Rapport de projet

Wollenburger Antoine, Chénais Sébastien

Sommaire

Introduction

Il nous a été demandé d'écrire un compilateur, en OCAML, d'un langage de notre cru vers du svg (format pour décrire des images vectorielles). Pour mener à bien cette tâche, nous nous reposerons sur les notions que nous avons vu en cours.

Chapitre 1

Évolution du projet

1.1 Définition d'une grammaire

La toute première étape d'une telle tâche est la définition d'une grammaire. Afin de pourvoir y arriver, nous avons décidé de commencer avec une grammaire ne permettant que quelques actions basiques. A savoir utiliser des variables, dessiner des points et tracer des lignes.

1.2 Lexer

Nous avons alors écrit l'analyseur syntaxique, comprenant les mots de notre grammaire. Cette étape est de loin la plus facile.

1.3 Parser

Ensuite, nous avons transformé le fichier d'entrée en un arbre binaire, modèle de représentation intermédiaire, qui nous permettra l'écriture du fichier svg. Nous avons décidé d'implémenter nos arbres comme suit :

Listing 1.1– Type arbre

```
type operation =
    Drawing
  | DrawingSize
  | BlocEmbrace
  | Declaration
  | BlocBrace
  | Parameters
  | Point
  | Line
  | Instruction
  | Draw
  | Var of (string)
```

```
  | Number of (int);;

type t_arbreB = Empty | Node of node
        and node = { value: operation; left: t_arbreB; right: t_arbreB };;
```

1.4 Et le bonheur arrive.

Viens alors la partie de loin la plus fastidieuse. En effet, pour transformer cet arbre binaire en svg, nous avons besoin d'une implémentation d'un automate à pile. Pour ceci point de choix. Nous passerons par une table shift-reduce. A la vue de la taille du vocabulaire, cette table est tout sauf petite.