Korrekte Software: Grundlagen und Methoden
Vorlesung 3 vom 18.04.16: Operationale Semantik

Serge Autexier, Christoph Lüth

Universität Bremen

Sommersemester 2016

Fahrplan

- Einführung
- Die Floyd-Hoare-Logik
- Operationale Semantik
- Denotationale Semantik
- Äquivalenz der Semantiken
- Verifikation: Vorwärts oder Rückwärts?
- Korrektheit des Hoare-Kalküls
- Einführung in Isabelle/HOL
- Weitere Datentypen: Strukturen und Felder
- Funktionen und Prozeduren
- Referenzen und Zeiger
- Frame Conditions & Modification Clauses
- Ausblick und Rückblick

Zutaten

```
// GGT(A,B)
if (a == 0) r= b;
else {
  while (b != 0) {
   if (a <= b)
      b = b - a;
   else a = a - b;
  }
  r = a;
}
```

- Programme berechnen Werte
- Basierend auf
  - Werte sind Variablen zugewiesen
  - Evaluation von Ausdrücken
- Folgt dem Programmablauf

Unsere Programmiersprache

Wir betrachten einen Ausschnitt der Programmiersprache C (C0).

Ausbaustufe 1 kennt folgende Konstrukte:

- Typen: **int**;
- Ausdrücke: Variablen, Literale (für ganze Zahlen), arithmetische Operatoren (für ganze Zahlen), Relationen (==, !=, <=, ...), boolsche Operatoren (&&, ||);
- Anweisungen:
  - Fallunterscheidung (**if**...**else**...), Iteration (**while**), Zuweisung, Blöcke;
  - Sequenzierung und leere Anweisung sind implizit

Semantik von C0

Systemzustände

- Ausdrücke werten zu Werten **Val** (hier ganze Zahlen) aus.
- Adressen **Loc** sind hier Programmvariablen (Namen)
- Ein Systemzustand bildet Adressen auf Werte ab: $\Sigma = \mathbf{Loc} \rightharpoonup \mathbf{Val}$
- Ein Programm bildet einen Anfangszustand möglicherweise auf einen Endzustand ab (wenn es terminiert).
- Zusicherungen sind Prädikate über dem Systemzustand.

C0: Ausdrücke und Anweisungen

Aexp $a ::= \mathbf{N} \mid \mathbf{Loc} \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 * a_2 \mid a_1/a_2$

Bexp $b ::= \mathbf{0} \mid \mathbf{1} \mid a_1 == a_2 \mid a_1 != a_2$
$\mid a_1 <= a_2 \mid !b \mid b_1 \,\&\&\, b2 \mid b_1 \,||\, b_2$

Exp $e := \mathbf{Aexp} \mid \mathbf{Bexp}$

Stmt $c ::=$ **Loc** = **Exp**;
$\mid$ **if** (b) c_1 **else** c_2
$\mid$ **while** (b) c
$\mid \{c^*\}$

Eine Handvoll Beispiele

```
// {y = Y ∧ y ≥ 0}
x= 1;
while (y != 0) {
 y= y-1;
 x= 2*x;
}
// {x = 2^Y}
```

```
// {a ≥ 0 ∧ b ≥ 0}
r= b;
q= 0;
while (b <= r) {
   r= r-y;
   q= q+1;
   }
// {a = b*q + r ∧ r < b}
```

```
p = 1;
c = 1;
while (c<=n) {
   c = c+1;
   p = p*c;
}
// {p = n!}
```

```
// {0 ≤ a}
t = 1;
s = 1;
i = 0;
while (s <= a) {
   t = t + 2;
   s = s + t;
   i = i + 1;
   }
// {i^2 ≤ a ∧ a < (i+1)^2}
```


Operationale Semantik: Arithmetische Ausdrücke

Ein arithmetischer Ausdruck a wertet unter gegebenen Zustand σ zu einer ganzen Zahl n (Wert) aus oder zu einem Fehler $\perp$.

- **Aexp** $a ::= \mathbf{N} \mid \mathbf{Loc} \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 * a_2 \mid a_1 \,/\, a_2$
- Zustände bilden Adressen/Programmvariablen auf **Werte** ab (σ)

$$\langle a, \sigma\rangle \rightarrow_{Aexp} n|\perp$$

Regeln

$$\langle n, \sigma\rangle \rightarrow_{Aexp} n$$

$$\frac{X \in \mathbf{Loc}, X \in Dom(\sigma), \sigma(X) = v}{\langle X, \sigma\rangle \rightarrow_{Aexp} v} \qquad \frac{X \in \mathbf{Loc}, X \notin Dom(\sigma)}{\langle X, \sigma\rangle \rightarrow_{Aexp} \perp}$$

$$\frac{\langle a_1, \sigma\rangle \rightarrow_{Aexp} n_1 \qquad \langle a_2, \sigma\rangle \rightarrow_{Aexp} n_2 \qquad n_i \in \mathbf{N}, n \text{ Summe } n_1 \text{ und } n_2}{\langle a_1 + a_2, \sigma\rangle \rightarrow_{Aexp} n}$$

$$\frac{\langle a_1, \sigma\rangle \rightarrow_{Aexp} n_1 \qquad \langle a_2, \sigma\rangle \rightarrow_{Aexp} n_2 \qquad \text{falls } n_1 = \perp \text{ oder } n_2 = \perp}{\langle a_1 + a_2, \sigma\rangle \rightarrow_{Aexp} \perp}$$

