Technische Universität München
Institut für Informatik
Prof. Dr. Ernst W. Mayr

SS 1998
Aufgabenblatt 4
8. Juni 1998

Praktikum Algorithmen-Entwurf

(Abgabetermin: Montag, den 15. Juni 1998, $24.^{00}$Uhr)

Kürzeste und längste Pfade

Aufgabe 1 Algorithmus von Dijkstra

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit positiven Kantengewichten, bei dem jeder Knoten von jedem anderen aus erreichbar ist. Implementieren und animieren Sie den Algorithmus von Dijkstra so, daß die kürzesten Pfade von einem vom Benutzer gewählten Startknoten zu allen anderen Knoten berechnet und dargestellt werden. Bei jedem Knoten soll die Länge des berechneten Pfades zu diesem Knoten als Knotenlabel angezeigt werden.

Während der Algorithmus abläuft, sollen zu jeder Zeit die schon erledigten Knoten und die in der Priority-Queue befindlichen Knoten farblich gekennzeichnet sein. Ebenso sollen die bereits gefundenen kürzesten Pfade gut erkennbar sein. Für jeden Knoten in der Priority-Queue soll zusätzlich diejenige Kante markiert werden, über die er momentan am günstigsten von einem schon erledigten Knoten aus erreicht werden kann.

Aufgabe 2 Azyklische Graphen

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit Kantengewichten, der keine Kreise und genau einen Knoten v ohne eingehende Kanten enthält. Von v aus seien alle anderen Knoten erreichbar. Entwerfen, implementieren und animieren Sie einen Algorithmus, der *in linearer Zeit*, also in Zeit $O(|V| + |E|)$ statt $O(|V| \log |V| + |E|)$ wie beim Algorithmus von Dijkstra, die kürzesten Pfade von v zu allen anderen Knoten berechnet.

Erweitern Sie das Programm so, daß der Benutzer wählen kann, ob die kürzesten oder die längsten Pfade von v zu allen anderen Knoten berechnet werden sollen. In beiden Fällen sollen die Pfade in linearer Zeit berechnet werden.

Hinweise

Sie können den Algorithmus aus Aufgabe 1 mit den Graphen `pos1.gw` bis `pos4.gw` und den Algorithmus aus Aufgabe 2 mit den Graphen `wdag1.gw` bis `wdag4.gw` testen. All diese Graphen stehen wie üblich sowohl im WWW als auch im `graphen`-Unterverzeichnis der LEDA-Verzeichnisse in der HP-Halle bzw. auf den hpmayr-Rechnern zur Verfügung. Die Gewichte der Kanten sind wieder als Strings im User-Label gespeichert und sollten in ein `edge_array<double>` eingelesen werden.

Beachten Sie, daß die bearbeiteten Graphen dieses Mal gerichtet sind. Ihre Programme sollten daher ein `gw.set_directed(true)` enthalten und *kein* `g.make_undirected()`.