Introduction to language theory and compiling
Project - Part 3

Antoine Passemiers
Alexis Reynouard

December 12, 2017

Contents

1. Assumptions

- Pas de gestion du scope. Tout est variable global, meme pour les appels de fonction.
- Les variables Imp sont nomms en llvm, toutes les variables temporaires sont non-nommees
- Tout est int32 (meme les fonctions ne renvoyant rien renvoient i32 0)

2. Augmenting Imp's syntax and grammar

2.1 Regular expressions

On s'est simplifi la vie pour funcName, moduleName. Mots cls faciles du genre import.

2.2 Grammar rules

arglist, paramlist... assign -¿ call

3. Implementation

3.1 Improvements in both lexer and parser

3.1.1 Error handling

erreur de syntax, nom de module non existant, fonction non definie...

3.2 LLVM code generator

recursive descent code generator ?

On a hard code une methode par variable de la grammaire. Quand c'est necessaire, une methode renvoie le nom d'une variable temporaire (non nommee) qui stocke le resultat de l'expression.

Exemple: a := (7 + 9)

7 est stocke dans une variable non nommee, puis de meme pour 9, puis 7 + 9 encore dans une autre, puis cette derniere est stockee dans la variable %a avec un store.

4. Bonus features

- Fonctions (definition et appel, arite au choix)
- Mot cles rand, import, function
- Gestion de librairie standard
- Gestion d'exceptions
- Optimization (faudrait qu'on checke les flags de llvm pour qu'il optimise les trucs immondes qu'on genere)
- Si tu trouves des trucs facile a ajouter, ne te prives pas