Programmierung

Aufgabe 1 (AGS 15.17 b, 15.1 a)

(a) Gegeben sei folgender AM_1-Code:

```
1:  INIT 1;            10:  MUL;               19:  READ(global,1);
2:  CALL 18;           11:  STOREI(-3);        20:  LOADA(global,1);
3:  INIT 0;            12:  LOAD(lokal,-2);    21:  PUSH;
4:  LOAD(lokal,-2);    13:  LIT 1;             22:  LOAD(global,1);
5:  LIT 0;             14:  SUB;               23:  PUSH;
6:  GT;                15:  STORE(lokal,-2);   24:  CALL 3;
7:  JMC 17;            16:  JMP 4;             25:  WRITE(global,1);
8:  LIT 2;             17:  RET 2;             26:  JMP 0;
9:  LOADI(-3);         18:  INIT 0;
```

Dokumentieren Sie 12 Schritte der AM_1 mit der Startkonfiguration $\sigma = (22, \varepsilon, 1 : 3 : 0 : 1, 3, \varepsilon, \varepsilon)$.

(b) Gegeben sei folgendes C_1-Programm:

```
1  #include <stdio.h>
2  int b;
3  void f(int a, int *b) {
4    int c;
5    c = a;
6    while (c > 0) {
7      *b = *b * 2;
8      c = c - 1;
9    }
10 }
11
12 void main() {
13   int a;
14   scanf("%i", &a);
15   b = 1;
16   f(a, &b);
17   printf("%d", b);
18 }
```

Übersetzen Sie das Programm mittels *trans* in ein AM_1-Programm mit Baumstrukturierten Adressen. Geben Sie zunächst die Symboltabellen $tab_{\texttt{main}}$ und $tab_{\texttt{f}}$ zur Übersetzung der Statements in den Funktionen `main` bzw. `f` mittels *stseqtrans* an. Geben Sie keine weiteren Zwischenschritte an.

Aufgabe 2 (AGS 15.16 a, AGS 15.18 b)

(a) Gegeben sei folgendes Fragment eines C_1-Programms:

```
1  #include <stdio.h>
2
3  int x, y;
4
5  void f(...) {...}
6  void main() {...}
7  void g(int a, int *b) {
8    int c;
9    c = 3;
10   if (c == *b)
11     while (a > 0) f(&a, b);
12 }
```

Übersetzen Sie die Sequenz der Statements im Rumpf von g in entsprechenden AM_1-Code mit baumstrukturierten Adressen (mittels *stseqtrans*). Sie brauchen keine Zwischenschritte anzugeben. Geben Sie zunächst die benötigte Symboltabelle tab_g an.

(b) Gegeben sei folgender AM_1-Code:

```
1:  INIT 1;
2:  CALL 13;
3:  INIT 0;
4:  LOADI(-2);
5:  LIT 2;
6:  GT;
7:  JMC 12;
8:  LOADI(-2);
9:  LIT 2;
10: DIV;
11: STOREI(-2);
12: RET 1;
13: INIT 0;
14: READ(global, 1);
15: LOADA(global, 1);
16: PUSH;
17: CALL 3;
18: WRITE(global, 1);
19: JMP 0;
```

Erstellen Sie ein Ablaufprotokoll der AM_1, indem Sie sie schrittweise ablaufen lassen, bis die Maschine terminiert. Die Anfangskonfiguration sei $(14, \varepsilon, 0 : 0 : 1, 3, 4, \varepsilon)$. Sie müssen nur Zellen ausfüllen, deren Wert sich im Vergleich zur letzten Zeile geändert hat.

Zusatzaufgabe 1 (AGS 15.28 $\star$)

(a) Gegeben sei folgendes Fragment eines C_1-Programms.

```
1  #include <stdio.h>
2
3  int a, b;
4
5  void g(int *x);
6
7  void f(...)  {...}
8  void main()  {...}
9  void g(int *x) {
10   int y;
11   while (b != 1) {
12     b = *x - 1;
13     g(&y);
14   }
15   f(y, x);
16 }
```

Übersetzen Sie die Statements im Rumpf von g (Zeilen 11–15) mittels *stseqtrans* in AM_1-Code mit baumstrukturierten Adressen. Geben Sie keine Zwischenschritte an. *Geben Sie zunächst die dazu benötigte Symboltabelle an.*

(b) Gegeben sei folgender AM_1-Code:

```
 1:  INIT 1;             8:  STORE(global,1);  15:  LOADA(global,1);
 2:  CALL 10;            9:  RET 2;            16:  PUSH;
 3:  JMP 0;             10:  INIT 1;           17:  CALL 4;
 4:  INIT 1;            11:  READ(lokal,1);    18:  WRITE(global,1);
 5:  LOAD(lokal,-3);    12:  READ(global,1);   19:  RET 0;
 6:  LOADI(-2);         13:  LOAD(lokal,1);
 7:  ADD;               14:  PUSH;
```

Führen Sie jede der drei AM_1-Konifgurationen um jeweils vier Schritte weiter:

- $(12, \varepsilon, 0 : 3 : 0 : 7, 3, 5, \varepsilon)$,
- $(17, \varepsilon, 5 : 3 : 0 : 7 : 7 : 1, 3, \varepsilon, \varepsilon)$,
- $(8, 12, 5 : 3 : 0 : 7 : 7 : 1 : 18 : 3 : 0, 8, \varepsilon, \varepsilon)$.

Sie müssen nur Zellen ausfüllen, deren Wert sich im Vergleich zur vorherigen Zeile geändert hat.