Operationale Semantik: Arithmetische Ausdrücke

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_i \in \mathbf{N}, n \text{ Differenz } n_1 \text{ und } n_2}{\langle a_1 - a_2,\sigma\rangle \to_{Aexp} n}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad \text{falls } n_1 = \bot \text{ oder } n_2 = \bot}{\langle a_1 - a_2,\sigma\rangle \to_{Aexp} \bot}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_i \in \mathbf{N}, n \text{ Produkt } n_1 \text{ und } n_2}{\langle a_1 * a_2,\sigma\rangle \to_{Aexp} n}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad \text{falls } n_1 = \bot \text{ oder } n_2 = \bot}{\langle a_1 * a_2,\sigma\rangle \to_{Aexp} \bot}$$

Operationale Semantik: Arithmetische Ausdrücke

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_i \in \mathbf{N}, n_2 \neq 0, n \text{ Quotient } n_1 \text{ und } n_2}{\langle a_1/a_2,\sigma\rangle \to_{Aexp} n}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad \text{falls } n_1 = \bot, n_2 = \bot \text{ oder } n_2 = 0}{\langle a_1 + a_2,\sigma\rangle \to_{Aexp} \bot}$$

Beispiel Ableitungen

Sei $\sigma(X) = 6, \sigma(Y) = 5$.

$$\frac{\dfrac{\langle X,\sigma\rangle \to_{Aexp} 6 \quad \langle Y,\sigma\rangle \to_{Aexp} 5}{\langle X+Y,\sigma\rangle \to_{Aexp} 11} \quad \dfrac{\langle X,\sigma\rangle \to_{Aexp} 6 \quad \langle Y,\sigma\rangle \to_{Aexp} 5}{\langle X-Y,\sigma\rangle \to_{Aexp} 1}}{\langle (X+Y)*(X-Y),\sigma\rangle \to_{Aexp} 11}$$

$$\frac{\dfrac{\langle X,\sigma\rangle \to_{Aexp} 6 \quad \langle X,\sigma\rangle \to_{Aexp} 6}{\langle X*X,\sigma\rangle \to_{Aexp} 36} \quad \dfrac{\langle Y,\sigma\rangle \to_{Aexp} 5 \quad \langle Y,\sigma\rangle \to_{Aexp} 5}{\langle Y*Y,\sigma\rangle \to_{Aexp} 25}}{\langle (X*X)-(Y*Y),\sigma\rangle \to_{Aexp} 11}$$

Operationale Semantik: Boolesche Ausdrücken

- **Bexp** $b ::= 0 \mid 1 \mid a_1 == a_2 \mid a_1 <= a_2 \mid !b \mid b_1 \&\& b2 \mid b_1 \mid\mid b_2$

Rules

$$\langle \mathbf{1},\sigma\rangle \to_{Bexp} \mathbf{1} \qquad \langle \mathbf{0},\sigma\rangle \to_{Bexp} \mathbf{0}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_i \neq \bot, n_1 \text{ und } n_2 \text{ gleich}}{\langle a_1 == a_2,\sigma\rangle \to_{Bexp} 1}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_i \neq \bot, n_1 \text{ und } n_2 \text{ ungleich}}{\langle a_1 == a_2,\sigma\rangle \to_{Bexp} 0}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_1 = \bot \text{ or } n_2 = \bot}{\langle a_1 == a_2,\sigma\rangle \to_{Bexp} \bot} \quad \langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_i \neq \bot, n_1 \text{ größer als } n_2}{\langle a_1 <= a_2,\sigma\rangle \to_{Bexp} 0}$$

$$\frac{\langle a_1,\sigma\rangle \to_{Aexp} n_1 \quad \langle a_2,\sigma\rangle \to_{Aexp} n_2 \quad n_1 = \bot \text{ or } n_2 = \bot}{\langle a_1 <= a_2,\sigma\rangle \to_{Bexp} \bot} \quad \frac{\langle b,\sigma\rangle \to_{Bexp} 1}{\langle !b,\sigma\rangle \to_{Bexp} 0} \quad \langle b,\sigma\rangle$$

$$\frac{\langle b_1,\sigma\rangle \to_{Bexp} t_1 \quad \langle b_2,\sigma\rangle \to_{Bexp} t_2}{\langle b_1 \,\&\&\, b_2,\sigma\rangle \to_{Bexp} t}$$

$$\frac{\langle b_1,\sigma\rangle \to_{Bexp} t_1 \quad \langle b_2,\sigma\rangle \to_{Bexp} t_2}{\langle b_1 \mid\mid b_2,\sigma\rangle \to_{Bexp} t}$$

