Rapport : Interpréteur du Calcul Lambda

Haotian XUE
Master 2 Sciences et Technologies du Logiciel (STL)

17 novembre 2024

Introduction

Ce projet consiste à développer un interpréteur en OCaml pour le -calcul enrichi. L'objectif principal est de permettre l'évaluation d'expressions incluant des variables, des abstractions, des applications, des opérations arithmétiques, et des listes, tout en assurant un système de typage fiable.

Fonctionnalités principales

Langage et syntaxe

Le langage supporte :

— Les expressions classiques du -calcul : variables, abstractions (`x. x`) et applications (`(f x)`).
— Les opérations sur les entiers (`+`, `-`, `*`) et les listes (`head`, `tail`, `length`).
— Les branches conditionnelles (`if ... then ... else`).
— La déclaration de variables (`let x = ... in ...`).

Évaluateur et système de typage

L'évaluateur utilise une stratégie *Call-by-Value* et gère les *-réductions*. Le système de typage vérifie les expressions avant l'évaluation, en s'appuyant sur un environnement associant variables et types.

Interface interactive (REPL)

Le REPL permet d'écrire des expressions, de voir leur type inféré et leur résultat après évaluation. Les erreurs de syntaxe et de typage sont signalées avec des messages clairs.

Exemples d'utilisation

Expressions simples et conditionnelles

```
> x . x
Expression : x . x
Type : (Int -> Int)
Resultat : x . x

> if 0 = 0 then 1 else 2
Expression : if 0 = 0 then 1 else 2
Type : Int
Resultat : 1
```

Listes

```
> [1, 2, 3]
Expression : [1, 2, 3]
Type : [Int]
Resultat : [1, 2, 3]

> head([1, 2, 3])
Expression : head([1, 2, 3])
Type : Int
Resultat : 1
```

Conclusion

L'interpréteur atteint ses objectifs en supportant un sous-ensemble du -calcul avec des extensions pratiques comme les listes et les branches conditionnelles. Malgré quelques limitations, comme l'absence de types polymorphes, ce travail pose une base robuste pour des améliorations futures.