Wintersemester 2015/16
Prof. Dr. S. Hougardy
Dr. U. Brenner

Forschungsinstitut für Diskrete Mathematik
Universität Bonn

Einführung in die Diskrete Mathematik
2. Programmierübung

Implementieren Sie den Goldberg-Tarjan-Algorithmus zur Berechnung eines maximalen s-t-Flusses. Ihr Programm sollte höchstens $O(n^2m)$ Laufzeit haben, idealerweise nur $O(n^2\sqrt{m})$.

Einlesen der Daten: Dem Programm muss beim Aufruf der Name einer Datei übergeben werden. Ein Aufruf hat also die Form

```
<programmname> <dateiname>
```

Eine gültige Datei, die einen Graphen beschreibt, hat das folgende Format:

```
Knotenanzahl
Knoten0a Knoten0b Kapazitaet0
Knoten1a Knoten1b Kapazitaet1
...
```

Die Einträge der Datei sind ausschließlich ganze Zahlen. Sie können voraussetzen, dass die Summe aller Zahlen in der Eingabe kleiner als 2^{31} ist. In der ersten Zeilen steht eine einzelne natürliche Zahl, welche die Anzahl der Knoten angibt. Jede weitere Zeile spezifiziert genau eine Kante. Die ersten beiden Einträge einer Zeile sind zwei verschiedene nichtnegative ganze Zahlen, welche die Nummern der Endknoten der Kante sind. Dabei nehmen wir an, dass, wenn wir n Knoten haben, die Knoten von 0 bis $n-1$ durchnumeriert sind. Der dritte Eintrag in der Zeile ist eine positive ganze Zahl, die die Kapazität der Kante bezeichnet. Der Index einer jeden Kante ist durch ihre Zeilennummer in der Eingabedatei gegeben: Zeile i kodiert die Kante mit Index $i-2$ (für $i = 2, \ldots, m+1$, wobei m die Zahl der Kanten sei). Die Kanten sind dadurch auch von 0 bis $m-1$ durchnumeriert. Parallele Kanten können vorkommen.
Sie können voraussetzen, dass es mindestens 2 Knoten gibt. Der Knoten s hat dabei Nummer 0 und der Knoten t Nummer 1.

Ausgabeformat: Das Programm muss in der ersten Zeile der Ausgabe den Wert eines maximalen s-t-Flusses ausgeben. Die weiteren Zeilen müssen jeweils genau einen Index einer Kante und den zugehörigen Flusswert enthalten. Es werden dabei nur die Kanten mit positivem Fluss ausgegeben, und die Zeilen sollen nach dem Kantenindex aufsteigend sortiert sein.

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

```
5
0 1 2
2 1 2
2 3 2
3 1 1
0 2 3
3 4 2
```

Die Ausgabe des Programms muss dann so aussehen:

```
5
0 2
1 2
2 1
3 1
4 3
```

Auf der Homepage der Übung:

`http://www.or.uni-bonn.de/lectures/ws15/edm_uebung_ws15.html`

finden Sie die Dateien `graph.cpp` und `graph.h`, die eine Datenstruktur enthalten, mit der ein Graph gespeichert werden kann. Auf der Seite steht auch ein kleines Testprogramm `testgraph_flow.cpp`, das einen gerichteten Graphen einliest. Die Einleseroutine ist mit dem oben beschriebenen Format kompatibel. Sie können diese Dateien (die auch in der Vorlesung "Algorithmische Mathematik I" im Wintersemester 2014/2015 Verwendung fanden) benutzen und geeignet abändern.

Die Testprogramme basieren auf C++-11, d.h. man muss beim Kompilieren die Option "-std=c++11" verwendet werden, also z.B.:

```
g++ -std=c++11 -Wall *.cpp
```

Das Programm muss in C oder C++ geschrieben sein. Es muss korrekt arbeiten und ohne Fehlermeldung kompiliert werden können. Der Code muss auf einem gängigen Linuxsystem funktionieren. Für die Sortierung dürfen Sie `qsort` oder `std::sort` verwenden. Andere Algorithmen aus externen Bibliotheken dürfen nicht verwendet werden. Achten Sie auch darauf, dass Sie Ihr Programm ausreichend mit Kommentaren versehen.
Testinstanzen befinden sich auf der Seite

`http://www.or.uni-bonn.de/lectures/ws15/edm_uebung_ws15.html`

Abgabe: Der Quelltext des Programms muss bis Donnerstag, 21. Januar 2016, 16:15 Uhr per E-Mail beim jeweiligen Tutor eingegangen sein.