Korrekte Software: Grundlagen und Methoden
Vorlesung 2 vom 10.04.18: Operationale Semantik

Serge Autexier, Christoph Lüth

Universität Bremen

Sommersemester 2018

Fahrplan

- Einführung
- Operationale Semantik
- Denotationale Semantik
- Äquivalenz der Operationalen und Denotationalen Semantik
- Die Floyd-Hoare-Logik
- Invarianten und die Korrektheit des Floyd-Hoare-Kalküls
- Strukturierte Datentypen
- Modellierung und Spezifikation
- Verifikationsbedingungen
- Vorwärts mit Floyd und Hoare
- Funktionen und Prozeduren
- Referenzen
- 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 ($==, <, \ldots$), boolsche Operatoren (&&, ||);
- Anweisungen:
  - Fallunterscheidung (**if**...**else**...), Iteration (**while**), Zuweisung, Blöcke;
  - Sequenzierung und leere Anweisung sind implizit

Semantik von C0

- Die (operationale) Semantik einer imperativen Sprache wie C0 ist ein **Zustandsübergang**: das System hat einen impliziten Zustand, der durch Zuweisung von **Werten** an **Adressen** geändert werden kann.

Systemzustände

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

C0: Ausdrücke und Anweisungen

$$
\begin{array}{rll}
\textbf{Aexp} & a ::= & \mathbf{Z} \mid \mathbf{Idt} \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 * a_2 \mid a_1/a_2 \\
\textbf{Bexp} & b ::= & \mathbf{1} \mid \mathbf{0} \mid a_1 == a_2 \mid a_1 < a_2 \mid !\, b \mid b_1 \,\&\&\, b_2 \mid b_1 \,||\, b_2 \\
\textbf{Exp} & e := & a \mid b \\
\textbf{Stmt} & c ::= & \mathbf{Idt} = \mathbf{Exp} \\
 & \mid & \textbf{if}\ (b)\ c_1\ \textbf{else}\ c_2 \\
 & \mid & \textbf{while}\ (b)\ c \\
 & \mid & c_1; c_2 \\
 & \mid & \{\,\}
\end{array}
$$

NB: Nicht die **konkrete** Syntax.

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

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

Regeln:

$$\frac{}{\langle n, \sigma\rangle \to_{Aexp} n}$$

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

Operationale Semantik: Arithmetische Ausdrücke

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

$$\langle a, \sigma \rangle \to_{Aexp} n \underline{\perp}$$

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

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

$$\frac{\langle a_1, \sigma \rangle \to_{Aexp} n_1 \qquad \langle a_2, \sigma \rangle \to_{Aexp} n_2 \qquad n_i \in \mathbf{Z}, n \text{ Diff. } n_1 \text{ und } n_2}{\langle a_1 - a_2, \sigma \rangle \to_{Aexp} n}$$

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

Operationale Semantik: Arithmetische Ausdrücke

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

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

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

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

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

$$\frac{\langle a_1, \sigma \rangle \to_{Aexp} n_1 \qquad \langle a_2, \sigma \rangle \to_{Aexp} n_2 \qquad \text{falls } n_1 = \perp, n_2 = \perp \text{ oder } n_2 = 0}{\langle a_1 / a_2, \sigma \rangle \to_{Aexp} \perp}$$

Beispiel-Ableitungen

Sei $\sigma(x) = 6, \sigma(y) = 5$.

$$\frac{\dfrac{\dfrac{\langle x, \sigma \rangle \to_{Aexp} 6 \qquad \langle y, \sigma \rangle \to_{Aexp} 5}{\langle x + y, \sigma \rangle \to_{Aexp} 11} \qquad \dfrac{\langle x, \sigma \rangle \to_{Aexp} 6 \qquad \langle y, \sigma \rangle \to_{Aexp} 5}{\langle x - y, \sigma \rangle \to_{Aexp} 1}}{\langle (x + y) * (x - y), \sigma \rangle \to_{Aexp} 11}$$

$$\frac{\dfrac{\dfrac{\langle x, \sigma \rangle \to_{Aexp} 6 \qquad \langle x, \sigma \rangle \to_{Aexp} 6}{\langle x * x, \sigma \rangle \to_{Aexp} 36} \qquad \dfrac{\langle y, \sigma \rangle \to_{Aexp} 5 \qquad \langle y, \sigma \rangle \to_{Aexp} 5}{\langle y * y, \sigma \rangle \to_{Aexp} 25}}{\langle (x * x) - (y * y), \sigma \rangle \to_{Aexp} 11}$$

