言語処理工学 A　期末テスト

2017 年 2 月 3 日

井上克郎

ノート教科書持ち込みなし。[1]は解答用紙表紙、[2]は折りたたんだ内側 2 枚に、［3］は裏表紙に解答を書くこと。間違った場所に書いた場合は減点する。

［1］次のコードは、教科書の C のサンプルコンパイラの while 文の処理部分である。①～⑤を適当なものを補って完成させよ。なお while 文の構文は 'while (***exp***) ***st*** ' で、exp は条件式、st は文(statement)を表す非終端記号である。（1 ページめに書くこと）

```
void while-st(){
      int lab1, lab2;
      if(tok != SLPAREN) error(" '(' Expected");
      tok =   ①  ;
      lab1 = get-inlabel();
      gen-code("②", ③);
      exp();
      lab2 = get-inlabel();
      gen-code(" TST.W R0");  // check accumulator R0
      gen-code(" BEQ  ④", ⑤); // Exit if not true
      if(tok != SRPAREN) error(" ')' Expected);
      tok = scan();
      st(lab2, lab1)  ;
      gen-code(" BRA  ④", ⑥); // Go back to loop top
      gen-code("②", ⑦); //Exit point
}
```

問題は裏面に続く

[2] 次の 3 番地コードに関して答えよ。(解答用紙２－３ページに答えを書くこと)

① a = 1
② b = 0
③ L1: b = a + b
④ if a > 10 goto L2
⑤ a = a + 1
⑥ goto L1
⑦ L2: a = 1
⑧ if b < 100 goto L1
⑨ print a, b

(注:print は引数を出力するマクロで、引数を参照するのみで変更しない)

(1) このプログラムの基本ブロックに分け、フローグラフを書け。プログラムの上から順に、各ブロックに番号 B1～をつけ、各ブロックに入るコードを行番号で明示せよ。

(2) 得られたフローグラフの支配木（dominator tree）を書け。

(3) フローグラフ中の各バックエッジを挙げ、それぞれが構成するループの各頂点を示せ。

(4) 各基本ブロックの Gen 集合、Kill 集合を求めよ。

(5) 各基本ブロックの入り口（IN)、出口（OUT）で出現しうる変数定義の行番号の集合（データフロー方程式の解）を示せ。データフロー方程式を立てて、それを漸近的に解く途中経過も示せ。

[3] ネストを許す参照スコープを持つ手続き型言語（例えば Pascal など）の実行を行うために実行フレームにはどのような情報を格納する必要があるか。必要な情報の名前とそれが必要な理由をそれぞれについて簡単に述べよ。(4 ページに書くこと)

解答

[1] 次のコードは、教科書の C のサンプルコンパイラの while 文の処理部分である。①～⑤を適当なものを補って完成させよ。なお while 文の構文は 'while (***exp***) ***st*** ' で、exp は条件式、st は文(statement)を表す非終端記号である。

```
void while-st(){
      int lab1, lab2;

      if(tok != SLPAREN) error(" '(' Expected");
      tok =  ① scan()  ;
      lab1 = get-inlabel();
      gen-code("②L%d:", ③lab1);
      exp();
      lab2 = get-inlabel();
      gen-code(" TST.W R0");  // check accumulator R0
      gen-code(" BEQ  ④L%d", ⑤lab2); // Exit if not true
      if(tok != SRPAREN) error(" ')' Expected);
      tok = scan();
      st(lab2, lab1)  ;
      gen-code(" BRA  ④L%d", ⑥lab1); // Go back to loop
top
      gen-code("②L%d:", ⑦lab2); //Exit point
}
```

[3]教科書参照

[2]

L: リーダー

(1)

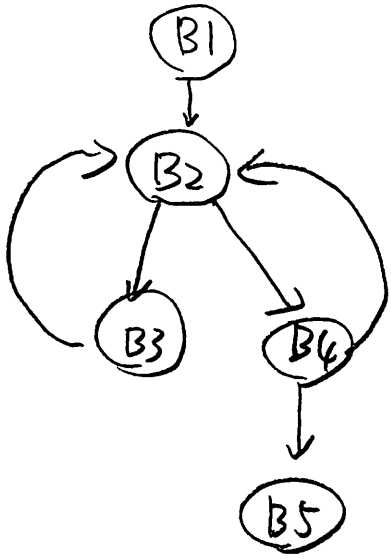
L ① a = 1
② b = 0 — B1

L ③ L1: b = a + b
④ if a > 10 goto L2 — B2

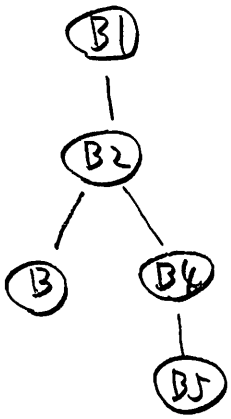
L ⑤ a = a + 1
⑥ goto L1 — B3

L ⑦ L2: a = 1
⑧ if b < 100 goto L1 — B4

L ⑨ print a, b — B5



(2)



(3) バックエッジ ループ

B3 → B2 {B2, B3}

B4 → B2 {B2, B4}

(4)

	GEN	KILL
B1	{①,②}	{③,⑤,⑦}
B2	{③}	{②}
B3	{⑤}	{①,⑦}
B4	{⑦}	{①,⑤}
B5	∅	∅

(5)

	step ∅ IN	step ∅ OUT	step 1 IN	step 1 OUT	STEP 2 IN	STEP 2 OUT	STEP 3 IN	STEP 3 OUT
B1	∅	{①,②}	∅	{①,②}	∅	{①,②}	∅	{①,②}
B2	∅	{③}	{①,②,⑤,⑦}	{①,③,⑤,⑦}	{①,②,③,⑤,⑦}	{①,③,⑤,⑦}	{①,②,③,⑤,⑦}	{①,③,⑤,⑦}
B3	∅	{⑤}	{③}	{③,⑤}	{①,③,⑦}	{③,⑤}	{①,③,⑦}	{③,⑤}
B4	∅	{⑦}	{①}	{②,⑦}	{①,③,⑤,⑦}	{③,⑦}	{①,③,⑤,⑦}	{③,⑦}
B5	∅	∅	{①}	{①}	{③,⑦}	{③,⑦}	{③,⑦}	{③,⑦}

同じなので step 3 で停止