Technische Universität München **WS 2006/07**
Institut für Informatik
Prof. Manfred Broy **6. November 2006**
A. Gruler, B. Spanfelner **Musterlösung zu Blatt 2**

Übungen zu „Formale Methoden“

Aufgabe 1 Datenmodellierung [LÖSUNG]

Im Hinblick auf Teilaufgabe b.) betrachten wir gleich den Fall einer speziellen Prioritässchlange, deren Elemente natürliche Zahlen (anstatt allgemeiner Elemente) sind. Genauer betrachten wir eine Menge Pq endlicher Mehrfachmengen natürlicher Zahlen. Die leere Menge ϵ gehöre zu Pq, und für $s \in Pq$ sei das Einfügen einer natürlichen Zahl und das Auswählen sowie das Löschen einer größten Zahl aus s definiert. Eine solche Rechenstruktur heißt Prioritätsschlange, in Anlehnung an Warteschlangen, in denen die Elemente eine Priorität – hier festgelegt durch den Wert des Elements – besitzen.

(a) Prioritätsschlangen haben die Signatur $\Sigma = (S, F)$.

Die Menge S der Sortennamen besteht aus Pq, der Sorte der Prioritätsschlangen, Nat, der Sorte der natürlichen Zahlen, und $Bool$, der Sorte der Wahrheitswerte. Somit ist $S = \{Pq, Nat, Bool\}$.

Die Menge der Funktionsnamen mit ihren Funktionalitäten ist

$$
\begin{array}{lllll}
F & = \{ & epq: & Pq, & \text{„leere Prioritätsschlange“} \\
 & & enq: & Pq, Nat \rightarrow Pq, & \text{„Einfügen“} \\
 & & next: & Pq \rightarrow Nat, & \text{„Auswählen“} \\
 & & deq: & Pq \rightarrow Pq, & \text{„Löschen“} \\
 & & \geq: & Nat, Nat \rightarrow Bool & \text{„Ordnung der natürlichen Zahlen“} \\
 & & ...\} & &
\end{array}
$$

(b) Zwei Modelle: Bei der Angabe von Rechenstrukturen müssen wir uns entscheiden, ob wir mit totalen oder partiellen Funktionen spezifizieren. Wir entscheiden uns für totale Funktionen und müssen daher die Trägermengen um ein $\perp$-Element komplettieren. Alle Funktionen sollen strikt definiert werden, d.h. sie sollen als Ergebnis $\perp$ liefern, wenn ein Argument $\perp$ ist. Wir müssen uns ferner entscheiden, ob wir mit der Sorte *Bool* ebenso verfahren, also eine dreiwertige Logik verwenden wollen. Dies tun wir nicht; wir verlangen also, dass alle Funktionen mit booleschen Ergebnissen stark definiert sind, d.h. *true* oder *false* und nie $\perp$ liefern. Z.B. sei $(\perp = \perp)$ wahr und es gelte $\forall\, n : Nat \;:\; n \neq \perp \;\wedge\; n \geq \perp$.

Modell P der absteigend geordneten Sequenzen.

Eine Sequenz $s \in \mathbb{N}^*$ ist absteigend geordnet, g.d.w.

$$x, y \in \mathbb{N}^*,\; m, n \in \mathbb{N} : \; s = x \circ < m > \circ < n > \circ y \rightarrow m \geq n$$

Nach dieser Definition sind insbesondere die leere Sequenz ϵ und alle einelementigen Sequenzen geordnet. Denn für alle möglichen Zerlegungen $x\circ < m > \circ < n > \circ y$ gilt $m \geq n$. Es gibt nämlich in diesen Fällen keine solche Zerlegung. Die Trägermenge sei $\quad Pq^P = \{s \in \mathbb{N}^* : s \text{ ist absteigend geordnet}\} \cup \{\perp\}$.

