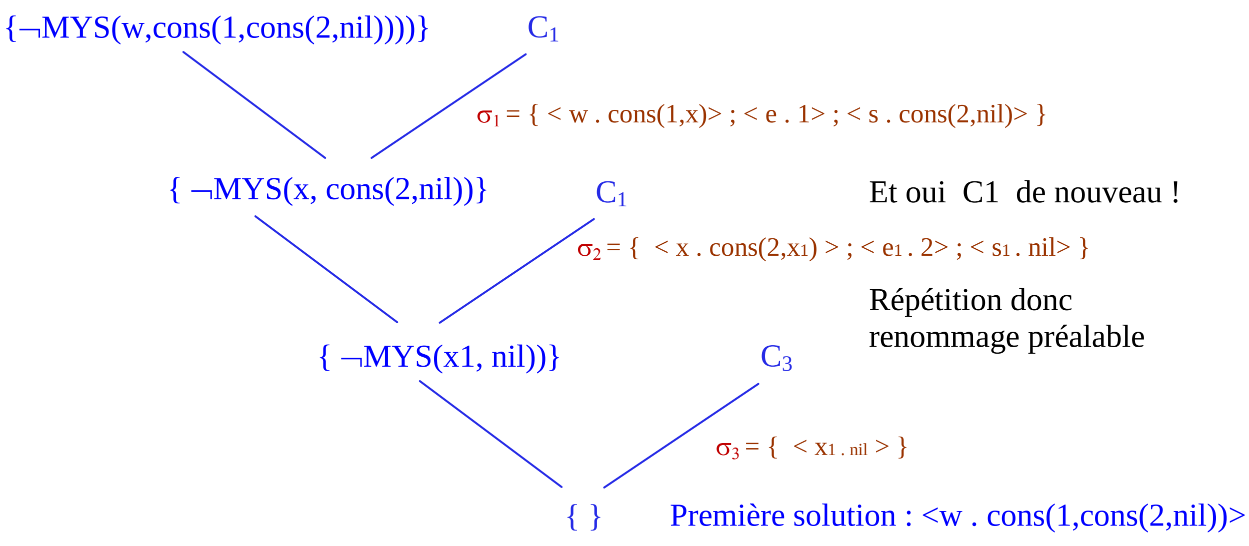
C_2 : { ¬MYS(y,t) ∨ MYS(y,cons(f,t)) }

C_3 : { MYS(nil,nil) }

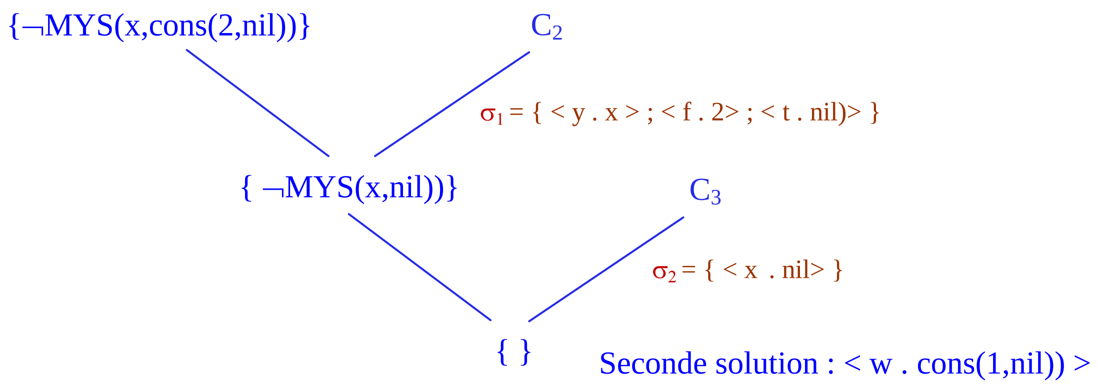
et enfin la négation de la question :

¬Q : { ¬MYS(w, cons(1,cons(2,nil))) }

L'ordre est donc désormais à respecter tel quel, et, à chaque cycle, la première clause qui pourra être associée à une clause-but est la bonne, qu'on le veuille ou non. On va donc partir de ¬*Q*, et finalement ce qui nous intéressera prioritairement est de connaître la liaison sur *w*. Autre point qu'il faut rappeler, la stratégie Prolog est en profondeur d'abord, jusqu'à ce que le graphe de recherche aboutisse à une impasse, lorsqu'aucune clause du programme ne peut être associée à la clause-but courante, ou jusqu'à ce qu'il aboutisse à la clause vide.



On fait alors un retour-arrière, niveau par niveau, qui nous amène au premier résolvant qui a été inféré.



Les deux solutions demandées ayant été obtenues, nous n'irons donc pas plus avant.

<u>Question b</u> : dès lors, il est possible de suspecter quelles pourraient être les autres réponses si l'on poursuivait le développement du graphe de recherche jusqu'à épuisement (des possibilités d'appariement, pas de ceux qui font la simulation…, encore que…). On se doute qu'il s'agit de : < w . nil> et < w . cons(2,nil)>

<u>Question c</u> : rappelons, comme on l'a vu en TD, que l'emploi de *cons* est un artifice qui permet de recréer des listes tout en restant dans le langage L1. Autrement dit, ayant précisé que *nil* désigne la liste vide, les solutions renvoyées lors du problème précédent sont les listes :

2 . 1 . nil, 1 . nil, puis nil et enfin 2 . nil,

c'est-à-dire <u>toutes les sous-listes de la liste passée en argument</u>, l'ordre relatif étant toutefois maintenu.

L'interprétation des règles est alors la suivante :
- Pour la règle R_3, évidemment : "la liste vide est une sous-liste de la liste vide",
- Pour la règle R_1 : "si *x* est une sous-liste de *s*, *e* . *x* est une sous-liste de *e* . *s*", ce qu'on ne saurait discuter.
- Pour la règle R_2 : "si *x* est une sous-liste de *s*, *x* est une sous-liste de *e* . *s*", ce qui est tout aussi pertinent.

Rappelons enfin que ces règles sont employées en sens inverse de l'implication logique, en tant que règles de réécriture d'un but en un but plus simple.