

Aufgabenblatt 7

letzte Aktualisierung: 13. June, 12:16 Uhr

Ausgabe: 12.6.2016

Abgabe: 22.6.2016 23:59

Thema: Flüsse, Ford-Fulkerson

Abgabe

Die folgenden Datei muss für eine erfolgreiche Abgabe im svn Ordner
 Tutorien/txx/Studierende/deinname@TU-BERLIN.DE/Abgaben/
 eingecheckt sein:

Geforderte Datei:

Blatt07/src/Network.java

Wichtige Ankündigungen

- Unit tests dürfen geteilt werden. Lösungen dürfen auf keinen Fall geteilt werden. Wir testen auf Plagiate.

1. Aufgabe: Flussgraphen - Einführung

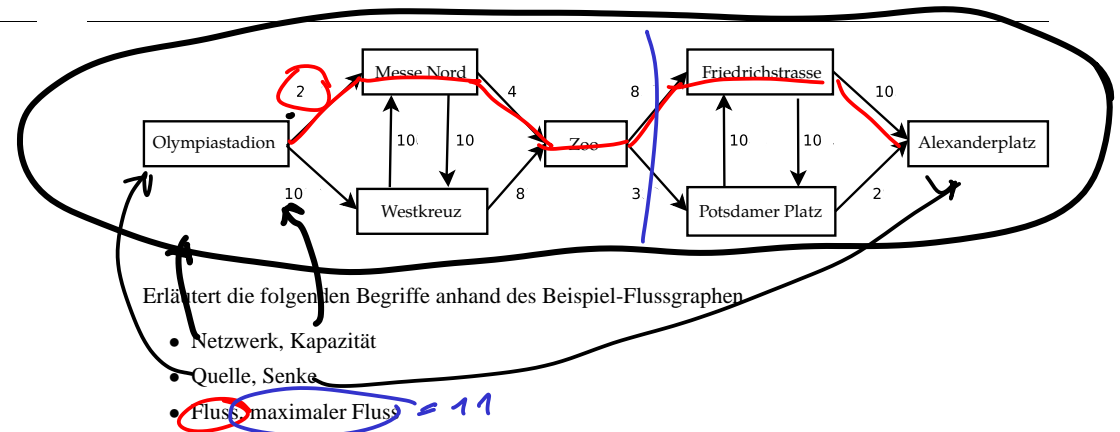
Flussgraphen sind gerichtete Graphen mit Kantengewichten, wobei das Gewicht einer Kante die Kapazität des Flusses entlang der Kante angibt.

Flussgraphen modellieren verschiedene Transportprobleme wie z.B. Verkehr in einem Strassennetz, Produktionswege, Routing in einem Netzwerk oder die verfügbare Leistung im elektrischen Energieversorgungsnetz.

Auf diesem Zettel werdet ihr den Ford-Fulkerson Algorithmus benutzen, um maximale Flüsse zu berechnen.

1.1. (Tut) Begriffe - Grundlagen

Nach dem Pokalfinale fahren 75.000 Menschen mit U- und S-Bahn vom Olympiastadion Richtung Alexanderplatz zur Feier. Der Graph unten zeigt einen Ausschnitt des Verkehrsnetzes und die verschiedenen möglichen Wege, modelliert als gerichteter Graph. Die Kantengewichte geben an, wie viele Fahrgäste pro Stunde über jede Teilstrecke transportiert werden können.



1.2. (Tut) Restflussgraph

Was ist der Restflussgraph? Wie sieht der Restflussgraph für die beiden Beispielgraphen in Bildern 3 und 4 aus?



1.3. (Tut) Ford-Fulkerson Algorithmus

Erläutert die beiden Schritte des Ford-Fulkerson Algorithmus am Beispiel-Flussgraphen im Bild 4

- Finde einen augmentierenden Pfad im Restflussgraph
- Update den Restflussgraph mit dem augmentierenden Pfad.

Wendet den Ford-Fulkerson Algorithmus auf die beiden Graphen an.

