Algorithmen und Datenstrukturen

14. Übungsblatt

Zeitraum: 29. Januar – 2. Februar 2018

Übung 1 (AGS 2.2.52)

(a) Sei $\Sigma = \{a, b, c, d\}$. Geben Sie die Mengen V und R einer EBNF-Definition $\mathcal{E} = (V, \Sigma, S, R)$ an, so dass $W(\mathcal{E}) = \left\{ a^{n+m} b^m c^\ell d^{2n+1} \;\middle|\; n, \ell \geq 0, m \geq 1 \right\}$.

(b) Sei $\Sigma = \{a, b\}$ und $\mathcal{E} = (V, \Sigma, X, R)$ eine EBNF-Definition mit $V = \{X, Y\}$ sowie

$$R = \left\{ X ::= \hat{[} aY \hat{]}, \quad Y ::= \hat{(} X b \hat{|} X b b \hat{)} \right\}.$$

Geben Sie die für die Fixpunktsemantik von $\mathcal{E}$ zu iterierende Funktion f an. Dokumentieren Sie die ersten fünf Iterationsschritte der Fixpunktsemantik von $\mathcal{E}$.

(c) Geben Sie die Sprache $W(\mathcal{E})$ an, die durch die EBNF-Definition $\mathcal{E}$ beschrieben wird.

Übung 2 (AGS 3.2.33)

Gegeben sei die folgende Typdefinition für binäre Bäume:

```
typedef struct node *tree;
struct node { int value;  tree left, right; };
```

(a) Schreiben Sie eine Funktion `int ident(tree t1, tree t2)`, die den Wert 1 zurückgibt, wenn die Bäume `t1` und `t2` gleich sind, ansonsten den Wert 0 (*gleich* heißt: die gleiche Struktur und an entsprechenden Positionen die gleichen Werte).

(b) Schreiben Sie eine Funktion `int internal(tree t)`, die die Anzahl der Knoten von `t` zählt, die *keine* Blattknoten sind.

(c) Schreiben Sie eine Funktion `void lRot(tree *tp)`, die an der Wurzel des Eingabebaums eine einfache Linksrotation (im Sinne der AVL-Bäume) durchführt. Sie können davon ausgehen, dass der Wurzelknoten einen rechten Kindknoten hat. *Beachten Sie, dass sich der Wurzelknoten des Eingabebaums durch diese Funktion ändern kann.*

Übung 3 (AGS 6.1.12)

Wenden Sie den Quicksort-Algorithmus auf die Folge 2, 1, 3, 6, 5, 4 an. Die Zahlen sollen aufsteigend sortiert werden.

Übung 4 (AGS 6.2.10)

Wenden Sie auf die Folge: 5, 7, 25, 13, 6, 12, 27, 20, 8, 4 den Heapsort-Algorithmus an. Sie müssen in Phase zwei nur zwei Doppelschritte ausführen!

Übung 5 (AGS 4.21)

Gegeben sei folgendes C-Programm:

```c
1  #include <stdio.h>
2
3  void g(int d, int* e);
4
5  void f(int* a, int* b) {
6    int c;
7    c = *a;
8    /* label1 */
9    while (c < *b) {
10     c = c + 1;
11     /* label2 */
12     g(c, b); /* $1 */
13     *a = c;
14     /* label3 */
15   }
16 }
17
18 void g(int d, int* e) {
19   /* label4 */
20   if (d < *e) {
21     *e = *e - 1;
22     /* label5 */
23     f(&d, e); /* $2 */
24   }
25   /* label6 */
26 }
27
28 int main() {
29   int m, n;
30   m = 2;
31   n = 4;
32   /* label7 */
33   f(&m, &n); /* $3 */
34   /* label8 */
35   return 0;
36 }
```

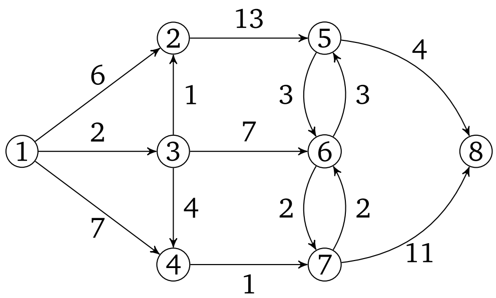
(a) Bestimmen Sie den Gültigkeitsbereich jedes Objektes. Nutzen Sie dazu die Zeilennummern.

(b) Setzen Sie das folgende Speicherbelegungsprotokoll fort.

