Bergische Universität GH Wuppertal
Fachbereich Mathematik
Prof. Dr. A. Frommer, Holger Arndt

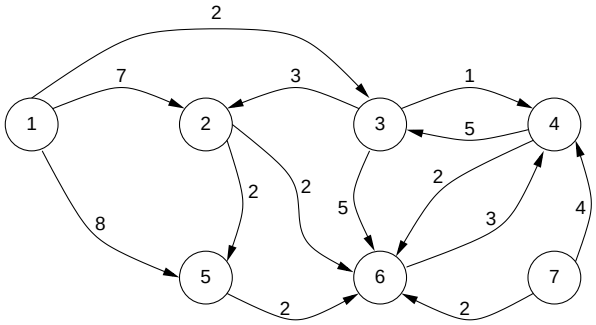
Informatik II: Algorithmen und Datenstrukturen

SS 2002

13. Übungsblatt

Aufgabe 38 (Algorithmus von Dijkstra, 4T)

Auf den unten skizzierten Graphen soll der Algorithmus von Dijkstra mit Startknoten 1 angewendet werden. Geben Sie dazu für jeden Schritt, in Form einer Tabelle, jeweils die bisher berechneten Entfernungen an. Markieren Sie außerdem in jedem Schritt die Entfernungen, welche bereits als minimal bekannt sind.

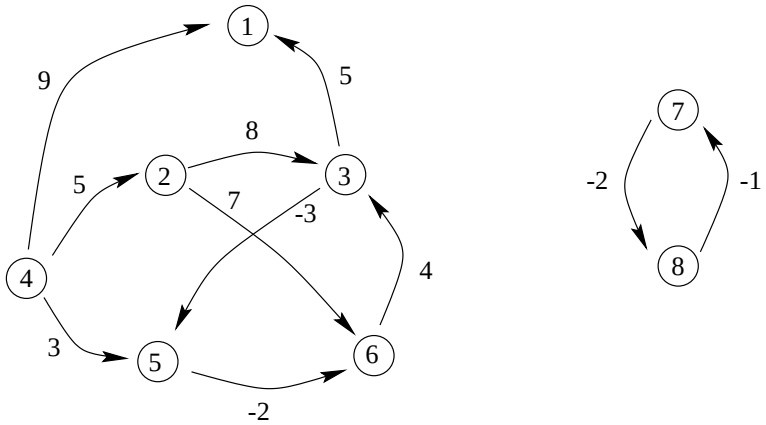


Aufgabe 39 (Algorithmus von Ford/Bellman, 4T+5P)

Sei $G = (V, E, g)$ ein Kanten-gewichteter (gerichteter oder ungerichteter) Graph mit $g : E \longrightarrow \mathbb{R}$ und $v \in V$.

Für $x \in V$ seien $\delta_v^g(x)$ und $L_v^g(x)$ wie in Abschnitt 4.3.2 der Vorlesung definiert.

a) Wenden Sie den Algorithmus von Ford/Bellman auf folgendes Beispiel an:



Wählen Sie dabei $v = 4$ und geben Sie nach jedem Durchlauf der i-Schleife die Δ-Werte aller Knoten und die Menge D_i an.

b) Implementieren Sie den Algorithmus von Ford und Bellman und erweitern Sie ihn so, dass bei Graphen ohne negativen Zyklus für jeden Knoten $x \in V$ außer dem Abstand $\delta_v^g(x)$ auch noch ein gewichtsminimaler Kantenzug von v nach x ausgegeben wird (sofern $\delta_v^g(x) \neq +\infty$). Die Gesamtzahl der Operationen soll dabei weiterhin $\mathcal{O}(|V| \cdot (|V| + |E|))$ sein.

Hinweis: Diese Komplexität kann nicht mit einer Darstellung des Graphen durch eine Adjazenzmatrix erreicht werden.
Für jeden Knoten x muss nicht der gesamte (bisher als gewichtsminimal betrachtete) Kantenzug von v nach x abgespeichert werden, sondern nur der Vorgänger von x in einem solchen Kantenzug.

