IAO2 MEDIAN-2017

CLASSONS

Ecrire les assertions suivantes en Langage des Prédicats du premier ordre, soit L1. On pourra à cet effet utiliser les prédicats suivants :

tiroir(x) : "x est un tiroir"
feuille(x) : "x est une feuille"
classeur(x) : "x est un classeur"
poly(x) : "x est un polycopié »
dans(x,y) : "x est dans y"

1- Tous les tiroirs contiennent des feuilles.
2- Aucun des classeurs ne contient de polycopiés.
3- Dans l'un des classeurs, il n'y a que des polycopiés.
4- S'il y a des feuilles dans un tiroir, on ne peut y trouver de polycopiés.

Phrase 1 :
Formule suffisamment simple ici pour pouvoir être écrite directement…
$\forall x\, (\text{tiroir}(x) \Rightarrow \exists y\, (\text{feuille}(y) \wedge \text{dans}(y,x)))$

Phrase 2 :
Pareil, sans trop d'efforts, avec une traduction quasi littérale.
$\neg\exists x\, (\text{tiroir}(x) \wedge \exists y\, (\text{poly}(y) \wedge \text{dans}(y,x)))$
ou encore : $\neg\exists x\, \exists y\, (\text{tiroir}(x) \wedge \text{poly}(y) \wedge \text{dans}(y,x))$
ou encore : $\forall x\, (\text{tiroir}(x) \Rightarrow \forall y\, (\text{poly}(y) \Rightarrow \neg\text{dans}(y,x)))$, etc….

Phrase 3 :
N'exigeant pas à mon avis le découpage en étapes.
$\exists x\, (\text{classeur}(x) \wedge \forall y\, (\text{dans}(y,x))) \Rightarrow \text{poly}(y)))$

Phrase 4 :
Un petit peu plus difficile sans doute, et là il était pertinent de décomposer :
$\forall x\, ((\text{tiroir}(x) \wedge P(x)) \Rightarrow \exists y\, (\text{feuille}(y) \wedge \text{dans}(y,x)))$
Avec P(x) : "il y a des feuilles dans x"
Soit : $P(x) : \exists y\, (\text{feuille}(y) \wedge \text{dans}(y,x))$
Finalement :
$\forall x\, ((\text{tiroir}(x) \wedge \exists y\, (\text{feuille}(y) \wedge \text{dans}(y,x))) \Rightarrow \exists y\, (\text{feuille}(y) \wedge \text{dans}(y,x)))$

INTERPRÉTONS

On considère les formules suivantes :

A : $\forall x\, (P(x) \Rightarrow Q(x,f(x)))$
B : $\forall x\, (P(x) \Rightarrow \exists y\, Q(x,y))$
C : $\exists y\, \forall x\, (P(x) \Rightarrow Q(x,y))$

On considère alors l'interprétation I_1 de domaine $D = \{a\ ;\ b\}$, où les symboles de prédicats P et Q, et le symbole de fonction f sont définis comme suit :

$I_1(P) : D \to \{V\ ;\ F\}$	$I_1(Q) : D^2 \to \{V\ ;\ F\}$	$I_1(f) : D \to D$
$a \mapsto V$	$(b,a) \mapsto V$	$a \mapsto a$
$b \mapsto F$	$(b,b) \mapsto V$	$b \mapsto b$
	F pour les autres	

puis l'interprétation I_2 de domaine $\{a\ ;\ b\}$, avec :

$I_2(P) : D \to \{V\ ;\ F\}$	$I_2(Q) : D^2 \to \{V\ ;\ F\}$	$I_2(f) : D \to D$
$a \mapsto V$	$(a,b) \mapsto V$	$a \mapsto b$
$b \mapsto V$	$(b,b) \mapsto V$	$b \mapsto b$
	F pour les autres	

Donner alors la valeur de vérité des formules A, B et C pour les interprétations I_1 et I_2. Quelques justifications seront les bienvenues.

