Algorithmen und Datenstrukturen

4. Übungsblatt

Zeitraum: 10. November – 14. November 2014

Übung 1 (AGS 2.2.38)
Sei $\mathcal{E} = (V, \Sigma, S, R)$ mit $V = \{S, B\}$, $\Sigma = \{b\}$ und $R = \{\ S ::= [B]\ b,\ B ::= Sb\}$ eine EBNF–Definition. Berechnen Sie die syntaktischen Kategorien $W(\mathcal{E}, S)$ und $W(\mathcal{E}, B)$ mit Hilfe der Fixpunktsemantik. Gehen Sie dazu in den folgenden Schritten vor:

(a) Dokumentieren Sie mindestens 5 Iterationsschritte,
(b) und schreiben Sie in Mengenschreibweise die Sprachen $W(\mathcal{E}, S)$ und $W(\mathcal{E}, B)$ auf.
(c) Zeigen Sie mithilfe der Semantik von EBNF-Termen, dass die Sprachen $W(\mathcal{E}, S) = W(\mathcal{E}, A) = \{(ba)^n b \mid n \in \mathbb{N}\}$ die EBNF–Regel $S ::= \hat{\{}ba\hat{\}}A$ erfüllen.

Übung 2 (AGS 2.2.43)
Für die folgenden Teilaufgaben nehmen wir das Alphabet $\Sigma = \{a, b, c\}$ an.

(a) Gegeben seien die Menge $V = \{A\}$ der syntaktischen Variablen sowie die Funktion $\rho : V \to \mathcal{P}(\Sigma^*)$ mit $\rho(A) = \{a^n b \mid n \geq 0\}$. Zeigen Sie die Gültigkeit der Gleichung

$$[\![\hat{(}\hat{[}a\hat{]}A \hat{|}\ b\hat{)}]\!](\rho) = \{a\}^* \cdot \{b\},$$

indem Sie *schrittweise* die über den induktiven Aufbau von EBNF-Termen definierten Regeln zur Bestimmung der Objektsprache anwenden und letztere mathematisch zulässig umformen!

(b) Wir betrachten die Sprache

$$L = \{(ab)^n w c^{2n+1} \mid n \geq 0, w \in \Sigma^*\}\ .$$

Entwerfen Sie ein endliches System $\mathcal{U}$ von Syntaxdiagrammen, so dass $\mathcal{U}$ die Sprache L erzeugt! Kennzeichnen Sie das entsprechende Startsymbol.

(c) Nun seien die syntaktischen Variablen $V = \{S, A, B\}$ gegeben, sowie die EBNF-Definition $\mathcal{E} = (V, \Sigma, S, R)$ mit Regelmenge R wie folgt:

$$S ::= \hat{\{}A\hat{\}}A$$
$$A ::= \hat{(}cAc\ \hat{|}\ B\hat{)} \qquad\qquad B ::= \hat{[}a\hat{]}b$$

Wandeln Sie $\mathcal{E}$ gemäß der Übersetzungsvorschrift *trans* aus der Vorlesung in ein System von Syntaxdiagrammen um, welches die gleiche Sprache $W(\mathcal{E})$ erzeugt. Sie müssen *keine Zwischenschritte* angeben!

Übung 3 (AGS 3.1.5)
Schreiben Sie für die Berechnung der Fakultätsfunktion $n! \mapsto 1 \cdot \ldots \cdot n$, $n \in \mathbb{N}$, ein C-Programm, welches eine natürliche Zahl als Eingabe fordert und den zugeordneten Funktionswert ausgibt.

Übung 4

Ein ganzzahliger Geldbetrag in Euro soll in die Grundeinheiten 500, 200, 100, 50, 20, 10, 5, 2 und 1 zerlegt werden.

Schreiben Sie ein C-Programm, welches einen Geldbetrag als Eingabe fordert, dann die Vielfachheiten der oben genannten Grundeinheiten berechnet und das Ergebnis auf dem Bildschirm ausgibt. Entwickeln Sie einen solchen Algorithmus, der ein Minimum an Geldscheinen und Münzen erzeugt. Geben Sie zunächst den Algorithmus in Pseudocode an.

Zusatzaufgabe 1 (AGS 2.2.39)

(a) Geben Sie eine EBNF-Definition $\mathcal{E}'$ an, so dass gilt: W($\mathcal{E}'$)= $\{a^{i+j}b^{j+k+l}ac^{2l} \mid i,j,k,l \geq 0\}$

(b) Sei $\mathcal{E} = (V, \Sigma, S, R)$ mit $V = \{S, A\}$, $\Sigma = \{a, b\}$ und $R = \{\ S ::= aAb,\ A ::= [S] \mid b\}$. Berechnen Sie die syntaktischen Kategorien $W(\mathcal{E}, S)$ und $W(\mathcal{E}, A)$ mit Hilfe der Fixpunktsemantik. Gehen Sie dazu in den folgenden Schritten vor:

- Dokumentieren Sie 5 Iterationsschritte.
- Schreiben Sie in Mengenschreibweise die Sprachen $W(\mathcal{E}, S)$ und $W(\mathcal{E}, A)$ auf.

Zusatzaufgabe 2 (AGS 3.1.9)

Es soll geprüft werden, ob der Inhalt des Zeichenfeldes `feld` der Länge `l` ein Palindrom ist, das heißt ob die Zeichenkette sowohl von vorne als auch von hinten gelesen gleich lautet. Ist die Zeichenkette ein Palindrom, so soll der Parameter `korrekt` den Wert `true` repräsentieren, sonst `false`.

(a) Von folgender Funktion wird behauptet, dass sie diese Aufgabe löst:

```
palindrom1(char feld[], int l, int korrekt) {
  int i;
  i = 1;
  l = l - 1;
  while (i < l && korrekt) {
    korrekt = feld[i] == feld[l];
    i = i + 1;
  }
  return korrekt;
}
```

Prüfen Sie diese Funktion. Korrigieren Sie eventuell enthaltene Fehler.

(b) Schreiben Sie eine Funktion `palindrom2`, die rekursiv arbeitet. Überlegen Sie sich hierbei als erstes einen geeigneten Funktionskopf.