Übungen zu Model-Checking

Sommersemester 2008 / Blatt 2

Wir besprechen die Beispiele gemeinsam am 29.5.
Bei Fragen zum Übungsbetrieb oder den Beispielen, schicken Sie mir bitte eine email (`schalla@in.tum.de`).

Beispiel 2.1: Nichtdeterminismus in Büchi-Automaten

Zeigen Sie, dass nichtdeterministische Büchi-Automaten strikt ausdrucksstärker sind als deterministische Büchi-Automaten.

a. Sei $L \subseteq \Sigma^\omega$ die Sprache über dem Alphabet $\Sigma = \{a, b\}$, welche die Wörter mit endlich vielen b-s beinhaltet:

$$L = \{\sigma \in \Sigma^\omega \mid \#_b(\sigma) < \infty\}$$

Geben Sie einen nichtdeterministischen Büchi-Automaten an, der L akzeptiert.

b. Zeigen Sie, dass es keinen deterministischen Büchi-Automaten gibt, der L akzeptiert.

Hinweis: Nehmen Sie an, es gäbe einen solchen Automaten $\mathcal{B}$ mit k Zuständen, und betrachten Sie den akzeptierenden Lauf für das Wort $w_0 = a^\omega$ (der Automat muss das Wort akzeptieren).

Fügen Sie nun hinter dem ersten akzeptierenden Zustand dieses Laufes ein b ein, um ein Wort w_1 zu erhalten: Wenn der Automat nach Lesen des Präfixes a^{l_1} zum ersten Mal in einen akzeptieren Zustand übergeht, setzen Sie $w_1 = a^{l_1}ba^\omega$. Beachten Sie, dass $w_1 \in L$ gilt. Welche Eigenschaft muss der Zustand, der nach Lesen des Präfixes $a^{l_1}b$ erreicht wird, erfüllen? Und wie können Sie diese Eigenschaft nützen, um mittels einer Reihe von Wörtern $w_0, \ldots, w_{k+1}$ einen Widerspruch zu abzuleiten?

Beispiel 2.2: Übersetzung von LTL nach Büchi-Automaten

Geben Sie für die folgenden **LTL**-Formeln möglichst kleine Büchi Automaten an (und benützen Sie dafür nicht die Konstruktion aus der Vorlesung):

a. $(\mathbf{G}p) \rightarrow (p \,\mathbf{U}\, q)$

b. $\mathbf{G}(p \rightarrow \mathbf{X}(\neg p \,\mathbf{U}\, q))$

c. $p \,\mathbf{R}\, q$

Beispiel 2.3: Token Ring in Spin

Ein Token Ring besteht aus m unabhängigen Prozesse, die in einem Kreis angeordnet sind, so dass jeder Prozesse genau einen linken und einen rechten Nachbar hat. Die Prozesse in dem Token Ring benützen ein Token (am besten durch eine Message in einem Channel dargestellt), um sich zu synchronisieren. Nach jeden Berechnungsschritt wird das Token an einen der Nachbarn weitergereicht. Ein Prozess darf nur dann in der Critical Section sein, wenn er über das Token verfügt.

a. Implementieren Sie einen Token Ring in Promela mit $m = 4$, wobei das Token zu einem der beiden Nachbarn nichtdeterministisch weitergereicht wird. Simulieren Sie den Token Ring interaktiv.

b. Benützen Sie SPIN, um zu überprüfen, dass zu jedem Zeitpunkt höchstens ein Prozess in der Critical Section ist.

c. Benützen Sie SPIN, um zu überprüfen, dass zumindest ein Prozess unendlich oft in die Critical Section eintritt.

d. Wiederholen Sie die obigen Schritte für ein deterministisches Modell, wobei das Token immer nach links weitergereicht wird.

e. Benützen Sie SPIN, um für die deterministische und nichtdeterministische Variante zu überprüfen, ob *jeder* Prozess unendlich oft in die Critical Section kommt.

Hinweis: Benützen Sie einen Promela Prozesstyp, um die Token Ring Prozesse zu modellieren:

```
proctype process(chan left; chan right)
```

Dieser Prozess erhält als Parameter jeweils einen Channel zum linken und rechten Nachbaren. Starten Sie dann vier Prozesse dieses Typs in einem init-Prozess mittels

```
run process(t1,t2);
run process(t2,t3);
run process(t3,t4);
run process(t4,t1)
```

wobei Sie vorher die Channels mittels

```
mtype {
  TOKEN
}

chan t1 = [1] of { mtype }
chan t2 = [1] of { mtype }
chan t3 = [1] of { mtype }
chan t4 = [1] of { mtype }
```

deklarieren.