Operationale Semantik: Anweisungen

- **Stmt** $c ::= \mathbf{Loc} = \mathbf{Exp};$ | $\{c^*\}$ | **if** (b) c_1 **else** c_2 | **while** (b) c

Regeln

$$\frac{\langle c,\sigma\rangle \to_{Stmt} \sigma'}{\langle X = 5,\sigma\rangle \to_{Stmt} \sigma'}$$

wobei $\sigma'(X) = 5$ und $\sigma'(Y) = \sigma(Y)$ für alle $Y \neq X$

Definiere :

$$\sigma[m/X](Y) := \begin{cases} m & \text{if } X = Y \\ \sigma(Y) & \text{sonst} \end{cases}$$

$$\langle X = 5,\sigma\rangle \to_{Stmt} \sigma[5/X] \qquad \langle \{\,\},\sigma\rangle \to_{Stmt} \sigma$$

$$\frac{\langle a,\sigma\rangle \to_{Aexp} n \in \mathbf{N}}{\langle X = a,\sigma\rangle \to_{Stmt} \sigma[n/X]} \quad \frac{\langle a,\sigma\rangle \to_{Aexp} \bot}{\langle X = a,\sigma\rangle \to_{Stmt} \bot} \quad \frac{\langle c,\sigma\rangle \to_{Stmt} \sigma' \neq \bot \quad \langle \{c_s\},\sigma'\rangle \to_{Stmt} \sigma'' \neq \bot}{\langle \{c\ c_s\},\sigma\rangle \to_{Stmt} \sigma''}$$

$$\frac{\langle c,\sigma\rangle \to_{Stmt} \bot}{\langle \{c\ c_s\},\sigma\rangle \to_{Stmt} \bot}$$

Beispiel

wobei $t = 1$ wenn $t_1 = t_2 = 1$;
$t = 0$ wenn $t_1 = 0$ oder ($t_1 = 1$ und $t_2 = 0$);
$t = \bot$ sonst

wobei $t = 0$ wenn $t_1 = t_2 = 0$;
$t = 1$ wenn $t_1 = 1$ oder ($t_1 = 0$ und $t_2 = 1$);
$t = \bot$ sonst

```
x= 1;
while (y != 0) {
  y = y-1;
  x = 2*x;
}
// x = 2^y
```

$\sigma(y) = 3$

Äquivalenz arithmetischer Ausdrücke

Gegeben zwei Aexp a_1 and a_2

- Sind sie gleich?

$$a_1 \sim_{Aexp} a_2 \text{ gdw } \forall \sigma, n.\langle a_1,\sigma\rangle \to_{Aexp} n \Leftrightarrow \langle a_2,\sigma\rangle \to_{Aexp} n$$

(X*X) + 2*X*Y + (Y*Y) und (X+Y) * (X+Y)

- Wann sind sie gleich?

$$\exists \sigma, n.\langle a_1,\sigma\rangle \to_{Aexp} n \Leftrightarrow \langle a_2,\sigma\rangle \to_{Aexp} n$$

X*X und X*X+2
X*X und X*X+1

Äquivalenz Boolscher Ausdrücke

Gegeben zwei Bexp-Ausdrücke b_1 and b_2

- Sind sie gleich?

$$b_1 \sim_{Bexp} b_2 \text{ iff } \forall \sigma, b.\langle b_1,\sigma\rangle \to_{Bexp} b \Leftrightarrow \langle b_2,\sigma\rangle \to_{Bexp} b$$

A || (A && B) und A

Beweisen

Zwei Programme c_0, c_1 sind äquivalent gdw. sie die gleichen Zustandsveränderungen bewirken. Formal definieren wir

Definition

$$c_0 \sim c_1 \text{ iff } \forall \sigma, \sigma'.\ \langle c_0, \sigma\rangle \to_{Stmt} \sigma' \Leftrightarrow \langle c_1, \sigma\rangle \to_{Stmt} \sigma'$$

Ein einfaches Beispiel:

Lemma

Sei $w \equiv$ ***while*** $(\ b\ )\ \ c$ *mit* $b \in$ ***Bexp***, $c \in$ ***Stmt***.
Dann gilt: $w \sim$ ***if*** $(\ b\ )\ \{c; w\}$ ***else*** $\{\}$

Beweis an der Tafel

Zusammenfassung

- Operationale Semantik als ein Mittel für Beschreibung der Semantik
- Auswertungsregeln arbeiten entlang der syntaktischen Struktur
- Werten Ausdrücke zu Werten aus und Programme zu Zuständen (zu gegebenen Zustand)
- Fragen zu Programmen: Gleichheit