Fachhochschule Wedel

Seminararbeit

Thema:

Flussprobleme in Graphen und ihre Anwendung auf 0/1-Netzwerken

Eingereicht von:	Claudia Padberg (wi4409) An der Windmühle 4 22880 Wedel Tel. (04103) 9881967 E-Mail: c.padberg@gmx.de
Erarbeitet im:	10. Semester
Abgegeben am:	11. Januar 2005
Referent:	Prof. Dr. Sebastian Iwanowski Fachhochschule Wedel Feldstrasse 143 22880 Wedel

Inhaltsverzeichnis

1. Einleitung

1.1 Problemstellung

Eine Vielzahl von praktischen Anwendungen können als Netzwerke dargestellt werden, in welchen die Problembehandlung, welche stets um Optimierung bemüht ist, stattfindet.

Gerade aus wirtschaftlichen Gründen, ist es von besonderem Interesse, die zur Verfügung stehenden Kapazitäten optimal auszunutzen.

Da die Netzwerke meist eine enorme Größe und Komplexität aufweisen, ist die Lösungsfindung nicht mehr ohne die Anwendung von geeigneten Verfahren möglich.

Als Beispiele für betrachtete Netzwerke können der Fluss von Öl durch Pipelines, der Transport von Gütern auf bestimmten Handelswegen oder auch der Fluss von Fahrzeugen auf einem Verkehrsnetz gesehen werden.

In dieser Arbeit geht es also um den Transport von bestimmten Datenmengen durch ein Netzwerk, welcher später als so genannter „Fluss“ bezeichnet wird.

Die Fragen, die hierbei beantwortet werden sollen, lauteten:

- Wie viel Durchsatz einer Datenmenge kann maximal erzielt werden?
- Wie viele verschiedene Wege gibt es, welche sich zu keiner Zeit Kapazitäten teilen müssen?

Zur Beantwortung dieser Fragen muss beachtet werden, dass die Netzwerke auf ihren Wegen unterschiedliche, obere Kapazitätsbeschränkungen besitzen, welche nicht überschritten werden dürfen.

Diese Arbeit beschäftigt sich im Wesentlichen mit der Erläuterung zweier Verfahren zur Bestimmung von maximalen Flüssen in Netzwerken.

1.2 Konzeption

Zunächst wird im zweiten Kapitel ein Überblick über die Grundlagen der Graphentheorie gegeben, welcher sich von der einfachen Darstellung und Definition eines Netzwerkes bis hin zur Erläuterung von Erweiterungswegen, die man für die späteren Verfahren zur maximalen Flusssuche benötigt, erstreckt.

Als weitere Grundlage wird der Satz von Ford und Fulkerson angeführt, wobei in dieser Arbeit auf die Beweisführung dessen verzichtet wird.

Im dritten Kapitel wird gezeigt, wie man in einem Netzwerk die bereits erwähnten Erweiterungswege mit Hilfe einer Breitensuche bestimmen kann.

Die Kapitel 4 und 5 bilden den Hauptteil der Arbeit. Sie beschreiben die Algorithmen von Edmonds und Karp sowie von Dinic. Das Beispiel, welches zur Veranschaulichung der einzelnen Schritte der Algorithmen benutzt wird ist aus der Literatur von Volker Turau, Algorithmische Graphentheorie, entnommen und weitergeführt worden.

In Kapitel 6 wird schließlich noch auf die Anwendung dieser Algorithmen auf 0/1-Netzwerken eingegangen.

2. Theoretische Grundlagen

2.1 Ein q-s-Netzwerk

Ein *q*-*s*-Netzwerk ist ein kantenbewerteter, gerichteter Graph G mit zwei ausgezeichneten Ecken.
Dieser Graph G setzt sich aus einer Eckenmenge E und einer Kantenmenge K zusammen. Die beiden besonderen Ecken in diesem Graphen sind die Startecke q (Quelle) und die Endecke s (Senke). Alle übrigen Ecken sind Zwischenstationen auf dem Weg von q nach s.
Alle Kanten, die eine Verbindung von zwei Ecken im Netzwerk darstellen besitzen eine Kapazität c. Sie gibt an, wie viel maximal durch jede Kante fließen kann.

