Technische Universität München
Institut für Informatik
Lehrstuhl für Effiziente Algorithmen
Volker Heun

SS 1999
Übungsblatt 3
1. Juni 1999

Grundlegende Algorithmen

Abgabetermin: 8. Juni

Aufgabe 1

Man kann einen neuen Heap auch wie folgt aufbauen. Am Anfang ist ein Baum mit einem Knoten (der Wurzel) ein Heap. Nun ergänzt man in jeder Phase den fast vollständigen binären Baum um ein Blatt. Den Inhalt dieses Blattes läßt man nun in dem Baum soweit durch Vertauschungen nach oben wandern, bis die Heap-Eigenschaft wiederhergestellt ist.

Beschreiben Sie diesen Algorithmus zum Aufbau eines Heaps möglichst detailliert (etwa in Pseudo-C) und analysieren Sie dessen Komplexität (Anzahl der benötigten Vergleiche).

Aufgabe 2

Sei M eine n-elementige Menge ganzer Zahlen. Zeigen Sie, daß $n-1$ Vergleiche ausreichen, um das Minimum von M zu bestimmen. Zeigen Sie außerdem, daß im allgemeinen mindestens $n-1$ viele Vergleiche notwendig sind.

Aufgabe 3

Betrachten Sie die folgende Idee für einen Sortieralgorithmus auf einem Feld der Größe n. Zu Beginn wird das minimale Element aller Elemente bestimmt und dieses minimale Element mit dem ersten Element des Feldes vertauscht. Dann wird das Minimum der restlichen Elemente bestimmt und dieses Element mit dem zweiten Element des Feldes vertauscht, u.s.w. Allgemein bestimmt man also in der i-ten Phase das Minimum der $n-i+1$ größten Elemente und vertauscht dieses mit dem i-ten Element des Feldes.

Beschreiben Sie diesen Algorithmus möglichst detailliert (etwa in Pseudo-C) und analysieren Sie dessen Komplexität (Anzahl der benötigten Vergleiche).

Was hat dieser Algorithmus mit HeapSort zu tun?

Aufgabe 4

Betrachten Sie den folgenden Sortieralgorithmus auf $n = k^m$ Elementen (für eine Konstante k). Zuerst werden die Elemente in k^{m-1} Gruppen erster Ordnung mit jeweils k Elementen aufgeteilt. Diese Gruppen erster Ordnung werden dann in k^{m-2} Gruppen zweiter Ordnung mit jeweils k Gruppen erster Ordnung aufgeteilt, u.s.w. Allgemein gibt es also k^{m-i} Gruppen i-ter Ordnung, die jeweils aus k Gruppen $(i-1)$-ter Ordnung gebildet wurden.

Zuerst wird in jeder Gruppe das Minimum gebildet. Dabei ist das Minimum einer Gruppe erste Ordnung das Minimum seiner Elemente. Das Minimum einer Gruppe i-ter Ordnung (für $i > 1$) ist das Minimum der Minima der Gruppen $(i-1)$-ter Ordnung, die in dieser Gruppe i-ter Ordnung enthalten sind. Damit ist das Minimum der Gruppe m-ter Ordnung genau das Minimum aller Elemente. Dieses wird nun entfernt und in allen Gruppen, die dieses Element enthielten, das Minimum neu gebildet.

Beschreiben Sie diesen Algorithmus möglichst detailliert (etwa in Pseudo-C) und analysieren Sie dessen Komplexität (Anzahl der benötigten Vergleiche).

Was hat dieser Algorithmus mit HEAPSORT zu tun?