编译器速成指南

2019年3月

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

编译器速成指南

张凯羿

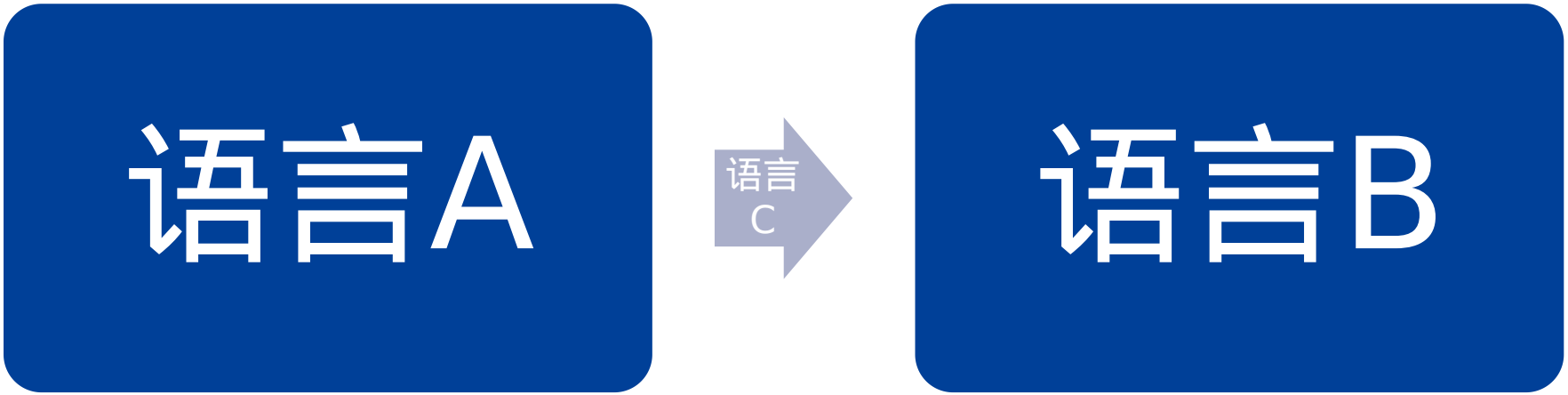
2019年3月

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

为什么我们要写编译器

- 1.锻炼能力
  - A.代码能力
  - B.学习能力
  - ~~C.抱大腿能力~~
- 2.能力证明

什么是编译器



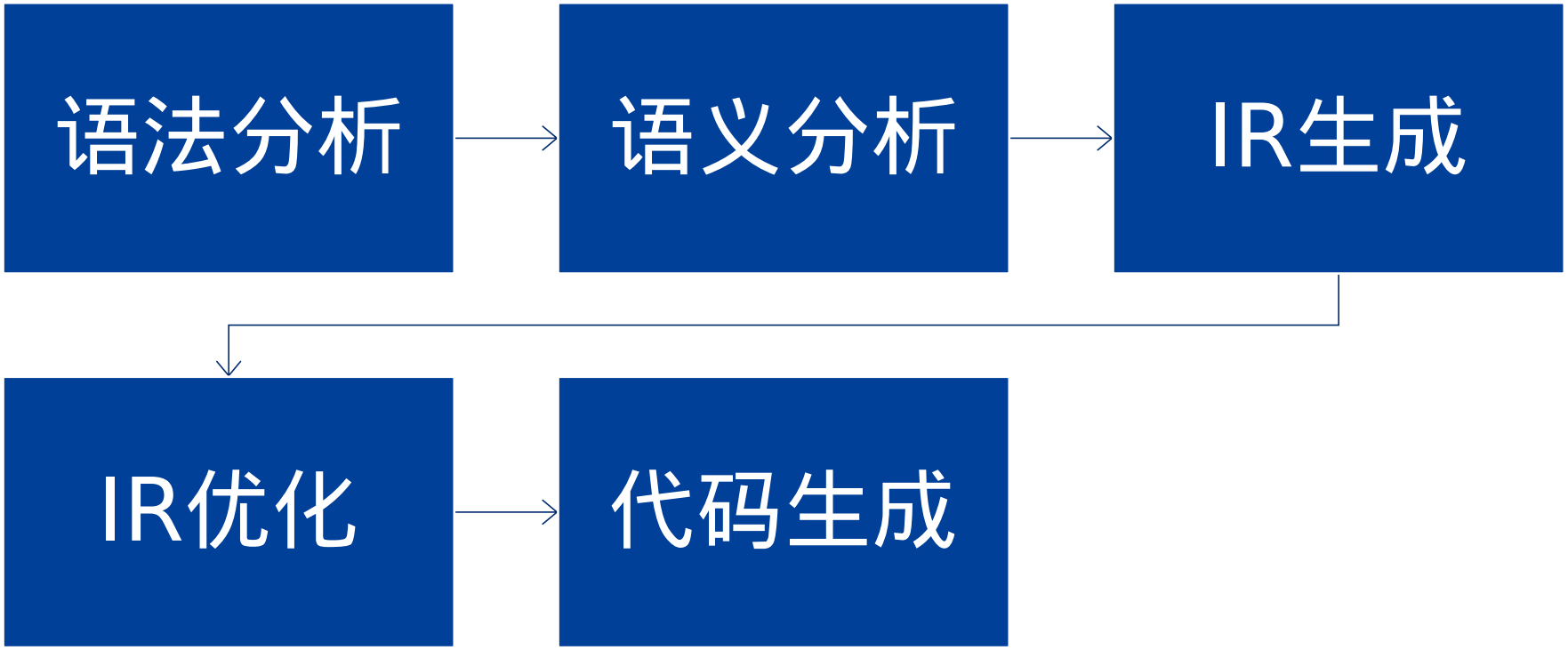
什么是编译器

Mx*

Java

X64汇编

编译器流程



语法分析(1)-Lexing

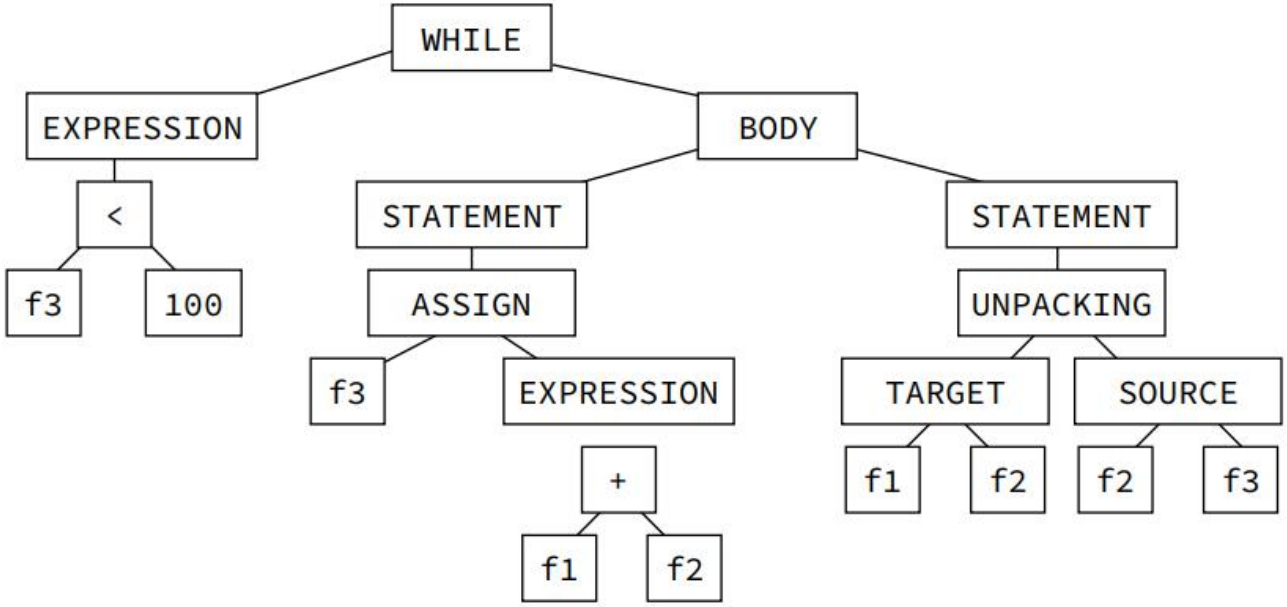
```
while f3 < 100 {
   f3 = f1 + f2;
   f1, f2 = f2, f3;
}
```

```
KEYWORD     while
IDENTIFIER  f3
SYMBOL      <
LITERAL     100
SYMBOL      {
IDENTIFIER  f3
SYMBOL      =
IDENTIFIER  f1
SYMBOL      +
IDENTIFIER  f2
SYMBOL      ;
IDENTIFIER  f1
SYMBOL      ,
IDENTIFIER  f2
SYMBOL      =
IDENTIFIER  f2
SYMBOL      ,
IDENTIFIER  f3
SYMBOL      ;
SYMBOL      }
```

语法分析(2)-Parsing

```
while f3 < 100 {
  f3 = f1 + f2;
  f1, f2 = f2, f3;
}
```

```
KEYWORD     while
IDENTIFIER  f3
SYMBOL      <
LITERAL     100
SYMBOL      {
IDENTIFIER  f3
SYMBOL      =
IDENTIFIER  f1
SYMBOL      +
IDENTIFIER  f2
SYMBOL      ;
IDENTIFIER  f1
SYMBOL      ,
IDENTIFIER  f2
SYMBOL      =
IDENTIFIER  f2
SYMBOL      ,
IDENTIFIER  f3
SYMBOL      ;
SYMBOL      }
```