2. Aufgabe: Edmonds-Karp

2.1. (Tut) Edmonds-Karp-Algorithmus: Theorie Wie verbessert der Algorithmus von Edmonds und Karp die Laufzeit vom Ford-Fulkerson-Algorithmus beim Maximalen-Fluss-Problem?

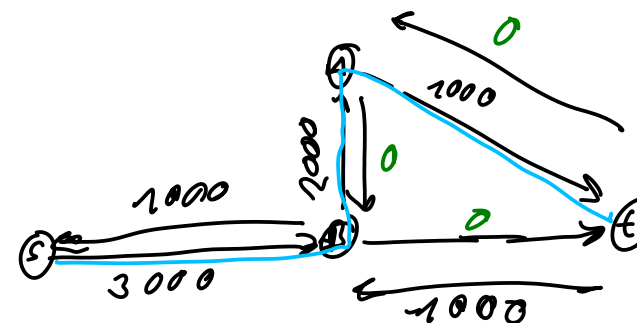
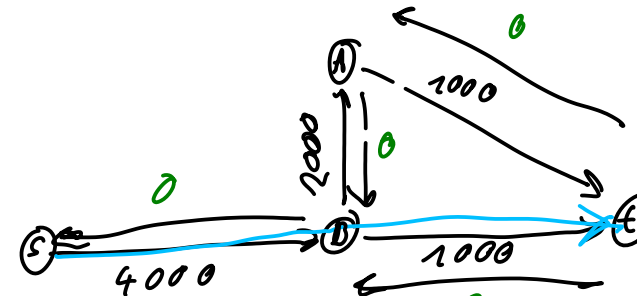
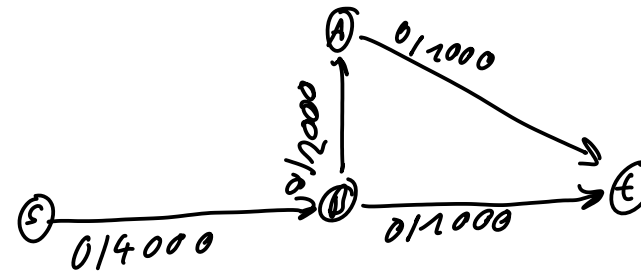
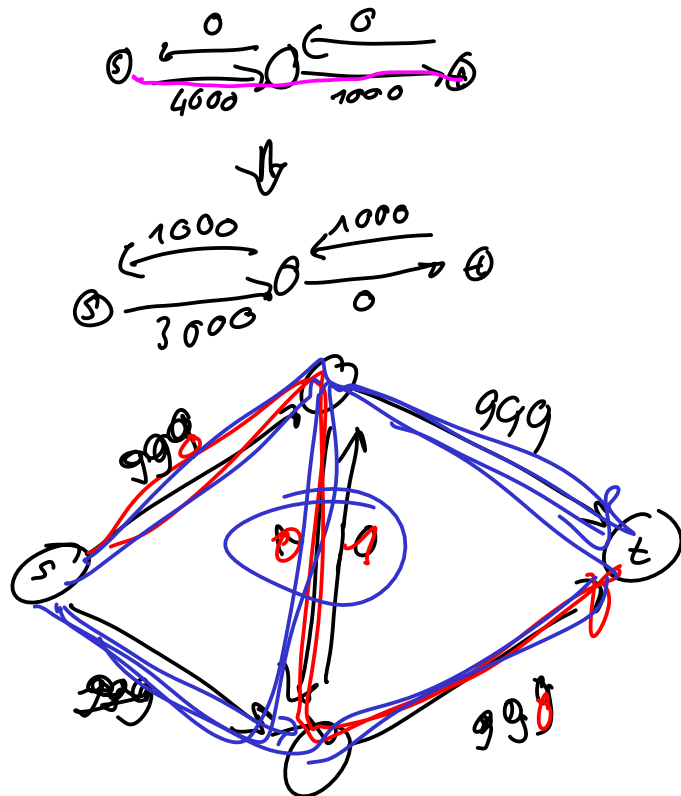
3. Aufgabe: Flussgraphen - Ford-Fulkerson

