Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr
Matthias Baumgart

Wintersemester 2007/2008
Übungsblatt 7
30. November 2007

Effiziente Algorithmen und Datenstrukturen I

Abgabetermin: 07.12.2007 **vor** *der Vorlesung*

Aufgabe 1 (10 Punkte)

Mit Hilfe einer Priority Queue PQ lassen sich n Schlüssel sortieren, indem man diese zunächst in PQ einfügt und anschließend n mal ExtractMin ausführt.

Implementieren Sie einen solchen Sortieralgorithmus mit Hilfe von Binomial-Heaps.

Die zu sortierende Schlüsselmenge sei in einer Datei enthalten, deren Name beim Aufruf angeben wird. Jede Zeile dieser Datei enthält genau einen Schlüssel. Die Ausgabe soll anschließend in einer Datei gespeichert werden, deren Name ebenfalls beim Ausführen angeben wird. Senden Sie Ihren Quelltext per Email an `baumgart@in.tum.de`.

Aufgabe 2 (10 Punkte)

(a) Fügen Sie die Schlüssel $2, 3, 6, 7, 5, 1, 4$ in dieser Reihenfolge in einen anfangs leeren Splay-Tree ein. Zeichnen Sie den Baum jeweils nach dem Einfügen.

(b) Betrachten Sie den Splay-Tree aus Aufgabe (a) nach dem Einfügen des Schlüssels 5. Geben Sie eine Folge von Zugriffsoperationen an, die diesen Splay-Tree degenerieren lässt, das heißt, wir erhalten durch diese Folge von Operationen eine lineare Liste.

Zeichnen Sie die dabei auftretenden Splay-Trees.

Aufgabe 3 (10 Punkte)

Schlagen Sie eine Erweiterung der 2-Level-Buckets PQ vor, so dass auch Werte effizient eingefügt werden können, die unter dem minimalen Wert liegen. Führen Sie eine amortisierte Analyse Ihrer Lösung mit einer geeignet gewählten Potentialfunktion durch.

Aufgabe 4 (10 Punkte)

Verallgemeinern Sie die Idee der 2-Level-Buckets auf k-Level-Buckets. Die amortisierte Laufzeit von Insert und DecreaseKey soll wieder $O(1)$ sein, die von ExtractMin nur noch $O(\sqrt[k]{C})$, wobei C wieder die obere Schranke für die Differenz zwischen dem größten und dem kleinsten Schlüssel sein soll.

Hinweis: Sie können zur Vereinfachung annehmen, dass nur Schlüssel aus der Menge $\{0, \ldots, C-1\}$ in den k-Level-Bucket eingefügt werden