Rapport projet traduction des langages

Aymen Ben Abdallah

janvier 2020

Contents

1 Introduction

Ce document représente un résumé des tentatives de résolution des problèmes abordés par ce projet et des différentes difficultés que j'ai rencontré.

Un compilateur est un programme traitant des instructions dans un langage donné et de les traduire en un autre langage de programmation de plus bas niveau (langage machine).

Ce projet consiste à ajouter de nouvelles fonctionnalités au langage Rat tel que les pointeurs, les types énumérés, la surcharge de fonctions ... Il fallait donc modifier les différentes passes afin de les adapter à chacun de ces changements.

Dans la suite de ce rapport, on citera les différents choix de conception qu'on a été amené à prendre pour intégrer ces nouvelles fonctionnalités.

2 Les pointeurs

Afin d'ajouter les pointeurs, on s'est décidé de prendre le concept établi ci dessous comme support:

- créer un type **affectable** qui peut être soit un **pointeur** soit un **ident**
- créer un type **Pointeur of Typ** dans type.ml
- rajouter les constructeurs suivants au type expression :
  - **Null** qui représente le pointeur null
  - **Val of affectable** afin d'accéder à la valeur de l'affectable
  - **Adresse of string** afin d'accéder à l'adresse de l'affectable
  - **Nouveau of typ** pour créer un pointeur

2.1 gestion des identifiants

Comme on a ajouté le nouveau type **affectable**, il a fallu réviser la structure de la tds. Un pattern matching devait donc nous permettre d'identifier si un affectable est un ident ou un pointeur (ou un pointeur de pointeur...). Dans le dernier cas, on fait un appel récursif à la fonction devrait donc permettre de trouver l'identifiant dernier du pointeur et ainsi en créer une **info_ast**.

Ce plan, bien que théoriquement correcte, nous a empéché d'avancer plus loin dans le projet et nous a bloqué malgrès les effort qui y sont investis. Les difficultés rencontrés seront expliqués plus tard

2.2 gestion des types

Dans cette passe, nous avons juste ajouté une fonction qui traite le type pointeur (ou affectable) qui se charge de boucler sur un pointeur pour retourner son type (pointeur sur un entier par example). D'autres difficultés ont aussi été rencontrés dans cette passe énoncés plus tard aussi

3 Surcharge de fonctions

3.1 Démarche

Cette étape a été résolu (partiellement car pas terminée) en ajoutant une table de hashage qui se charge de stocker les fonctions déclarées (vérifier que les types de ses paramètres ne matche pas ceux d'une autre fonction).

4 Difficultés rencontrées

En ce qui concerne les pointeurs, leur implantation a été un succès dans les tests mais a quand même fait apparaître de nouvelles erreurs: si par exmple un rationnel est déclaré en fonction d'une variable int : rat a = b/c, le compilateur ne reconnaîtra pas alors b et c et retournera une exception **MauvaiseUtilisationIdentifiant**. les erreurs levées dans les test de typage sont de même nature.

La surcharge de fonction n'est elle pas implantée complaitement faute de temps. En effet, quelques exception se lèvent dans les tests de type "Match Failure" et cela car la passe n'est pas complètement implantée et la partie **AppelFonction** dans la **passeTypeRat** a été écrite n'importe comment juste pour permettre la bonne compilation du code.

On note par contre que cette passe est valide dans le code de la version de tp jointe avec le code du projet.

5 Conclusion

Le projet en tout n'a pas été trop compliqué mais faute de temps et de capacité de débugger les erreurs survenues on n'a pas pu compléter les autres fonctionnalités. Cependant une idée assez claire est en tête de la manière dont on aurait pu procéder pour implanter ces fonctionnalités là.