

编译原理 - 作业(5) : 代码生成与优化

截至时间: 2022.6.19/周一 23:59:59

提交方式: 超算习堂 (<https://easyhpc.net/course/144>)

Q1: (p408, Exercises 6.6.1) Add rules to the syntax-directed definition of Fig. 6.36 for the following control-flow constructs:

- A repeat-statement **repeat** S **while** B .
- A for-loop **for** $(S_1; B; S_2) S_3$.

PRODUCTION	SEMANTIC RULES
$P \rightarrow S$	$S.next = newlabel()$ $P.code = S.code \parallel label(S.next)$
$S \rightarrow \text{assign}$	$S.code = \text{assign.code}$
$S \rightarrow \text{if} (B) S_1$	$B.true = newlabel()$ $B.false = S_1.next = S.next$ $S.code = B.code \parallel label(B.true) \parallel S_1.code$
$S \rightarrow \text{if} (B) S_1 \text{ else } S_2$	$B.true = newlabel()$ $B.false = newlabel()$ $S_1.next = S_2.next = S.next$ $S.code = B.code$ $\parallel label(B.true) \parallel S_1.code$ $\parallel gen('goto' S.next)$ $\parallel label(B.false) \parallel S_2.code$
$S \rightarrow \text{while} (B) S_1$	$begin = newlabel()$ $B.true = newlabel()$ $B.false = S.next$ $S_1.next = begin$ $S.code = label(begin) \parallel B.code$ $\parallel label(B.true) \parallel S_1.code$ $\parallel gen('goto' begin)$
$S \rightarrow S_1 S_2$	$S_1.next = newlabel()$ $S_2.next = S.next$ $S.code = S_1.code \parallel label(S_1.next) \parallel S_2.code$

Figure 6.36: Syntax-directed definition for flow-of-control statements.

Q2: (p541, Exercises 8.5.1&2) For the basic block

$d = b * c$

$e = a + b$

$b = b * c$

$a = e - d$

- Construct the DAG of the block.
- Simplify the three-address code of the block, assuming
 - Only a is live on exit from the block.
 - a , b , and c are live on exit from the block.

Q3: For the code segment below:

```
1:    x = 0
2:    y = 0
3:  L0: if n / 2 goto L1
4:    x = x + n
5:    y = y + 1
6:    goto L2
7:  L1: y = y + n
8:    c = 4 / 2
9:    t1 = x * c
10:   t2 = c - 1
11:   x = x + t2
12: L2: n = n - 1
13:   if n > 0 goto L0
14:   return x
```

- a. Partition the code segment into basic blocks.
- b. Construct the control flow graph.
- c. For lines 7-11, list two optimization techniques.
- d. Suppose the whole segment is from a function '*int Func(int n)*', where *n* is the argument, *x* and *y* are local variables, then how to retrieve *n*, *x* and *y* when *Func()* is called in the final target code? Hint: consider *\$fp* and *\$sp*.