Formale Grundlagen der Informatik 3

Prof. Dr. Christoph Walther / Visar Januzaj, Nathan Wasser

Technische Universität Darmstadt — Wintersemester 2011/12

Lösungsvorschlag zu Hausübung 1

Aufgabe 1.1 (Implementieren und Spezifizieren) (12 Punkte)

Betrachten Sie die folgende Datenstruktur für S-Expressions.

```
structure sexpr[@I] <=
  nil,
  atom(data : @I),
  cons(car : sexpr[@I], cdr : sexpr[@I])
```

(a) Schreiben Sie eine Prozedur `maximum(s : sexpr[ℕ]) : ℕ`, die das größte Element eines S-Expressions natürlicher Zahlen berechnet. Falls der S-Expression keine Elemente besitzt, soll 0 geliefert werden.

Lösungsvorschlag

```
function maximum(s : sexpr[ℕ]) : ℕ <=
if ?nil(s)
  then 0
  else if ?atom(s)
         then data(s)
         else if maximum(car(s)) > maximum(cdr(s))
                then maximum(car(s))
                else maximum(cdr(s))
              end_if
       end_if
end_if
```

(b) Schreiben Sie eine Prozedur `isIn(i : @I, s : sexpr[@I]) : bool`, die überprüft, ob `i` im S-Expression `s` vorkommt.

Lösungsvorschlag

```
function isIn(i : @I, s : sexpr[@I]) : bool <=
if ?nil(s)
  then false
  else if ?atom(s)
         then i = data(s)
         else if isIn(i, car(s))
                then true
                else isIn(i, cdr(s))
              end_if
       end_if
end_if
```

(c) Formulieren Sie die folgenden Eigenschaften von `maximum` durch $\mathcal{L}$-Lemmata:

- Das Maximum eines S-Expressions ist 0, oder Element des S-Expressions.
- Das Maximum eines S-Expressions ist größer oder gleich jedem Element des S-Expressions.

Lösungsvorschlag

```
lemma maximum is in s-expression or zero <= ∀ s : sexpr[ℕ]
  if{isIn(maximum(s), s), true, ?0(maximum(s))}

lemma maximum is greater or equal <= ∀ s : sexpr[ℕ], n : ℕ
  if{isIn(n, s), ¬ n > maximum(s), true}
```

Aufgabe 1.2 (Kalküle) (17 Punkte)

Betrachten Sie die Sprache $L = \{w \mid |w|_a = |w|_b = |w|_c\} \subset \{a,b,c\}^*$. Sie sollen Kalküle entwerfen, um das Wortproblem von L zu lösen. In Ihren Kalkülregeln dürfen $|w|_a, |w|_b$ und $|w|_c$ **nicht** vorkommen.

(a) Geben Sie einen nicht-deterministischen Kalkül an, um für Wörter aus $\{a,b,c\}^*$ festzustellen, ob das Wort in L ist. Als Sprache ihres Kalküls soll $\{a,b,c\}^*$ genommen werden.

Lösungsvorschlag

Wir definieren die folgenden Regeln:

$$\text{(abc)}\ \frac{\alpha\beta\gamma\delta}{\alpha a\beta b\gamma c\delta} \quad \text{(acb)}\ \frac{\alpha\beta\gamma\delta}{\alpha a\beta c\gamma b\delta} \quad \text{(bac)}\ \frac{\alpha\beta\gamma\delta}{\alpha b\beta a\gamma c\delta} \quad \text{(bca)}\ \frac{\alpha\beta\gamma\delta}{\alpha b\beta c\gamma a\delta} \quad \text{(cab)}\ \frac{\alpha\beta\gamma\delta}{\alpha c\beta a\gamma b\delta} \quad \text{(cba)}\ \frac{\alpha\beta\gamma\delta}{\alpha c\beta b\gamma a\delta}$$

Eine Herleitung zu einem Wort α ist eine Sequenz von Wörtern aus Σ^*, die mit α endet und in der jedes Wort aus dem vorhergehenden durch Anwendung einer Regel hervorgeht.

Ein Wort ist in L gdw. es eine mit dem leeren Wort beginnende Herleitung dazu gibt.

(b) Geben Sie einen deterministischen Kalkül an, um für Wörter aus $\{a,b,c\}^*$ festzustellen, ob das Wort in L ist. Als Sprache ihres Kalküls soll $\{a,b,c\}^* \times \mathbb{N}^3$ genommen werden. Ihr Kalkül soll zeichenweise vorgehen.

Lösungsvorschlag

Wir definieren die folgenden Regeln: $$(1)\ \frac{a\alpha, (x,y,z)}{\alpha, (x+1,y,z)} \quad (2)\ \frac{b\alpha, (x,y,z)}{\alpha, (x,y+1,z)} \quad (3)\ \frac{c\alpha, (x,y,z)}{\alpha, (x,y,z+1)}$$

Eine Herleitung zu einem Wort α ist eine Sequenz von Paaren aus Wörtern und Zahlentripeln, die mit $\alpha, (0,0,0)$ beginnt und in der jedes weitere Paar aus der vorhergehenden durch Anwendung einer Regel hervorgeht.

Ein Wort α ist in L genau dann, wenn es eine Herleitung zu α gibt, die mit dem leeren Wort und einem Tripel der Form x, x, x endet.

(c) Lösen Sie die folgenden Instanzen des Wortproblems für L mit Hilfe Ihrer Kalküle, d. h. geben Sie jeweils an, ob das Wort in L ist und belegen Sie dies durch Angabe geeigneter Herleitungen beider Kalküle.