Phrase A : $\forall x\ (P(x) \Rightarrow Q(x,f(x)))$

- $I_1(A) = F$. En effet, quand x=a, P(a) est V pour I_1. Mais, $I_1(f(a))$ étant égal à a, on constate que $I_1(Q(a,f(a)))$ est F car Q(a,a) est F pour I_1.
- $I_2(A) = V$. En effet, quand x=a, P(a) est V pour I_2, et a est bien en relation par Q avec f(a), car ici f(a) est interprété en b. De même, pour x=b, P(b) est V, et b est bien en relation avec f(b) car ici f(b) est interprété en b.

Phrase B : $\forall x\ (P(x) \Rightarrow \exists y\ Q(x,y))$

- $I_1(B) = F$. Pour x=a, on constate que la partie gauche de l'implication est V, alors que la partie droite est F : en effet, il n'existe pas de y tel que a soit en relation avec cet y par Q.
- $I_2(B) = V$. Pour x=a, on constate que la partie gauche de l'implication est V. La partie droite est V : en effet, a est en relation avec b par Q, et b est en relation avec b.

Phrase C : $\exists y\ \forall x\ (P(x) \Rightarrow Q(x,y))$

- $I_1(C) = F$. Pour x=a : on constate une fois de plus que la partie droite est F car a n'est en relation avec rien pour I_1.
- $I_2(C) = V$. Cet y est en fait b, car on constate que a et b sont en relation par Q avec b.

FLIPPER

1) Traduire en Langage des Prédicats du Premier Ordre L1 chacune des assertions suivantes :

   H_1 Les dauphins gris savent nager.

   H_2 Si tous ses enfants savent nager, un dauphin est joueur.

   H_3 Un dauphin qui a au moins un parent gris ou blanc est nécessairement gris.

   Vous utiliserez les prédicats suivants :

   gris(x) : "x est gris"

   blanc(x) : "x est blanc"

joueur(x) : “x est joueur”

parent(x,y) : “x est parent de y”

nage(x) : “x nage”

Remarques très importantes à lire préventivement :

- Le prédicat *dauphin(x)*, commun à toutes les assertions, n’a pas été jugé nécessaire à l’expression des connaissances. Si cela peut vous aider, vous pouvez l’introduire initialement dans vos formules, quitte à le retirer proprement par la suite (ou à le garder, sachant qu’alors les calculs seront plus longs).
- Prenez votre temps pour écrire ces formules. De leur justesse dépend celle de la suite !!!
- A cet égard, il est très fortement conseillé de déterminer les formules H_2 et H_3 en procédant en deux étapes !!!
- Vous ferez également très attention au parenthésage dont l’absence ou une mauvaise disposition pourrait avoir des conséquences désastreuses lors de l’étape de clausification.

On va donc procéder avec méthode, par prudence.

H1 : très facile.

$\forall x\ (gris(x) \ \Rightarrow\ nage(x))$

H2 : on va donc procéder avec méthode, par prudence.
On traduit d’abord : "si un dauphin x vérifie la propriété P, alors il est joueur".
$\forall x\ (\ (dauphin(x)\ \wedge\ P(x))\ \ \Rightarrow\ joueur(x))$
Avec P(x) : "tous les enfants de x savent nager".
Soit P(x) : $\forall x\ ((parent(x,y))\ \Rightarrow\ nage(y))$
Donc finalement :
$\forall x\ (\ (dauphin(x)\ \wedge\ \forall y\ (parent(x,y))\ \Rightarrow\ nage(y))\ \Rightarrow\ joueur(x))$
Et si on rend le prédicat dauphin implicite, on obtient en définitive :
$\forall x\ (\ (\forall y\ (parent(x,y)) \Rightarrow\ nage(y))\ \Rightarrow\ joueur(x))$
Il faut remarquer ici que la présence des parenthèses, en rouge, est capitale !

