Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Datenstrukturen und Algorithmen (SS 2013)

Übungsblatt 2

Abgabe: Montag, **29.04.2013**, 14:00 Uhr

- Die Übungen sollen in Gruppen von zwei bis drei Personen bearbeitet werden.
- Schreiben Sie die Namen jedes Gruppenmitglieds sowie alle Matrikelnummern auf die abgegebenen Lösungen.
- Schreiben Sie die Namen jedes Gruppenmitglieds sowie alle Matrikelnummern auch in die Quellcode-Dateien.
- Geben Sie Ihre Lösungen am **Anfang** der Globalübung, montags, 14:00 Uhr, ab.
- Schicken Sie den jeweiligen Quellcode bitte per **E-Mail** direkt an Ihre/n Tutor/in.
- Geben Sie außerdem den ausgedruckten Quellcode zusammen mit den schriftlichen Lösungen ab.
- Zu spät abgegebene Lösungen werden nicht bewertet.
- Sofern nicht anders gefordert, müssen alle Lösungen und Zwischenschritte kommentiert werden.

Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Aufgabe 1 (*Abstrakte Datentypen* [**10 Punkte**])

Gegeben sei folgende Signatur eines abstrakten Datentyps:

Sorten: $\mathbb{R}$, A

Funktionen:

$$\begin{array}{lll} \mathrm{c} & : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} & \to \mathrm{A} \\ \mathrm{s} & : \mathbb{R} \times \mathrm{A} & \to \mathrm{A} \\ + & : \mathrm{A} \times \mathrm{A} & \to \mathrm{A} \\ \times & : \mathrm{A} \times \mathrm{A} & \to \mathrm{A} \\ \mathrm{d} & : \mathrm{A} & \to \mathbb{R} \\ \mathrm{t} & : \mathrm{A} & \to \mathrm{A} \end{array}$$

Axiome: $\forall a \in \mathrm{A}\ \forall i, j \in \mathbb{R}$

$$\begin{array}{lll} (1) & \mathrm{s}(f, \mathrm{c}(w,x,y,z)) & = \mathrm{c}(f \cdot w, f \cdot x, f \cdot y, f \cdot z) \\ (2) & +(\mathrm{c}(w,x,y,z), \mathrm{c}(i,j,k,l)) & = \mathrm{c}(w+i, x+j, y+k, z+l) \\ (3) & \times(\mathrm{c}(w,x,y,z), \mathrm{c}(i,j,k,l)) & = \mathrm{c}(w \cdot i + x \cdot k, \\ & & \quad w \cdot j + x \cdot l, \\ & & \quad y \cdot i + z \cdot k, \\ & & \quad y \cdot j + z \cdot l) \\ (4) & \mathrm{d}(\mathrm{c}(w,x,y,z)) & = w \cdot z - x \cdot y \\ (5) & \mathrm{t}(\mathrm{c}(w,x,y,z)) & = \mathrm{c}(w,y,x,z) \end{array}$$

Die Grundrechenarten können dabei als elementar betrachtet werden.

(a) Beweisen oder widerlegen Sie folgende Behauptung [**2 Punkte**]:

$$\forall i,j,k,l,w,x,y,z \in \mathbb{R} : \times(\mathrm{c}(w,x,y,z), \mathrm{c}(i,j,k,l)) = \times(\mathrm{c}(i,j,k,l), \mathrm{c}(w,x,y,z))$$

(b) Beweisen oder widerlegen Sie folgende Behauptung [**3 Punkte**]:

$$\begin{aligned} \forall w,x,y,z,f \in \mathbb{R} : \ & \mathrm{s}\Big(f, \mathrm{s}\Big(f, \mathrm{s}\Big(\mathrm{d}\big(\mathrm{t}(\mathrm{c}(w,x,y,z))\big), \mathrm{c}\big(1,0,0,1\big)\Big)\Big)\Big) \\ = \ & \mathrm{s}\Big(\mathrm{d}\Big(\mathrm{s}\big(f, \mathrm{c}(w,x,y,z)\big)\Big), \mathrm{c}(1,0,0,1)\Big) \end{aligned}$$

(c) Beschreiben Sie den Datentyp und dessen Funktionen mit eigenen Worten. [**2 Punkte**]

(d) Definieren Sie einen ADT $\mathrm{STACK_A}$, welcher einen Stack für Objekte des Datentyps A darstellt. Neben den üblichen Stackoperationen soll der Datentyp des Weiteren eine Funktion $\mathrm{contains_d}$ bereitstellen, welche prüft, ob der Stack für einen Skalar $f \in \mathbb{R}$ ein Objekt $a \in \mathrm{A}$ enthält mit $\mathrm{d}(a) = f$. Geben Sie sämtliche Sorten, Funktionen und Axiome an, welche für die Definition des ADT $\mathrm{STACK_A}$ notwendig sind. [**3 Punkte**]

