Contrôle Continu 3

Exercice 1: A propos du programme absurde

On considère le programme propositionnel Π_1 suivant (vu en cours).

(R0) m.
(R8) m, not a $\rightarrow$ a.

Question 1.1 Montrez à l'aide de l'algorithme ASPERIX vu en cours (arbre de dérivations) que ce programme Π_1 n'a pas de modèle stable.

Question 1.2 Donnez un contre-exemple (et justifiez qu'il s'agit bien d'un contre-exemple) à l'affirmation suivante:

Affirmation 1 *Si un programme* Π *contient la règle* ($R8$)*, alors aucun modèle stable de* Π *ne peut contenir l'atome* m*.*

Question 1.3 Reformulez l'affirmation 1 de façon à la rendre valide (vous n'avez pas à démontrer la validité).

Exercice 2: Echauffement

On considère le programme propositionnel Π_2 suivant:

(R1) l1.
(R2) c1.
(R3) s1.
(R4) l1, not c2, not s2 $\rightarrow$ l2.
(R5) c1, not l2, not s2 $\rightarrow$ c2.
(R6) s1, not l2, not c2 $\rightarrow$ s2.
(R7) l1, c1, s2 $\rightarrow$ m.
(R8) m, not a $\rightarrow$ a.

Question 2.1 Montrez en utilisant la définition par point fixe (basée sur le programme réduit) que $\{l1, c1, s1, c2\}$ est un modèle stable de Π_2.

Question 2.2 Pour chacun des ensembles d'atomes suivants, dites (en le justifiant soigneusement) si il s'agit oui ou non d'un modèle stable de Π_2. Il ne sera pas forcément nécessaire de calculer le programme réduit ni d'utiliser l'algorithme ASPERIX. Les résultats de l'exercice 1 pourront éventuellement être utiles...

(E1) $\{l1, c1, s2\}$
(E2) $\{l1, c1, s1, s2, m\}$
(E3) $\{l1, c1, s1\}$
(E4) $\{l1, c1, s1, c2, s2\}$

(E5) $\{l1, c1, s1, l2\}$

Question 2.3 Le programme Π_2 est-il stratifiable? Justifiez votre réponse sans avoir à dessiner le graphe de dépendance.

Exercice 3: Le loup, la chèvre et la salade

Un fermier veut traverser une rivière avec son loup, sa chèvre et sa salade. Il doit utiliser un bateau (que lui seul peut piloter), mais ne peut faire traverser qu'un seul passager à la fois (même la salade prend trop de place). Son problème est que, si il les laisse sans surveillance (si il n'est pas du même coté de la rivière), le loup mange la chèvre et la chèvre mange la salade...

Le but de cet exercice est d'écrire un programme dont les modèles stables sont toutes les façons de faire traverser la rivière aux 3 passagers sans que personne ne soit mangé. Notre programme contiendra les faits suivants (les noms de constantes commencent par une minuscule, les noms de variables commenceront par une majuscule). Notons aussi que nous faisons l'hypothèse du nom unique, en particulier loup $\neq$ chevre sera vrai dans toutes les interprétations.

(F1) pilote(fermier), passager(loup), passager(chevre), passager(salade).
(F2) berge(initial), berge(final).
(F3) position(fermier, initial, 0), position(loup, initial, 0).
(F4) position(chevre, initial, 0), , position(salade, initial, 0).
(F5) mange(loup, chevre), mange(chevre, salade).

L'atome position(X, B, N) veut dire que X est sur la berge B à l'étape N. Une règle de la forme
position(X, B, N), berge(B), berge(B'), B $\neq$ B', ... $\rightarrow$ position(X, B', N+1)
pourra gérer le changement de position de X à l'étape N, si il se déplace.

Question 1 Ecrire une (ou plusieurs) règle dont la conclusion est gagné(N) si, à l'étape N, le fermier et tous ses passagers sont sur la berge finale.

Question 2 Ecrire une (ou plusieurs) règle dont la conclusion est perdu(N) si, à l'étape N, le loup peut manger la chèvre ou la chèvre peut manger la salade.

Question 3 Il serait tout à fait possible que le fermier fasse traverser la chèvre, puis revienne avec la chèvre, puis refasse traverser la chèvre... Pour éviter les dérivations infinies, il faudra détecter (puis interdire) les répétitions. Ecrire une (ou plusieurs) règle dont la conclusion est redondance(N) si, à l'étape N, on observe la même situation qu'à une étape précédente (vous pourrez utiliser le prédicat $<$).

Question 4 On ne souhaite pas arriver à un modèle stable si on a déduit perdu(N). De la même façon, on ne souhaite pas arriver à un modèle stable si on a déduit redondance(N) (car une dérivation sans redondance suffit, et sera gérée dans une autre branche de l'arbre de recherche). Ecrire une (ou plusieurs) règle qui empechera d'aboutir à un modèle stable si un de ces deux atomes a été déduit.

Question 5 Ecrire une (ou plusieurs) règle qui modélise le fait qu'à chaque étape N, tant que le but n'a pas été atteint, le fermier traversera soit seul (on

déduira traverse1(N)) soit accompagné (on déduira traverse2(N), sans décider pour l'instant de qui il va faire traverser). Attention, on ne peut traverser à 2 que si un passager est disponible...

Question 6 Ecrire une (ou plusieurs) règle qui choisit le passager à transporter parmi les passagers disponibles. L'atome traverse(X, N) sera généré pour indiquer que le fermier fait traverser X à l'étape N.

Question 7 Ecrire une (ou plusieurs) règle pour gérer l'évolution des positions (le frame problem). Tout ce qui a traversé doit se retrouver sur la berge opposée à l'étape suivante, sinon la position ne change pas à l'étape suivante.