Zusammengefasst gilt für ein *q*-*s*-Netzwerk folgende Form:

- $N = (G, q, s, c)$, mit
    - $G = (E, K)$
    - $E = \{e_1, e_2, \ldots, e_n\}$
    - $K = \{k_1, k_2, \ldots, k_m\}$
    - $c(e)$ = Kapazität c der Kante e
    - q = Quelle
    - s = Senke

Für ein *q*-*s*-Netzwerk gelten weiterhin folgende Bedingungen:

1. Die Kapazitäten der Kanten sind nichtnegative reelle Zahlen.
2. Es gilt $g^-(q) = g^+(s) = 0$, d.h. der Innengrad von q sowie der Aussengrad von s ist gleich 0.

Da die im späteren Verlauf vorgestellten Algorithmen auch ohne die zweite Bedingung korrekt arbeiten, wird auf diese zunächst verzichtet.

Folgend wird ein Netzwerk graphisch dargestellt. Die Quelle und Senke sind festgelegt und die Ecken der Zwischenstationen mit den Ziffern 1 bis 6 bezeichnet. Die Zahlen auf den Kanten geben ihre jeweiligen Kapazitäten an.

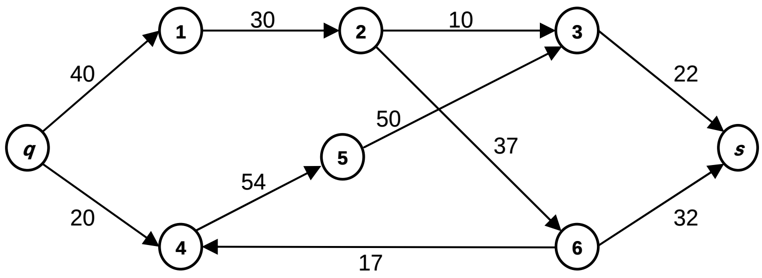


Abbildung 1 Ein *q*-*s*-Netzwerk

2.2 Ein q-s-Fluss

Um den Datentransport durch das Netzwerk darzustellen, wird zusätzlich der Begriff des *q*-*s*-Flusses eingeführt.
Durch eine Funktion *f* wird jeder Kante des Netzwerkes eine nichtnegative reelle Zahl zugeordnet.
Es entsteht ein Fluss durch das Netzwerk, welcher in *q* startet, über die Kanten und Zwischenstationen fließt und schließlich in *s* endet.

An einen *q*-*s*-Fluss werden die folgenden beiden Bedingungen gestellt:

1. $0 \leq f(k) \leq c(k)$, d.h. der Fluss durch eine Kante entspricht maximal der Kapazität dieser Kante.

2. $\sum_{k=(j,e)\in K} f(k) = \sum_{k=(e,j)\in K} f(k)$, d.h. die Datenmenge, die pro Zeiteinheit in eine Ecke hinein fließt muss auch in der gleichen Zeiteinheit wieder hinaus fließen. (Flusserhaltung)

Die folgende Darstellung zeigt wiederum das Netzwerk, welches um einen gültigen Fluss erweitert wird. Der aktuelle Fluss ist jeweils auf den Kanten in rot beschrieben. Die Zahlen dahinter zeigen weiterhin die maximalen Kapazitäten.

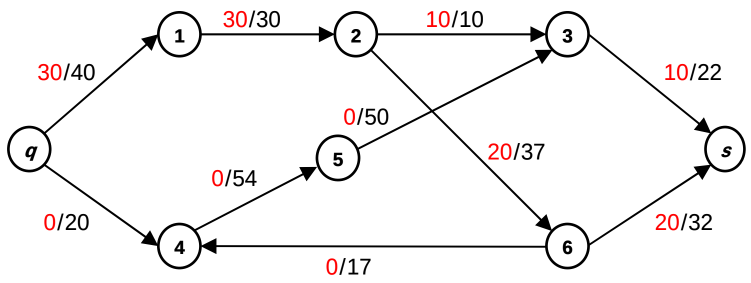


Abbildung 2 Ein *q*-*s*-Fluss

2.3 Der Wert eines Flusses

Der Wert eines Flusses $|f|$ gibt an, wie viel Durchsatz einer Datenmenge auf dem aktuellen Fluss durch das Netzwerk pro Zeiteinheit erzielt wird.