Die Funktionen sind wie folgt definiert:

$$
\begin{aligned}
epq^P &= \epsilon \\
enq^P(s,m) &= \begin{cases} \perp & \text{falls } s = \perp \ \vee \ m = \perp \\ s \circ < m > & \text{falls } s = \epsilon \ \vee \ \exists s' \in \mathbb{N}^*,\ n \in \mathbb{N} : \\ & \qquad\qquad s = s' \circ < n > \ \wedge \ (n \geq m) \\ enq^P(s',m) \circ < n > & \text{falls } s = s' \circ < n > \ \wedge \ \neg(n \geq m) \end{cases} \\
next^P(s) &= \begin{cases} \perp & \text{falls } s = \epsilon \ \vee \ s = \perp \\ n & \text{falls } s = < n > \circ s' \end{cases} \\
deq^P(s) &= \begin{cases} \perp & \text{falls } s = \epsilon \ \vee \ s = \perp \\ s' & \text{falls } s = < n > \circ s' \end{cases}
\end{aligned}
$$

Modell Q der beliebigen Sequenzen.

Die Trägermenge sei $\quad Pq^Q = \mathbb{N}^* \cup \{\perp\}$.

Die Funktionen sind wie folgt definiert:

$$
\begin{aligned}
epq^Q &= \epsilon \\
enq^Q(s,m) &= \begin{cases} \perp & \text{falls } s = \perp \ \vee \ m = \perp \\ s \circ < m > & \text{sonst} \end{cases} \\
next^Q(s) &= \begin{cases} \perp & \text{falls } s = \epsilon \ \vee \ s = \perp \\ m & \text{falls } s = < m > \circ s' \\ & \qquad\qquad \wedge \ (s' = \epsilon \ \vee \ m \geq next^Q(s')) \\ next^Q(s') & \text{falls } s = < m > \circ s' \ \wedge \ \neg(m \geq next^Q(s')) \end{cases} \\
deq^Q(s) &= \begin{cases} \perp & \text{falls } s = \epsilon \ \vee \ s = \perp \\ s' & \text{falls } s = < m > \circ s' \\ & \qquad\qquad \wedge \ (s' = \epsilon \ \vee \ m \geq next^Q(s')) \\ < m > \circ \ deq^Q(s') & \text{falls } s = < m > \circ s' \ \wedge \ \neg(m \geq next^Q(s')) \end{cases}
\end{aligned}
$$

Im Modell P der geordneten Sequenzen ist das Einfügen aufwändig, das Auswählen und Löschen dafür einfach. Im Modell Q ist das umgekehrt.

(c) Wann und in welcher Sicht sind in den Modellen zwei Prioritätsschlangen gleich?

Als Sequenzen sind zwei Prioritätsschlangen gleich, wenn sie gleich lang sind und elementweise übereinstimmen.
In Q sind die Prioritätsschlangen $< 17 \ \ 5 \ \ 23 >$ und $< 23 \ \ 17 \ \ 5 >$ als Sequenzen verschieden. In der Zugriffssicht (Nutzungssicht) sind sie hingegen gleich. In P gibt es nur geordnete Sequenzen. Die Sequenzengleichheit gibt auch die Zugriffssicht wieder.

(d) Typische Eigenschaften von Prioritätsschlangen:

Wir stellen auf systematische Weise Eigenschaften zusammen: Pq-Terme werden durch die Funktionen epq und enq aufgebaut und durch die Funktionen $next$ und deq zerlegt. Wir suchen nach Gleichungen, die (von links nach rechts angewendet) Terme $next(t)$ und $deq(t)$ reduzieren. Dazu setzen wir für t zum einen epq und zum anderen $enq(s, n)$ ein. Auf diese Weise erhalten wir das folgende Axiomensystem für Prioritätsschlangen:

$s \ : \ Pq \setminus \{\perp\}, \ n : Nat \ :$

$$\begin{array}{rcl} next(enq(epq, n)) & = & n, \\ deq(enq(epq, n)) & = & epq, \\[1ex] (n \ \geq \ next(s)) & \rightarrow & next(enq(s, n)) \ = \ n, \\ (\neg(n \ \geq \ next(s))) & \rightarrow & next(enq(s, n)) \ = \ next(s), \\[1ex] (n \ \geq \ next(s)) & \rightarrow & deq(enq(s, n)) \ = \ s, \\ (\neg(n \ \geq \ next(s))) & \rightarrow & deq(enq(s, n)) \ = \ enq(deq(s), n). \end{array}$$

Die letzten vier bedingten Gleichungen schließen den Fall $s = epq$ ein, wenn

$$next(epq) = \perp \quad \wedge \quad deq(epq) = \perp$$

gefordert wird. Denn nach der Festlegung gilt dann $n \ \geq \ next(s)$ für $s = epq$ und für alle $n \ : \ Nat$.

