Institut für Informatik
der Universität München
Prof. M. Hofmann
Dipl. Inf. Hermann Gruber

SS 2007

15. Mai 2006

Übungen zur Vorlesung
Effiziente Algorithmen

Blatt 4

Aufgabe P-5: Zeigen Sie, dass jeder vergleichsbasierte Sortieralgorithmus im *durschnittlichen* Fall $\Omega(n \log n)$ Vergleiche benötigt, um eine Permutation der Liste $(1, 2, \ldots, n)$ zu sortieren.

Hinweis: Finden Sie eine untere Schranke für die Summe der Längen aller Pfade von der Wurzel zu einem Blatt in einem beliebigen Binärbaum mit N Blättern.

Hausaufgaben

Aufgabe H-13: Zeigen Sie, dass es *kein* vergleichbasiertes Sortierverfahren gibt, das mindestens die Hälfte der $n!$ möglichen Eingaben der Länge n in linearer Zeit sortiert. Gibt es ein Verfahren, das jede $n$te Eingabe der Länge n in linearer Zeit sortiert? Oder wenigstens jede 2^nte? *Hinweis:* Verallgemeinern Sie die Technik, die Sie in der Vorlesung gehört haben (Sie dürfen annehmen, dass sich der Algorithmus auf den restlichen Eingaben beliebig verhält).
(6 Punkte)

Aufgabe H-14:

a) Beschreiben und analysieren Sie einen Algorithmus mit (Worst-Case-) Zeitkomplexität $O(n)$, der alle Listen korrekt sortiert, deren Einträge nur aus Booleschen Werten, d.h. 0 und 1 bestehen.

b) Verallgemeinern Sie diesen Algorithmus zu einem Algorithmus mit Laufzeit $O(n)$ und Platzbedarf $O(n)$, der alle Listen korrekt sortiert, deren Einträge alle in der Menge $\{0, 1, 2, \ldots, c-1, c\}$ (für c konstant) liegen. Argumentieren Sie, warum der Algorithmus korrekt ist, und beweisen Sie die obere Schranke an Laufzeit und Platzbedarf.

c) Widerspricht die Laufzeit ihres Algorithmus der in der Vorlesung bewiesenen unteren Schranke von $\Omega(n \log n)$? Begründen Sie Ihre Antwort.

(8 Punkte)

Aufgabe H-15: Es sei das folgende Problem gegeben: Gegeben eine Liste von n verschieden großen Elementen und eine Zahl i, gib die kleinsten i Elemente aus. Analysieren Sie die Zeitkomplexität der folgenden Lösungsvorschläge bezüglich n und i:

a) Man sortiert die Liste, und gibt dann die i ersten Elemente aus.

b) Man konstruiert eine Priority Queue und extrahiert die i kleinsten Elemente.

c) Man bestimmt ein Element von Rang i, und partitioniert geeignet.

(6 Punkte)

Abgabe: Bis Dienstag, 22.5.2006, 09:15 Uhr im Abgabekasten in der Resi-39.