28.11.2008

Praktikum: Graphen und Netzwerke

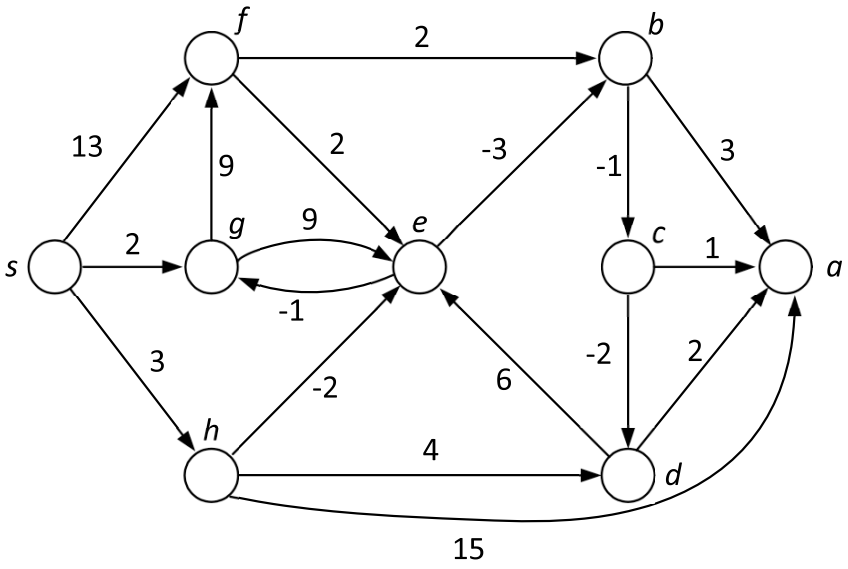
Aufgabe 1 → Experimentierumgebung für Graphen und ihre Visualisierung
Aufgabe 2 → Kapitel 5.1 – Bellman-Ford-Algorithmus
Aufgabe 3 → Kapitel 5.4 – Constraint-Graphen und Bellman-Ford-Algorithmus
Aufgabe 4 → Kapitel 7 – Flüsse in Netzwerken

Aufgabe 1 (Graphen)

Implementieren Sie einen ADT für gerichtete, gewichtete Graphen G = (V, E). Verwenden Sie für die Speicherung des Graphen eine Adjazenzmatrix, in welche die Knoten V und die gegebenenfalls gewichteten Kanten E von einem File eingelesen werden können.
Entwickeln Sie eine interaktive Testumgebung und eine geeignete GUI, in der Graphen visualisiert, in die verschiedene Algorithmen wie *sssP*, *Alg. von Dinitz* eingebunden und in der die Algorithmen ausgeführt werden können.

Aufgabe 2 (Bellman-Ford-Algorithmus)

Implementieren Sie den Bellman-Ford-Algorithmus und wenden Sie ihn auf den folgenden Graphen G und den Startknoten **s** an. Führen Sie den Algorithmus Schritt für Schritt so aus, dass die einzelnen Relaxationsschritte in der Visualisierung des Graphen beobachtet werden können. Markieren Sie die Kanten, die nach jedem der |V|- 1 Durchläufe den aktuellen Kürzeste-Wegebaum zeigen. Stellen Sie in den Knoten v die jeweils aktuellen d[v]-Werte dar. Wie immer, gute Doku anfertigen.



Aufgabe 3 (Constraint-Graphen)

Berechnen Sie mit Hilfe der Constraint-Graphen-Methode und des Bellman-Ford-Algorithmus eine zulässige Lösung für die folgenden beiden Systeme von Differenzbedingungen oder argumentieren Sie, dass keine zulässige Lösung existiert.

$x_1 - x_2 \leq 1$
$x_1 - x_4 \leq -4$
$x_2 - x_3 \leq 2$
$x_2 - x_5 \leq 7$
$x_2 - x_6 \leq 5$
$x_3 - x_6 \leq 10$
$x_4 - x_2 \leq 2$
$x_5 - x_1 \leq -1$
$x_5 - x_4 \leq 3$
$x_6 - x_3 \leq -8$

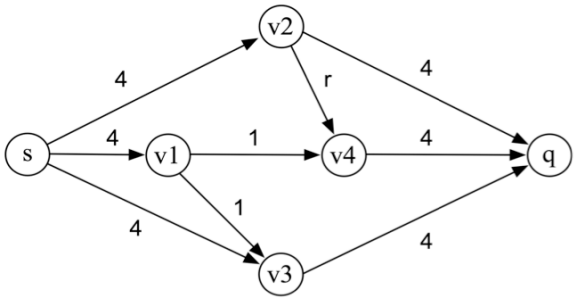
$x_1 - x_2 \leq 1$
$x_1 - x_5 \leq 5$
$x_2 - x_4 \leq -6$
$x_3 - x_2 \leq 1$
$x_4 - x_1 \leq 3$
$x_4 - x_3 \leq 5$
$x_4 - x_5 \leq 10$
$x_5 - x_3 \leq -4$
$x_5 - x_4 \leq -8$

Aufgabe 4 (Netzwerkflüsse)

Zum Berechnen von Flüssen in Netzwerken mit n Knoten und m Kanten erweitert der Algorithmus von Dinitz den Netzwerkfluss entlang gleichlanger kürzester Wege **gleichzeitig**. Dazu wird ein so genannter Niveaugraph verwendet. Darunter versteht man Folgendes:

- *niveau(v)* ist Länge des kürzesten Weges von s nach v.
- Niveaugraph, abgeleitet aus dem Restgraph R mit Kanten (v, w), wobei *niveau(w)* = *niveau(v)* + 1. Der Niveaugraph kann in O(n + m) Schritten durch Breitensuche berechnet werden.
- Blockierender Fluss: Jeder Pfad von s nach q enthält gesättigte Kante (f(e) = cap(e)).

a) Untersuchen Sie folgendes Beispiel für die mögliche Nichtkonvergenz des F&F-Algorithmus für irraationale Zahlen.



$$r = \frac{\sqrt{5}-1}{2} < 1$$

Wenn für r der goldene Schnitt, also eine irrationale Zahl, gewählt wird, dann ist es möglich, dass der **Ford & Fulkerson**-Algorithmus nicht gegen ein Optimum konvergiert. Betrachte dazu eine Folge zunehmender Wege, welche Restkapazitäten der Kanten (v2, v4), (v1, v4) und (v1, v3) benutzen, um die Berechnung der Folge $\{f_n\}$ zu simulieren, die durch die Differenzgleichung

$$f_{n+2} = f_n - f_{n+1}, \quad f_0 = 1, \; f_1 = r$$

definiert ist. Beobachte, dass $f_n = r^n$ für alle n, wenn r als goldener Schnitt gewählt wird. Bei jeder Iteration wird genau eine der erwähnten Kanten die Restkapazität 0 erhalten, während die anderen f_n und f_{n+1} erhalten. Implementieren Sie den F&F-Algorithmus und wenden Sie ihn auf das Beispiel an. Versuchen Sie die o.g. zunehmenden Wege zu „erzwingen“, um das Verhalten des Algorithmus, d.h. das Nicht-Konvergieren gegen ein Optimum step-by-step beobachten zu können.

b) Wenden Sie den Algorithmus von Dinitz auf folgendes Flussnetzwerk an, um den maximalen Fluss zu berechnen. Stellen Sie die einzelnen Schritte mit der in Aufgabe 1 entwickelten GUI dar.