H3 : la démarche est semblable à la précédente, en plus simple.
On traduit d’abord : "si un dauphin x vérifie la propriété P, alors il est joueur". Et on obtient :
$\forall x\ (\ (dauphin(x)\ \wedge\ \exists y\ (parent(y,x)\ \wedge\ (gris(y) \vee\ blanc(y))))\ \Rightarrow\ gris(x))$
D’où finalement :
$\forall x\ (\ (\exists y\ (parent(y,x)\ \wedge\ (gris(y) \vee\ blanc(y))))\ \Rightarrow\ gris(x))$

2) Montrer tout d’abord que “*tout dauphin sans enfant est joueur*" (formule α), à l’aide d’un argument simple en termes de modèles et portant sur l’une des formules. Remarque : à défaut, on aurait pu prendre la méthode de réfutation, mais au prix d’un calcul assez long.

En fait, il suffit de considérer l'hypothèse H_2. En effet, soit un dauphin quelconque x. Ce dauphin étant sans enfant, parent(x,y), la partie gauche de l'implication interne est toujours F, et par conséquent cette implication est toujours V. En conséquence, la partie gauche de l'implication H_2 est vraie, et comme H_2 est elle-même considérée comme V, la seule issue est que la partie droite soit V. Par conséquent, le dauphin x est joueur.

3) Mettre les formules H_1, H_2, H_3 sous forme clausale. Les calculs explicites de ces clauses seront effectués sur la feuille de gauche.
Indications : H_1 donne une clause C_1 ; H_2 donne deux clauses C_2 et C_3 et enfin H_3 donne deux clauses C_4 et C_5.

Réglons le cas de H1 en quelques secondes :

C_1 : $\{\neg gris(x) \vee nage(x)\}$

Pour H2 et H3, donnons le détail des opérations.

H2 : $\forall x\ ((\forall y\ (parent(x,y) \Rightarrow nage(y)) \Rightarrow joueur(x))$
Là il fallait faire très attention à ce qu'est la partie gauche de l'implication.
$\forall x\ (\neg(\forall y\ (\neg parent(x,y) \vee nage(y))) \vee joueur(x))$
$\forall x\ ((\exists y\ (parent(x,y) \wedge \neg nage(y))) \vee joueur(x))$
Skolémisation du second type :
$\forall x\ ((parent(x,f(x)) \wedge \neg nage(f(x))) \vee joueur(x))$
$(parent(x,f(x)) \wedge \neg nage(f(x))) \vee joueur(x)$
$(parent(x,f(x)) \vee joueur(x)) \wedge (\neg nage(f(x)) \vee joueur(x))$
Soit finalement deux clauses :

C_2 : $\{parent(y,f(y)) \vee joueur(y)\}$
C_3 : $\{\neg nage(f(z)) \vee joueur(z)\}$

H3 : $\forall x\ ((\exists y\ (parent(y,x) \wedge (gris(y) \vee blanc(y)))) \Rightarrow gris(x))$
Là aussi, attention à la partie gauche de l'implication !
$\forall x\ (\neg(\exists y\ (parent(y,x) \wedge (gris(y) \vee blanc(y)))) \vee gris(x))$
$\forall x\ ((\forall y\ (\neg parent(y,x) \vee (\neg gris(y) \wedge \neg blanc(y)))) \vee gris(x))$
Propagation de la négation (étape 2) :
$\neg parent(y,x) \vee (\neg gris(y) \wedge \neg blanc(y)))) \vee gris(x)$
Passons à l'étape 6, celle de la distributivité.
$(\neg parent(y,x) \vee \neg gris(y) \vee gris(x)) \wedge (\neg parent(x,y) \vee \neg blanc(y) \vee gris(x))$
En définitive, on obtient deux clauses :

C_4 : $\{\neg parent(v,u) \vee \neg gris(v) \vee gris(u)\}$
C_5 : $\{\neg parent(r,s) \vee \neg blanc(s) \vee gris(r)\}$

4) On suppose que les hypothèses H_1, H_2, H_3 sont vraies. Vous devrez montrer que : "*tous les dauphins gris sont joueurs*" (formule β). Pour ce faire, vous adopterez la <u>méthode de réfutation par résolution</u> : aucune stratégie ne vous est imposée. Remarque : il n'est pas certain que toutes les clauses soient utiles…….

