Technische Universität München
Institut für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Angelika Steger

WS 1996/7
Übungsblatt 8
8. Januar 1996

Effiziente Algorithmen und Datenstrukturen

Abgabe am 15. Januar 1996 bis 8:00 Uhr im Briefkasten

Aufgabe 29

Als eine Anwendung der Backward-Radixsort-Idee betrachten Sie den folgenden Kartentrick: Aus einem Stapel von 27 Karten merkt sich der Zuschauer eine bestimmte; außerdem nennt er dem Zauberer eine Zahl n zwischen 1 und 27. Der Zauberer macht nun drei Runden lang folgendes: Er macht aus dem gegebenen Stapel der 27 mit dem Rücken nach oben liegenden Karten drei neue Stapel, indem er eine Karte nach der anderen vom gegebenen Stapel nimmt und sie der Reihenfolge nach mit dem Gesicht nach oben auf die drei neu zu bildenden Stapel legt, die erste links, die zweite in die Mitte, die dritte rechts, die vierte links, die fünfte in die Mitte, usw., bis die drei neuen Stapel mit jeweils neun Karten gefüllt sind. Dann läßt er sich vom Zuschauer sagen, in welchem der drei Stapel die gemerkte Karte liegt, woraufhin er die drei Stapel einfach zu einem Stapel aufeinander stellt, diesen Stapel umdreht (so daß die Karten wieder mit dem Rücken nach oben liegen) und mit der nächsten Runde fortfährt. Nach Beenden der dritten Runde (die Karten bilden einen Stapel mit dem Rücken nach oben) nimmt der Zauberer unter Mitzählen die Karten der Reihe nach vom Stapel; die n-te Karte ist genau diejenige, die sich der Zuschauer gemerkt hat.
Erklären Sie, wie dieser Trick funktioniert, insbesondere, wie der Zauberer in jeder der drei Runden die drei Stapel aufeinanderstellen muß, damit die gewünschte Karte am Ende an der gewünschten Position zu finden ist.

Aufgabe 30

a) Zeigen Sie: Ist T ein Binärbaum mit k Blättern, so ist die durchschnittliche Tiefe von T mindestens $\log_2 k$.
(Die Tiefe eines Blattes b ist die Anzahl der Kanten des Pfades von b zur Wurzel. Die durchschnittliche Tiefe eines Baumes ist die Summe der Tiefen seiner Blätter geteilt durch die Anzahl der Blätter.)

b) Zeigen Sie: Jedes auf Vergleichsoperationen beruhende Verfahren zum Sortieren von n verschiedenen Schlüsseln benötigt *im Durchschnitt* $\Omega(n \log n)$ Schlüsselvergleiche.
HINWEIS: Versuchen Sie, Teil a) anzuwenden.

Aufgabe 31

Betrachten Sie die folgende Variante von QUICKSORT:

if $r - l \leq m$ then
    INSERTIONSORT(a, l, r)
else
    $\langle$wie bisher$\rangle$

Für die durchschnittliche Anzahl C_n von Schlüsselvergleichen gilt dann:

$$C_n = \begin{cases} n + 1 + \frac{1}{n} \sum_{j=1}^{n} (C_{j-1} + C_{n-j}) & \text{für } n > m \\ \frac{1}{4} n(n-1) & \text{für } n \leq m \end{cases}$$

a) Lösen Sie diese Rekursion analog zu dem in der Vorlesung vorgestellten Verfahren.
b) Bestimmen Sie eine Funktion $f(m)$, so daß die durchschnittliche Anzahl der Vergleiche $2n \log_2 n + n \cdot f(m) + o(n)$ ist. Wie sollte man also m wählen?

Aufgabe 32

Implementieren Sie den in der Vorlesung vorgestellten Algorithmus für Forward-Radixsort. Diese Aufgabe entspricht der ersten Aufgabe des Programmierwettbewerbes, mit dem Unterschied, daß nun die Laufzeit des Referenzprogrammes nicht geschlagen werden braucht, dafür aber vorgeschrieben ist, welcher Algorithmus verwendet werden muß. Genauere Spezifikationen finden Sie im WWW unter der URL

`http://wwwmayr.informatik.tu-muenchen.de/lehrva/1996WS/ea-ueb/ea-sort.html`.

Das Referenzprogramm, einen Rahmen für die Ein- und Ausgabe (den Sie nicht unbedingt verwenden müssen) sowie eine Test-Eingabe finden Sie auf den hphalle-Rechnern unter

`~kuehnle/programmierwettbewerb/sortieren/`.