Wintersemester 2017/18 Forschungsinstitut für Diskrete Mathematik
Prof. Dr. J. Vygen Universität Bonn
Dr. U. Brenner

Einführung in die Diskrete Mathematik
9. Übung

1. Sei (G, u, b, c) eine Instanz des MINIMUM-COST-FLOW-PROBLEMS. Man nennt eine Funktion $\pi : V(G) \to \mathbb{R}$ ein *optimales Potential*, falls es einen b-Fluss f in (G, u) mit minimalen Kosten gibt, so dass π ein zulässiges Potential bezüglich (G_f, c) ist.

   (a) Man beweise, dass eine Funktion $\pi : V(G) \to \mathbb{R}$ genau dann ein optimales Potential ist, wenn für jedes $X \subseteq V(G)$ die folgende Ungleichung gilt:

   $$b(X) + \sum_{e \in \delta^-(X): c_\pi(e) < 0} u(e) \leq \sum_{e \in \delta^+(X): c_\pi(e) \leq 0} u(e). \qquad (*)$$

   (b) Man zeige, wie man in polynomieller Zeit für eine gegebene Funktion $\pi : V(G) \to \mathbb{R}$ entweder eine die Bedingung $(*)$ verletzende Menge X findet oder entscheidet, dass es keine solche gibt.

   (c) Zeigen Sie, wie man für ein gegebenes optimales Potential einen b-Fluss mit minimalen Kosten in $O(m + n^3)$ Zeit findet. (3+2+2 Punkte)

2. Wir betrachten ein Verfahren, das aus dem SUKZESSIVE-KÜRZESTE-WEGE-ALGORITHMUS entsteht, indem man stets um $\gamma' := \min\left\{ \min_{e \in E(P)} u_f(e), \quad \max\{b'(s), -b'(t)\}\right\}$ augmentiert. Zeigen Sie, dass dieser Algorithmus bei ganzzahligen b-Werten und Kapazitäten ebenfalls nach höchstens $\frac{1}{2} \sum_{v \in V(G)} |b(v)|$ Augmentierungen terminiert. (3 Punkte)

3. Das gebrochene b-Matching-Problem wird folgendermaßen definiert: Gegeben seien ein ungerichteter Graph G, Kapazitäten $u : E(G) \to \mathbb{R}_+$, Zahlen $b : V(G) \to \mathbb{R}_+$ und Gewichte $c : E(G) \to \mathbb{R}$. Man finde eine Abbildung $f : E(G) \to \mathbb{R}_+$ mit $f(e) \leq u(e)$ für alle $e \in E(G)$ und $\sum_{e \in \delta(v)} f(e) \leq b(v)$ für alle $v \in V(G)$, die $\sum_{e \in E(G)} c(e) f(e)$ maximiert.

   (a) Man zeige, wie man dieses Problem durch Zurückführung auf ein MINIMUM-COST-FLOW-PROBLEM lösen kann.

   (b) Man zeige, dass, wenn b und u ganzzahlig sind, stets eine halb-ganzzahlige Lösung f existiert (d.h. $2f(e)$ muss für alle $e \in E(G)$ ganzzahlig sein). (2+2 Punkte)

4. Implementieren Sie den SUKZESSIVE-KÜRZESTE-WEGE-ALGORITHMUS, um das ZUORDNUNGSPROBLEM zu lösen. Das Programm soll Laufzeit $O(n^3)$ haben, wobei $n = |V(G)|$ sei. Ein genauere Beschreibung der Aufgabe finden Sie umseitig. (12 Punkte)

Abgabe der Theorieaufgaben: Dienstag, den 19.12.2017, vor der Vorlesung.

Zur Programmieraufgabe:
Das Programm muss in C oder C++ geschrieben sein. Es wird empfohlen, C++ zu verwenden. In diesem Fall kann man zum Einlesen und Speichern der Graphen die Klasse `Graph` aus der Vorlesung "Algorithmische Mathematik I" aus dem Wintersemester 2016/2017 verwenden. Alle Datenstrukturen, die in dieser Vorlesung vorgestellt wurden, finden Sie hier:

`http://www.or.uni-bonn.de/~hougardy/alma/alma.html`

Sie können alle diese Programme und Datenstrukturen verwenden. Außerdem dürfen Sie die STL verwenden, aber keine weiteren externen Bibliotheken.

Eingabeformat:
Die Einträge der Datei sind ausschließlich ganze Zahlen. Sie können voraussetzen, dass die Summe der Absolutbeträge aller Zahlen in der Eingabe kleiner als 2^{31} ist. In der ersten Zeile steht eine einzelne positive gerade Zahl n, welche die Anzahl der Knoten angibt. Die Knoten werden von 0 bis $n-1$ durchnumeriert. Jede folgende Zeile kodiert genau eine Kante. Die ersten beiden Einträge einer Zeile sind die Nummern der Endknoten der Kante (wobei die Kante vom jeweils ersten angegebenen Knoten zum zweiten gerichtet sei). Die erste Nummer liegt dabei in $\{0, \dots, \frac{n}{2}-1\}$ und die zweite in $\{\frac{n}{2}, \dots, n-1\}$. Der dritte Eintrag in der Zeile gibt die Kosten der Kante an.
Parallele Kanten kommen in den Instanzen nicht vor.

Ausgabeformat: Das Programm muss in der ersten Zeile der Ausgabe die Kosten der berechneten Lösung ausgeben. Danach folgen $\frac{n}{2}$ weitere Zeilen, die jeweils genau zwei Zahlen enthalten und eine Kante der berechneten Lösung kodieren. Die Nummern in der jeweiligen Zeile geben dabei die Nummern der Endknoten an (wobei die kleinere Nummer zuerst stehen soll).

Beispiel: Eine Eingabedatei für einen Graphen mit sechs Knoten und sechs Kanten kann so aussehen:

```
6
2 5 37
0 3 12
0 4 -14
1 4 8
2 3 4
1 5 2
```

Die Ausgabe der Programms kann dann so aussehen (die letzten drei Zeilen können anders sortiert sein):

```
-8
0 4
2 3
1 5
```

Abgabe: Der Quelltext des Programms muss bis **9. Januar, 16:15 Uhr** per E-Mail beim jeweiligen Tutor eingegangen sein. Außerdem ist bis zu diesem Zeitpunkt ein Ausdruck des Quelltextes zusammen mit den Theorieaufgaben abzugeben.

Testinstanzen befinden sich auf der Seite

`http://www.or.uni-bonn.de/lectures/ws17/edm_uebung_ws17.html`

Sollten weitere Hinweise zu der Programmierübung notwendig sein, werden diese ebenfalls auf dieser Homepage bekanntgegeben.