编译系统

第八章
代码优化

哈尔滨工业大学 陈鄞

第17、18讲（代码优化_2、3）要点

- 数据流分析技术及其主要应用
  - 到达定值
  - 活跃变量
  - 可用表达式
  - 支配结点

数据流分析的主要应用

- 到达定值　∃
  - 定义：如果存在一条从紧跟在x的定值d后面的点到达某一程序点p的路径，而且在此路径上d没有被“杀死”(如果在此路径上有对变量x的其它定值d'，则称定值d被定值d'“杀死”了)，则称定值d到达程序点p
  - 用来解决的主要问题
    - 程序点p处使用的x的值可能是在哪里定值的
      - 循环不变计算的检测
      - 常量合并
      - 变量未经定值就被引用
      - ud链 (引用-定值链)

数据流分析的主要应用

- 到达定值 ∃ 正向

$OUT[ENTRY] = \Phi$	$IN[B] = \cup_{P是B的一个前驱} OUT[P]$
$OUT[B] = \Phi$	$OUT[B] = gen_B \cup (IN[B]-kill_B)$

- 活跃变量 ∃
  - 对于变量x和程序点p，如果在流图中沿着从p开始的某条路径会引用变量x在p点的值，则称变量x在点p是活跃(*live*)的，否则称变量x在点p不活跃(*dead*)
  - 用来解决的主要问题
    - x在程序点p处的定值在哪里会被使用
      - 删除无用赋值
      - 寄存器分配
      - *du*链 (定值-引用链)

数据流分析的主要应用

- 到达定值 ∃ 正向
- 活跃变量 ∃ 逆向

$OUT[ENTRY] = \Phi$ $OUT[B] = \Phi$	$IN[B]=\cup_{P是B的一个前驱} OUT[P]$ $OUT[B] = gen_B \cup (IN[B]-kill_B)$
$IN[EXIT] = \Phi$ $IN[B] = \Phi$	$OUT[B] = \cup_{S是B的一个后继} IN[S]$ $IN[B] = use_B \cup (OUT[B] - def_B)$

- 可用表达式 ∀
  - 如果从流图的首节点到达程序点 p 的每条路径都对表达式 x op y 进行计算，并且从最后一个这样的计算到点 p 之间没有再次对 x 或 y 定值，那么表达式 x op y 在点 p 是可用的(*available*)
  - 主要用途
    - 消除全局公共子表达式
    - 复制传播

数据流分析的主要应用

- 到达定值　∃　正向
- 活跃变量　∃　逆向
- 可用表达式　∀　正向
- 支配结点　∀

$OUT[ENTRY] = \Phi$ $OUT[B] = \Phi$	$IN[B] = \cup_{P\text{是}B\text{的一个前驱}} OUT[P]$ $OUT[B] = gen_B \cup (IN[B] - kill_B)$
$IN[EXIT] = \Phi$ $IN[B] = \Phi$	$OUT[B] = \cup_{S\text{是}B\text{的一个后继}} IN[S]$ $IN[B] = use_B \cup (OUT[B] - def_B)$
$OUT[ENTRY] = \Phi$; $OUT[B] = U$;	$IN[B] = \cap_{P\text{是}B\text{的一个前驱}} OUT[P]$ $OUT[B] = e\_gen_B \cup (IN[B] - e\_kill_B)$

- 如果从流图的入口结点到结点n的每条路径都经过结点d，则称结点d支配(*dominate*)结点n，记为$d\ dom\ n$

数据流分析的主要应用

➢ 到达定值	∃	正向	$OUT[ENTRY] = \Phi$ $OUT[B] = \Phi$	$IN[B] = \cup_{P是B的一个前驱} OUT[P]$ $OUT[B] = gen_B \cup (IN[B]-kill_B)$
➢ 活跃变量	∃	逆向	$IN[EXIT] = \Phi$ $IN[B] = \Phi$	$OUT[B] = \cup_{S是B的一个后继} IN[S]$ $IN[B] = use_B \cup (OUT[B] - def_B)$
➢ 可用表达式	∀	正向	$OUT[ENTRY] = \Phi;$ $OUT[B] = U;$	$IN[B] = \cap_{P是B的一个前驱} OUT[P]$ $OUT[B] = e\_gen_B \cup (IN[B] - e\_kill_B)$
➢ 支配结点	∀	正向	$OUT[ENTRY] = \{ENTRY\}$ $OUT[B] = N$	$IN[B] = \cap_{P是B的一个前驱} OUT[P]$ $OUT[B] = IN[B] \cup \{B\}$

结束