Institut für Informatik
der Universität München
Prof. M. Hofmann
Dipl. Inf. Andreas Abel
Dipl. Math. Stefan Schimanski

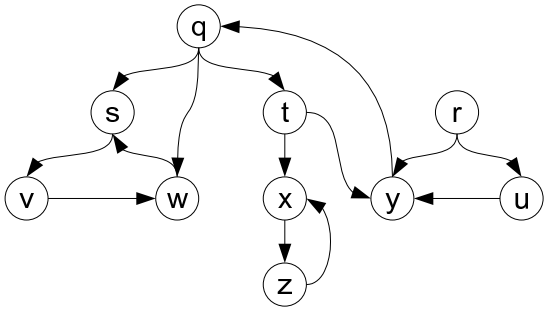
SS 2006

26. Juni 2006

Übungen zur Vorlesung
Effiziente Algorithmen

Blatt 9

Aufgabe H-28:



a) Geben Sie die Adjazenzmatrix und Adjazenzlisten an, wobei die Knoten alphabetisch angeordnet sind.

b) Zeigen Sie den Ablauf der Tiefensuche bei alphabetisch sortierten Knoten. Geben Sie für jeden Knoten die *discovery time* und *finishing time* an, und klassifizieren Sie die Kanten des Graphen.

c) Entfernen Sie aus dem Graphen die Kanten (y, q), (z, x) und (w, s). Geben Sie eine topologische Sortierung des entstehenden Graphen an unter Verwendung der *finishing time*.

d) Entfernen Sie aus dem Graphen im Gegensatz zu c) nur die Kanten (y, q) und (z, x). Zeigen Sie, dass es keine topologische Sortierung geben kann.

(8 Punkte)

Aufgabe H-29: Tiefensuche kann verwendet werden, um einen ungerichteten Graphen in seine Zusammenhangskomponenten zu zerlegen: Modifizieren Sie den Algorithmus DFS so, dass er für jeden Knoten v eine Zahl $cc[v]$ zwischen 1 und k, wobei k die Zahl der Zusammenhangskomponenten des Graphen ist,

mit ausgibt, derart dass $cc[u] = cc[v]$ genau dann gilt, wenn u und v in der selben Komponente liegen. (4 Punkte)

Aufgabe H-30:

a) Lösen sie das Problem aus der vorigen Aufgabe unter Verwendung der Datenstruktur UNION-FIND, d.h. zeigen Sie, wie für einen ungerichteten Graphen $G = (V, E)$ ein effizienter Algorithmus $areConnected(u, v)$ gewonnen werden kann, der für $u, v \in V$ entscheidet, ob es in G einen Pfad zwischen u und v gibt. Wie komplex ist $areConnected(u, v)$ und wie komplex ist der Aufbau der Datenstruktur zuvor?

b) Vergleichen Sie die Komplexität mit der der Lösung, die Tiefensuche verwendet. Welchen Vorteil hat der Algorithmus aus a) mit UNION-FIND?

(4 Punkte)

Aufgabe H-31: Eine andere Möglichkeit, einen gerichteten azyklischen Graphen topologisch zu sortieren, ist die folgende: man sucht einen Knoten vom Eingangsgrad 0, gibt diesen aus, und entfernt ihn und alle von ihm ausgehenden Kanten, und wiederholt dies bis der Graph leer wird. Der Graph sei als Liste von Knoten $v_1, ..., v_n$ und Liste von Kanten $e_1, ..., e_m$, $e_i = (v_{i_1}, v_{i_2})$ gegeben.

Geben Sie einen Algorithmus an, der diese Idee umsetzt und in Zeit $O(|V|+|E|)$ läuft. (4 Punkte)

Abgabe bis Montag, 3. Juli, 14.00 Uhr in einer der Vorlesungen oder Übungen oder im dafür vorgesehenen Briefkasten in der Oettingen- oder Theresienstraße. Oder zu Beginn der Montags-Übung (14.15 Uhr).