Programmierung

10. Übungsblatt

Zeitraum: 29. Juni – 03. Juli 2015

Übung 1 (AGS 13.9)

(a) Geben Sie für das folgende C_0-Programm die bereits linearisierte Übersetzung an. Zwischenschritte der Übersetzung brauchen Sie nicht anzugeben. Schreiben Sie je Zeile nur einen Befehl.

```
#include <stdio.h>

int main() {
  int x, y;
  scanf("%i", &x);
  y = 0;
  while (x > 0) {
    x = x - 1;
    y = y + 1;
  }
  printf("%d", y);
  return 0;
}
```

(b) Folgende linearisierte Übersetzung $prog_0$ sei gegeben:

```
1: READ 2;     6: GT;       11: STORE 2;  16: JMP 4;
2: LIT 0;      7: JMC 17;   12: LOAD 1;   17: WRITE 1;
3: STORE 1;    8: LOAD 2;   13: LIT 1;
4: LOAD 2;     9: LIT  1;   14: ADD;
5: LIT 5;      10: SUB;     15: STORE 1;
```

Berechnen Sie $\mathcal{P}[\![prog_0]\!](0)$ durch Ablaufenlassen der AM_0.

Übung 2 (AGS 14.10)

(a) Folgendes Fragment eines C_1-Programms sei bekannt:

```
#include <stdio.h>

int a, b;

void h(...)  {...}

void g(int *x)
{ int y;
  while (b != 1) {b = *x - 1; h(&y);}
  f(y,x);
}

void f(...)  {...}
void main()  {...}
```

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 dazu benötigte Symboltabelle $tab_{\text{g+lDecl}}$ an.

(b) Gegeben sei folgender AM_1-Code:

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

Betrachten Sie nun die AM_1, die sich bereits im Zustand

$$\sigma = (12,\ \varepsilon,\ 0:3:0,\ 3,\ 9,\ \varepsilon)$$

befindet. Lassen Sie die AM_1, beginnend mit σ, auf dem oben gegebenen AM_1-Code solange ablaufen, bis die Maschine stoppt. Dokumentieren Sie den Zustand der AM_1 nach Ausführung jedes Befehls.

Übung 3 (AGS 14.15)

(a) Gegeben sei die Symboltabelle

$$tab_{\text{g+lDecl}} = [\text{g}/(\text{proc}, 1), \text{x}/(\text{var}, \text{global}, 1), \text{p}/(\text{var-ref}, -2), \text{i}/(\text{var}, \text{lokal}, 1)],$$

sowie folgendes Fragment aus der Funktion g des zugehörigen C_1-Programms:

```
if (x > 0) {
  g(&x);
  i = x / 2;
} else {
  x = *p;
}
printf("%d", x);
```

Übersetzen Sie dieses Fragment anhand $tab_{\text{g+lDecl}}$ in entsprechenden AM_1-Code mit baumstrukturierten Adressen (mittels *stseqtrans*). Nehmen Sie an, `if` sei das dritte Statement in g. Sie brauchen keine Zwischenschritte anzugeben.

(b) Gegeben sei folgender AM_1-Code:

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

Protokollieren Sie 16 Schritte des schrittweisen Ablaufes dieses Programms auf der AM_1, ausgehend vom Startzustand $\sigma = (15, \varepsilon, 0:0:3:0:0, 4, 1, \varepsilon)$.

Zusatzaufgabe 1 (AGS 14.3 ⋆)

(a) Übersetzen Sie nachfolgende C_1-Statements in entsprechenden AM_1-Code mit baumstrukturierten Adressen. Zwischenschritte brauchen Sie keine anzugeben.
Die zugehörige Symboltabelle ist: tab $= [f/(\text{proc}, 1), d/(\text{var}, \text{global}, 1), x/(\text{var}, \text{global}, 2)]$.

```
...
d = 4;
f(d, &x);
printf("%d", x);
...
```

(b) Gegeben sei folgender AM_1-Code:

```
 1:  INIT 1;              8:  STOREI(-2);          15:  READ(global, 1);
 2:  CALL 10;             9:  RET 1;               16:  LOADA(global, 1);
 3:  JMP 0;               10: INIT 2;              17:  PUSH;
 4:  INIT 0;              11: LIT 0;               18:  CALL 4;
 5:  LOAD(global, 1);     12: STORE(lokal, 1);     19:  WRITE(lokal, 1);
 6:  LIT 42;              13: LIT 4;               20:  RET 0;
 7:  SUB;                 14: STORE(lokal, 2);
```

Lassen Sie diesen auf der AM_1 ablaufen, bis der Befehlszähler den Wert 20 erreicht hat. Die AM_1 befindet sich bereits im Zustand

$$\sigma = (18,\ \varepsilon,\ 7:3:0:0:4:1,\ 3,\ \varepsilon,\ \varepsilon).$$

Dokumentieren Sie den Zustand der AM_1 nach Ausführung jedes Befehls.

(c) Stellen Sie den Aufbau des Laufzeitkellers für den Zustand σ graphisch dar.
Markieren Sie dabei insbesondere die Aktivierungsblöcke sowie den Referenzzeiger REF.

Zusatzaufgabe 2 (AGS 12.2.3 ⋆)

(a) Wenden Sie den Unifikationsalgorithmus auf die Terme t_1 und t_2 an, und ermitteln Sie deren allgemeinsten Unifikator:
$t_1 = \sigma(\sigma(\gamma(x_1), x_2), \gamma(\gamma(\alpha)))$
$t_2 = \sigma(\sigma(\gamma(\alpha), \gamma(\gamma(x_1))), \gamma(x_3))$
Geben Sie dabei jeweils die benutzte Regel an.

(b) Identifizieren Sie die Symbole σ, γ und α mit den Typkonstruktoren $(\,,\,)^2$, [ ] und `Int`.
Geben Sie nun zu t_1, t_2 aus (a) die äquivalenten Typterme und Typausdrücke an.