Karlsruhe Institute of Technology

Lehrstuhl für Programmierparadigmen
Andreas Lochbihler
andreas.lochbihler@kit.edu

Semantik von Programmiersprachen – SS 2010

http://pp.info.uni-karlsruhe.de/lehre/SS2010/semantik

Blatt 6: Erweiterungen zu While Besprechung: 25.05.2010

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

(a) c_1 `or` c_2 und c_2 `or` c_1 sind äquivalent bzgl. der Big-Step-Semantik.
(b) c_1 `or` c_2 und c_2 `or` c_1 sind äquivalent bzgl. der Small-Step-Semantik.
(c) `x := 0; y := 0; while (y == 0) do (x := x + 1 or y := 1)` terminiert immer.
(d) `x := 5 or x := 6` und `x := 5 || x := 6` sind semantisch äquivalent.
(e) c_1 `||` `(`c_2 `||` c_3`)` = `(`c_1 `||` c_2`)` `||` c_3
(f) c_1 `||` c_2 und c_2 `||` c_1 sind äquivalent bzgl. der Small-Step-Semantik.
(g) Die Big-Step-Semantik von While_B ist nicht deterministisch.
(h) Nach Auführung von `{ var x = 1; y := x + 1; { var y = 3; x := y + 2; { var x = 6; z := x + y }; y := z }; z := x + y + z }` hat `z` den Wert 24.
(i) `{ var z = 142; { var x = x + 1; z := x }; x := z - 1 }` ist semantisch äquivalent zu `skip`.

2. Blöcke und Parallelität (H)

In dieser Aufgabe seien die Erweiterungen zur Paralellität While_{PAR} und zu lokalen Variablen mittels Blöcken While_B kombiniert. Was sind die möglichen Endzustände des folgenden Programms in der kombinierten Small-Step-Semantik für den Anfangszustand $[\mathtt{x} \mapsto 1]$?

```
({ var y = 1; x := x + 1; y := y + 1; x := x + 2; y := y + 2; z := y }) ||
({ var y = 1; x := x * 3; y := y * 3; x := x * 4; y := y * 4; z := y })
```

3. Exceptions (Ü)

Die Sprache While soll um Exception-Handling erweitert werden. Dazu werden zwei neue Anweisungen zu While hinzugefügt:

`raise` und `try` c_1 `catch` c_2

Wenn in c_1 eine Exception mittels `raise` erzeugt wird, soll der Rest von c_1 nicht mehr abgearbeitet, sondern der Exception-Handler c_2 der Anweisung `try` c_1 `catch` c_2 ausgeführt werden. Wird keine Exception ausgelöst, wird c_2 ignoriert.[1]

(a) Ergänzen Sie die Small-Step-Semantik von While um neue Regeln für die neuen Konstrukte.
(b) Zeigen Sie nun, dass Sie obige Regeln sinnvoll gewählt haben. Formulieren Sie dazu formal, dass in Ihrer Semantik $\langle$`raise`, $\sigma\rangle$ und $\langle$`skip`, $\sigma\rangle$ die einzigen blockierten Konfigurationen sind. Beweisen Sie Ihre Behauptung (vgl. Lem. 3).
(c) Wie muss man die Big-Step-Semantik anpassen, um Exceptions zu integrieren? Kann man die erweiterten Big-Step- und Small-Step-Semantiken als äquivalent bezeichnen?

[1] Dieser Mechanismus ähnelt den Exceptions von Java, es gibt allerdings nur eine einzige Sorte von Exceptions.