TECHNISCHE UNIVERSITÄT DRESDEN

PROGRAMMIERUNG

Übung 11: C_1 und AM_1

Eric Kunze
eric.kunze@mailbox.tu-dresden.de

TU Dresden, 28. Juni 2019

C_1 und AM_1

- **bisher:** Implementierung von C_0 mit AM_0
- **jetzt:** Erweiterung auf C_1 mit AM_1
  - Erweiterung um Funktionen ohne Rückgabewert
  - Einschränkungen von C_0 bleiben erhalten
- **Implementierung** durch
  - Syntax von C_1
  - Befehle und Semantik einer abstrakten Maschine AM_1
  - Übersetzer $C_1 \leftrightarrow AM_1$

Befehlssemantik der AM_1

$b \in \{\text{global}, \text{lokal}\}$
r . . . aktueller REF

$$adr(r, b, o) = \begin{cases} r + o & \text{wenn } b = \text{lokal} \\ o & \text{wenn } b = \text{global} \end{cases}$$

Befehl	**Auswirkungen**
LOAD(b,o)	Lädt den Inhalt von Adresse $adr(r, b, o)$ auf den Datenkeller, inkrementiere Befehlszähler
STORE(b,o)	Speichere oberstes Datenkellerelement an $adr(r, b, o)$, inkrementiere Befehlszähler
WRITE(b,o)	Schreibe Inhalt an Adresse $adr(r, b, o)$ auf das Ausgabeband, inkrementiere Befehlszähler
READ(b,o)	Lies oberstes Element vom Eingabeband, speichere an Adresse $adr(r, b, o)$, inkrementiere Befehlszähler

Befehlssemantik der AM_1

Befehl	Auswirkungen
LOADI(o)	Ermittle Wert (= b) an Adresse $r+o$, Lade Inhalt von Adresse b auf Datenkeller, inkrementiere Befehlszähler
STOREI(o)	Ermittle Wert (= b) an Adresse $r+o$, nimm oberstes Datenkellerelement, speichere dieses an Adresse b, inkrementiere Befehlszähler
WRITEI(o)	Ermittle Wert (= b) an Adresse $r+o$, schreibe den Inhalt an Adresse b auf Ausgabeband, inkrementiere Befehlszähler
READI(o)	Ermittle Wert (= b) an Adresse $r+o$, lies das oberste Element vom Eingabeband, speichere es an Adresse b , inkrementiere Befehlszähler
LOADA(b,o)	Lege $adr(r, b, o)$ auf Datenkeller, inkrementiere Befehlszähler
PUSH	oberstes Element vom Datenkeller auf Laufzeitkeller, Befehlszähler inkrementieren
CALL adr	Befehlszählerwert inkrementieren und auf LZK legen, Befehlszähler auf adr setzen, REF auf LZK legen, REF auf Länge des LZK ändern
INIT n	n-mal 0 auf den Laufzeitkeller legen
RET n	im LZK alles nach REF-Zeiger löschen, oberstes Element des LZK als REF setzen, oberstes Element des LZK als Befehlszähler setzen, n Elemente von LZK löschen

Aufgabe 1 – Teil (a)

```
while (*p > i) { f(p); i = i + 1; }
p = &i;
```

$$tab_{g+\text{lDecl}} = \{\texttt{f}/(\text{proc}, 1), \texttt{g}/(\text{proc}, 2), \texttt{i}/(\text{var}, \text{lokal}, 1), \texttt{p}/(\text{var-ref}, -2)\}$$

Lösung.

```
2.2.1  LOADI(-2) ; LOAD(lokal,1) ; GT ; JMC 2.2.2 ;
       LOAD(lokal,-2) ; PUSH ; CALL 1 ;
       LOAD(lokal,1) ; LIT 1 ; ADD ; STORE(lokal,1) ; JMP 2.2.1 ;

2.2.2  LOADA(lokal,1) ; STORE(lokal,-2) ;
```

Aufgabe 1 – Teil (b)

Gegeben ist folgender AM_1-Code:

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

Führen Sie 11 Schritte der AM_1 auf der Konfiguration $\sigma = (22, \varepsilon, 1 : 3 : 0 : 1, 3, \varepsilon, \varepsilon)$ aus.

Aufgabe 1 – Teil (b)

BZ	DK	LZK	REF	Inp	Out
22	ε	1:3:0:1	3	ε	ε
23	1	1:3:0:1	3	ε	ε
24	ε	1:3:0:1:1	3	ε	ε
3	ε	1:3:0:1:1:25:3	7	ε	ε
4	ε	1:3:0:1:1:25:3	7	ε	ε
5	1	1:3:0:1:1:25:3	7	ε	ε
6	0:1	1:3:0:1:1:25:3	7	ε	ε
7	1	1:3:0:1:1:25:3	7	ε	ε
8	ε	1:3:0:1:1:25:3	7	ε	ε
9	2	1:3:0:1:1:25:3	7	ε	ε
10	1:2	1:3:0:1:1:25:3	7	ε	ε
11	2	1:3:0:1:1:25:3	7	ε	ε
12	ε	2:3:0:1:1:25:3	7	ε	ε
13	1	2:3:0:1:1:25:3	7	ε	ε
14	1:1	2:3:0:1:1:25:3	7	ε	ε

Aufgabe 2

```
#include <stdio.h>
int x, y;

void f(...) ...

void g(int a, int *b) {
   int c;
   c = 3;
   if (c == *b) while (a > 0) f(&a, b);
}

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

Aufgabe 2

$$lokal - tab_{\mathrm{g}} = \text{[f/(proc,1) , g/(proc,2) , x/(var,global,1) , y/(var,global,2) ,}$$
$$\text{a/(var,lokal,–3) , b/(var-ref,–2) , c/(var,lokal,1)]}$$

Lösung.

```
            LIT 3 ; STORE(lokal,1);
            LOAD(lokal,1) ; LOADI(-2) ; EQ ; JMC 2.2.1 ;
 2.2.2.1:   LOAD(lokal,-3) ; LIT 0 ; GT ; JMC 2.2.2.2 ;
            LOADA(lokal,-3) ; PUSH ; LOAD(lokal,-2) ; PUSH ; CALL 1 ;
            JMP 2.2.2.1
 2.2.2.2:   2.2.1:
```

Aufgabe 3

Gegeben ist folgender AM_1-Code:

```
 1: INIT 1;
 2: CALL 1§;
 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;
```

Gesucht ist das Ablaufprotokoll der AM_1 mit der Anfangskonfiguration
$\sigma = (14, \varepsilon, 0 : 0 : 1, 3, 4, \varepsilon)$.

Aufgabe 3

BZ	DK	LZK	REF	Inp	Out
14	ε	0:0:1	3	4	ε
15	ε	4:0:1	3	ε	ε
16	1	4:0:1	3	ε	ε
17	ε	4:0:1:1	3	ε	ε
3	ε	4:0:1:1:18:3	6	ε	ε
4	ε	4:0:1:1:18:3	6	ε	ε
5	4	4:0:1:1:18:3	6	ε	ε
6	2:4	4:0:1:1:18:3	6	ε	ε
7	1	4:0:1:1:18:3	6	ε	ε
8	ε	4:0:1:1:18:3	6	ε	ε
9	4	4:0:1:1:18:3	6	ε	ε
10	2:4	4:0:1:1:18:3	6	ε	ε
11	2	4:0:1:1:18:3	6	ε	ε
12	ε	2:0:1:1:18:3	6	ε	ε
18	ε	2:0:1	3	ε	ε
19	ε	2:0:1	3	ε	2