Aufgabe 2 Termerzeugte / initiale / terminale Σ–Algebra. Homomorphismus [LÖSUNG]

(a) Die Modelle P und Q sind jeweils termerzeugt, da in beiden Modellen sämtliche Elemente jeder Trägermenge durch die Funktionen erzeugt werden können (epq und enq erzeugen alle Elemente der Trägermenge zu Pq; Durch Hinzunahme von $next$ lassen sich auch noch sämtliche natürlichen Zahlen erzeugen.). Die Eigenschaft der Termerzeugtheit ist im Allgemeinen durch **generated_by** gezeigt.
Wenn wir ein Modell R definieren mit Pq^Q als Trägermenge und mit den Generatoren von Modell P und den Diskriminatoren von Modell Q, wird R nicht termerzeugt, da durch diese Funktionen nur geordnete Sequenzen geliefert werden und nicht alle Elemente der Trägermenge (also auch beliebige Sequenzen) als Terme aus diesen Funktionen dargestellt werden können.

(b) Der Homomorphismus φ bildet beliebige Sequenzen vom Modell Q auf geordnete Sequenzen im Modell P ab:

$$\varphi : \ Pq^Q \cup \mathbb{N}^\perp \ \rightarrow Pq^P \cup \mathbb{N}^\perp$$

Sei φ auf $\mathbb{N}^\perp$ die Identität, d.h. für die Homomorphiefunktion zur Sorte Nat gilt:

$$\varphi_{Nat}(n) = n, \quad n \in \mathbb{N}^\perp$$

Auf Pq^Q lauten die Bedingungen der Funktionenverträglichkeit:

$$\varphi(epq^Q) = epq^P$$

$$\varphi(enq^Q(s,n)) = enq^P(\varphi(s), n)$$

$$\varphi(next^Q(s)) = next^P(\varphi(s))$$

$$\varphi(deq^Q(s)) = deq^P(\varphi(s))$$

Die folgende Funktion $\varphi(s)$ erfüllt die Abbildung, indem sie die Elemente einer beliebigen Schlange $s \in Pq^Q$ absteigend ordnet (Aus einer beliebigen, nichtleeren Prioritätsschlange s wird dabei jeweils das Element höchster Priorität entfernt und an den Anfang der rekursiv sortierten Rest-Schlange angefügt.)

$$\varphi(s) = \begin{cases} s & \text{, falls} \quad s = \epsilon \vee s = \perp \\ < next^Q(s) > \circ \varphi(deq^Q(s)) & \text{, sonst} \end{cases}$$

(c)
- Eine Σ–Rechenstruktur A heißt *initial* in einer Klasse von Σ–Algebren C, falls für jede Σ–Algebra B in C genau ein Homomorphismus von A nach B existiert.
- Eine Σ–Rechenstruktur Z heißt *terminal* (*final*) in einer Klasse C von Σ–Algebren, falls für jede Σ–Algebra B in C genau ein Homomorphismus von B nach Z existiert.

In einer initialen Σ–Algebra sind zwei Elemente verschieden, wenn ihre Gleichheit nicht explizit (ausgedrückt durch Axiome) gefordert ist.

In einer terminalen Σ–Algebra sind zwei Elemente gleich, wenn ihre Verschiedenheit nicht explizit (ausgedrückt durch Axiome) gefordert ist.

In der Menge der Σ-Algebren für Prioritätsschlangen ist Q initial und P terminal.

Aufgabe 3 Datenmodellierung [LÖSUNG]

(a) Stacks haben die Signatur $\Sigma = (S, F)$.

Die Menge S der Sortennamen besteht aus St, der Sorte der Stacks, Nat, der Sorte der natürlichen Zahlen, und $Bool$, der Sorte der Wahrheitswerte. Somit ist $S = \{St, Nat, Bool\}$.