$\beta : \forall x\ (\text{gris}(x) \Rightarrow \text{joueur}(x))$

Donc, $\neg\beta : \neg\forall x\ (\neg\text{gris}(x) \vee \text{joueur}(x))$

Formule qui donne deux clauses à l'issue du processus de clausification :

$\exists x\ \neg\ (\neg\text{gris}(x) \vee \text{joueur}(x))$

$\exists x\ (\text{gris}(x) \wedge \neg\text{joueur}(x))$

Skolémisation du premier type :

$\text{gris}(a) \wedge \neg\text{joueur}(a)$ où désigne un symbole de constante quelconque.

Soit enfin :

C_6 : {gris(a)}

C_7 : {¬joueur(a)}

Finalement, le problème posé se réduit à la découverte d'un graphe de réfutation à partir des clauses suivantes :

$C_1 : \{\neg\text{gris}(x) \vee \text{nage}(x)\}$

$C_2 : \{\text{parent}(y,f(y)) \vee \text{joueur}(y)\}$

$C_3 : \{\neg\text{nage}(f(z)) \vee \text{joueur}(z)\}$

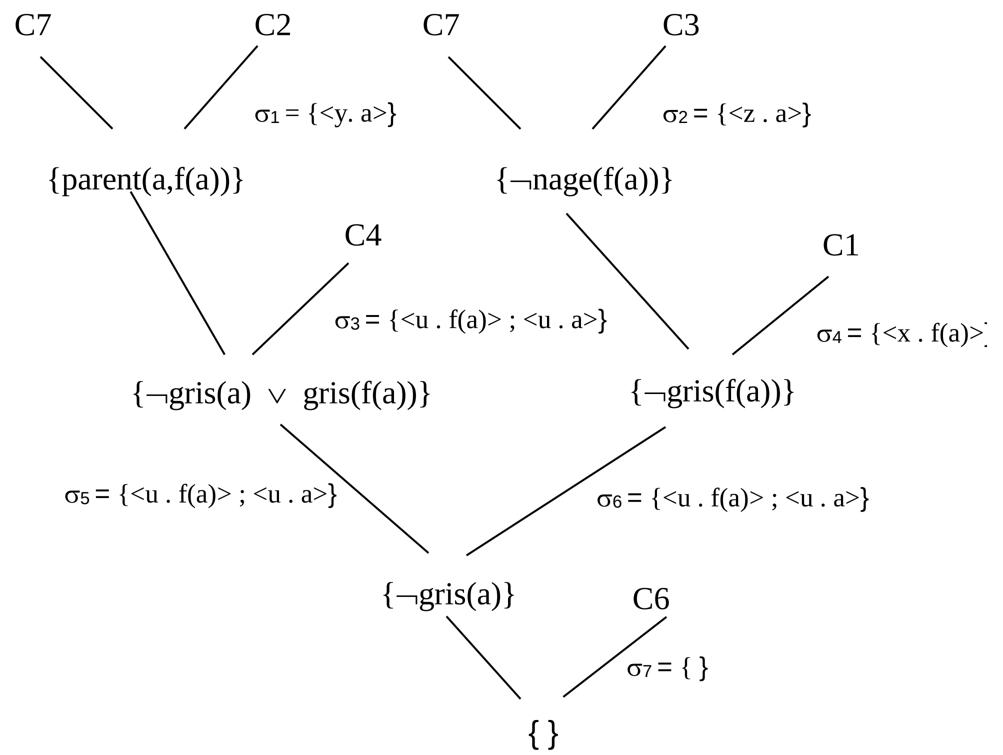
$C_4 : \{\neg\text{parent}(v,u) \vee \neg\text{gris}(v) \vee \text{gris}(u)\}$

$C_5 : \{\neg\text{parent}(r,s) \vee \neg\text{blanc}(s) \vee \text{gris}(r)\}$

$C_6 : \{\text{gris}(a)\}$

$C_7 : \{\neg\text{joueur}(a)\}$

<u>Un arbre de réfutation :</u>



C_5 était donc inutile, ce qui n'est pas une surprise, à cause de la présence du prédicat 'blanc'.