Operationale Semantik: Boolesche Ausdrücke

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

Regeln:

$$\langle b, \sigma \rangle \to_{Bexp} 1 | 0 | \perp$$

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

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

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

$$\frac{\langle a_1, \sigma \rangle \to_{Aexp} n_1 \qquad \langle a_2, \sigma \rangle \to_{Aexp} n_2 \qquad n_1 = \perp \text{ or } n_2 = \perp}{\langle a_1 == a_2, \sigma \rangle \to_{Bexp} \perp}$$

Operationale Semantik: Boolesche Ausdrücke

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

Regeln:

$$\langle b, \sigma \rangle \to_{Bexp} 1 | 0 | \perp$$

$$\frac{\langle b, \sigma \rangle \to_{Bexp} 1}{\langle !b, \sigma \rangle \to_{Bexp} 0} \qquad \frac{\langle b, \sigma \rangle \to_{Bexp} 0}{\langle !b, \sigma \rangle \to_{Bexp} 1} \qquad \frac{\langle b, \sigma \rangle \to_{Bexp} \perp}{\langle !b, \sigma \rangle \to_{Bexp} \perp}$$

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

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

Operationale Semantik: Boolesche Ausdrücke

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

Regeln:

$$\langle b, \sigma \rangle \to_{Bexp} 1 | 0 | \perp$$

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

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

Operationale Semantik: Anweisungen

- **Stmt** $c ::= \mathbf{Idt} = \mathbf{Exp} \mid \textbf{if } (b) \ c_1 \ \textbf{else} \ c_2 \mid \textbf{while } (b) \ c \mid c_1; c_2 \mid \{\}$

Beispiel:

$$\langle c, \sigma \rangle \to_{Stmt} \sigma' | \perp$$

$$\langle x = 5, \sigma \rangle \to_{Stmt} \sigma'$$

wobei $\sigma'(x) = 5$ und $\sigma'(y) = \sigma(y)$ für alle $y \neq x$

Operationale Semantik: Anweisungen

- **Stmt** $c ::= \mathbf{Idt} = \mathbf{Exp} \mid \textbf{if } (b) \ c_1 \ \textbf{else} \ c_2 \mid \textbf{while } (b) \ c \mid c_1; c_2 \mid \{\}$

Regeln:

Definiere:

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

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

Es gilt:

$$\forall \sigma, n, m, \forall x, y \ . \ x \neq y \Rightarrow \sigma[n/x][m/y] = \sigma[m/y][n/x]$$
$$\forall \sigma, n, m, \forall x \ . \ \sigma[n/x][m/x] = \sigma[m/x]$$

Operationale Semantik: Anweisungen

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

Regeln:

$$\langle \{\}, \sigma \rangle \to_{Stmt} \sigma$$

$$\frac{\langle a, \sigma \rangle \to_{Aexp} n \in \mathbf{Z}}{\langle x = a, \sigma \rangle \to_{Stmt} \sigma[n/x]} \qquad \frac{\langle a, \sigma \rangle \to_{Aexp} \bot}{\langle x = a, \sigma \rangle \to_{Stmt} \bot}$$

$$\frac{\langle c_1, \sigma \rangle \to_{Stmt} \sigma' \neq \bot \qquad \langle c_2, \sigma' \rangle \to_{Stmt} \sigma'' \neq \bot}{\langle c_1; c_2, \sigma \rangle \to_{Stmt} \sigma''}$$

$$\frac{\langle c_1, \sigma \rangle \to_{Stmt} \bot}{\langle c_1; c_2, \sigma \rangle \to_{Stmt} \bot}$$

$$\frac{\langle c_1, \sigma \rangle \to_{Stmt} \sigma' \neq \bot \qquad \langle \{c_2\}, \sigma' \rangle \to_{Stmt} \bot}{\langle c_1; c_2, \sigma \rangle \to_{Stmt} \bot}$$

