IFT 2035
Travail pratique 2 - Interpréteur H2035
Rapport

Marie-Anne Prud'Homme-Maurice (1054064)
Olivier Guénette (20154866)

13 décembre 2021

Tentons d'être logique

Dans le cadre du cours IFT 2035. Il nous a été demandé de concevoir un elaborateur et un evaluateur du langage h2035 en utilisant le langage de programmation logique Prolog. Le travail a pour but d'implémenter une fonction qui décompose les lignes de code pour faire une inférence des types ainsi que de remplacer les appels de variable pour utiliser les indices de Bruijn. Par la suite, une autre fonction eval permet d'effectuer les opérations appropriées.

Ce rapport décrit notre processus d'analyse, les problèmes rencontrés et nos solutions.

Apprentissage de Prolog

Comme une majorité de personnes dans la classe, cet exercice fût notre premier vrai travail pratique en programmation logique. Pour débuter le projet, nous avons passé un peu de temps à lire et faire des recherches sur le language pour venir palier à notre manque de connaissance.

Voici quelques ressources utilisées:

- https://www.tutorialspoint.com/prolog/index.htm
- https://ocw.upj.ac.id/files/Textbook-TIF212-Prolog-Tutorial-3.pdf
- https://www.geeksforgeeks.org/prolog-an-introduction/

Utilisation des indice de Bruijn

L'utilisation des indices de Bruijn est un concept interessant pour simplifier la référence aux valeurs des variables dans un environnment donné. Cependant, nous avions de la difficulté à faire les liens entre les valeurs des indices dans les exemples donnés. Notre intuition pensait que les indices devait posséder d'autres valeur que celles mentionnées.

Pour résoudre cette situation, nous avons fait des sous-routines de test pour nous donner plus de vision sur le contenue de l'environnement en imprimant sont contenu au moment de trouver l'indice de la variable.

De plus, lors de l'évaluation des fonctions, des erreurs sont survenue, ce qui nous a induit vers un processus de recherche pour corriger le problème de référence.

Association de valeur aux variables indéfinies

Dans prolog, des variables peuvent être déclarées, mais instanciées vraiment plus tard dans l'exécution du programme. Ceci était plus une surprise de notre part, car aillant plus travaillé avec Java, Python et Javascript, il nous était pas intuitif de passer des variables non définies dans des fonctions comme paramètres.

Un exemple concret serait l'élaboration des équation contenant un lambda. Dans le code fourni, la variable du lambda est insérée au début de l'environnment avec une référence vers une variable de type inconnu. Cette variable est ensuite utilisée pour créer le type du lambda. Cette utilisation de variable était un concept différent pour nous.

La solution a été de faire plus de petits exercices en utilisant les ressources mentionnées plus haut. Ainsi que de trouver une façon d'associer des valeurs à ces variables.

Détermination des ? par inférence de type

C'était difficile de bien saisir comment implémenter le ?, car nous sommes habitués d'implémenter des variables dont le type nous est déjà connu. En utilisant les indices de Bruijn il était plus facile de retrouver ces variables sans nom en utilisant le type attendu.

Pour surmonter cette difficulté, nous avons fait l'inférence des types sur une feuille de papier pour inférer nous même les foncions appropriées.

Implentation de let mutuellement recursif

Au début du travail pratique nous pensions qu'il n'y avait pas de différence entre la présence et l'absence des crochets dans un let, donc nous les avions implémentés de la même façon. Après de longues analyses, nous avions compris que ce n'était pas le cas et que la difficulté du let mutuellement récursif était que les variables pouvaient être placées dans n'importe quel ordre.

Notre solution a été d'insérée les noms des variables dans l'environnment. Par la suite, élaboré les définitions pour trouver les type et d'utiliser les indices de Bruijn lors de création de variables mutuellement récursives. Pour finalement, faire des fermetures de l'environnment des lamdbdas lors de ¡ l'évaluation des variables récursives.

Conclusion

En conclusion, ce travail pratique était encore plus difficile que le dernier car nous avions encore moins d'expérience avec Prolog que Haskell. Malgré plusieurs problèmes de compréhension et d'implémentation, nous avons su approfondir nos connaissances en langage de programmation logique.