$|f|$ ergibt sich aus der Summe aller Flüsse aus der Quelle sowie aus der Summe aller Flüsse in die Senke. Daraus ergibt sich, dass alles, was aus der Quelle heraus fließt auch in die Senke hinein fließt.

Da auf die zweite Bedingung eines *q*-*s*- Netzwerkes verzichtet wurde, müssen eventuelle Rückflüsse in die Quelle bzw. aus der Senke subtrahiert werden.

Damit wird der Wert eines Flusses wie folgt definiert:

$$|f| = \sum_{k=(q,i)\in E} f(k) - \sum_{k=(i,q)\in E} f(k)$$

Für den Fluss aus Abbildung 2 ergibt sich der Wert des Flusses aus $f(q,1) = 30$ und $f(q,4) = 0$ zu $|f| = 30$.

2.4 Ein q-s-Schnitt

Bei einen *q*-*s*-Schnitt in einem Netzwerk wird die Eckenmenge *E* in zwei disjunkte Teilmengen *X* und $\overline{X}$ zerlegt, so dass $q \in X$ und $s \in \overline{X}$.

In Abbildung 3 wird im Netzwerk ein Schnitt gemacht, so dass sich die Teilmengen $X = \{q, 1, 2, 4\}$ und $\overline{X} = \{3, 5, 6, s\}$ ergeben.

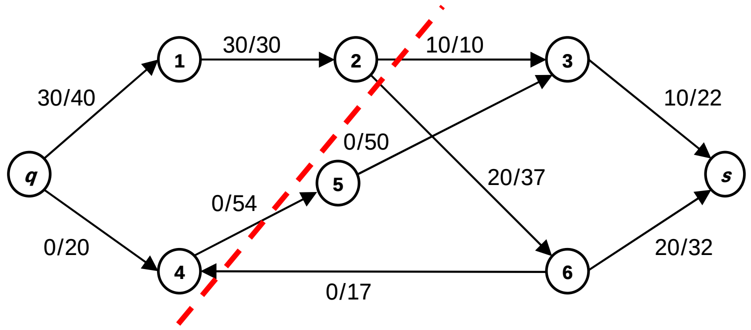


Abbildung 3 Ein *q*-*s*-Schnitt

Für einen *q*-*s*-Schnitt kann nun der Fluss sowie die Kapazität des Schnittes berechnet werden. Dafür werden alle Kanten betrachtet, die aus den Mengen von *X* nach $\overline{X}$ führen. In der Abbildung sind dies die Kanten (*2,3*), (*2,6*) und (*4,5*).

Die Kapazität des Schnittes ergibt sich aus der Summe der Kapazitäten der beschriebenen Kanten mit:

$$C(X, \overline{X}) = \sum_{\substack{k=(i,j)\in E \\ i\in X, j\in \overline{X}}} c(k)$$

Der Fluss des Schnittes ergibt sich entsprechend aus der Summe der Flüsse der betrachteten Kanten mit:

$$f(X, \overline{X}) = \sum_{\substack{k=(i,j)\in E \\ i\in X, j\in \overline{X}}} f(k)$$

Im Beispiel aus Abbildung 3 ist die Kapazität des Schnittes $C(X, \overline{X}) = 101$ und der Fluss des Schnittes $f(X, \overline{X}) = 30$.

2.5 Erweiterungswege

Von besonderer Bedeutung für die späteren Verfahren sind Erweiterungswege in einem Netzwerk. Dies sind Wege, auf denen der Fluss so verändert wird, dass sich der Gesamtfluss auf dem Netzwerk erhöht.

Zum Graphen G wird zunächst der ungerichtete Graph G^u gebildet.

Auf G^u können nun Wege erzeugt werden, welche die Kanten des Graphen auch in umgekehrter Richtung durchlaufen können.