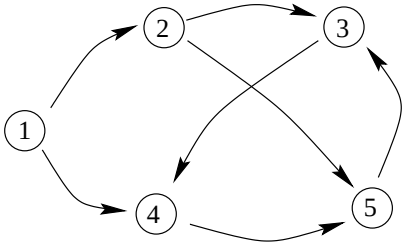
Ergänzen Sie die Funktion in dem auf der Webseite abgelegten Rahmenprogramm und testen Sie es an dem dort befindlichen Beispiel. Wählen Sie dabei als Startknoten $v = 0$.

Aufgabe 40 (Topologisches Sortieren, 4T)

Sei $G = (V, E)$ ein Graph mit Wurzel w. Nachfolgender Algorithmus ist eine Erweiterung des Depth-First-Durchlaufens aus der Vorlesung, bei der zusätzlich die Zeitpunkte $f(u)$ festgehalten werden, zu denen die Untersuchung eines Knotens u beendet wird. Falls beim Durchlaufen ein Zyklus gefunden wird, bricht der Algorithmus mit einer entsprechenden Meldung ab. Dabei seien anfänglich alle Knoten aus V mit *nicht-besucht* markiert und $t = 0$.

```
Algorithmus DepthFirstSuche(v)
markiere v als entdeckt
for alle von v ausgehenden Kanten (v,x) do
    if x noch als nicht-besucht markiert ist then
        DepthFirstSuche(x)
    else if x als entdeckt markiert ist then
        melde „Zyklus gefunden"; Ende
    end if
end for
markiere v als besucht
t = t + 1
f(v) = t
```

a) Auf welche Art bricht der Algorithmus für den folgenden Graphen ab, wenn Sie beim Knoten 1 starten?

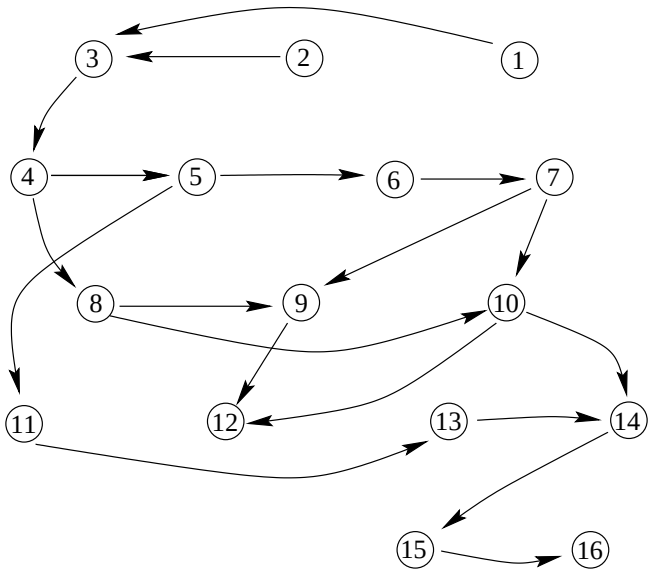


b) Wie muss man den Algorithmus abändern, damit er in einem zyklenfreien Graphen alle Knoten markiert?

c) Endet der Algorithmus aus b) ohne die Meldung „Zyklus gefunden", so erhält man eine topologische Sortierung durch Anordnen der Knoten nach absteigenden Werten von f. (Dies brauchen Sie nicht zu zeigen.)

Im folgenden „Bauherrn-Graphen“ existiert eine Kante vom Knoten v zum Knoten w genau dann, wenn die zu w gehörige Arbeit erst nach der von v durchgeführt werden kann. Sortieren Sie diesen Graphen mit dem Algorithmus aus b) topologisch.

1 Grundstück vorbereiten
2 Baumaterial anliefern
3 Baugrube ausheben
4 Fundamente gießen
5 Mauern
6 Dachstuhl bauen
7 Dach decken
8 Außeninstallationen
9 Außenputz
10 Fenster einsetzen
11 Decken einziehen
12 Garten anlegen
13 Inneninstallationen
14 Wände isolieren
15 Malerarbeiten
16 Umziehen



Abgabe: Mi., 17.07.2002, 14 Uhr