Datenstrukturen und Algorithmen: Blatt 6

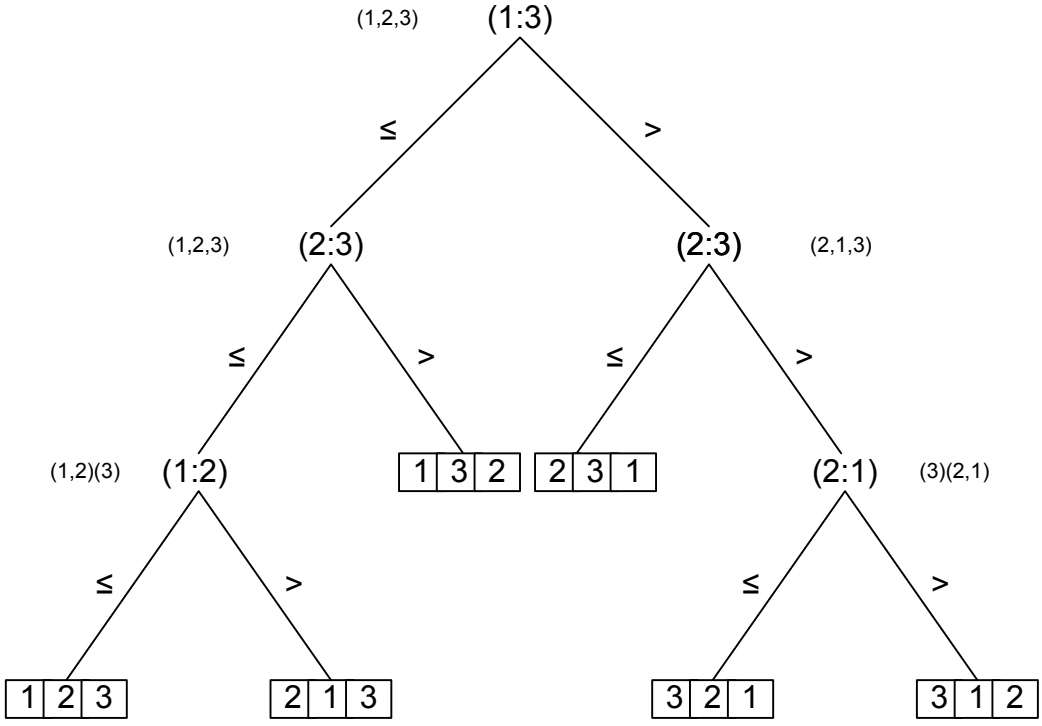
Bernhard Dietrich (6256800)
Lars Fernhomberg (6256030)
Sebastian Kniesburges (6257120)
Marcus Köthenbürger (6258550)

Übungsgruppe 11
Mittwoch, 11:00-13:00 Uhr in D1.303
Matthias Ernst

Aufgabe	1	2	3	Σ	Korrektor
Punkte					
von	6	6	6	18	

Aufgabe 1

Bestimmen Sie den Entscheidungsbaum für Quicksort bei $n = 3$.



Aufgabe 3

Gegeben seien n Zahlen im Intervall $[0...k]$. Beschreiben Sie einen Algorithmus, der in Zeit $O(n+k)$ Zusatzinformationen zur Eingabe berechnet, so dass künftige Anfragen nach der Zahl der Eingabezahlen in einem Intervall $[a...b]$, $0 \le a \le b \le k$, in Zeit $O(1)$ beantwortet werden kann.