Bei Kanten, die in G^u in der gleichen Richtung wie in G durchlaufen werden, handelt es sich um Vorwärtskanten. Entsprechend werden die Kanten, die in entgegen gesetzter Richtung zu G durchlaufen werden, als Rückwärtskanten bezeichnet.

Die Bedingungen für einen Erweiterungsweg *EW* sind wie folgt:

- Auf Vorwärtskanten eines *EW* ist $f(k) < c(k)$, d.h. es muss eine Restkapazität vorhanden sein.
- Auf Rückwärtskanten eines *EW* ist $f(k) > 0$, d.h. der aktuelle Fluss muss > 0 sein.

Die Abbildung 4 zeigt das Netzwerk, auf dem ein möglicher Erweiterungsweg hervorgehoben wird. Der Erweiterungsweg verläuft über *q, 4, 5, 3, 2, s*.

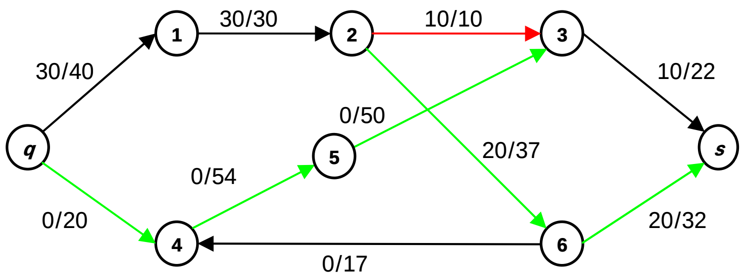


Abbildung 4 Ein Erweiterungsweg

Die Bedingungen für die Vorwärtskanten (grün) und die Rückwärtskante (rot) werden in diesem Beispiel erfüllt.

In diesem Fall wird ersichtlich, dass die zweite Bedingung für einen *q*-*s*-Fluss, die Erhaltungsbedingung, dadurch berücksichtigt wird, dass auf den Rückwärtskanten des Erweiterungsweges der Fluss wieder reduziert bzw. umgeleitet wird, um so die Kapazitäten der Folgekanten nicht zu überschreiten.

Zur Erhöhung des Flusses auf einem *EW* muss zunächst dessen „engste Stelle" $f\Delta$ gefunden werden. Hierzu werden zwei Größen eingeführt:

- $f_v = min\ \{\ c(k) - f\ (k)\ |\ k\ ist\ Vorwärtskante\ des\ EW\}$
- $f_r = min\ \{f\ (k)\ |\ k\ ist\ Rückwärtskante\ des\ EW\}$

f_v gibt die minimale Restkapazität der Vorwärtskanten an, f_r gibt den minimalen Fluss auf den Rückwärtskanten an. Wird im *EW* keine Rückwärtskante benutzt, so setzt man $f_r = \infty$.
$f\Delta = \min\ \{ f_v , f_r \}$ und gibt damit schließlich den Wert an, um den der Fluss entlang des *EW* auf den Vorwärtskanten erhöht und auf den Rückwärtskanten reduziert wird. Für den *EW* aus Abbildung 4 ergibt sich aufgrund der Restkapazität der Vorwärtskante (*6,s*) der Wert für $f_v = 12$ und für die Rückwärtskante (*3,2*) der Wert $f_r = 10$. Somit wird der Fluss um $f\Delta = 10$ erhöht.

Folgend ist die Veränderung des Flusses dargestellt:

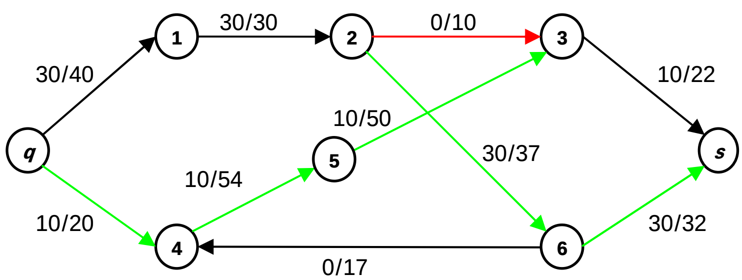


Abbildung 5 Veränderung des Flusses entlang des *EW*

2.6 Der Satz von Ford und Fulkerson

Der Satz, auch bekannt als Max-Flow-Min-Cut Theorem, wurde 1956 von Ford und Fulkerson entwickelt.
Er dient als Grundlage für die späteren Algorithmen.

