

## Übungen zu Model Checking

Besprechung am 14.06.05

### Aufgabe 6.1

```
void quicksort( int left , int right ) {
    int lo , hi , piv;
    if( left >= right ) return;
    piv = a[right]; lo = left; hi = right;
    while( lo <= hi ) {
        if( a[hi] > piv ) hi--;
        else {
            int tmp = a[lo]; a[lo] = a[hi]; a[hi] = tmp;
            lo++;
        }
    }
    quicksort(left , hi); quicksort(lo , right);
}
```

Modellieren Sie den Kontrollfluss der obigen Quicksortimplementierung als Pushdown-System. Vernachlässigen Sie hierbei die Variablen und Zuweisungen, und modellieren Sie Verzweigungen mit Hilfe von Nichtdeterminismus.

### Aufgabe 6.2     *pre\**, *post\**

$$\begin{array}{ll}
 P & := \{p_0, p_1, p_2\} & r_1 & := p_0a_0 \hookrightarrow p_1a_1a_0 \\
 \Gamma & := \{a_0, a_1, a_2\} & r_2 & := p_1a_1 \hookrightarrow p_2a_2a_0 \\
 \Delta & := \{r_1, r_2, r_3, r_4\} & r_3 & := p_2a_2 \hookrightarrow p_0a_1 \\
 & & r_4 & := p_0a_1 \hookrightarrow p_0\varepsilon
 \end{array}$$

a) Berechnen Sie die Menge aller Konfigurationen des obigen Pushdown-Systems, welche eine Konfiguration der Form  $p\varepsilon$  ( $p \in P$ ) erreichen können.

b) In der Vorlesung wurde weiterhin der *post\**-Algorithmus erwähnt, welcher dazu dient, die Nachfolgermenge ausgehend von einem gegebenen  $\mathcal{P}$ -Automaten zu berechnen.

Versuchen Sie einen entsprechenden Algorithmus anzugeben. Ergänzen Sie hierfür den anfänglich gegebenen  $\mathcal{P}$ -Automaten schrittweise um zusätzliche Transitionen:

- Betrachten Sie hierfür eine bereits vorhandene Transition  $q \xrightarrow{a} r$  ( $a \in \Gamma$ ) und leiten Sie hiermit ab, welche weiteren Konfigurationen der Automat auf Grund dieser Transition erkennen muss.
- Sollte eine Transition mit mehr als einem Stacksymbol ( $\Gamma$ ) beschriftet werden müssen, so spalten Sie diese Transition unter Verwendung zusätzlicher Zustände geeignet auf (*Hinweis*: Verwenden Sie für  $q \xrightarrow{ab} q'$  den Zwischenzustand  $(q, a)$ ).
- Vergessen Sie auch nicht,  $\varepsilon$ -Transitionen des aktuellen  $\mathcal{P}$ -Automaten geeignet zusammenzuziehen.

*Hinweis*: Sie müssen nicht die Korrektheit Ihres Algorithmus beweisen.

c) Berechnen Sie mit Hilfe Ihres *post\**-Algorithmus die Menge aller Nachfolger der Konfiguration  $p_0\gamma_0\gamma_2$ .

### Aufgabe 6.3     Repeating Heads

In der Vorlesung haben Sie das **LTL**-Modelchecking für Pushdown-Systeme kennengelernt. Hierfür wurde für eine unendlichen Berechnung  $\rho = (q_0, a_0\gamma_0)(q_1, a_1\gamma_1) \dots (q_i \in P, a_i \in \Gamma, \gamma_i \in \Gamma^*)$  die Teilsequenz  $(q_{i_k}, a_{i_k}\gamma_{i_k})_{(k \in \mathbb{N})}$  der Konfigurationen mit minimaler Stacklänge (bezüglich des jeweiligen Suffix  $\rho^{i_k-1}$ ) betrachtet ( $i_k := \min\{i > i_{k-1} \mid \forall j > i_{k-1} : \#\gamma_j \geq \#\gamma_{i_k}\}$ ).

