Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr
Tobias Lieber

Wintersemester 2010/11
Übungsblatt 10
18. Januar 2011

Effiziente Algorithmen und Datenstrukturen

Abgabetermin: 25. Januar 2011 vor der Vorlesung

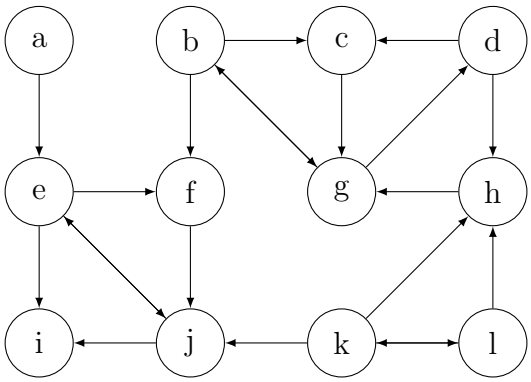
Aufgabe 1 (10 Punkte)

Sei ein Array $A[0..n-1]$ mit n paarweise verschiedenen natürlichen Zahlen gegeben. Ein Paar (i, j) heißt *Inversion* (bezüglich der aufsteigenden Sortierung), wenn $i < j$ und $A[i] > A[j]$ gilt.

Beschreiben Sie einen Algorithmus, der die Anzahl der Inversionen in einem Array $A[0..n-1]$ in Laufzeit $O(n \log n)$ bestimmt.

Aufgabe 2 (10 Punkte)

Führen Sie im folgenden gerichteten Graphen die Tiefensuche (DFS) aus.



Starten Sie die Tiefensuche mit einem bisher noch nicht besuchten Knoten neu, falls es keine ausgehenden Kanten mehr gibt.
Explorieren Sie dabei den Graphen entsprechend der *lexikographischen Ordnung* auf den Knotennamen, d.h. wenn Sie in einem Schritt unter mehreren Knoten wählen können, verwenden Sie den Knoten, der in der lexikographischen Ordnung vor allen anderen steht.

Beispiel: Angenommen man würde bei Knoten b starten, dann ist c der nächste betrachtete Knoten, da er unter allen Nachbarn von b (c, f und g) in der lexikographischen Ordnung der erste Knoten ist.

Aufgabe 3 (10 Punkte)

In der Vorlesung wurde angesprochen, dass der QuickSort-Algorithmus so implementiert werden kann, dass die repeat-until-Schleife nur $n - 1$ Elementvergleiche benötigt.

1. Berechnen Sie die Anzahl der Elementvergleiche an, die die repeat-until-Schleife in dem QuickSort-Algorithmus aus der Vorlesung im schlechtesten Fall benötigt.

Geben Sie auch ein passendes Beispiel der Größe 6 (mit Begründung) an, für die der schlechteste Fall erreicht wird.

2. Geben Sie eine Modifikation des vorgestellten QuickSort-Algorithmus an, so dass die repeat-until-Schleife nur $n - 1$ Elementvergleiche benötigt.

Aufgabe 4 (10 Punkte)

Eine Topologische Sortierung eines DAG $G = (V, A)$ weisst jedem Knoten $v \in V$ eine eindeutige Nummer $f(v) \in \{1, 2, \ldots, |V|\}$ zu, so dass für jede Kante $(u, v) \in A$ gilt: $f(u) < f(v)$.

- Geben Sie einen Linearzeit-Algorithmus in Pseudocode an, der nicht auf der Tiefensuche basiert, der eine topologische Sortierung eines DAGs G berechnet.

Nun nehmen wir an, dass uns weder die üblichen Graphoperationen zum Traversieren einer Kante oder Abfragen der Nachbarschaft eines Knotens noch irgendwelche „höheren Datenstrukuren“ wie z.B. Queues, Listen, PriorityQueues oder Stacks zur Verfügung stehen.
Die einzige Operation, die wir benutzen können, ist eine Tiefensuche (DFS), die Arrays `dfs_num` und `finish_num` zurückgibt, die zu den jeweiligen Knoten die entsprechenden Werte beeinhalten. Außerdem stehen uns `for`-Schleifen und die übliche Addition bzw. Subtraktion zur Verfügung.

a) Geben Sie einen Algorithmus (Pseudocode reicht) an der wie, der obige Algorithmus auch nur $\mathcal{O}(n)$ Zeit beansprucht.

b) Begründen Sie die Laufzeit Ihres Algorithmus kurz.

c) Beweisen Sie die Korrektheit Ihres Algorithmus.