

Semantik von Programmiersprachen – SS 2015

<http://pp.ipd.kit.edu/lehre/SS2015/semantik>

Blatt 13: Axiomatische Semantik

Besprechung: 06.07.2015

1. Welche der folgenden Aussagen sind richtig, welche falsch? (H)

- (a) $\{x == 0\} x := x + 1 \{0 \leq x\}$ ist eine Zusicherung.
- (b) In $\{x = n\} y := x * n \{y = n^2\}$ ist n eine logische Variable.
- (c) Wenn $\vdash \{P\} \text{skip} \{Q\}$, dann $P = Q$.
- (d) Es gibt ein c mit $\vdash \{\text{tt}\} c \{\text{ff}\}$.
- (e) Statt der Regel ASS_P wäre auch die Regel $\vdash \{P\} x := a \{P[x \mapsto \mathcal{A}[[a]]]\}$ sinnvoll.

2. Quadratwurzelberechnung (H)

Zeigen Sie, dass folgende Zusicherung in der axiomatischen Semantik ableitbar ist.

```
{ a ≥ 0 }
n := 0; m := 1; k := 0;
while (k <= a - 1) do (n := n + 1; k := k + m; m := m + 2)
{ n - 1 < √a ≤ n }
```

Hinweis: Die Invariante der Schleife ist etwas trickreich. Berechnen Sie ein paar einfache Beispiele und versuchen Sie herauszufinden, in welcher Beziehung m , n und k zueinander stehen.

3. Schnelle Division (Ü)

Arithmetische Ausdrücke in \mathbf{Aexp} enthalten keinen Divisionsoperator. In dieser Aufgabe sollen Sie einen `While`-Programm schreiben, das zwei ganze Zahlen mit logarithmischer Laufzeit (in der Größe der Zahlen) dividiert, und dieses korrekt beweisen.

- (a) Ergänzen Sie folgendes Programmskelett, so dass das vollständige Programm c folgende Spezifikation in Form einer Zusicherung erfüllt, die angegebenen Invarianten verwendet und die Eingabevariablen x und y unverändert lässt:

$$\{x \geq 0 \wedge y > 0\} c \{x = q \cdot y + r \wedge 0 \leq r < y\}$$

```
... ; // Invariante  $I_0$  sicherstellen
while (z <= x) do ( ... // Invariante  $I_0$  erhalten und z verdoppeln );
... ; // Invariante  $I_1$  sicherstellen
while (not (n == 0)) do ( ... // Invariante  $I_1$  erhalten und n verkleinern )
```

wobei

$$I_0(\sigma) \equiv \sigma(z) = \sigma(y) \cdot 2^{\sigma(n)} \wedge \sigma(n) \geq 0 \wedge \sigma(x) \geq 0 \wedge \sigma(y) > 0$$

$$I_1(\sigma) \equiv \sigma(x) = \sigma(q) \cdot \sigma(z) + \sigma(r) \wedge 0 \leq \sigma(r) < \sigma(z) \wedge \sigma(z) = \sigma(y) \cdot 2^{\sigma(n)} \wedge \sigma(n) \geq 0$$

Für diese Aufgabe können Sie neben den bisherigen arithmetischen Ausdrücken auch den neuen Verschiebe-Ausdruck $a \gg 1$ mit der Semantik

$$\mathcal{A} \llbracket a \gg 1 \rrbracket \sigma = \left\lfloor \frac{\mathcal{A} \llbracket a \rrbracket \sigma}{2} \right\rfloor$$

verwenden, wobei $\lfloor x \rfloor$ die rationale Zahl x auf die nächstkleinere ganze Zahl abrundet.

- (b) Weisen Sie nach, dass obige Zusicherung für Ihr Programm c ableitbar ist. Verwenden Sie dabei die angegebenen Invarianten für die Schleifen.
- (c) Terminiert Ihr Programm immer?
- (d) Eliminieren Sie jetzt den Operator $a \gg 1$ in Ihrem Programm, verwenden Sie bei Bedarf Schleifen. Wie müsste man den Korrektheitsbeweis anpassen? In welcher Komplexitätsklasse liegt Ihr Programm jetzt?