# Lab 2: Dataflow and optimizations

小组成员：张崴，刘梦馨

1. **实验完成目标：**
   1. Build Static Single Assignment (SSA) form.
   2. Perform SSA based constant propagation.
   3. Translate SSA back to non-SSA 3-address code.
2. **程序使用方法**

执行complish.sh完成程序编译。run.sh后加参数执行相应程序，-backend后接ssa，3addr,rep分别表示生成ssa代码，3addr代码和优化报告；-opt 后接ssa,scp表示运行简单常量传播。用重定向符’<’ 将指定程序3addr代码交给相应程序执行。

1. **程序设计思想**
2. 生成SSA
3. 生成Dominance Frontier
   1. 查找每个节点的父节点，将每个节点的父节点存入cfg\_par中，cfg\_par节点的数据结构为map<int,vector<int>>
   2. 针对每个节点存储的父节点数组中寻找每个节点的immediate dominator，初始化cfg\_idom=cfg\_par，将每个节点的父节点数组排序，将父节点数组中标号大于当前节点的父节点删除。当父节点数组中所有节点不相等时，寻找数组中最大的元素的父节点中最小的一个，替换原来的节点，将数组重新排序，直到数组中所有元素相等，元素值即为当前节点的immediate dominator。
   3. 对于每个节点如果有多于一个的父节点，对每个父节点查找该节点的cfg\_idom是否为其某个父节点，如果不是将该节点加入其父节点的df中，递归加入dominate tree其他父节点的df中。
4. 插入φ-nodes
   1. 对于每个变量的每次赋值，将节点加入worklist
   2. 当worklist不为空，取出一个节点查找df集合，如果df集合中的节点不在hasalready集合中，插入φ节点，hasalready中插入该节点，如果节点不在everonworklist集合中，将该节点加入everonworklist和worklist中
5. Rename变量
   1. 如果该节点被访问过，返回
   2. 对每个φ节点，从counter中提取编号，给φ节点左部变量变量更名，将编号入栈，counter自增
   3. 对节点中每个表达式如果表达式右部存在变量，从栈中提取编号，对变量更名；如果变大时左部存在变量名，从counter中提取编号给右部变量更名，将编号入栈，counter自增
   4. 对每个后继结点中的φ节点，提取栈中的编号，将相应位置的变量名更名
   5. 递归对每个后继结点执行rename操作
   6. 将编号出栈
6. 常量传播
7. 对每个变量的每次赋值，如果赋值为常数，将所有出现用到该变量的地方替换为该常量
8. 针对每个φ函数，观察φ函数中可选的值是否相同如果相同常量，将赋值替换为该常量
9. 重复1）和2）直到不再发生变化
10. 将SSA转换为non-SSA三地址格式

由于只对局部long型变量进行了SSA操作，没有对全局变量和数组变量进行操作，lab1中的翻译程序对局部的long型变量可以不考虑后面的地址码进行翻译，所以在由SSA转换为non-SSA三地址格式中不对变量的地址进行还原，不影响后续的操作。只需要遍历扫描所有修改后的三地址代码，将含有φ语句的指令设为nop，如果该语句下一条语句为move语句，且源地址和目标地址分别为φ指令和对应的变量，则把下一条指令也置为nop