Operationale Semantik: Anweisungen

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

Regeln:

$$\frac{\langle b, \sigma \rangle \to_{Bexp} 1 \qquad \langle c_1, \sigma \rangle \to_{Stmt} \sigma'}{\langle \textbf{if } (b) \ c_1 \ \textbf{else} \ c_2, \sigma \rangle \to_{Stmt} \sigma'}$$

$$\frac{\langle b, \sigma \rangle \to_{Bexp} 0 \qquad \langle c_2, \sigma \rangle \to_{Stmt} \sigma'}{\langle \textbf{if } (b) \ c_1 \ \textbf{else} \ c_2, \sigma \rangle \to_{Stmt} \sigma'}$$

$$\frac{\langle b, \sigma \rangle \to_{Bexp} \bot}{\langle \textbf{if } (b) \ c_1 \ \textbf{else} \ c_2, \sigma \rangle \to_{Stmt} \bot}$$

Operationale Semantik: Anweisungen

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

Regeln:

$$\frac{\langle b, \sigma \rangle \to_{Bexp} 0}{\langle \textbf{while } (b) \ c, \sigma \rangle \to_{Stmt} \sigma}$$

$$\frac{\langle b, \sigma \rangle \to_{Bexp} 1 \qquad \langle c, \sigma \rangle \to_{Stmt} \sigma' \qquad \langle \textbf{while } (b) \ c, \sigma' \rangle \to_{Stmt} \sigma''}{\langle \textbf{while } (b) \ c, \sigma \rangle \to_{Stmt} \sigma''}$$

$$\frac{\langle b, \sigma \rangle \to_{Bexp} 1 \qquad \langle c, \sigma \rangle \to_{Stmt} \bot}{\langle \textbf{while } (b) \ c, \sigma \rangle \to_{Stmt} \bot} \qquad \frac{\langle b, \sigma \rangle \to_{Bexp} \bot}{\langle \textbf{while } (b) \ c, \sigma \rangle \to_{Stmt} \bot}$$

Beispiel

```
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 `9*x+22`
`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

Beweis

Gegeben beliebiger Programmzustand σ. Zu zeigen ist, dass sowohl w also auch **if** (b) $\{c; w\}$ **else** $\{\}$ zu dem selben Programmzustand auswerten oder beide zu einem Fehler. Der Beweis geht per Fallunterscheidung über die Auswertung von Teilausdrücken bzw. Teilprogrammen.

1. $\langle b, \sigma \rangle \to_{Bexp} 0$:

$$\langle \textbf{while } (b) \ c, \sigma \rangle \to_{Stmt} \sigma$$
$$\langle \textbf{if } (b) \ \{c; w\} \ \textbf{else} \ \{\}, \sigma \rangle \to_{Stmt} \langle \{\}, \sigma \rangle \to_{Stmt} \sigma$$

2. $\langle b, \sigma \rangle \to_{Bexp} 1$:
   1. $\langle c, \sigma \rangle \to_{Stmt} \sigma'$

$$\overbrace{\langle \textbf{while } (b) \ c, \sigma \rangle}^{w} \to_{Stmt} \langle c, \sigma \rangle \to_{Stmt} \sigma'$$
$$\langle w, \sigma' \rangle \to_{Stmt} \sigma''$$
$$\langle \textbf{if } (b) \ \{c; w\} \ \textbf{else} \ \{\}, \sigma \rangle \to_{Stmt} \langle \{c; w\}, \sigma \rangle \to_{Stmt} \langle c, \sigma \rangle \to_{Stmt} \sigma'$$
$$\langle w, \sigma' \rangle \to_{Stmt} \sigma''$$

Beweis II

2. $\langle b, \sigma\rangle \to_{Bexp} 1$:

   2.2. $\langle c, \sigma\rangle \to_{Stmt} \bot$

$$\langle \overbrace{\textbf{while } (b)\ c}^{w}, \sigma\rangle \to_{Stmt} \langle c, \sigma\rangle \to_{Stmt} \bot$$
$$\langle \textbf{if } (b)\ \{c; w\}\ \textbf{else}\ \{\}, \sigma\rangle \to_{Stmt} \langle \{c; w\}, \sigma\rangle \to_{Stmt} \langle c, \sigma\rangle \to_{Stmt} \bot$$

3. $\langle b, \sigma\rangle \to_{Bexp} \bot$:

$$\langle \textbf{while } (b)\ c, \sigma\rangle \to_{Stmt} \bot$$
$$\langle \textbf{if } (b)\ \{c; w\}\ \textbf{else}\ \{\}, \sigma\rangle \to_{Stmt} \bot$$

Zusammenfassung

- Operationale Semantik als ein Mittel zur 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