3.1. Ford-Fulkerson (100 Punkte) Implementiert die vorgestellte Variante des Ford-Fulkerson Algorithmus in der Methode `int fordFulkerson()` der Klasse `Network.java`. Nutzt dabei die vorgegebenen Methodenköpfe:

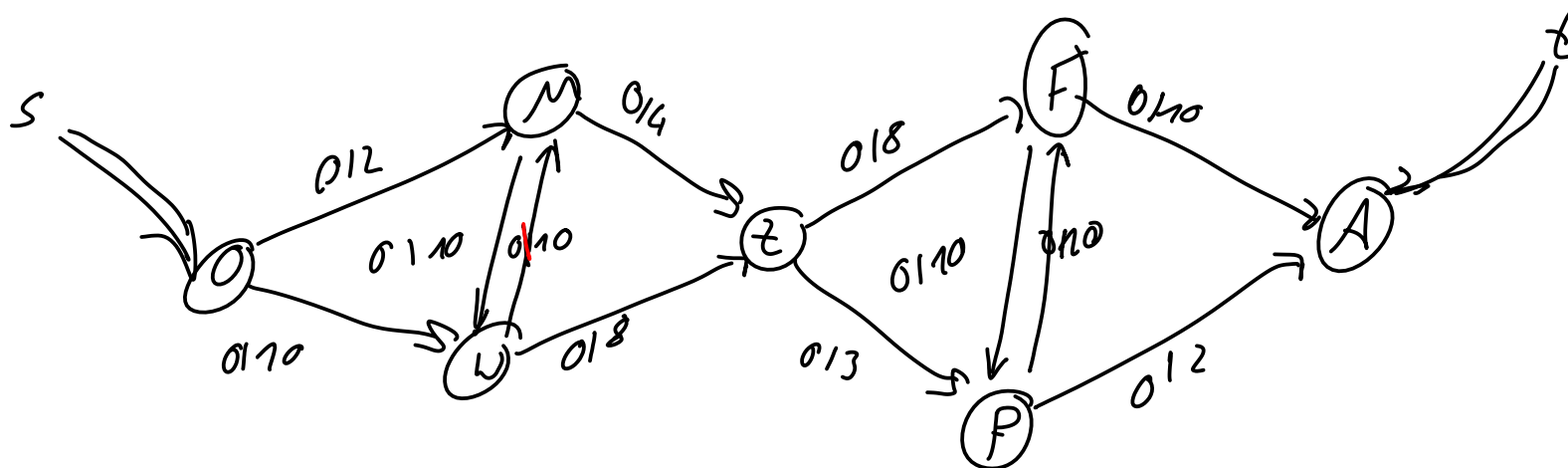
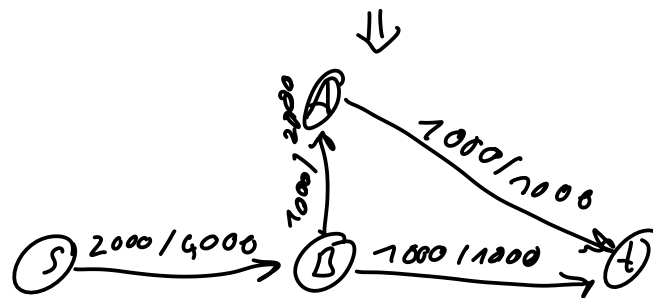
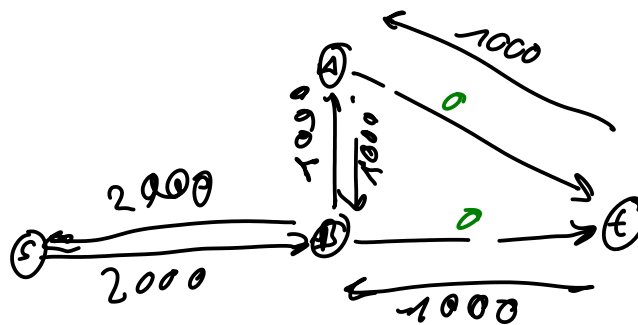
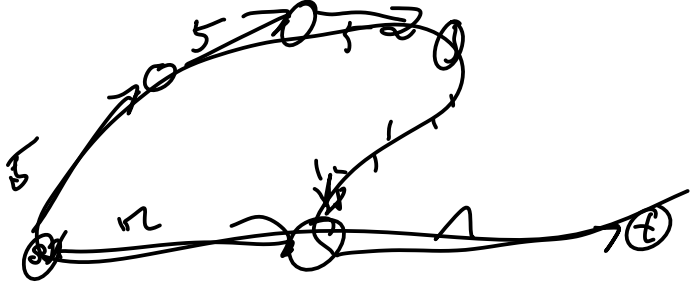
- `LinkedList<Node> findAugmentingPath(int start, int end)` findet einen flussverstärkenden (augmentierenden) Pfad im Graph. Falls keiner existiert, gibt die Methode eine leere Liste zurück. Bei nicht-existierenden Start- oder Endknoten wirft die Methode eine Exception.

