Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr
Matthias Baumgart

Wintersemester 2008/09
Aufgabenblatt 1
20. Oktober 2008

Praktikum Algorithmen-Entwurf

Letzter Abgabetermin: Montag, den 27.10.2008, 14^{00} Uhr

Aufgabe 1 (LEDA (First Steps))

Machen Sie sich mit der Benutzung der LEDA-Bibliothek bei der Erstellung von C++-Programmen vertraut. Legen Sie ein Verzeichnis an, in dem Sie im Verlauf des Semesters Ihre Praktikumsaufgaben bearbeiten und Ihre Lösungen erstellen werden, und kopieren Sie das Makefile und die Quellen dfs.C und control.h von der Praktikumswebpage. Das Programm dfs.C sollte sich nun mittels `make dfs` übersetzen lassen.

Aufgabe 2 (Breitensuche)

Implementieren und animieren Sie Breitensuche in zusammenhängenden, ungerichteten Graphen unter Verwendung von LEDA. Verwenden Sie zur Darstellung des Graphen auf dem Bildschirm die Klasse `GraphWin`. Benutzen Sie das Programm dfs.C als Vorlage. Der Benutzer soll per Mausklick einen Startknoten s auswählen können, und Ihr Programm soll dann mittels einer Breitensuche den Graphen durchlaufen und dabei für alle anderen Knoten v die Länge eines kürzesten Pfades (bzgl. Anzahl der Kanten) von s zu v berechnen.

Es soll am Bildschirm gut mitverfolgt werden können, in welcher Reihenfolge das Programm die Knoten besucht, welche Knoten bereits besucht wurden, und welche Knoten gegenwärtig in der Queue gespeichert sind. Verwenden Sie zu diesem Zweck die Möglichkeiten, die `GraphWin` bzgl. der Darstellung von Knoten und Kanten anbietet.

Die Abstände der Knoten vom Startknoten s sollen als User-Labels auf dem Bildschirm dargestellt werden. Die Kanten, die im Breitensuch-Baum enthalten sind, sollen ebenfalls hervorgehoben werden.

Aufgabe 3 (Topologisches Sortieren)

Implementieren und animieren Sie einen Algorithmus, der einen beliebigen gerichteten Graphen $G = (V, E)$ als Eingabe bekommt und der versucht, jedem Knoten $v \in V$ eine eindeutige Nummer $f(v) \in \{1, 2, \ldots, |V|\}$ zuzuordnen, so dass für jede Kante $(u, v) \in E$ gilt: $f(u) < f(v)$. (Das heißt, der Anfangsknoten jeder Kante muss eine kleinere Nummer haben als der Endknoten der Kante.)

Eine solche Nummerierung existiert übrigens genau dann, wenn der Graph keinen Zyklus enthält. Falls der Eingabegraph doch einen Zyklus enthält, soll Ihr Programm dies erkennen und einen Zyklus am Bildschirm farbig markieren. Bemühen Sie sich um eine möglichst effiziente Implementierung!

Hinweise

Die Abgabe der Lösungen muss jeweils bis Montag 14^{00} Uhr per E-Mail an die Adresse `algoprak@in.tum.de` erfolgen. Jede Zweiergruppe soll nur eine Lösung abzugeben. Um den Einstieg zu erleichtern, wird jedoch empfohlen, dass jeder Student eine eigene Lösung zur Aufgabe 2 anfertigt und dann aus diesen beiden Lösungen eine ausgewählt und abgeschickt wird.

Es sind nur die C++-Sourcen der erstellten Programme abzugeben, d.h. in diesem Fall Source-Code zu Aufgabe 2 und und 3.
Aktuelle Informationen, Testdaten für Ihre Programme und PostScript-Dateien mit den Praktikumsunterlagen finden Sie im WWW unter folgender URL:

`http://wwwmayr.in.tum.de/lehre/2008WS/algoprak/data`

Anforderungen an abgegebene Programme

Die abgegebenen Lösungen werden nach den folgenden Kriterien bewertet:

- Korrektheit der berechneten Ergebnisse (bei den Beispielgraphen und beliebigen weiteren Eingaben)
- Effizienz der Implementierung (Vermeidung ineffizienter Konstrukte, so dass bei Entfernung der Animationsroutinen die ggf. geforderte Worst-Case-Laufzeit eingehalten wird)
- Qualität der Animation (die Arbeitsweise des Algorithmus soll anschaulich visualisiert werden)
- Lesbarkeit des Quelltextes (ausreichend Kommentare, die das Verständnis erleichtern)

Außerdem legen wir großen Wert darauf, dass die Lösungen selbst erarbeitet und nicht Programme anderer Gruppen als "Vorlage" verwendet werden. Falls Probleme bei der Implementierung einer Lösung auftreten, sollte keinesfalls auf die evtl. schon fertigen Lösungen einer anderen Gruppe zurückgegriffen werden. (Diskussionen über Implementierungsmöglichkeiten sind natürlich erlaubt und erwünscht, die direkte Weitergabe von Source-Codes ist jedoch nicht gestattet.) Stattdessen können Verständnis- oder Implementierungsprobleme mit dem jeweiligen Betreuer in der Sprechstunde besprochen werden.