Compilation.

Didier Rémy
Octobre 2000

http://cristal.inria.fr/~remy/poly/compil/
http://w3.edu.polytechnique.fr/profs/informatique/Didier.Remy/compil/

Travaux dirigés avec Luc Maranget

Slide 1

Programme C :

```
int fact (int x) {
  if (x == 0) return 1;
  else return x * fact (x-1);
}
```

Code Pentium :

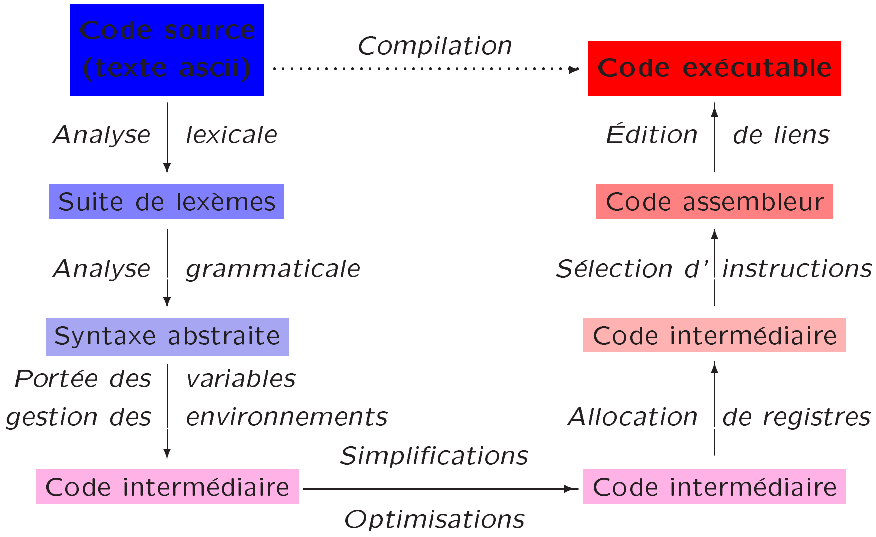
```
fact:
    pushl %ebp
    movl %esp,%ebp
    cmpl $0,8(%ebp)
    jne .L2
    movl $1,%eax
    jmp .L1
    .align 4
    jmp .L3
    .align 4
.L2:
    movl 8(%ebp),%eax
    decl %eax
    pushl %eax
    call fact
    addl $4,%esp
    movl %eax,%eax
    movl %eax,%edx
    imull 8(%ebp),%edx
    movl %edx,%eax
    jmp .L1
    .align 4
.L3:
.L1:
    leave
    ret
```

Code optimisé

```
fact:
    pushl %ebp
    movl %esp,%ebp
    pushl %ebx
    movl 8(%ebp),%ebx
    testl %ebx,%ebx
    je .L2
    leal -1(%ebx),%eax
    pushl %eax
    call fact
    imull %ebx,%eax
    jmp .L4
    .align 4
.L2:
    movl $1,%eax
.L4:
    movl -4(%ebp),%ebx
    leave
    ret
```

La chaîne complète de compilation

Slide 2



Démystifier l'évaluation des programmes

- Comprendre les principes
- Appréhender la pratique.
- Apprendre quelques techniques.

Contenu

Slide 3

- Langages formels, automates finis et analyse lexicale.
- Grammaires *context-free* et analyse syntaxique.
- Architecture des machines, code machine, assembleur.
- Sémantiques opérationnelles, interpréteurs.
- Code intermédiaire. Portée des variables. Environnements.
- Optimisation de code. Durée de vie, allocation de registres.
- Sélection d'instructions, génération de code.
- Analyses statiques des programmes.

Outils

– Langage de programmation : Ocaml
– Simulateur de la machine MIPS R2000 : Spim
(développé à l'université de Wisconsin, USA)

Projet de programmation

Slide 4

– Écrire un petit compilateur complet pour un petit langage,
– Approfondir un bout de la chaîne de compilation.

Quelques exemples des années passées

– Ajout d'un gestionnaire mémoire ;
– Ré-implémentation de l'allocation de registres ;
– Extensions du langage source ;
– Compilation vers les processeurs x86.

Code produit pour fact en cours...

Slide 5

```
      fact_f =8
fact:
      subu $sp, $sp, fact_f
      sw   $s0, 0($sp)
      sw   $ra, 4($sp)
      li   $v0, 1
      ble  $a0, $v0, L8
      move $s0, $a0
      subu $a0, $a0, 1
      jal  fact
      mul  $v0, $s0, $v0
      j    L10
L8:
      li   $v0, 1
L10:
      lw   $s0, 0($sp)
      lw   $ra, 4($sp)
      addu $sp, $sp, fact_f
      j    $ra
```