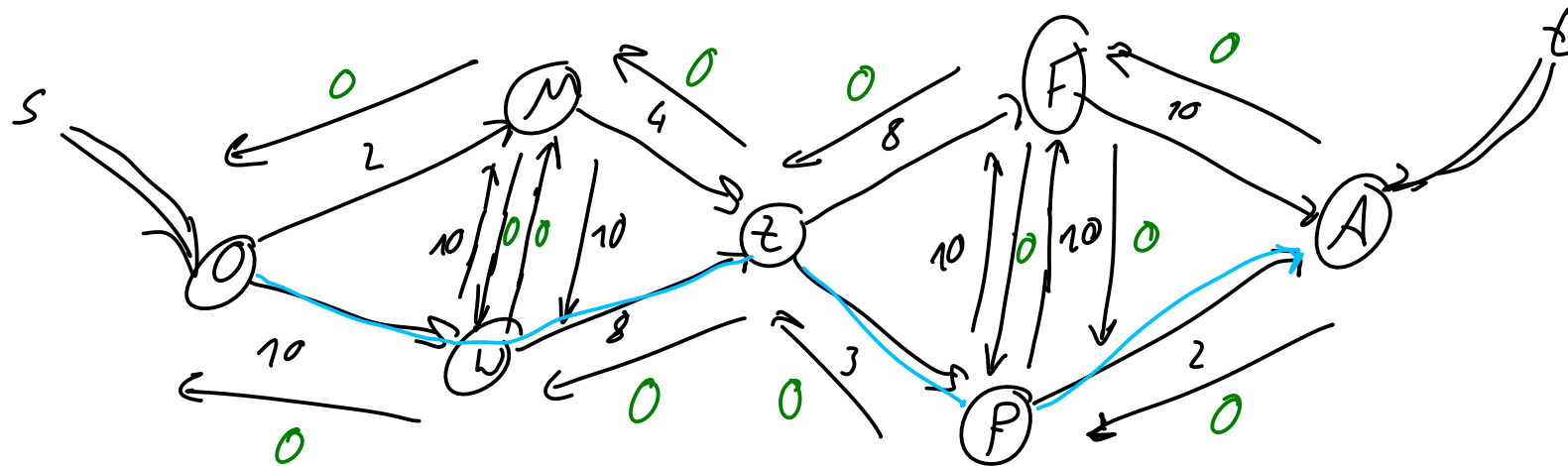
- `int findMinCapacity(LinkedList<Node> path)`
findet die minimale Restkapazität auf dem Pfad und gibt diese zurück. Ein nicht existenter Pfad (leere Knotenliste) hat die Restkapazität 0.
- `void updateResidualCapacity(int minCapacity, LinkedList<Node> p)`
updates die Kapazität des Restflussgraphens mit dem gegebenen Fluss über den gegebenen (augmentierenden) Pfad.
- `int fordFulkerson()`
Wendet den Ford-Fulkerson Algorithmus an, um den maximalen Fluss zu berechnen. Nach Ausführung des Algorithmus sollten die `currentFlow` Variablen jeder Kante einen gültigen Fluss tragen, der maximal ist. Die Methode gibt ausserdem den Wert des Flusses zurück.

1.2



$$1000 + 1000 = 2000$$





$$2+1+4+3+1=11$$

