SOM2IF15 – Compilation

Frédéric Loulergue

2022

Outline

Nom et contact

- Frédéric Loulergue
- frederic.loulergue@univ-orleans.fr
- Réponse habituellement dans les 24h (jours et horaires ouvrés)
- Bâtiment IIIA, bureau C01

Postes

- 2000-05: Maître de conférences, Université Paris-Est Créteil
- 2005-16 : Professeur, Université d'Orléans
  - 2007 : Professeur invité (Université de Tokyo)
  - 2014-15 : Détachement Inria (Paris)
  - 2015 : Professeur invité (NII, Tokyo)
- 2016-20 : Full Professor, Northern Arizona University
- Depuis fin 2020 : Professeur, directeur adjoint du LIFO
  Équipe LMV: Langage Modèles et Vérification

Outline

Compilation ?

Compiler

Transformer un programme dans un langage non directement executable en un programme dans un autre langage executable.

Quelques livres de référence récents

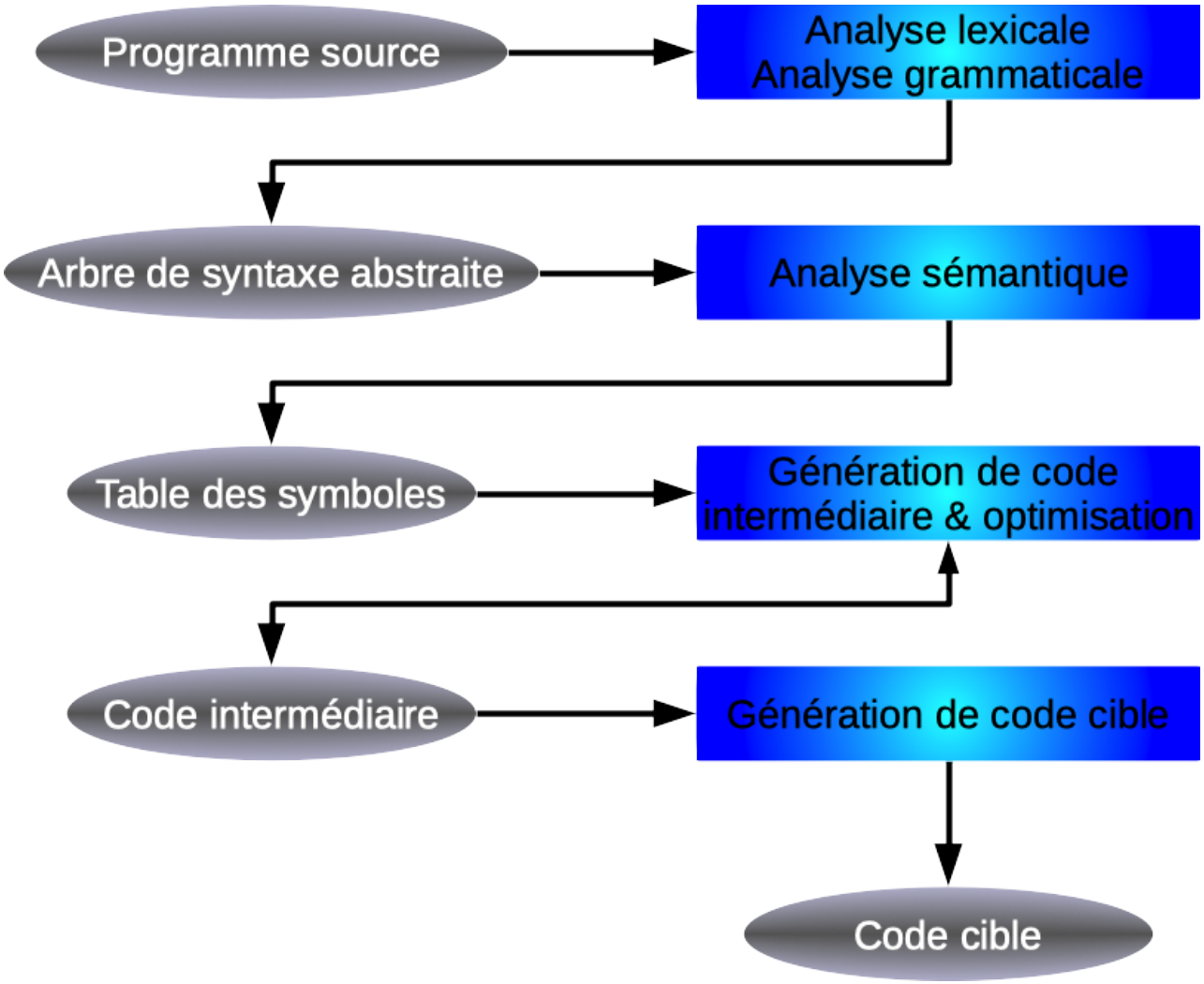
- A. Appel, Modern compiler implementation in Java 500 pp.
- C. Fisher et al, Crafting a Compiler 700 pp.
- K. Cooper, Engineering: A Compiler 800 pp.

$\Longrightarrow$ conception et développement d'un compilateur complet

Comment faire en 12h CM + 18h TP ?

- Il faut faire des choix !

Vue schématique d'un compilateur



- Analyse syntaxique : approche pratique
- Optimisations : très peu

Réalisation d'un compilation complet

Langages ?

- Source : à définir ensemble ! (*par défaut* : impératif)
- Cible : *par défaut* assembleur MIPS
- Implantation : *par défaut* Java

Organisation

- Par équipes
- Cours : 8 $\times$ 1h30
- TP : 9 $\times$ 2h

Évaluation RNE: CC intégral

- Code : entre 8000 et 10000 lignes de Java
  (non généré, hors tests)
  - Une partie sera fournie
  - Une partie sera écrite collectivement en TP
  - Une partie sera écrite par groupe en TP ou hors séances
  - Rendus réguliers (pull automatique des dépôts `git`)
    Échec du *build*: 0
  - Évaluation : code qui compile, code correct, code bien architecturé, effort personnel (commits `git`)
- Présentation (soutenance) du compilateur à la fin du semestre

Évaluation RSE

- CT 2h (nécessite d'avoit étudié le code du projet)

Outils

Par défaut

- IntelliJ (avec plugin ANTLR v4)
- build avec Gradle (DSL Kotlin)
- ANTLR: `https://www.antlr.org`
- SPIM: `http://spimsimulator.sourceforge.net`

Outils

Est-ce que mon équipe peut

- utiliser un autre langage de développement ?
  - OCaml, F#, Reason : oui
  - Langage s'interfaçant avec Java (Scala, Kotlin, ...) : oui
  - Python : oui (mais ce n'est pas conseillé)
  - autre : à discuter
- utiliser un autre IDE ?
  oui, mais si vous avez des problèmes, vous vous débrouillez seuls
- utiliser un autre système de build :
  - oui, mais si vous avez des problèmes, vous vous débrouillez seuls
  - il faut que l'outil soit supporté par IntelliJ et que les dépendances et les tâches de build soient opérationnelles en quelques clics de ma part (et que ça soit bien documenté).
- compiler vers autres chose que MIPS/SPIM ?
  ça dépend, à discuter pour les détails