Die Ausgangsidee unseres Algorithmus ist, dass in einem ersten Schritt das gegebenen Array aus n Zahlen durchgegangen wird, und dabei in einem zweiten Array, welches $k+1$ Elemente (entsprechend dem Eingabeintervall von 0 bis k) hat, die Häufigkeit der einzelnen Zahlen gespeichert wird. In einem zweiten Schritt wird nun das zweite – mit der Häufigkeit gefüllte Array – durchlaufen, wobei in jedem Feld des Arrays gespeichert wird, wie oft die Zahlen von 0 bis einschließlich dem jeweils aktuellem Feld aufgetreten sind. Da die Arrays im Pseudocode mit 1 anfangen, lässt sich hierbei allerdings eine Verschiebung nicht vermeiden, so dass sich die Häufigkeit der Zahl 0 an der Position 1, die Häufigkeit der Zahl 1 an der Position 2, …, die Häufigkeit der Zahl k an der Position $k+1$ befindet. Die

Häufigkeit eines bestimmten Intervalls $[a...b]$ mit $0 \leq a \leq b \leq k$ lässt sich anschließend durch $B[b+1] - B[a+1]$ ermitteln, wobei B das angesprochene zweite Array ist.

Eingabe: Ein Array A, in welchem Zahlen aus dem Intervall $[0...k]$ gespeichert sind, sowie die größte im Array befindliche Zahl.

Ausgabe: Array B, in welchem die kumulierte Häufigkeit der einzelnen Zahlen gespeichert ist, wobei die einzelnen Zahlen um eine Stelle verschoben sind (vgl. Erklärungstext)

```
NUMBERCOUNT(A,K)
1.    for i ← 1 to length[A]
2.          do B[A[i]+1] ← B[A[i]+1] + 1
3.    for j ← 2 to length[B]
4.          do B[j] ← B[j] + B[j-1]
5.    return B
```

Korrektheitsbeweis:

1. for-Schleife

Invariante: Das Array B enthält die Häufigkeitsverteilung für die im Intervall $[1...i]$ des Arrays A vorkommenden Zahlen.

Initalisierung: Vor Beginn der for-Schleife wurde noch kein Bereich des Arrays A bearbeitet, so dass Array B noch keine Häufigkeitsverteilung enthalten kann.

Erhaltung: Bei jedem Durchlauf wird die Häufigkeit für die jeweilige Zahl `A[i]` im Array B um eins erhöht, wobei die Zahl `A[i]` ihrem Nachfolger `A[i]+1` zugeordnet wird. Alle anderen Elemente des Arrays B bleiben unberührt.

Terminierung: Nach dem Schleifendurchlauf enthält das Array B die Häufigkeit des Vorkommens der jeweiligen Zahlen in Array A.

2. for-Schleife

Invariante: Das Array B enthält die aufsummierte Anzahl von Vorkommen der Zahlen $[0...j]$ aus dem Array B.

Initalisierung: Für $j = 1$ gibt es keine Vorgänger, die aufsummiert werden können, so dass `B[1]` unverändert bleibt.

Erhaltung: Dem Element `j` wird der Wert seines Vorgängers hinzuaddiert (wobei dieser, außer bei `j=1`, bereits in einem vorherigen Schleifendurchlauf aufsummiert wurde).

Terminierung: Da bei jedem Schleifendurchlauf eine Summe aus der Anzahl aller vorherigen Vorkommen und der jeweils aktuellen Anzahl aller Vorkommen zugewiesen wird, enthält das Array nach Beendigung der Schleife die kumulierte Häufigkeit bis zu dem jeweiligen Element.

Da sowohl die erste, als auch die zweite Schleife wie spezifiziert arbeiten, arbeitet der Algorithmus insgesamt korrekt.

Aufwandsabschätzung: Die erste `for`-Schleife (Zeile 1 & 2) wird n mal durchlaufen, sodass sich hierfür $O(n)$ ergibt. Die zweite `for`-Schleife (Zeile 3 & 4) wird k mal durchlaufen, da $k+1 = length[B]$. Da die Schleife jedoch von mit dem Wert $j = 2$ startet und dann bis $k+1$ läuft, wird die Schleife insgesamt k mal durchlaufen, so dass sich $O(k)$ ergibt.

Es ergibt sich somit insgesamt $O(n) + O(k) = O(n+k)$.

Aufgabe 2

Was ist die geringste Tiefe eines Blattes in einem Entscheidungsbaum eines Vergleichssortieres für n Eingabezahlen?