Dokumentation

Inhalt

[Einleitung 3](#_Toc54260153)

[Vorstellung des Problems oder der Problemstellung 4](#_Toc54260154)

[Beschreibung der gewählten Lösung. 5](#_Toc54260155)

[Ungünstige Pfadwahl 5](#_Toc54260156)

[Reflektion 6](#_Toc54260157)

# Einleitung

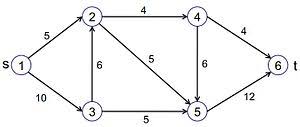
* Das Problem in einen grösseren Kontext einordnen
* Motivation für das Problem (Wofür ist die Lösung gut? Was wird damit gelöst?)
* Grobe Beschreibung des Problems

# Vorstellung des Problems oder der Problemstellung

Das Problem, mit dem wir uns beschäftigen, ist der Ford Fulkerson Algorithmus, auch der maximale Fluss genannt. Dieser Algorithmus regelt den Transport von Ressourcen von der Quelle zum Ziel. Auf dem Weg zum Ziel gibt es dann Einschränkungen für die Anzahl der transportierenden Ressourcen. Die Aufgabe ist es herauszufinden wie viele Ressourcen maximal von der Quelle zum Ziel kommen können, unter Einhaltung der Einschränkungen zu transportieren. Anders gesagt, man muss den maximalen Fluss finden.

Als klassisches Beispiel wird oft eine Verkehrssituation verwendet. Autos wollen von A nach B fahren, jedoch hat es auf allen Strassen (Kanten) eine bestimmte Anzahl Spuren (Kapazität). Kreuzungen werden Knoten genannt und der Start- und Endpunkt (A und B) Quelle und Ziel. Andere Beispiele wären zum Beispiel die Verteilung elektrischer Energie auf einem Stromnetzwerk, oder das Senden von Daten.[[1]](#footnote-1)

https://www-m9.ma.tum.de/graph-algorithms/flow-ford-fulkerson/index\_de.html

Das ist ein typisches Beispiel einer Problemstellung. S steht für Source (Quelle) und t für Target (Ziel). Die umkreisten Zahlen (1 bis 6) sind die verschiedenen Knoten. Die Pfeile dazwischen sind die Kanten und geben die Richtung des Flusses an. Jede Kante ist mit einer Zahl beschriftet, welche für die Kapazität dessen steht.

1 Darstellung einer Problemstellung

Bild:

https://de.wikiversity.org/wiki/Kurs:Algorithmen\_und\_Datenstrukturen/Vorlesung/Flussproblem\_Ford-Fulkerson

**//Erklärung der Kanten und Kapazitäten (siehe** [**ungünstige Pfadwahl**](#_Ungünstige_Pfadwahl)**)**

**Mögliche Lösungen**

* Welche Lösungsansätze (Algorithmen) gibt es?
* Wenn es verschiedene Lösungsansätze gibt, dann ihre Vor- und Nachteile erläutern.

# Beschreibung der gewählten Lösung.

* Erläuterung des verwendeten Algorithmus' (Pseudocode, etc.)
* UML Klassendiagramm/ MVC
* Welche Datenstrukturen wurden benutzt?
* Besonderheiten der implementierten Lösung
* Wichtige Codeteile (Umsetzung des Algorithmus, GUI, Schnittstellen) werden vorgestellt.

**//Erklärung der Lösung**

## Ungünstige Pfadwahl

Nach einiger Zeit ist uns die ungünstige Pfadwahl entgegnet. Das bedeutet, dass beim Wählen eines Weges ein anderer Weg blockiert wird, durch die vollständige Ausnützung einer Kante. Dadurch kann es passieren, dass man nicht den maximalen Fluss findet. Wie in diesem Beispielbild dargestellt:

Der erste gewählte Weg ist 1-3-2-4, wodurch 5 Ressourcen ins Ziel gelangen. Der maximale Fluss beträgt jedoch mehr. Das liegt daran, dass die, jetzt ausgelastete, Kante 3 die Wege 1-3-4 und 1-2-4 blockiert. Würde man die anderen zwei Wege wählen enthält man einen maximalen Fluss von 10.

Beispiel Bild von der Präsentation

-> Ungünstige Pfadwahl

Dieses Problem haben wir wie folgt gelöst.

# Reflektion

* Was erfüllt das Programm
* Was sind die Grenzen des Programms (was kann es nicht?)
* Wie wurde das Programm getestet (Was ist Input, was Output)
* Was ist noch nicht gelöst?
* Wie könnte das Programm noch erweitert werden?
* Wo gab es Schwierigkeiten in der Umsetzung?

1. www-m9.ma.tum.de (2020) [↑](#footnote-ref-1)