- abac

Lösungsvorschlag

Das Wort ist nicht in der Sprache, denn keine Herleitung endet mit dem leeren Wort und einem geeigneten Tripel:
$(abac, (0,0,0)) \Rightarrow_{(1)} (bac, (1,0,0)) \Rightarrow_{(2)} (ac, (1,1,0)) \Rightarrow_{(1)} (c, (2,1,0)) \Rightarrow_{(3)} (\epsilon, (2,1,1))$

Das Wort ist nicht in der Sprache, denn keine Herleitung beginnt mit dem leeren Wort: $a \Rightarrow_{bac} abac$
$a \Rightarrow_{abc} abac$.

- bcaacb

Lösungsvorschlag

Das Wort ist in der Sprache, wie die folgende Herleitung belegt:
$(bcaacb, (0,0,0)) \Rightarrow_{(2)} (caacb, (0,1,0)) \Rightarrow_{(3)} (aacb, (0,1,1)) \Rightarrow_{(1)} (acb, (1,1,1)) \Rightarrow_{(1)} (cb, (2,1,1)) \Rightarrow_{(3)} (b, (2,1,2)) \Rightarrow_{(2)} (\epsilon, (2,2,2))$

$\epsilon \Rightarrow_{bac} bac \Rightarrow_{cab} bcaacb$

(d) **(Bonus)** Geben Sie einen deterministischen Kalkül an, um für Wörter aus $\{a,b,c\}^*$ festzustellen, ob das Wort in L ist. Als Sprache ihres Kalküls soll $\{a,b,c\}^* \times \{A,B,C\}$ genommen werden. Ihr Kalkül soll zeichenweise vorgehen.

Lösungsvorschlag

Wir definieren die folgenden Regeln:

$$\text{(A1)}\ \frac{a\alpha, A}{\alpha, B} \quad \text{(A2)}\ \frac{x\alpha, A}{\alpha x, A}\ ,\ x \in \{b,c\}$$

$$\text{(B1)}\ \frac{b\alpha, B}{\alpha, C} \quad \text{(B2)}\ \frac{x\alpha, B}{\alpha x, B}\ ,\ x \in \{a,c\}$$

$$\text{(C1)}\ \frac{c\alpha, C}{\alpha, A} \quad \text{(C2)}\ \frac{x\alpha, C}{\alpha x, C}\ ,\ x \in \{a,b\}$$

Eine Herleitung zu einem Wort α ist eine Sequenz von Paaren aus Wörtern und Elementen von $\{A,B,C\}$, die mit α, A beginnt und in der jedes weitere Paar aus der vorhergehenden durch Anwendung einer Regel hervorgeht.

Ein Wort α ist in L genau dann, wenn es eine Herleitung zu α gibt, die mit (ε, A) endet.

Aufgabe 1.3 (Testen von Lemmata) (11 Punkte)

Betrachten Sie die Prozedur `mix`:

```
function mix(k : list[@I], l : list[@I]) : list[@I] <=
if ?ø(k)
  then l
  else if ?ø(l)
         then k
         else hd(k) :: mix(tl(l), tl(k))
       end_if
end_if
```

Die folgenden Lemmata lassen sich in ✓`eriFun` mit „Generate Lemma“ aus dem Menü „Program“ für `mix` erzeugen. Zur Lösung dieser Aufgabe sollten Sie jedoch ✓`eriFun` nicht benötigen.

(a) `lemma mix is associative <= ∀ x, y, z : list[@I]`
```
mix(mix(x, y), z) = mix(x, mix(y, z))
```

(b) `lemma mix is commutative <= ∀ x, y : list[@I]`
```
mix(x, y) = mix(y, x)
```

(c) `lemma mix is idempotent <= ∀ x, y : list[@I]`
```
if{x = y, mix(x, y) = x, true}
```

(d) `lemma mix is left-cancelable <= ∀ x1, x2, y, z : list[@I]`
```
if{x1 = x2, if{mix(x1, y) = mix(x2, z), y = z, true}, true}
```

(e) `lemma mix is right-cancelable <= ∀ x1, x2, y, z : list[@I]`
```
if{x1 = x2, if{mix(y, x1) = mix(z, x2), y = z, true}, true}
```

(f) `lemma mix is injective <= ∀ x1, x2, y1, y2 : list[@I]`
```
if{mix(x1, x2) = mix(y1, y2), if{x1 = y1, x2 = y2, false}, true}
```

Geben Sie für jedes dieser Lemmata an, ob es gilt. Geben Sie für jedes ungültige Lemma ein Gegenbeispiel in Form einer widerlegenden Substitution an.

Lösungsvorschlag

(a) Das Lemma gilt.

(b) Das Lemma gilt nicht: $\{x/0 :: \emptyset, y/1 :: \emptyset\}$

(c) Das Lemma gilt.

(d) Das Lemma gilt nicht: $\{x1/0 :: \emptyset, x2/0 :: \emptyset, y/\emptyset, z/0 :: \emptyset\}$

(e) Das Lemma gilt nicht: $\{x1/0 :: \emptyset, x2/0 :: \emptyset, y/\emptyset, z/0 :: \emptyset\}$

(f) Das Lemma gilt nicht: $\{x1/0 :: \emptyset, x2/\emptyset, y1/\emptyset, y2/0 :: \emptyset\}$