Die Menge der Funktionsnamen mit ihren Funktionalitäten ist

$$
\begin{array}{llll}
F = \{ & estack: & St, & \text{„leerer Stack“} \\
& push: & St, Nat \rightarrow St, & \text{„Push“} \\
& top: & St \rightarrow Nat, & \text{„Oberstes Element“} \\
& pop: & St \rightarrow St, & \text{„Löschen“} \\
& isEStack: & St \rightarrow Bool, & \text{„Überprüfung ob leer“} \\
& ...\} & &
\end{array}
$$

Lässt man die Bezeichnungen der Sorten und der Funktionssymbole ausser Acht, so erkennt man, dass die Funktionssignaturen mit denen der Prioritätsschlangen übereinstimmen. Somit lässt sich allein aus der Signatur noch kein Unterschied erkennen.

(b) Wir geben nun zwei Modelle R und S an, die zu der Signatur passen. Für die St wählen wir in beiden Modellen Sequenzen über natürlichen Zahlen $\mathbb{N}$ Nat wird mit $\mathbb{N}$ und Bool mit $\mathbb{B}$ belegt.

Modell R der nach oben wachsenden Stacks.

Die Funktionen sind wie folgt definiert:

$$
\begin{aligned}
estack^R &= \epsilon \\
push^R(s, m) &= s \circ < m > \\
top^R(s) &= n && \text{falls } s = s' \circ < n > \\
pop^R(s) &= s' && \text{falls } s = s' \circ < n > \\
isEStack^R(s) &= true && \text{falls } s = estack \\
isEStack^R(s) &= true && \text{sonst}
\end{aligned}
$$

Modell S der beliebigen Sequenzen. Die Funktionen sind wie folgt definiert:

$$
\begin{aligned}
estack^S &= \epsilon \\
push^S(s, m) &= < m > \circ s \\
top^S(s) &= n && \text{falls } s = < n > \circ s' \\
pop^S(s) &= s' && \text{falls } s = < n > \circ s' \\
isEStack^S(s) &= true && \text{falls } s = estack \\
isEStack^S(s) &= true && \text{sonst}
\end{aligned}
$$

(c) Typische Eigenschaften von Prioritätsschlangen:

Wir stellen auf systematische Weise Eigenschaften zusammen: St-Terme werden durch die Funktionen *estack* und *push* aufgebaut und durch die Funktionen *pop* und *top* zerlegt. Wir suchen nach Gleichungen, die (von links nach rechts angewendet) Terme $top(t)$ und $pop(t)$ reduzieren. Dazu setzen wir für t zum einen *estack* und zum anderen $push(s, n)$ ein. Auf diese Weise erhalten wir das folgende Axiomensystem für Prioritätsschlangen:

$s \ : \ Pq \setminus \{\perp\}, \ n : Nat \ :$

$$top(push(s,n)) = n,$$
$$pop(push(s,n)) = n,$$

$$isEStack(estack()) = true$$
$$isEStack(push(s,n)) = true$$

(d) Homomorphismus

Der Homomorphismus φ bildet aufsteigende Stacks vom Modell R auf absteigende Stacks im Modell S ab:

$$\varphi : \ St^R \cup \mathbb{N} \ \rightarrow St^S \cup \mathbb{N}$$

Sei φ auf $\mathbb{N}$ die Identität, d.h. für die Homomorphiefunktion zur Sorte Nat gilt:

$$\varphi_{Nat}(n) = n, \quad n \in \mathbb{N}$$

Auf St^R lauten die Bedingungen der Funktionenverträglichkeit:

$$\varphi(estack^R) = estack^S$$

$$\varphi(push^R(s,n)) = push^P(\varphi(s), n)$$

$$\varphi(top^R(s)) = top^S(\varphi(s))$$

$$\varphi(pop^R(s)) = pop^S(\varphi(s))$$

Die folgende Funktion $\varphi(s)$ erfüllt die Abbildung, indem sie die Elemente eines aufsteigenden Stacks $s \in St^R$ absteigend ordnet

$$\varphi(s) = \begin{cases} s & \text{, falls} \quad s = \epsilon \\ top^R(s) \circ \varphi(pop^R(s)) & \text{, sonst} \end{cases}$$