Prof. Dr. Leif Kobbelt
Thomas Ströder, Fabian Emmes
Sven Middelberg, Michael Kremer

Informatik 8
RWTH Aachen

Lösungsvorschlag

(a) Anwendung von Axiom (3) auf beiden Seiten führt zu:

$$\begin{aligned} & \mathrm{c}(w \cdot i + x \cdot k, w \cdot j + x \cdot l, y \cdot i + z \cdot k, y \cdot j + z \cdot l) \\ = \ & \mathrm{c}(i \cdot w + j \cdot y, i \cdot x + j \cdot z, k \cdot w + l \cdot y, k \cdot x + l \cdot z) \ . \end{aligned} \tag{1}$$

Sei nun z. B. $w = y = z = i = j = l = 0$ und $x = k = 1$. Einsetzen in Gleichung (1) liefert:

$$\mathrm{c}(1,0,0,0) \ = \ \mathrm{c}(0,0,0,1) \ .$$

Da dies offensichtlich falsch ist, ist die Behauptung widerlegt. □

(b) Anwendung der Axiome auf beiden Seiten führt zu:

$$\begin{aligned}
& \mathrm{s}\Big(f, \mathrm{s}\Big(f, \mathrm{s}\Big(\mathrm{d}\big(\mathrm{t}(\mathrm{c}(w,x,y,z))\big), \mathrm{c}\big(1,0,0,1\big)\Big)\Big)\Big) \\
= \ & \mathrm{s}\Big(\mathrm{d}\Big(\mathrm{s}\big(f, \mathrm{c}(w,x,y,z)\big)\Big), \mathrm{c}(1,0,0,1)\Big) \\
\Rightarrow \quad & \mathrm{s}\Big(f, \mathrm{s}\Big(f, \mathrm{s}\Big(\mathrm{d}\big(\mathrm{c}(w,y,x,z)\big), \mathrm{c}\big(1,0,0,1\big)\Big)\Big)\Big) && \text{Axiom (5)} \\
= \ & \mathrm{s}\Big(\mathrm{d}\Big(\mathrm{c}(f \cdot w, f \cdot x, f \cdot y, f \cdot z)\Big), \mathrm{c}(1,0,0,1)\Big) && \text{Axiom (1)} \\
\Rightarrow \quad & \mathrm{s}\Big(f, \mathrm{s}\Big(f, \mathrm{s}\Big(w \cdot z - x \cdot y, \mathrm{c}\big(1,0,0,1\big)\Big)\Big)\Big) && \text{Axiom (4)} \\
= \ & \mathrm{s}\Big(f^2 \cdot (w \cdot z - x \cdot y), \mathrm{c}(1,0,0,1)\Big) && \text{Axiom (4)} \\
\Rightarrow \quad & \mathrm{s}\Big(f, \mathrm{s}\Big(f, \mathrm{c}(w \cdot z - x \cdot y, 0, 0, w \cdot z - x \cdot y)\Big)\Big) && \text{Axiom (1)} \\
= \ & \mathrm{c}(f^2 \cdot (w \cdot z - x \cdot y), 0, 0, f^2 \cdot (w \cdot z - x \cdot y)) && \text{Axiom (1)} \\
\Rightarrow \quad & \mathrm{s}\Big(f, \mathrm{c}(f \cdot (w \cdot z - x \cdot y), 0, 0, f \cdot (w \cdot z - x \cdot y))\Big) && \text{Axiom (1)} \\
= \ & \mathrm{c}(f^2 \cdot (w \cdot z - x \cdot y), 0, 0, f^2 \cdot (w \cdot z - x \cdot y)) \\
\Rightarrow \quad & \mathrm{c}(f^2 \cdot (w \cdot z - x \cdot y), 0, 0, f^2 \cdot (w \cdot z - x \cdot y)) && \text{Axiom (1)} \\
= \ & \mathrm{c}(f^2 \cdot (w \cdot z - x \cdot y), 0, 0, f^2 \cdot (w \cdot z - x \cdot y))
\end{aligned}$$

Damit ist die Behauptung bewiesen. □

(c) Der Datentyp repräsentiert 2×2 Matrizen, wobei die Funktionen den folgenden Matrixoperatoren entsprechen:

c(w, x, y, z) Konstruiert die 2×2 Matrix $\begin{pmatrix} w & x \\ y & z \end{pmatrix}$.

s(f, M) Skaliert die Matrix M uniform mit Faktor f.

