

Institut für Wirtschaftstheorie
und Operations Research
Prof. Dr. Klaus Neumann
Cord-Ulrich Fündeling
Collegium am Schloß, Bau IV, Zi. 102.2
Tel.: (+49) 721 608 3809
e-mail: fuendeling@wior.uni-karlsruhe.de

Karlsruhe, den 1. April 2003

Vorlesung “Optimierung auf Graphen und Netzwerken II”
im Sommersemester 2003

Rechnerübungsblatt 3

Programmierhinweis zu den Aufgaben 4 bis 6:

Zur Bearbeitung der Aufgaben 4 bis 6 benötigen Sie die Datenstruktur *ForwardStar* zur Speicherung von Digraphen (vgl. Rechnerübungsblatt 1 (WS 2002/2003), Aufgabe 1 bzw. Neumann/Morlock, S. 194f) sowie eine entsprechende Einleseroutine für Dateien im *ForwardStar*-Format (vgl. Rechnerübungsblatt 2 (WS 2002/2003), Aufgabe 3).

Versehen Sie den Programmcode zu jeder der drei Aufgaben mit einer eigenen `main`-Methode, so dass jeder Algorithmus separat aufgerufen und getestet werden kann. Die Übergabe der zu bearbeitenden Problem Instanz soll jeweils als Kommandozeilenparameter erfolgen. Beispielsweise soll also die Kommandozeile `java TripleTest.dat` zum Aufruf des Tripelalgorithmus mit der in `Test.dat` enthaltenen *ForwardStar*-Instanz führen.

Alle Programmausgaben sollen auf dem Bildschirm erfolgen.

Aufgabe 4 (1 Punkt): *Tripelalgorithmus*

Implementieren Sie den Tripelalgorithmus zur Bestimmung kürzester Wege zwischen allen Knoten (vgl. Neumann/Morlock, S. 219ff.). Der Algorithmus soll für einen übergebenen Digraphen \vec{G} (in Form der Datenstruktur *ForwardStar*) die Matrix \mathcal{D} der kürzesten Distanzen sowie die Matrix \mathcal{P} der Vorgänger auf den kürzesten Wegen bestimmen und ausgeben. Falls in \vec{G} Zyklen negativer Länge existieren, soll der Algorithmus abbrechen und die Zeile `Zyklus negativer Laenge entdeckt` sowie einer dieser Zyklen ausgegeben werden.

Aufgabe 5 (3 Punkte): *Sukzessive Einbeziehung von Knoten*

Implementieren Sie das Verfahren Sukzessive Einbeziehung von Knoten in der Variante „Weiteste Knoten zuerst“ (vgl. Neumann/Morlock S. 445ff). Der Algorithmus soll das

symmetrische HRP für einen übergebenen Digraphen \vec{G} (in Form der Datenstruktur *ForwardStar*) lösen. Sie können davon ausgehen, dass \vec{G} immer einem symmetrischen Digraphen entspricht. Die Vollständigkeit von \vec{G} sowie die Gültigkeit der Dreiecksungleichung kann hingegen nicht vorausgesetzt werden. Zur Bestimmung der Kantenbewertungen des \vec{G} entsprechenden vollständigen Graphen \vec{G} können Sie den Tripelalgorithmus aus Aufgabe 4 verwenden.

Implementieren Sie eine oder mehrere Datenstrukturen zur Speicherung von Rundreisen und Hamilton'schen Kreisen. Die Datenstruktur(en) sollte(n) es erlauben, Knoten sukzessive zu einem bestehenden (Teil-)Kreis bzw. zu einer bestehenden (Teil-)Rundreise hinzuzufügen.

Bestimmen Sie für jeden möglichen Startknoten $i \in V$ jeweils eine Rundreise und speichern Sie die beste dabei gefundene Lösung.

Als Ausgabe des Programms wird die gefundene Rundreise in \vec{G} , der entsprechende Hamilton'sche Kreis im vollständigen Graphen \vec{G} sowie die Länge der Rundreise erwartet.

Aufgabe 6 (2 Punkte): *2-opt-Verfahren*

Implementieren Sie das 2-opt-Verfahren zur Verbesserung einer gegebenen Anfangslösung für das symmetrische HRP (vgl. Neumann/Morlock S. 451 ff.).

Das Verfahren soll allgemein mit einer *ForwardStar*-Instanz sowie einer beliebigen Anfangslösung aufrufbar sein (verwenden Sie dabei die in Aufgabe 5 entwickelte Datenstruktur zur Speicherung Hamilton'scher Kreise). Beim Aufruf der `main`-Methode von Aufgabe 6 soll zunächst eine solche Anfangslösung mit Hilfe des Verfahrens Sukzessive Einbeziehung von Knoten aus Aufgabe 5 bestimmt und anschließend mit dem 2-opt-Verfahren verbessert werden.

Organisatorische Hinweise

Die Übungsaufgaben müssen in der Programmiersprache Java (J2SE 1.4.0) implementiert werden. Für die Bearbeitung der Aufgaben sind im CIP-Pool II der Fakultät für Wirtschaftswissenschaften (Geb. 11.40) am Mittwoch von 9.45 Uhr bis 11.15 Uhr 16 PCs reserviert.

Falls Sie an einer Benotung Ihrer Rechnerübung interessiert sind, so bilden Sie bitte Gruppen zu zwei oder drei Personen. Die Abgabe der Programme muss voraussichtlich bis spätestens Freitag, 11. Juli 2003, um 12:00 Uhr erfolgen, die Abnahme der Rechnerübung findet jeweils gruppenweise wahrscheinlich in der darauf folgenden Woche statt. Abzugeben sind sowohl alle Quellcode- (`*.java`) als auch die kompilierten Binärdateien (`*.class`). Die genauen Termine und Modalitäten werden rechtzeitig im Internet bzw. auf den (Tafel-)Übungsblättern bekanntgegeben. Eine gruppenübergreifende Zusammenarbeit bei der Lösung der Aufgaben ist nicht gestattet und führt zum Ausschluß aus dem Bewertungsverfahren. Insgesamt sind in beiden Semestern 12 Punkte erreichbar, von denen 6 auf das Winter- und 6 auf das Sommersemester entfallen. Die Punkte können in die **bestandene** Klausur zu den Vorlesungen „Optimierung auf Graphen und Netzwerken I + II“ am 31. Juli 2003 bzw. die **bestandene** Nachklausur Ende des Wintersemesters 2003/04 eingerechnet werden (in den Klausuren sind ca. 120 Punkte erreichbar).