Bourret-Mathieu Blaise, Kineider Guillaume, Laborier Nestor

Compilation : exemple de compilateur pour des expressions arithmétiques simples

Positionnement Thématique de notre TIPE

Informatique Théorique, Informatique Pratique

Mots-clés

Compilation	*Compiling*
Analyse Lexicale	*Lexical Analysis*
Analyse Syntaxique	*Syntaxical Analysis*
Calcul Arithmétique	*Arithmetic Calculus*
Interpréteur	*Interpreter*

Bibliographie commentée

L'apparition des premiers langages de programmation, c'est-à-dire les langages indépendants de la machine sur laquelle ils sont écrit (à dissocier des langages machine et assembleur), date des années 50. C'est à cette période que sont naturellement apparus les premiers compilateurs, indispensable au fonctionnement de ces langages.

Tout langage doit être interprété par la machine pour être exécuté, et pour cela il doit être compilé, c'est à dire traduit dans un langage compréhensible par la machine mais trop hostile à l'homme pour être utilisé directement. Selon le livre Compilateurs : Principes, Techniques et Outils **[1]**, *« un compilateur est un programme qui lit un programme écrit dans un premier langage – le langage source – et le traduit en un programme équivalent écrit dans un autre langage – le langage cible »*. Le compilateur ne traduit donc pas nécessairement en langage machine mais vers un autre langage quel qu'il soit. Le compilateur lui-même est écrit dans un langage, qui peut même être celui qu'il traduit ! On parle alors de *boostrapping* **[1]**. Nous utiliserons pour notre part le langage Caml pour nos expériences **[4]**. Le compilateur sert aussi à détecter des erreurs dans le programme source **[5]** mais nous nous intéresserons uniquement à sa fonction de traducteur.

Pour comprendre le programme source, il s'appuie principalement sur deux analyses : lexicale et syntaxique. Pour programmer ces analyses nous utiliserons deux outils fournis par Caml, CamlLex pour l'analyse lexicale, CamlYacc pour la syntaxique **[3,4]**. Les structures et les syntaxes spécifiques de ces 2 outils, détaillées principalement dans Formation au Langage Caml de Claude Marché **[3]**, nous ont amenés à mieux comprendre les mécanismes de la compilation.

Pour limiter notre étude, nous nous sommes intéressés à deux langages en particulier. Pour le langage source, nous avons choisi Scheme et en langage cible, nous nous sommes portés sur Forth. Le choix de ces langages n'est pas anodin, en effet, leur structure de base (notamment pour les opérations arithmétiques) est assez aisée à comprendre et leurs syntaxes sont assez éloignées pour bien apercevoir la mise en œuvre des mécanismes de la compilation. Nous nous sommes initiés à Scheme grâce à l'ouvrage Structure And Interpretation of Computer Programs de H.Abelson et G.J.Sussman **[2]**. La documentation Le Langage Caml de Xavier Leroy **[6]** nous a aidé à comprendre la structure de pile utilisé par le langage Forth. Ces deux ouvrages avec l'aide de celui de Claude Marché **[3]** nous ont permis de créer, dans un premier temps, deux interpréteurs (Scheme et Forth), étape initiatique vers la création de notre compilateur Scheme-Forth.

Problématique retenue

Pour tout compilateur, les impératifs et le fonctionnement sont les mêmes. Il s'agira donc de comprendre le processus théorique de la compilation puis une méthode de réalisation en pratique.

Objectifs du TIPE

1. Principe théorique : je détaillerai le modèle théorique ainsi que les différentes étapes du processus de compilation.
2. Je présenterai rapidement les langages auxquels nous nous sommes intéressé, c’est-à-dire les langages Forth, Scheme et Caml Light avec les raisons de ce choix.
3. Nous réaliserons un petit compilateur pour des expressions arithmétiques ayant pour langage source Scheme et pour langage cible Forth.

Références bibliographiques

[1] A. AHO, R. SETHI, J. ULLMAN: *Compilateurs : principes, techniques et outils.* Pearson Education , 2007.

[2] H.ABELSON, G.J.SUSSMAN et : *Structure and Interpretation of Computer Programs.* Intereditions, 1989.

[3] C. MARCHÉ et R.TREINEN : *Formation au langage Caml*, https://www.lri.fr/~paulin/LGL/docs/polycaml-etuds.pdf

[4] X. LEROY : *Caml Light Manual*, https://caml.inria.fr/pub/docs/manual-caml-light/index.html

[5] L.MARANGET : *Cours de compilation*, http://gallium.inria.fr/~maranget/X/compil/poly/index.html

[6] X. LEROY : Le Langage Caml 2ème Edition