The Compiler Of Language MFChen

Bai YiWei

May 10, 2016

词法分析和语法分析

- 使用 Antlr4 来完成词法分析和语法分析
- 处理运算符优先级方法如下 (~~未充分利用 Antlr4 特性~~)

```
expr            ::= logical_and_expr
log_and_expr    ::= log_and_expr && log_and_expr
                  | relational_expr
relation_expr   ::= relation_expr < relation_expr
                  | additive_expr
additive_expr   ::= additive_expr + additive_expr
                  | ...
```

符号表

- 基本思路是虎书的符号表（双链表实现）
- 对于每一个 Symbol，需要存 $pair < scope, Property >$

~~AST~~CST

- Antlr4 的 Visitor 下， CST 跟 AST 没有本质差别
- 非常遗憾没能借助 AST 练习下设计模式的一些东西 T-T

语义分析

扫微小的三遍

- 处理出所有可能的 Type (用户定义的 Class)
- 处理出函数的性质和 Class 性质
- 检查程序语义

中间表示

- 采用 Eac ILOC 规范的三地址码
- 既然有了无限寄存器，那么要 load store 指令有和用？
- 魔改了一下，对虚拟寄存器进行了分类 $\Rightarrow$ 大锅

代码生成

- 先写一个 Memory to Memory 内存模型的程序，简单、易查错
- 程序架构完全靠自己思考，后期转换困难

内建函数实现

- 全部自己手写 MIPS 实现，采用内联方式
- 感受到了汇编语言不一样的思维

- 为什么不能拿寄存器当缓存来实现一下 Cache 呢？
- 先对每个基本块按照使用频率分配寄存器，当不够的时候踢掉前面那个使用频率最低的

魔改的代价

- 调不对 T-T

Memory To Memory 相关优化

- 利用寄存器多的用不了，优化 MIPS 伪指令 e.g *li*
- 分析每一个基本快的无用 load、store 进行删除

PrettyPrint

- 在 CST 上走一遍即可实现，利用 scope 处理缩进问题

丑陋的代码

```
class test{
     int a;
    int b;
        int d;
}

int f(int a, int    b) {
    int dd = a +b+ 1;
    return dd<< 2;
}

int main() {
    int a;
     int b;
    int c;
int d= 2;
int i;
 int sum;
    for (i= 0;i <10;    i++) {
      sum =sum+ i;
    }
    while(sum<= 10) {
     sum++;
    }
    if (sum >9) {
            sum = sum %10;
    }
    else
    sum = sum*10;
    return f(sum,sum+1);
```

PrettyPrint 后的代码

```
class test{
    int a;
    int b;
    int d;
}
int f(int a, int b)
{
    int dd = a + b + 1;
    return dd >> 2;
}
int main()
{
    int a;
    int b;
    int c;
    int d = 2;
    int i;
    int sum;
    for (i = 0; i < 10; i++)
    {
        sum = sum + i;
    }
    while(sum <= 10)
    {
        sum++;
    }
    if (sum > 9)
    {
        sum = sum % 10;
    }
    else
        sum = sum * 10;
    return f(sum, sum + 1);
}
```

感想

- 感谢马老师的课程
- 感谢助教们的努力工作和对我们的帮助