Fiche de TP no. 3

Rappels

Urls de `Prover9` et `Mace4` :

http://www.cs.unm.edu/~mccune/mace4/
http://www.cs.unm.edu/~mccune/mace4/manual/2009-11A/

`Prover9` essaye de montrer que $\Gamma \models \phi$, c'est-à-dire ϕ est une conséquence logique de Γ, en utilisant des méthodes basés sur la résolution. `Mace4` essaye de montrer que $\Gamma \not\models \phi$, en construirant un modèle de Γ qui rend ϕ fausse.
Utilisation via la ligne de commande :

```
prover9-mace4-gui
```

Skolemisation et résolution

Exercice 1 : *Trop des modèles.* Considérez le fichier input suivant :

```
assign(domain_size,5).
assign(max_models,-1).

formulas(sos).

S(0,1) & S(1,2) & S(2,3) & S(3,4).
S(x,y) -> x < y.
x < y  & y < z -> x < z.
x < y | x = y | y < x.
-(x<x).
S(x,y) -> - (exists z (x < z & z < y)).
exists x exists y x < y.

end_of_list.
```

`Mace4` prétend qu'il y a 10 modèles différents de taille 5 de cet ensemble de formules. Expliquez pourquoi.

Exercice 2. Récupérez le fichier Château Letot (TP 2) et demandez encore une fois à `Prover9` de prouver que Agate a tué Agate. Visualisez la preuve construite par `Prover9` par le biais de l'outil `gvzify`.

1. Quelles sont les clauses obtenues par `Prover9` à partir de formules dans les fenêtres des assomptions et des buts ? Quel est le langage de ces clauses ?
2. Analysez toute inférence obtenue par la règle de résolution : pour chaque inférence, quelle est la substitution utilisée ?
3. De même, pour la règle de factorisation.
4. Il y a t-il des inférences que vous ne pouvez pas reconduire à la règle de resolution ou à la règle de factorisation ?
5. Demandez à `Prover9` d'utiliser le calcul de la résolution pur (sans autres règles) ; pour ce faire, ajoutez dans la page des options du langage la commande suivante : `redeclare(equality,"--").` Que se passe-t-il ?

Maths sur ordinateur

Exercice 3 : *Ensembles.* En théorie des ensembles, deux ensembles sont considéré égaux si et seulement si tout élément de l'un est élément de l'autre. Cette définition de l'égalité est formalisée par l'hypothèse :

```
all x all y (x = y <-> all z (element(z,x) <-> element(z,y))) .
```

On dit qu'un ensemble est sous-ensemble d'un autre ensemble si tout élément de l'un est élément de l'autre.

1. Formalisez cette définition en notant par la formule `sousens(x,y)` le fait que x est sous-ensemble de y.
2. Montrez, grâce à `Prover9`, que ces deux définitions impliquent que la relation `sousens` est une relation d'ordre (réflexive, transitive, antisymétrique).
3. Montrez, grâce à `Mace4`, que la définition de l'égalité ci dessus est nécessaire afin que `sousens` soit une relation d'ordre. Identifiez quelle propriété, parmi réflexivité, transitivité, antisymétricite, n'est pas conséquence logique de la seule définition de `sousens`.
4. Montrez, grâce à `Mace4`, que la relation `sousens` n'est pas une relation d'équivalence. Étudiez le contre-modèle du but construit par `Mace4`.

Exercice 4 : *Partitions.* Rappelons que :
— une relation d'équivalence est une relation binaire R réflexive, symétrique et transitive ;
— une partition d'un ensemble D est une collection de sous-ensembles de D non vides $\{ P_i \mid i \in I \}$ telle que $P_i \neq P_j$ implique $P_i \cap P_j = \emptyset$, et $D = \bigcup_{i \in I} P_i$.

1. Utilisez `Prover9` pour montrer que si R est une relation d'équivalence sur D, alors la collection de sous-ensembles $\{ P_d \mid d \in D \}$—où $P_d = \{ x \in D \mid R(d,x) \}$—est une partition de D.
2. Utilisez `Prover9` pour montrer que si $\{ P_d \mid d \in D \}$ est une partition de D telle que $d \in P_d$, pour tout $d \in D$, alors la relation R definie par $R(x,y)$ ssi il existe $d \in D$ tel que $x, y \in P_d$ est une relation d'équivalence sur D.
3. Argumentez donc que les relations d'équivalence R sur D sont en bijection avec les partitions $\{ P_d \mid d \in D \}$ telles que $d \in P_d$, pour tout $d \in D$.
4. Utilisez `Mace4` pour compter le nombre de partitions sur un ensemble donné, de taille 3 jusqu'à la taille 10.
   Vérifiez votre résultat en recherchant la suite de nombres ainsi construite sur *The On-Line Encyclopedia of Integer Sequences*, `http://oeis.org/`.

Jeux

Exercice 5 : *Échecs.* Considérons un échiquier de taille $n \times n$. L'échiquier contient k pions blancs et k pions noirs (avec $k \leq n$). Voici les règles d'un jeu très simple :

1. Au debut du jeu, les pions blancs se trouvent à hazard sur la première ligne, et les pions noirs se trouvent à hazard sur la dernière ligne.
2. Les pions blancs et noirs peuvent avancer d'une case seulement (ne peuvent pas manger) ; s'il rencontrent un pion en face de l'autre couleur, alors il ne peuvent plus avancer.
3. Le blanc et le noir peuvent enchaîner des coups, sans donner la main à l'adversaire.

Choisissez un langage approprié et décrivez une position via un ensemble de formules de la logique du premier ordre, de façon qu'il y aura une correspondance bijective entre les positions et les modèles $\mathcal{M}$ de cet ensemble de formules tels que $D_{\mathcal{M}} = \{ 0, 1, \ldots, n-1 \}$.
Utilisez `Mace4` pour compter combien il y a de positions possibles dans le jeu si $n = 3$ et $k = 2$.
Combien de solutions on a si $n = 3$ et $k = 3$?