$+(M_1, M_2)$ Addiert die Matrizen M_1 und M_2.

$\times(M_1, M_2)$ Multipliziert die Matrizen M_1 und M_2.

d(M) Berechnet die Determinante der Matrix M.

t(M) Transponiert die Matrix M.

(d) Definition des Datentyps STACK_A:

Sorten: $\mathbb{R}$, BOOL, A, STACK_A

Funktionen:

create	:		$\rightarrow \text{STACK}_\text{A}$
push	:	$\text{A} \times \text{STACK}_\text{A}$	$\rightarrow \text{STACK}_\text{A}$
pop	:	STACK_A	$\rightarrow \text{STACK}_\text{A}$
top	:	STACK_A	$\rightarrow$ A
empty	:	STACK_A	$\rightarrow$ BOOL
contains_d	:	$\mathbb{R} \times \text{STACK}_\text{A}$	$\rightarrow$ BOOL

Axiome: $\forall a \in \text{A} \;\; \forall i, j \in \mathbb{R}$

(1)	pop(create())	$=$ error
(2)	pop(push(a, s))	$= s$
(3)	top(create())	$=$ error
(4)	top(push(a, s))	$= a$
(5)	empty(create())	$=$ true
(6)	empty(push(a, s))	$=$ false
(7)	$\text{contains}_\text{d}(f, \text{create}())$	$=$ false
(8)	$\text{contains}_\text{d}(f, \text{push}(a, s))$	$\Leftrightarrow (\text{d}(a) = f \;\vee \text{contains}_\text{d}(f, s))$

Aufgabe 2 (*Labyrinth* [**10 Punkte**])

In der Vorlesung wurde ein iterativer Algorithmus zum Lösen von Labyrinthen vorgestellt, der nur unter der Annahme funktioniert, dass das Labyrinth keine Zyklen enthält.
Erweitern Sie den Algorithmus, so dass er auch auf Labyrinthen mit Zyklen funktioniert.

(a) Geben Sie Ihren erweiterten Algorithmus in Pseudocode an und beschreiben Sie die Idee Ihrer Erweiterung knapp. [**5 Punkte**]

(b) In der auf der Homepage bereitgestellten Datei `Labyrinth.java` findet sich eine Implementierung des Algorithmus der Vorlesung und ein beispielhaftes Labyrinth. Das Labyrinth enthält allerdings einen Zyklus, daher terminiert der ursprüngliche Algorithmus hier nicht. Erweitern Sie die Methode `sucheIt`, welche den Algorithmus der Vorlesung implementiert, um Ihre Idee. Testen Sie, ob der Algorithmus nun das Ziel findet. [**5 Punkte**]

Bitte geben Sie nur die von Ihnen bearbeitete Datei `Labyrinth.java` ab.

Lösungsvorschlag

(a) Eine mögliche Lösung ist es, den Algorithmus um ein Array mit Booleschen Werten zu ergänzen, welche markieren, ob ein Feld schon besucht wurde.

Wir führen ein Hilfsprädikat $\mathsf{V} : \mathsf{W} \times \mathsf{R} \times \mathsf{Bool}^{\mathsf{m}\times\mathsf{n}} \to \mathsf{Bool}$ ein ((m, n) ist die Größe des Labyrinths), so dass $\mathsf{V}(p, r, v)$ genau dann wahr ist, wenn das Nachbarfeld von Feld p in Richtung r in v als besucht markiert ist.

Der folgende Pseudocode zeigt den erweiterten Algorithmus. Die ergänzten Stellen sind hierbei unterstrichen.

```
P ← Pbegin
S ← Create()
r ← 'N'
v ← Bool[n][m] /* Boolesches Array, initialisiert mit false. */
while P ≠ Pend and ((L(P,r) and V(P,r,v))
                    or r < 'W' or not Empty(S)) do
    while P ≠ Pend and L(P,r) and V(P,r,v) and
                        (S = ∅ or r ≠ −Top(S)) do
        S = Push(r,S)
        v[P.x][P.y] ← true
        P ← Go(P,r)
        r ← 'N'
    if P ≠ Pend then
        while r = 'W' and not Empty(S) do
            r = Top(S)
            P ← Go(P,−r)
            S ← Pop(S)
```

```
if r < 'W' then
  r ← next(r)
```

(b) Die Datei `Labyrinth.java` finden Sie zum Download auf der Homepage.