

Série d'exercices #10 $\frac{1}{2}$

IFT-2035

November 25, 2009

1 Arbres de preuve

Soit la relation `membre` qui définit quand un élément est dans une liste:

```
membre(E, [E | _]).  
membre(E, [_ | L]) :- membre(E, L).
```

1. Montrer l'arbre de preuve des étapes importantes par lesquelles passe le système Prolog pour essayer de satisfaire la requête:

```
membre(X, [1, 1]), X == 2.
```

Les étapes importantes sont celles où la recherche tombe sur une impossibilité (et doit donc faire un retour-arrière).

2. Corriger la relation avec l'aide de `\==` de sorte qu'elle évite la redondance.
3. En utilisant votre nouvelle définition, dessiner l'arbre de preuve qui satisfait la requête `membre(1, [2, 1, 3])`.
4. De même montrer les arbres de preuve importants construits pour essayer de satisfaire la requête `membre(1, [2, 3, 4])`.

2 L'incontournable

Définir la règle de tri `quicksort(X, Y)` qui dit que `Y` contient les mêmes éléments que la liste `X`, mais triés par ordre croissant. Cette version utilisera un opérateur de comparaison fixe, la relation `<`.

Une règle auxiliaire `partition(X, L, S, G)` sera nécessaire qui dit que la liste `S` contient les éléments de la liste `L` qui sont plus petits que `X`, alors que `G` contient ceux qui sont plus grand.

Utiliser la règle prédéfinie `append(X, Y, Z)` qui dit que `Z` est la concaténation des listes `X` et `Y`.