Haltepunkt	RM	1	2	3	4	5	6	7	8	9	10
label7	–	m 2	n 4								

Übung 6 (AGS 9.5.10)

Der kantenbewertete Graph G sei durch folgende graphische Darstellung gegeben:



(a) Berechnen Sie mit Hilfe des Dijkstra-Algorithmus die minimalen Entfernungen vom Knoten mit der Nummer 1 zu allen erreichbaren Knoten. Protokollieren Sie schrittweise die aktuelle Randknotenmenge und den zugehörigen Auswahlknoten. Geben Sie abschließend für alle berechneten kürzesten Wege die jeweils zu durchlaufende Knotenfolge (Pfadtabelle) an.

(b) Geben Sie die Pfadtabelle für eine weitere Lösung an, die die gleichen minimalen Entfernungen wie Teilaufgabe (a) besitzt. An welcher Stelle des Algorithmus im Protokoll von **(a)** ist diese weitere Lösung erkennbar?

Übung 7 (AGS 7.12)

(a) Bestimmen Sie die mit Hilfe des KMP-Algorithmus berechnete Verschiebetabelle für das Pattern `aabaaabb`.

(b) Mit Hilfe des KMP-Algorithmus ist die unten stehende unvollständige Verschiebetabelle berechnet worden. Die mit einem „?“ markierte Einträge sind unbekannt. Vervollständigen Sie das aus den Symbolen `a`, `b` und `c` bestehende Pattern.

Position	0	1	2	3	4	5
Pattern	b					c
Tabelle	-1	?	1	-1	?	2

Übung 8

(a) Bestimmen Sie die Levenshtein-Distanz $d(\text{ros}, \text{horse})$. Geben Sie die Berechnungsmatrix vollständig an.

(b) Wieviele Backtraces enthält die Berechnungsmatrix?

(c) Geben Sie alle Alignments zwischen ‘ros’ und ‘horse’ an.

Übung 9 (AGS 10.14)

Die Personen A und B spielen ein Spiel. In jeder Runde werden zwei Zufallszahlen x und y bestimmt, wobei x Werte aus der Menge $\{1,2\}$ und y aus $\{1,2,3\}$ annehmen kann. Der Spieler A gewinnt die Runde, falls die Summe $x + y$ eine ungerade Zahl ist. Ansonsten gewinnt B die Runde. Die Menge der möglichen Ergebnisse ist somit $X = \{1,2\} \times \{1,2,3\}$.

(a) Geben Sie den Analysator A für dieses Szenario an, also A(„A gewinnt“) und A(„B gewinnt“).

Die Spieler spielen 87 Runden des Spiels und A gewinnt dabei 60 Mal. Geben Sie den Korpus h mit unvollständigen Daten an, den sie erzeugen, also h(„A gewinnt“) und h(„B gewinnt“).

(b) Wir wollen aus diesem Korpus h mit dem EM-Algorithmus die Wahrscheinlichkeitsverteilungen der beiden Zufallszahlen bestimmen. Die initiale Wahrscheinlichkeitsverteilung $q_0 = q_0^x \times q_0^y$ ist gegeben durch $q_0^x(1) = 2/3$ und $q_0^y(1) = q_0^y(2) = 1/5$. Dabei ist q_0^x die Wahrscheinlichkeitsverteilung von x und q_0^y die von y. Bestimmen Sie die Werte $q_0(i, j)$ für $1 \leq i \leq 2$ und $1 \leq j \leq 3$. Hinweis: $q_0(2,3) = 1/5$.

(c) Führen Sie den E-Schritt aus, vervollständigen Sie also den Korpus h zum Korpus h_1. Geben Sie für alle $1 \leq i \leq 2$ und $1 \leq j \leq 3$ die Werte $h_1(i, j)$ an. Hinweis: $h_1(2,3) = 30$.

(d) Führen Sie nun den M-Schritt aus. Bestimmen Sie dafür zunächst die Teilkorpora h_1^x und h_1^y für x bzw. y. Geben Sie also für alle $1 \leq i \leq 2$ und $1 \leq j \leq 3$ die Werte $h_1^x(i)$ bzw. $h_1^y(j)$ an.

(e) Schätzen Sie nun die Wahrscheinlichkeitsverteilungen q_1^x sowie q_1^y von x bzw. y, indem Sie die relative Häufigkeit der Teilkorpora bestimmen. Geben Sie dafür wieder für alle $1 \leq i \leq 2$ und $1 \leq j \leq 3$ die Werte $q_1^x(i)$ bzw. $q_1^y(j)$ an.