Conséquence : Si l'on ajoute l'hypothèse que "*Flipper a un parent blanc* », que peut-on en déduire sans calcul ? On justifiera la réponse bien entendu.

D'après la formule H_5, Flipper ayant un parent blanc, on peut en déduire que celui-ci est gris. Donc il sait nager, selon l'hypothèse H_1, et il est joueur selon la déduction précédente.

Remarque : si, sait-on jamais, en fin d'examen, il vous reste du temps et de l'énergie pour appliquer la méthode de réfutation à cette question, vous pourrez vous y coller … (bonus à la clef car c'est loin d'être simple…)

VOISINAGE

1) Traduire en formules du langage L1, puis en clauses, les énoncés suivants :

   R1 : si un habitant est proche d'un deuxième habitant lui-même proche d'un troisième habitant, alors le premier est proche du troisième.

   R2 : tout voisin d'un habitant est proche de cet habitant.

   R3 : tout habitant a un voisin.

   Le prédicat *habitant* étant implicite, on prendra seulement les prédicats suivants :

   voisin(x,y) : "x est voisin de y"

   proche(x,y) : "x est proche de y"

Très facile, donc sans commentaires.

R1 : $\forall x\ \forall y\ \forall z\ (\ (\text{proche}(x,y)\ \wedge \text{proche}(y,z))\ \Rightarrow\ \text{proche}(x,z))$

ou l'alternative suivante : $\forall x\ \forall z\ (\ (\exists y\ (\text{proche}(x,y)\ \wedge \text{proche}(y,z)))\ \Rightarrow\ \text{proche}(x,z))$

R2 : $\forall x\ \forall y\ (\text{voisin}(x,y)\ \Rightarrow\ \text{proche}(x,y))$

R3 : $\forall x\ \exists y\ \text{voisin}(x,y)$

D'où tout aussi simplement les clauses, puisque déjà, à l'origine, ces règles sont des clauses :

C1 : $\{\neg\text{proche}(x,y)\ \vee\ \neg\text{proche}(y,z)\ \vee\ \text{proche}(x,z)\}$

C2 : $\{\neg\text{voisin}(u,v) \vee\ \text{proche}(u,v)\}$

C3 : $\{\text{voisin}(w,f(w))\}$

2) On décide de suivre la stratégie Prolog pour répondre à la question Q suivante : « *Marc est-il proche de quelqu'un ?* ».

   Peu après le début de la réfutation, vous devez constater ou plutôt anticiper un problème : décrivez-le, et proposez un moyen pour y remédier.

La question Q est : ∃x proche(M,x). Sa négation donne la clause suivante ¬Q : {¬proche(M,s)}. Selon la stratégie Prolog qui en particulier impose de choisir toujours la première clause parmi les clauses d'entrée, on constate immédiatement que c'est toujours la clause C1 qui va être sollicitée pour être le parent associé à la clause issue de ¬Q. On constate donc le développement d'une branche infinie qui interdit l'obtention de la clause vide. Par conséquent, l'ordre adopté pour les clauses n'est pas pertinent. La solution est extrêmement simple : il suffit d'inverser l'ordre des deux premières règles, le prédicat proche étant alors décomposer en prédicat voisin, C3 jouant alors le rôle d'une condition d'arrêt.

3) Cette modification ayant été effectuée, répondez à la question ci-dessus, en appliquant à la lettre une stratégie Prolog et en arrêtant la réfutation dès l'obtention d'une <u>deuxième solution</u>. Remarque : les renommages seront faits "à la Prolog".

Le programme modifié est donc finalement :

C1 : {¬voisin(u,v) ∨ proche(u,v)}

C2 : {¬proche(x,y) ∨ ¬proche(y,z) ∨ proche(x,z)}

C3 : {voisin(w,f(w))}

Avec ¬Q : {¬proche(M,s)}