Technische Universität Braunschweig

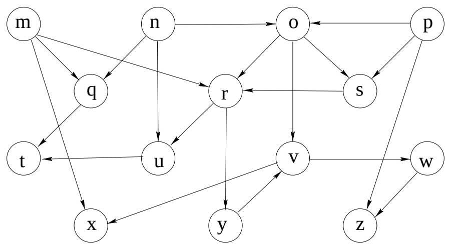
Institut für
Betriebssysteme und Rechnerverbund

Prof. Dr. H. Langendörfer · Prof. Dr. M. Zitterbart

Übungen zu Algorithmen und Datenstrukturen II

Blatt 8

20. Aufgabe: Topologische Sortierung



Für die topologische Sortierung eines gerichteten Graphen kann die Tiefensuche eingesetzt werden:

for $(v \in V)$ farbe$[v]$ = weiss;
for $(v \in V)$
  if (farbe$[v]$ == weiss)
    Tiefensuche (v);

Die Reihenfolge des Schwarz-werdens stellt eine topologische Sortierung dar.

(a) Zeigen Sie die Ordnung der Kanten, die entsteht, wenn man die topologische Sortierung des obigen Graphen mit Hilfe der Tiefensuche berechnet. Gehen Sie dabei davon aus, daß die Nachbarknoten in alphabetischer Reihenfolge gespeichert sind.

(b) Es existieren viele verschiedene Kantenordnungen eines gerichteten Graphen G, die topologische Sortierungen sind. Zeigen Sie, daß nicht alle möglichen topologischen Sortierungen eines Graphen mit Hilfe der Tiefensuche ermittelt werden können, unabhängig in welcher Reihenfolge die Knoten gespeichert sind.

21. Aufgabe: Transitive Hülle

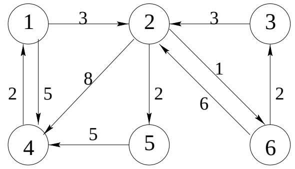
Der in der Vorlesung vorgestellte Algorithmus von Warshal zur Berechnung der transitiven Hülle eines Graphen, kann durch geringfügige Modifaktion auf gewichtete Graphen erweitert werden. Die Kanten sind hierbei bewertet, wobei ein Kantenwert die 'Länge' der Kante repräsentieren soll. Der Warshal-Algorithmus kann dadurch die kürzesten Wege (und deren Längen) zwischen allen Knotenpaaren (i, j) berechnen. Es ist hierzu die Definition der Variablen $m^k_{i,j}$ zu ändern. Es seien $m^k_{i,j}$ jetzt die kürzesten Weglängen von i nach j, wobei als mögliche 'Zwischenknoten' nur die Knoten $\{1, \ldots, k\}$ zugelassen sind. Es gilt die rekursive Beziehung:

$$m^k_{i,j} = min(m^{k-1}_{i,j}, m^{k-1}_{i,k} + m^{k-1}_{k,j})$$

Daraus ergibt sich der folgende Algorithmus ($E[i,j] = \infty$ wenn zwischen i und j keine Kante besteht und $E[i,i] = 0$):

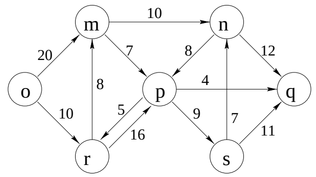
```
int  E[][] = new int [n+1][n+1];      // bewertete Adjazenzmatrix n*n
// Graph in E eintragen

for ( int k = 1; k <= n ; k++ )
  for ( int i = 1; i <= n ; i++ )
    for ( int j = 1; j <= n ; j++ )
      if ( E[i][k] + E[k][j] < E[i][j] )
        E[i][j] = E[i][k] + E[k][j];
```



Geben Sie die Matrizen für den obigen Graphen im Verlauf des Algorithmus an. Machen Sie hierzu eine "Momentaufnahme" vor jedem Erhöhen des k-Wertes.

22. Aufgabe: Flüsse in Netzwerken



Finden Sie in dem gegebenen Netzwerk den maximalen Fluß zwischen Knoten o und Knoten q. Verwenden Sie hierzu den Algorithmus von Ford-Fulkerson. Ermitteln Sie hierzu Erweiterungspfade mit einem zulässigen Fluß und bilden Sie die Restnetzwerke. Geben Sie die Erweiterungspfade und die entstehenden Restnetzwerke an. Was ist der maximale Fluß zwischen Quelle o und Senke s ?