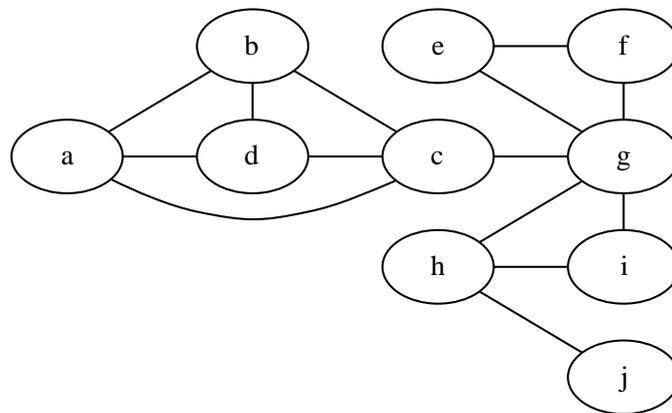


Theoretische Informatik I

6. Übung

Geben Sie die Lösungen der Aufgaben 1a, 1b und 2 bitte bis zum 23.11.2015 9:15 Uhr ab. (Briefkasten vorm Raum 1/266 oder per eMail an fal@informatik.tu-chemnitz.de, *Betreff: TI1 Hausaufgaben*)

1. Aufgabe: Der folgende Graph G sei Ihnen in Adjazenzlistendarstellung gegeben. Dabei sind alle Listen *alphabetisch* geordnet.



- Bestimmen Sie den l -Wert (=low-Wert) jedes Knotens. Führen Sie dazu die modifizierte Tiefensuche durch und beginnen Sie bei Knoten c .
- Begründen Sie anhand des low -Wertes, welche Knoten Artikulationspunkte sind.
- Geben Sie die zweifachen Zusammenhangskomponenten des Graphen so aus, wie es der Algorithmus der Vorlesung tut.

2. Aufgabe: Sei $G = (V, E)$ ein beliebiger ungerichteter Graph. *Widerlegen* Sie die folgenden Aussagen. Geben Sie dazu Beispielgraphen mit den dazugehörigen Tiefensuchbäumen sowie den d , f und low -Werten an.

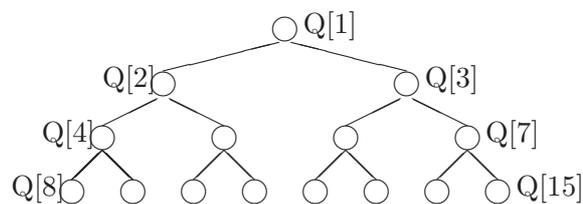
- Alle Knoten einer zweifachen Zusammenhangskomponente haben immer den gleichen low -Wert.
- Verschiedene Artikulationspunkte haben stets verschiedene low -Werte.
- Knoten in verschiedenen Komponenten haben immer verschiedene low -Werte.

3. Aufgabe: Wir betrachten einen Heap, der in einem Array $Q[1..n]$ implementiert ist. Zeigen Sie folgende *Vater-Sohn-Beziehung*:

Steht in einem Heap ein Element an Position i , dann steht der linke Sohn des Elements an Position $2 \cdot i$ und der rechte Sohn an Position $2 \cdot i + 1$.

4. Aufgabe: Zeigen Sie, dass der folgende Algorithmus die Laufzeit $O(n)$ hat.

Gegeben sei ein Feld $Q[1..n]$ von Elementen. Wir wollen aus Q einen *Heap* aufbauen. Wir nehmen $n = 2^k - 1$ an und betrachten Q als Baum (z. B. $k = 4$):



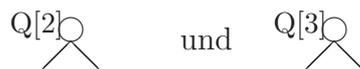
Der Heap wird nun *bottom-up* aufgebaut. Das heißt, zuerst wird in dem Teil



die Heap-Eigenschaft hergestellt, indem $Q[4]$ an die richtige Stelle sickert. Genauso verfahren wir mit den Teilen



Danach lassen wir $Q[2]$ bzw. $Q[3]$ an die richtige Stelle sickern, um in den Teilbäumen



die Heapeigenschaft herzustellen. Schließlich entsteht durch Sickern von $Q[1]$ ein korrekter Heap.

Hinweis: Überprüfen Sie, wie oft ein Knoten aus Ebene i maximal sickern kann, bis er an der richtigen Stelle steht. Summieren Sie diesen Wert über alle Knoten. Die entstehende Summenformel kann auf eine geometrische Reihe zurückgeführt werden, wenn man die Ungleichung $i < 1,5^i$ benutzt.