Lehrstuhl für Software and Computational Systems
Prof. Dr. Gidon Ernst, Marian Lingsch-Rosenfeld

Übungen zur Vorlesung
Formale Spezifikation und Verifikation

Wintersemester 2024/25

Übungsblatt 05

Bekanntgabe am 25.11.2023

Für diese Aufgabe werden Aufgaben 1.1 und 1.2 in den Tutorien vorgerechnet.

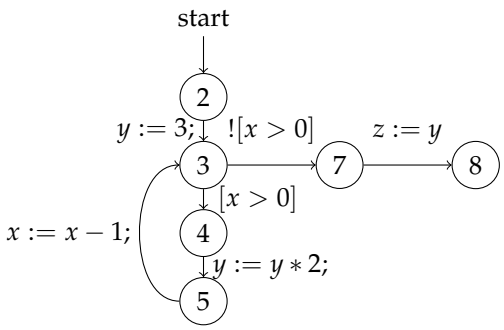
Aufgabe 2 ist zur Bearbeitung in Präsenz in den Tutorien vorgesehen.

1 Hoare-Logik

1.1 Einfache Exponentiation

Gegeben sei folgende Java-Methode/C-Funktion sowie der dazugehörige Kontrollflussautomat:

```
1 int exp(int x) {
2   int y = 1;
3   while (x>0) {
4     y = y * 2;
5     x = x - 1;
6   }
7   int z = y;
8   return z;
9 }
```



a) Beweisen Sie das Hoare-Tripel $\{x \geq 0 \wedge x = x_{init}\}\, P\, \{z = 2^{x_{init}}\}$, wobei P den Rumpf der Methode/Funktion bezeichnet. Überlegen Sie sich dazu zuerst eine geeignete Schleifeninvariante.

b) Welche Laufzeitkomplexität besitzt diese einfache Art der Exponentiation?

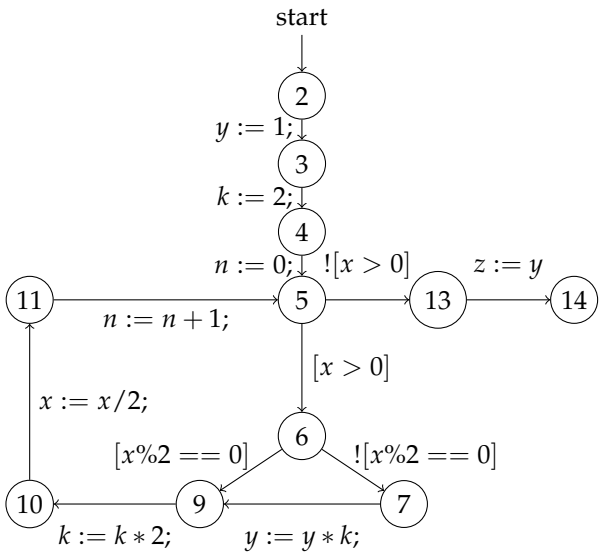
Hinweise:

- Es kann sein, dass die Invariante, die Ihnen als erstes in den Sinn kommt, nicht stark genug ist, um das Hoare-Tripel herzuleiten. In diesem Fall kann es nützlich sein, eine stärke Invariante zu fordern.

1.2 Binäre Exponentiation

Gegeben sei folgende Java-Methode/C-Funktion sowie der dazugehörige Kontrollflussautomat:

```
1  int fastexp(int x) {
2    int y = 1;
3    int k = 2;
4    int n = 0; // k == 2^(2^n)
5    while (x>0) {
6      if (!(x%2 == 0)) {
7        y = y * k;
8      }
9      k = k * k;
10     x = x / 2;
11     n = n + 1;
12   }
13   int z = y;
14   return z;
15 }
```



a) Beweisen Sie das Hoare-Tripel $\{x \geq 0 \wedge x = x_{init}\}\, P\, \{z = 2^{x_{init}}\}$, wobei P den Rumpf der Methode/Funktion bezeichnet. Nutzen Sie dazu die Schleifeninvariante I der Form $y \cdot k^x = 2^{x_{init}} \wedge k = 2^{2^n}$ und ergänzen Sie diese durch geeignete weitere Bedingungen, welche für den Beweis nötig sind.

b) Lässt sich die in a) gefundene Invariante abschwächen? Falls ja, geben Sie eine schwächere Invariante an, mit der das Hoare-Tripel dennoch bewiesen werden kann!

c) Welche Laufzeitkomplexität besitzt diese alternative Art der Exponentiation?

Hinweise:

- Die Variable n wurde nur hinzugefügt, um in der Invarante ohne Existenzquantor auszukommen. Natürlich liese sich das Hoare-Tripel auch ohne diese Hilfvariante herleiten.

2 Präsenzaufgabe: Dafny

Zeigen sie das die Programme aus Übungsblatt 4 und Aufgaben 1.1 und 1.2 korrekt sind, indem Sie deren Korrektheit in Dafny überprüfen. Bearbeiten sie diese soweit sie kommen. Als hilfestellung sehen sie das Programm zum addieren aus Übungsblatt 4 in Dafny in Listing 1.

```
1      method Add(a: int, b: int) returns (z: int)
2      requires true      // FIXME
3      ensures true       // FIXME
4  {
5      var y := a;
6      var bloc := b;
7      while (bloc > 0)
8          invariant true // FIXME
9          decreases bloc // required to proof termination
10     {
11         y := y + 1;
12         bloc := bloc - 1;
13     }
14     return y;
15 }
```

Listing 1: Dafny-Programm für die Addition