Compilation

Florent Mercier

1 Introduction à Lex et Yacc

1.1 Lex

Lex est un analyseur lexical, c'est à dire qu'il produit un automate reconnaissant les unités lexicales d'un programme.

Un fichier Lex est composé de trois parties :

Descriptions : permet de définir des expressions régulières

Règles : permet de définir des actions sur ces expressions régulières

Fonctions : permet de définir des fonctions utilisées dans ces actions

1.2 Yacc

Yacc est un analyseur syntaxique, c'est à dire qu'il reconnaît la structure d'un programme.

Un fichier Lex est composé de trois parties :

Descriptions : permet de définir les terminaux et non-terminaux d'une grammaire, les précédences et le start-symbol

Règles : permet de gérer les actions lors de règles reconnues et les accès aux sous-arbres

Fonctions : permet de définir des fonctions utilisées dans ces actions

1.3 Association Lex/Yacc

Lors du parcours d'un texte, Lex parcourt ce texte jusqu'à rencontrer un symbole non-terminal. Dans ce cas là, il l'envoie à Yacc qui le traite.
Cela permet de générer un analyseur complet de la grammaire.

2 Utilisation de Lex et Yacc dans le cadre du projet

L'utilisation de Lex et Yacc m'aurait permis de gérer l'ensemble de mes règles automatiquement plutôt que de devoir définir manuellement chaque comportement.

En effet, en indiquant mes règles de G0 dans ces analyseurs, ceux-ci auraient pu traiter l'ensemble des cinq arbres, bien que ce ne fut pas la partie la plus compliquée du projet.

Principalement, l'avantage que m'auraient procuré Lex et Yacc aurait été dans la reconnaissance de chaque élément d'une de mes lignes afin de traiter plus facilement leur type. En effet, là où j'ai rencontré le plus de difficultés a été dans la partie Scan, afin d'avancer dans une règle en ne prenant en compte que les éléments pertinents seulement j'ai eu des problèmes à identifier et délimiter les unités lexicales.

De plus, l'analyseur (principalement syntaxique) m'aurait permis à partir de mon jeu de règles de participer dans ma GPLaction à construire le code de mon programme.

3 Problèmes engendrés par l'utilisation de Lex et Yacc

Tout d'abord, l'analyseur lexical généré par Lex et l'analyseur syntaxique produit par Yacc ne fonctionnent qu'avec des programmes codés en langage Ada, Eiffel et C++, or j'ai décidé de réaliser mon projet en Java afin de pouvoir utiliser la flexibilité de ce langage pour créer mes noeuds, analyses et scanners. De plus, je ne dispose pas des compétences nécessaires en Ada, Eiffel ou C++ pour développer un compilateur dans ces langages.