Prof. Martin Hofmann
Dr. Ulrich Schöpp

Ludwig-Maximilians-Universität München
Institut für Informatik
SS 2014

Übungen zur Vorlesung
Formale Spezifikation und Verifikation

Blatt 7

Aufgabe 7-1 (3 Punkte) Geben Sie für folgendes Programm den Kontrollflussgraphen an und bestimmen Sie die Available Expressions für beide Programme, d.h. berechnen Sie jeweils die größte Lösung der Gleichungen für $AE_{entry}(l)$ und $AE_{exit}(l)$ für alle Programmpunkte l.

$$\texttt{while } [x * x + y * y < 4 \wedge i < 50]^1 \texttt{ do}$$
$$\left([z := x * x - y * y + cx]^2;\ [y := 2 * x * y - cy]^3;\ [x := z]^4;\ [i := i + 1]^5\right)$$

Aufgabe 7-2 Eine mögliche Anwendung der Available Expressions ist, die wiederholte Auswertung von Ausdrücken bei der Programmausführung zu vermeiden. Der Wert verfügbarer Ausdrücke kann gespeichert werden und dann später ohne Neuauswertung des Ausdrucks benutzt werden. Möchte man keinen zusätzlichen Speicher verwenden, so kann man sich auf verfügbare Ausdrücke beschränken, deren Wert in einer bestimmten Programmvariable verfügbar ist: Ein Ausdruck a ist an einem Programmpunkt in Variable x verfügbar, falls die Variable x an diesem Programmpunkt den Wert des Ausdrucks a speichert.

Passen Sie die Gleichungen für die Available Expressions (d.h. für $AE_{entry}(l)$ und $AE_{exit}(l)$) so an, dass für jeden Programmpunkt die Menge aller Paare (x, a) berechnet wird, für die an diesem Programmpunkt der Ausdruck a in Variable x verfügbar ist.

Aufgabe 7-3 Gegeben sei ein vollständiger Verband $(L, \sqsubseteq)$.

Dann ist die Menge $L \times L$ ebenfalls ein vollständiger Verband bezüglich der Ordnung

$$(x, y) \sqsubseteq (x', y') \quad \iff \quad x \sqsubseteq x' \text{ und } y \sqsubseteq y'.$$

Seien $F_1 \colon L \times L \to L$ und $F_2 \colon L \times L \to L$ zwei monotone Funktionen, d.h. aus $(x, y) \sqsubseteq (x', y')$ folgt $F_1(x, y) \sqsubseteq F_1(x', y')$ und analog für F_2.

Definiere $F, G \colon L \times L \to L \times L$ wie folgt:

$$F(x, y) = (F_1(x, y), F_2(x, y))$$
$$G(x, y) = (F_1(x, y), F_2(F_1(x, y), y))$$

Zeigen Sie:

a) $L \times L$ is ein vollständiger Verband.

b) Sowohl F als auch G sind monoton.

c) Der kleinste Fixpunkt von F und G ist gleich.

Aufgabe 7-4 (3 Punkte) In der Vorlesung wurde das Alternating Bit Protokoll in NuSMV implementiert und seine Korrektheit überprüft.

In der Implementierung des Protokolls wird angenommen, dass Sender und Empfänger über Kanäle kommunizieren, die Nachrichten entweder unverfälscht weitergeben oder unlesbar machen. Es werden zwei Arten von Kanälen verwendet: Ein Zwei-Bit-Kanal `two_bit_channel` überträgt in jedem Zeitschritt zwei Bit vom Sender zum Empfänger und ein Ein-Bit-Kanal `one_bit_channel` überträgt in jedem Zeitschritt ein Bit in die andere Richtung.

```
sen : sender();
rec : receiver();
out_c : two_bit_channel(sen.msg, sen.bit, rec.in0, rec.in1);
ret_c : one_bit_channel(rec.out, sen.in0);
```

Beim Zwei-Bit-Kanal `two_bit_channel` ist die Annahme, dass entweder beide Bits korrekt übertragen werden oder beide verfälscht werden.

Angenommen wir wollen das Alternating Bit Protokoll nur mit Ein-Bit-Kanälen implementieren und zwar so, dass die beiden Bits vom Sender zum Empfänger über zwei parallele Kanäle übertragen werden. Das System wird dann wie folgt zusammengesetzt:

```
sen : sender();
rec : receiver();
out_msg_c : one_bit_channel(sen.msg, rec.in0);
out_bit_c : one_bit_channel(sen.bit, rec.in1);
ret_c : one_bit_channel(rec.out, sen.in0);
```

Eine vollständige SMV-Datei `abp1.smv` mit dieser Änderung finden Sie auf der Vorlesungshomepage.

a) Warum ist `abp1.smv` keine korrekte Implementierung des Alternating Bit Protokolls?

b) Passen Sie die Module `sender` und `receiver` in `abp1.smv` so an, dass das Alternating Bit Protokoll korrekt implementiert wird. Ihre Implementierung soll alle in `abp1.smv` gegebenen Spezifikationen erfüllen. Die in Ihrer Implementierung auf den Kanälen gesendeten Nachrichten sollen von der Implementierung aus der Vorlesung nur in dem Fall abweichen, dass einer der Kanäle `out_msg_c` und `out_bit_c` in einem Zeitschritt eine Nachricht verfälscht, der andere aber nicht.