3. Übung Informatik I

Marcus Rickert

30. Dezember 1995

Aufgabe 2

Teil a) Algorithmus

Der Algorithmus entspricht im Prinzip exakt dem des Quicksort-Algorithmus. Er ist folgendermaßen aufgebaut:

- Am Anfang wird überprüft, ob die Menge der Schlüssel einelementig ist. Wenn dies der Fall ist, so steht der Schlüssel eindeutig fest und kann sofort an die übergeordnete Ebene zurüeckgegeben werden.
- Wenn die Menge größer ist, so wird nach dem üblichen Verfahren in drei Gruppen geteilt:
  1. Elemente $links$ bis $k-1$
  2. Element k und
  3. Elemente $k+1$ bis $rechts$,

  wobei alle Schlüssel einer Gruppe jeweils kleiner sind als die der nächsten.
- Nun wird der gewünschte Index des Schlüssels mit k verglichen und nur die Gruppe weiterbearbeitet, die den Index enthält. Die Routine wird rekursiv für diese Gruppe aufgerufen und nachher der Rückgabewert an die übergeordnete Ebene zurückgegeben.

Für die genaue Struktur siehe Listing.

Teil b) Laufzeit im worst-case

Die Laufzeit im worst-case entspricht der des Quicksort-Algorithmus wegen folgender Überlegung:

Sei eine Menge (Input) mit n Elementen gegeben. Im ungünstigsten Fall ist immer das linke Element der Menge das kleinste oder das größte. Dann braucht ein der Algorithmus $n-1$ Vergleiche bis die Menge aufgeteilt werden kann. Anschließend wird in eine 1-elementige und eine $n-1$-elementige Gruppe geteilt, wobei im worst-case der gewünschte Index in der größeren Gruppe liegt. Man erhält also

$$T(n) = \sum_{i=1}^{n-1} i - 1 = (n-1)^2$$

Vergleiche und kommt somit auf eine Laufzeit von $O(n^2)$.

Teil c) Durchschnittliche Laufzeit

Bei der durchschnittlichen Laufzeit kann man annehmen, daß normalerweise ebensoviele Schlüssel kleiner wie größer als der linke Schlüssel sind. Die Größe der Menge würde sich also bei jeder Ebene halbieren. Desweiteren kann man vereinfachend sagen, daß im Durchschnitt jedes zweite Element links mit jedem zweiten Element rechts vertauscht werden muß. Wenn n die Anzahl der Elemente ist, so ergibt sich für die Laufzeit:

$$T(n) = \sum_{i=0}^{\log(n)} \left(\frac{1}{2}\right)^i n = n \underbrace{\sum_{i=0}^{\log(n)} \left(\frac{1}{2}\right)^i}_{\leq 2} \leq 2n$$

Die Laufzeit nähert sich also schnell $O(n)$.

Aufgabe 3

Teil a)

Sei N die Anzahl der Knoten des Graphen G. Angenommen die Aussage ist falsch, dann hat jeder Knoten mindestens einen Vorgänger. Wähle einen Knoten $n_0 \in G$ aus. Suche für diesen den Vorgänger, der laut Annahme existiert, und nenne diesen $n_1 \in G$. Da der Graph nicht zyklisch ist, muß gelten $n_1 \in G/\{n_0\}$. Wiederhole diesen Schritt für $n_1, n_2, \ldots$ und so weiter. Es gilt immer, daß $n_i \in G/\{n_0, \ldots, n_{i-1}\}$. Bei $i > N$ führt dies jedoch zum Widerspruch: n_{N+1} existiert laut Annahme, die Menge $G/\{n_0, \ldots, n_N\}$ ist aber leer. Also war die Annahme falsch und es existiert ein Knoten aus G, der keinen Vorgänger hat. Q.E.D.