Probeklausur zur Vorlesung

Algorithmen und Programmierung III

Wolfgang Mulzer WiSe 2013/14

Bearbeitungszeit: 90 Minuten.

Aufgabe 1 Implementierungen eines ADT *2+2+3+3 Punkte*

Betrachten Sie die folgende Spezifikation eines abstrakten Datentypen: Sei $\mathcal{U}$ ein total geordnetes Universum. Es sollen Teilmengen $S \subseteq U$ gespeichert werden, so dass folgende Operationen möglich sind:

- `insert`(x): Voraussetzung: keine. Wirkung: $S \mapsto S \cup x$.
- `deleteMin`(): Voraussetzung: S ist nicht leer. Wirkung: $S \mapsto S \setminus \min S$.
- `deleteMax`(): Voraussetzung: S ist nicht leer. Wirkung: $S \mapsto S \setminus \max S$.

Sie dürfen annehmen, dass sich zwei Elemente aus $\mathcal{U}$ in konstanter Zeit vergleichen lassen. Für jede der folgenden vier Datenstrukturen, beschreiben Sie jeweils kurz, wie man die Operationen `deleteMin` und `deleteMax` implementieren kann, und welche asymptotischen Laufzeiten man erhält. Erklären Sie gegebenenfalls, welche zusätzlichen Annahmen nötig sind.

(a) Sortierte doppelt verkettete Liste mit Zeigern auf das erste und das letzte Element;

(b) AVL-Baum;

(c) binärer Min-Heap; und

(d) unkomprimierter Trie.

Aufgabe 2 Huffman-Kodes *1+6+3 Punkte*

(a) Wann heißt ein Kode $C : \Sigma \to \{0,1\}^*$ präfixfrei? Wieso ist diese Definition nützlich?

(b) Sei $\Sigma = \{a, b, c, d, e, f\}$ mit Häufigkeiten $a : 12$, $b : 62$, $c : 80$, $d : 74$, $e : 25$ und $f : 86$. Benutzen Sie den Algorithmus von Huffman, um einen Kode für Σ zu den gegebenen Häufigkeiten zu bestimmen. Zeigen Sie die einzelnen Schritte.

(c) Bewerten Sie die folgende Aussage: “Mit Huffmankodes kann man eine gegebene Datei optimal komprimieren.”

Aufgabe 3 Vermischtes *3+2+2+3 Punkte*

(a) Was ist ein Entwurfsmuster? Was ist ein Algorithmus? Nennen Sie eine Gemeinsamkeit und einen Unterschied.

(b) Was ist der Unterschied zwischen dem dynamischen und dem statischen Datentypen einer Variable? Geben Sie ein kurzes Beispielfragment in Java.

(c) Unter welchen Umständen ist eine Datenstruktur, die $O(\log n)$ *amortisierte* Laufzeit für eine Operation benötigt, einer Datenstruktur vorzuziehen, die $O(\log n)$ *worst-case* Laufzeit für die gleiche Operation benötigt?

(d) In der Vorlesung haben Sie mehrere Algorithmen gesehen, die dynamisches Programmieren verwenden. Warum findet man in diesen Algorithmen oft erst den *Wert* einer optimalen Lösung, bevor man eine optimale Lösung selbst berechnet?

Aufgabe 4 Skiplisten *4+3+3 Punkte*

(a) Beweisen Sie: Die erwartete Größe einer Skipliste mit n Schlüsseln ist $O(n)$.

(b) Seien L_1 und L_2 zwei Skiplisten, welche jeweils die Schlüsselmenge K_1 und K_2 speichern. Beschreiben Sie einen effizienten Algorithmus, der aus L_1 und L_2 eine Skipliste für die Menge $K_1 \cup K_2$ konstruiert. Analysieren Sie die erwartete Laufzeit Ihres Algorithmus. (Sie können dabei die Aussage von (a) verwenden.)

(c) Nehmen Sie nun bei (b) zusätzlich an, dass für alle $k_1 \in K_1$, $k_2 \in K_2$ gilt: $k_1 < k_2$. Das heißt, jeder Schlüssel in L_1 ist kleiner als alle Schlüssel in K_2. Geben Sie nun einen Algorithmus an, der L_1 und L_2 in erwarteter Zeit $O(\max\{\log|K_1|, \log|K_2|\})$ vereinigt, und beweisen Sie die Laufzeitgarantie.
Hinweis: Für alle $x \in (0,1)$ gilt: $\sum_{i=1}^{\infty} ix^i = x/(1-x)^2$.