Module « Paradigmes et langages de programmation »

Sujet proposé par Olivier Lecarme

2006–2007

Interprète pour une version minimale de Prolog

Le but est de réaliser une implémentation d'un Prolog minimal, c'est-à-dire d'un moteur d'unification de clauses de Horn de la forme la plus simple possible. Les spécifications sont données par les transparents 300 à 311 du cours des premières semaines. En première approche, vous pouvez ignorer les extensions expliquées dans les transparents 306 et 307, ainsi que le mécanisme de *coupure*.

Le programme à construire doit donc accepter des *clauses*, c'est-à-dire des *faits* et des *règles*, qui sont ajoutées à la base existante, initialement vide, et des *buts*, qui sont des questions posées.

On peut obtenir trois sortes de réponses :

- il n'y a pas de solution, c'est-à-dire que le but demandé ne s'efface pas ;
- il y a une solution, c'est-à-dire que le but s'efface inconditionnellement ;
- il y a plusieurs solutions, c'est-à-dire que le but s'efface pour différentes combinaisons de valeurs de ses variables ; dans ce cas, si l'utilisateur tape un point-virgule après avoir obtenu la réponse à un but, le programme doit fournir la prochaine solution, s'il y en a une.

Toutes les clauses du programme dont la tête peut s'unifier avec le but doivent être considérées dans la tentative d'effacement du but, leur ordre n'ayant pas d'importance.

Une partie importante du travail est donc la mise en œuvre de l'*unification* : il s'agit de tenter d'univer une expression, le but, avec un modèle, la tête de règle, dans un environnement courant. Si l'unification réussit, on obtient la substitution qui constitue la solution cherchée.

Le moteur de l'interprète a pour rôle de lire des buts à effacer, jusqu'à lire le but de sortie de l'interprète, et de tenter l'effacement de chaque but lu. L'effacement d'un but construit virtuellement un arbre de recherche des solutions. L'environnement courant de l'état de la recherche doit être mémorisé. La fonction d'effacement orend donc en compte la liste des buts à effacer, l'environnement courant et la liste des règles disponibles.

Vous pouvez rechercher sur le Web des cours de Prolog, ainsi que des indications sur la manière de programmer l'interprète. En revanche, la recherche de solutions toutes faites ne vous servira à rien, puisque le point capital est la comparaison des possibilités offertes par les trois langages d'implémentation. À vous de voir, en particulier, comment utiliser les spécificités de chaque langage : fonctions d'ordre supérieur en Caml, conteneurs et itérateurs en C++, évaluation dirigée par le but en Icon.