速成指南-语法

- 推荐
- 使用ANTLR
- 资料
- ANTLR官网
- ANTLR 4权威指南

- 其他
- 1.自己手写
- 2.Lex + Yacc
- 3.Flex + Bison

语义分析

```
int main(){
        int x=0;
        x=x+‘hello‘;
        return 0;
}
int main(){
        int x=0;
        x=x+y;
        return 0;
}
```

- CE!
- 在语义层面上出现问题

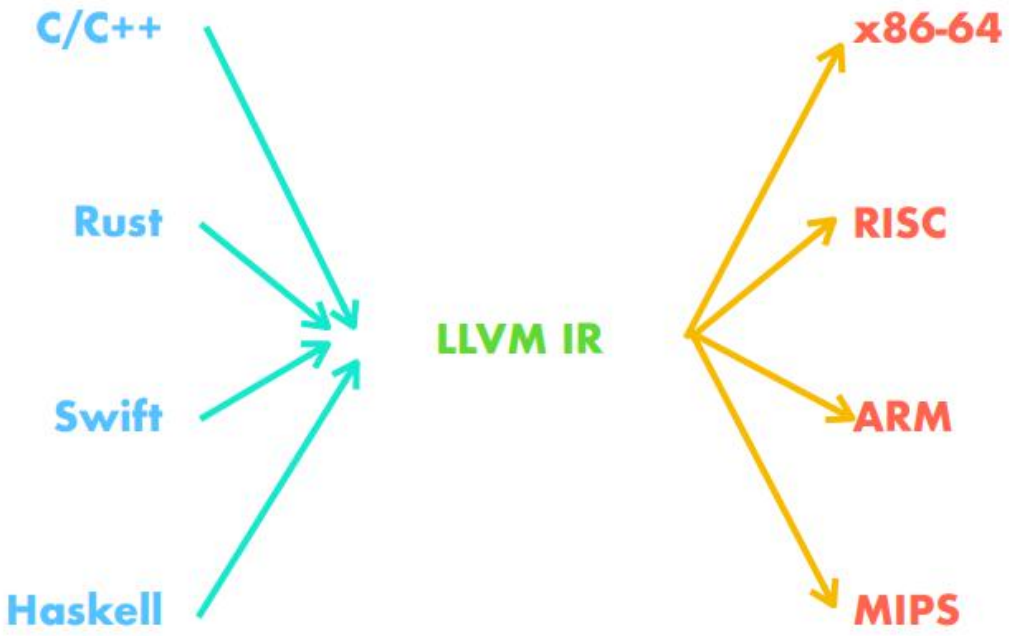
速成指南-语义

- 推荐
- 使用符号表
- 建立抽象语法树(AST)
- 学习学长代码
- 陈乐群
- 郑怜悯
- 丁尧尧

- 其他
- 1.不建立抽象语法树
- 2.自定义语法糖

中间代码生成-为什么我们需要IR？

- 1.简化源语言
- 2.更贴近目标语言
- 3.易于优化
- 4.减少针对不同平台的工作量*



中间代码生成

x=1+y*z;

- (move,v1,y,_)
- (move,v2,z,_)
- (mul,v3,v1,v2)
- (move,v4,1,_)
- (add,v5,v3,v4)
- (move,x,v5,_)

速成指南-中间代码生成

- 推荐
- 1.使用LLVM，或者
- 2.使用线性IR+CFG
- 资料
- 编译器设计
- 现代编译器的Java实现(第2版)

- 其他
- 1.不使用IR(难以优化)
- 2.树IR
- 3.图IR

速成指南-优化

- 推荐
- 寄存器分配!
- 寄存器分配!
- 寄存器分配!
- 良好的寄存器分配可以使你通过几乎所有数据
- 资料
- 编译器设计

- 其他
- 1.常量折叠
- 2.SSA
- 3.无关代码消除
- 4.函数内联
- 5. …

速成指南-代码生成

- 推荐
- 学习NASM，注意NASM和其他x86-64汇编稍有不同
- 使用libc减少工作量
- 使用c2nasm完成内建函数
- 资料
- 资料
- 其他
- 也没啥其他的了

速成指南-总结

- 1.使用ANTLR
- 2.建立抽象语法树和符号表
- 3.使用常见的IR
- 4.做性价比高的优化
- 5.使用libc和c2nasm
- 重点:
- 1.务必熟读手册，不要做无用功
- 2.设计好，再写代码，不要因设计失误反复推倒重来
- 3.有问题问同学、助教和搜索引擎

DDL

- 中期检查于3月30、31日举行
- 中期检查需在OJ上通过semantic测试，写report，还有code review
- 通过semantic的同学可以添加数据
- 中期检查前X天暂停数据添加，X待定
- DDL在5月初的某个周末
- 全部通过的同学可以添加数据
- DDL前Y天暂停数据添加，Y待定
- 得分将会以GCC-O0,O1,O2为基准测定
- DDL后第i天的提交会扣除$\frac{i(i+1)}{2}$的分数

Q&A