Es folgt  $\gamma_{i_k} \leq \gamma_{i_{k+1}}$ , d.h. in der Konfiguration  $(q_{i_k}, a_{i_k}\gamma_{i_k})$  können nur Regeln der Form  $q_{i_k}a_{i_k} \hookrightarrow a_{i_{k+1}} \mid ba_{i_{k+1}}$  ausgeführt werden.

Es lassen sich also alle unendlichen Berechnung auf ihre jeweilige Teilsequenz minimaler Stackinhalte reduzieren. Diese Teilsequenzen wiederum können dann durch einen endlichen Graphen  $\mathcal{G}$  mit Knoten  $(q, a)$  repräsentiert werden, da für diese Teilsequenzen offensichtlich nur das oberste Stackzeichen relevant ist, während der restliche Stackinhalt vergessen werden darf.

Eine Kante  $(q, a) \rightarrow (q', a')$  in  $\mathcal{G}$  existiert demnach genau dann, wenn Regeln der Form  $qa \hookrightarrow q'a' \mid pba'$  existieren, wobei in letzterem Fall zusätzlich  $pb \hookrightarrow^* q'\varepsilon$  gelten muss.

- a) Überlegen Sie sich, wie Sie mittels dem  $post^*$ - oder dem  $pre^*$ -Algorithmus *effizient* entscheiden können, ob  $pb \hookrightarrow^* q'\varepsilon$  gilt. Welchen der beiden Algorithmen würden Sie hierfür bevorzugen?

Die Kanten  $(q, a) \rightarrow (q', a')$  von  $\mathcal{G}$  sagen also gerade aus, dass es einen Pfad  $qa \hookrightarrow p_1\gamma_1a' \hookrightarrow p_2\gamma_2a' \hookrightarrow \dots \hookrightarrow q'a'$  gibt ( $\gamma_i \in \Gamma^*, p_i \in P$ ). Um zusätzlich die Büchi-Akzeptanzbedingung in  $\mathcal{G}$  zu kodieren, wird eine Kante  $(q, a) \rightarrow (q', a')$  von  $\mathcal{G}$  eingefärbt, falls auf solch einem Pfad vor Erreichen des Pfades  $q'a'$  ein akzeptierender Zustand besucht wird.

- b) Erweitern Sie den  $pre^*$ -Algorithmus derart, dass er Ihnen die zusätzliche Information liefert, ob eine erreichbare Konfiguration über einen akzeptierenden Zustand erreicht werden kann. Machen Sie sich zunächst klar, dass eine Transition (bzw. Pfad)  $q \xrightarrow{\gamma}^* q'$  mit  $q, q' \in P, \gamma \in \Gamma^*$  in dem durch den  $pre^*$ -Algorithmus erzeugten Automaten gerade bedeutet, dass es einen Pfad  $q\gamma \hookrightarrow^* q'\varepsilon$  gibt. Unterscheiden Sie dann geeignet im  $pre^*$ -Algorithmus zwischen zwei Typen von Transitionen.

*Hinweis:* Ein Korrektheitsbeweis ist nicht verlangt.

- c) Betrachten Sie nun folgendes Pushdown-System, stellen Sie den entsprechenden Graphen  $\mathcal{G}$  auf und entscheiden Sie, ob eine akzeptierende Berechnung existiert unter der Annahme, dass das System in der Konfiguration  $p_0a_0$  beginnt.

$$\begin{array}{ll}
 P & := \{p_0, p_1, p_2\} & r_1 & := p_0a_0 \hookrightarrow p_1a_1a_0 \\
 \Gamma & := \{a_0, a_1, a_2\} & r_2 & := p_1a_1 \hookrightarrow p_2a_2a_0 \\
 \Delta & := \{r_1, r_2, r_3, r_4\} & r_3 & := p_2a_2 \hookrightarrow p_0a_1 \\
 F & := \{p_2\} & r_4 & := p_0a_1 \hookrightarrow p_0\varepsilon
 \end{array}$$