Younes Chetouani

Rapport d’avancement : Évaluateur-Typeur de Lambda-Calcul

Encadrant : Romain Demangeon

17/11/2024

Younes Chetouani

L’objectif de ce projet est de développer un évaluateur et un typeur pour un λ-calcul enrichi. Le projet se divise en plusieurs parties :

- **Partie 2 : Évaluateur pour un λ-calcul pur**
- **Partie 3 : Système de typage pour le λ-calcul avec types simples**
- **Partie 4.1 : Extension de l’évaluateur pour un λ-calcul enrichi**
- **Partie 4.2 : Extension du système de typage pour les fonctionnalités avancées (entiers natifs, listes, tests conditionnels, etc.)**
- **Partie 5 : Traits impératifs**

L’IA a été utilisé dans la conception de tests ainsi que lors du débug. L’IA m’a également assisté lors de l’exécution des tests afin de détecter la cause du ou des problèmes en donnant à l’IA comme contexte les résultats obtenus et ceux attendus.

Partie 2 : Évaluateur pour un λ-calcul pur

- **État : Complétée et testée**
- **Code source : /lib/eval.ml**
- **Tests : /test/test_eval.ml**
- **Résumé** :
  - L’évaluateur implémente correctement les règles de réduction pour le λ-calcul pur, y compris les abstractions (`λx.M`), et les applications (`M N`),
  - Les tests sur des termes simples et complexes montrent que l’évaluation produit les résultats attendus.
- **Problèmes rencontrés** : Aucun problème majeur n’a été identifié dans cette partie.

Partie 3 : Système de typage avec types simples

- **État : Fonctionnel mais nécessite des corrections mineures**
- **Code source : /lib/type.ml**
- **Tests : /test/test_type.ml**
- **Résumé** :
  - Le système de typage repose sur un algorithme d’unification pour vérifier la cohérence des types dans un λ-calcul simplement typé.
  - Les tests sur des termes simples confirment que le typage est fonctionnel et génère les contraintes adéquates.
  - Cependant, un problème a été identifié au niveau de l’inférence des types : les types générés ne s’affichent pas correctement en sortie. Il s’agit vraisemblablement d’un problème lié à la gestion des substitutions dans les équations. J’ai suivi l’idée d’Amine pour retrouver le type inféré mais sans succès.
- **Problèmes rencontrés** :
  - Le typage est fonctionnellement correct mais le type inféré ne s’affiche pas de manière lisible, ce qui complique les validations manuelles.

Younes Chetouani

Partie 4.1 : Extension de l’évaluateur pour un λ-calcul enrichi

- **État : Complétée et testée**
- **Code source : /lib/eval_4.ml**
- **Tests : /test/test_eval_4.ml**
- **Résumé** :
    - L’évaluateur a été étendu pour gérer des fonctionnalités supplémentaires :
        - Entiers natifs avec les opérateurs d’addition et de soustraction.
        - Tests conditionnels (if zero then else) et opérations sur les listes (hd, tl, cons).
        - Gestion du point fixe (fix) pour la récursivité.
    - Les tests montrent que l’évaluateur gère correctement ces nouvelles constructions et qu’il n’y a pas de régression.
- **Problèmes rencontrés** : Problème rencontré lors de l’introduction des points fixes qui a été réglé grâce à l’IA.

Partie 4.2 : Extension du système de typage pour le λ-calcul enrichi

- **État : Implémentée mais non testée (time is up)**
- **Code source : /lib/type_4.ml**
- **Tests : none**
- **Résumé** :
    - Les règles de typage ont été étendues pour prendre en charge :
        - Le type N pour les entiers natifs et Lst pour les listes.
        - Les opérateurs arithmétiques, les tests conditionnels (if zero, if empty) et les opérations sur les listes (hd, tl, cons).
        - La quantification universelle (Forall) pour le polymorphisme.
    - Bien que le typeur a été implémenté, il n’a pas encore été soumis à des tests.
- **Problèmes rencontrés** :
    - Pas testée donc n’aura probablement pas le comportement attendu (et a toujours le même problème que en 3.)