编译原理第五次实验报告

201220069 周心同

实验介绍

在词法分析、语法分析、语义分析和中间代码生成程序 的基础上，使用数据流分析算法等消除效率低下和无法被轻易转化为机器代码的中间代 码，从而将C—源代码翻译成的中间代码转化为语义等价但是更加简洁高效的版本。

编译方式

实验环境：与实验要求相同。

在Code目录下make即可。

实验内容

本次实验使用了hby的框架代码。框架代码中的IR输入，基本块和控制流图已经在框架中完成，输入使用了实验一中lexical和bison结合的方法，主要空的内容为中间代码优化。

中间代码优化

框架中包括常量传播，可用表达式分析，复制传播，活跃变量分析（无用代码消除）这四个数据流分析的算法，这四个算法都属于龙书上给出的数据流分析算法。迭代方式我采用工作表算法，用一个集合储存下一次遍历会发生变化的基块，这样，已经达到不动点的基块就可以不用重复遍历了。这个算法本质上是图的广度优先遍历算法的变体，只是加入了一些剪枝的逻辑，每一轮只遍历可能会发生变化的结点，不发生变化的结点提前从遍历逻辑中去除。

循环不变式

此时OJ能有98分，再做提升就需要实现循环不变式和强度消减，因为性价比不高在尝试写过后放弃了（而且特别多逻辑判断 很难不出bug），结合实验指导书简短说一下我的思路。

1.找出支配结点

	支配结点
域	The power set of N
方向	Forwards
传递函数	$f_B(x) = x \cup \{B\}$
边界条件	$\text{OUT}[\text{ENTRY}] = \{\text{ENTRY}\}$
交汇运算($\wedge$)	$\cap$
方程式	$\text{OUT}[B] = f_B(\text{IN}[B])$ $\text{IN}[B] = \bigwedge_{P,pred(B)} \text{OUT}[P]$
初始化设置	$\text{OUT}[B] = N$

2.找出循环 由支配节点找出回边n->d，将两点添加进循环L中，若n!=d且n的父节点m不在L中，将m添加L中，重复该过程直至没有新节点添加进来

3.找出循环不变语句： 初始化标记下列语句：语句的运算分量为常数| 循环外定值

迭代进行标记：语句的运算分量为常数| 循环外定值| 已经标记为"不变"的到达定值（且只有一个到达定值）

重复该过程直至没有新的“不变”语句

4.循环不变式外提：首先判断上述循环不变式s: x = y + z是否满足：该语句必须是所有基本块出口结点的支配结点（即每个循环必定执行该语句），没有其他语句对x赋值，对x的引用仅由s到达（即循环中使用x的时候必须在s语句之后）。

对于满足上述条件的循环不变式，按照循环不变找出的次序提到外面（因为前面是迭代标记，所以要先找到已经“不变”的提出，才能提出后不变的）

5.外提算法：在循环前面加一个新块，将不变式提到该块内，前面进入循环的边改为先进入该块，再由该块进入循环。

强度消减

1.找出基础归纳变量：该语句必须是所有基本块出口结点的支配结点（即每个循环必定执行该语句），且不存在其他的赋值语句。

2.迭代找出归纳变量的家族：不存在循环外部赋值影响且必须被基础归纳变量支配。

3.恒等替换。

遇到的问题和解决方法

框架代码理解起来比较困难，我是一方面看了readme和一些软件分析的内容，同时对代码中的函数调用关系有了大致了解，清楚宏的用法，另一方面自己构造样例，注释部分算法来看整个代码的执行过程。同时框架代码的每个todo旁几乎都有提示，十分善解人意。

实验建议和感想

十分感谢hby的框架代码，地址在下面。

NJUCS-Compiler OJ (pascal-lab.net)