

Institut für Theoretische Informatik
Peter Widmayer
Thomas Tschager
Antonis Thomas

20. April 2016

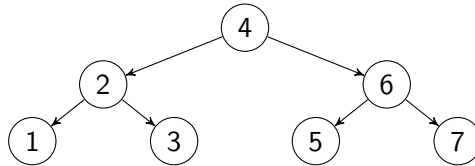
Datenstrukturen & Algorithmen

Blatt 8

FS 16

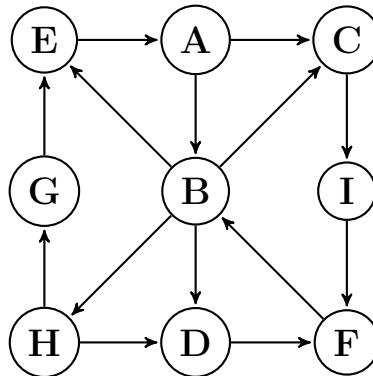
Aufgabe 8.1 *Optimale Suchbäume und Splay-Bäume.*

Gegeben sind die Schlüssel $1, 2, \dots, 7$ mit den Zugriffshäufigkeiten $1, 4, 1, 8, 1, 4, 1$. Auf Schlüssel 3 wird beispielsweise einmal zugegriffen und auf Schlüssel 4 achtmal. Es gibt keine Zugriffe auf Intervalle zwischen den Schlüsseln. Berechnen Sie einen häufigkeitsoptimalen Suchbaum und geben Sie eine Reihenfolge von 20 Anfragen an, welche die Zugriffshäufigkeiten berücksichtigen, sodass im anfangs wie unten dargestellten Splay-Baum weniger Vergleiche anfallen als im optimalen Suchbaum.



Aufgabe 8.2 *Tiefensuche und Breitensuche.*

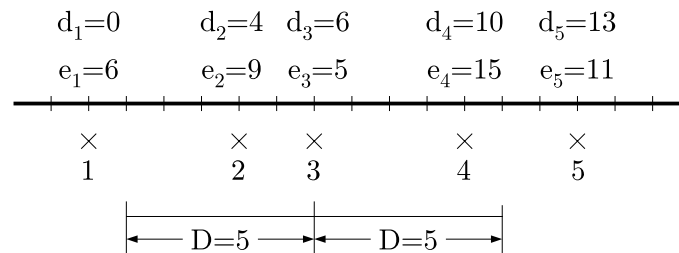
Betrachten Sie den folgenden Graphen $G = (V, E)$:



- Führen Sie für G eine Tiefensuche und eine Breitensuche aus, die jeweils im Knoten A startet. Hat ein Knoten mehr als einen möglichen Nachfolger, dann betrachten Sie diese in alphabetisch aufsteigender Reihenfolge.
- Welche Laufzeit haben Breiten- und Tiefensuche, wenn der Graph nicht in Adjazenzlisten-darstellung, sondern als Adjazenzmatrix gespeichert ist?

Aufgabe 8.3 *Dynamische Programmierung: Optimale Platzierung von Windrädern (Klausuraufgabe Jänner 2013).*

Entlang einer Strasse sollen Windräder zur Stromgewinnung aufgestellt werden. Aufgrund geographischer Gegebenheiten kommen n verschiedene Positionen in Betracht. Gesetze schreiben vor, dass zwischen zwei Windrädern ein Mindestabstand von D eingehalten werden muss. Die n möglichen Positionen sind als Koordinaten auf einer Linie gegeben, wobei die erste Position den Koordinaten-Wert 0 hat. Der Abstand der i -ten Position zur erstmöglichen Position beträgt d_i . Es gilt also $d_1 = 0$ und für alle $i \in \{1, \dots, n-1\}$ ist $d_i < d_{i+1}$. Wenn ein Windrad an der Position i aufgestellt wird, dann produziert es Energie $e_i > 0$. Die Aufgabe besteht nun darin, eine Positionierung von Windrädern mit maximaler gesamter Energieausbeute zu finden.



Beispiel: In der obigen Abbildung ist eine Situation für $n = 5$ mögliche Positionen dargestellt. Wird beispielsweise ein Windrad an der Position 3 aufgestellt, dann können an den Positionen 2 und 4 keine Windräder aufgestellt werden. Werden die Windräder an den Positionen 1, 3 und 5 aufgestellt, dann werden $6 + 5 + 11 = 22$ Einheiten Energie produziert. Dies ist aber nicht die optimale Lösung: Eine Installation auf den Positionen 2 und 4 erzeugt $9 + 15 = 24$ Einheiten Energie.

- (a) Zeigen Sie anhand eines (möglichst einfachen) Beispiels, dass die folgende Greedy-Strategie nicht notwendigerweise eine optimale Lösung liefert: *“Wähle so lange aus den möglichen Positionen diejenige mit maximaler Energieausbeute aus, bis keine weiteren Windräder mehr platziert werden können.”*
- (b) Beschreiben Sie einen möglichst effizienten Algorithmus der dynamischen Programmierung, der die maximal produzierbare Energie berechnet.
- (c) Geben Sie die Laufzeit Ihres Algorithmus an.
- (d) Beschreiben Sie im Detail, wie der Algorithmus aus (b) abgeändert werden muss, um zusätzlich eine Positionierung von Windrädern mit maximaler Energieausbeute zu berechnen.

Aufgabe 8.4 *Zyklen finden (Programmieraufgabe).*

In dieser Aufgabe wollen wir testen, ob ein ungerichteter Graph einen Zyklus enthält. Gegeben sei ein ungerichteter, zusammenhängender Graph G in Adjazenzlistendarstellung. Der Graph hat n Knoten, die mit den Zahlen $0, \dots, n-1$ bezeichnet werden. Ein Zyklus ist ein Weg, der an seinem Ausgangsknoten endet. Entwickeln Sie einen Algorithmus, der berechnet, ob G einen Zyklus enthält.

Eingabe Die erste Zeile der Eingabe enthält lediglich die Anzahl t der Testinstanzen. Danach folgen die Testinstanzen, wobei die erste Zeile einer Testinstanz die Anzahl n der Knoten des Graphs enthält und darauf n Zeilen mit den Adjazenzlisten der Knoten folgen. Dabei enthält die i -te Zeile der Adjazenzlisten eine Auflistung aller Knoten, die benachbart mit Knoten $i-1$ sind.

Ausgabe Für jede Testinstanz soll genau eine Zeile ausgegeben werden. Sie enthält den Buchstaben y , falls G einen Zyklus enthält, und andernfalls n .

Beispiel

Eingabe:

```
2
3
1
0 2
1
3
1 2
0 2
1 0
```

Ausgabe:

```
n
y
```

Hinweis Wir stellen eine Code-Vorlage zur Verfügung, in der das Einlesen der Eingabe bereits implementiert wurde. Der Graph ist in `ArrayList<Integer>[] graph` in Adjazenzlistendarstellung gespeichert, d.h. als Array (der Länge n) von ArrayLists, die Ganzzahlen speichern. Eine ArrayList speichert also die Adjazenzliste des entsprechenden Knoten.

Für diese Aufgabe gibt es lediglich ein Testset.

Abgabe: Am Mittwoch, den 27. April 2016 in Ihrer Übungsgruppe.