Satz:

1. Es sei f ein Fluss eines Netzwerkes. Genau dann ist f ein maximaler Fluss, wenn es keinen Erweiterungsweg bezüglich f gibt.
2. Der Wert des maximalen Flusses in einem Netzwerk ist gleich der minimalen Kapazität eines Schnittes.

Für die folgenden Kapitel ist der Punkt 1 von besonderem Interesse.
Sind also in einem Netzwerk alle Erweiterungswege gefunden und der Wert des Flusses wurde mit diesen erhöht, so ist der maximale Fluss erreicht.

3. Bestimmung von Erweiterungswegen

Es stellt sich nun die Frage, wie man die benötigten Erweiterungswege in einem Netzwerk bestimmen kann.

Zunächst wird aus dem Graphen G ein Restnetzwerk G_f konstruiert.
In G_f entspricht die Eckenmenge genau der des Ausgangsgraphen. Für die Kanten gilt:

- Eine Kante wird aus G übernommen, falls auf dieser noch eine Restkapazität vorhanden ist. Sie erhält die Bewertung $f(\mathrm{k}) - c(\mathrm{k})$.
- Zu einer Kante aus G wird eine Gegenkante erzeugt, falls der aktuelle Fluss auf dieser Kante > 0 ist. Sie erhält die Bewertung des aktuellen Flusses.

So können in G_f zwei Ecken durch antiparalelle Kanten verbunden werden, wenn bereits ein Fluss fließt, jedoch auch noch eine Restkapazität vorhanden ist.

Im Folgenden wird die Konstruktion des Restnetzwerkes am Beispiel des bekannten Graphen veranschaulicht.

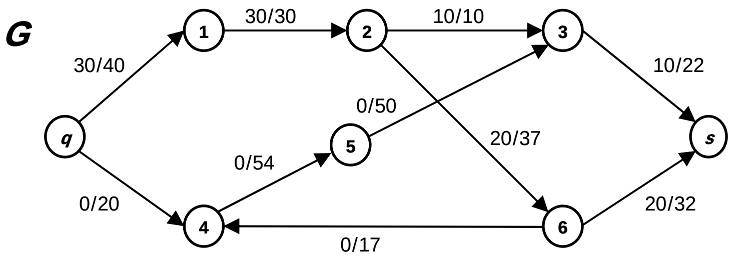


Abbildung 6 Der Ausgangsgraph *G*

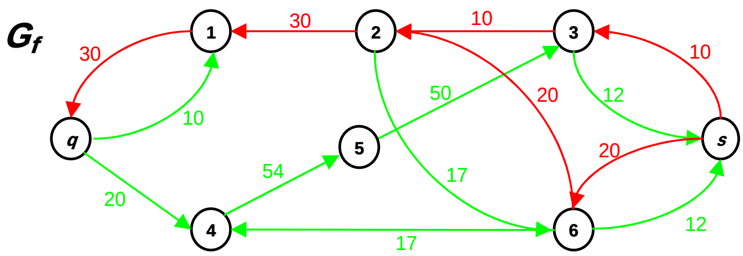


Abbildung 7 Das Restnetzwerk G_f

Nun können die Erweiterungswege für diesen Graphen mit Hilfe einer Breitensuche auf dem Restnetzwerk bestimmt werden.

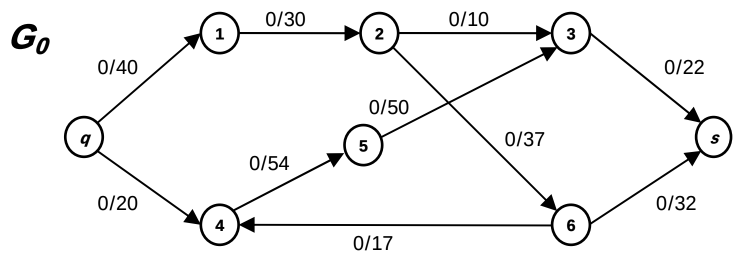
Die Breitensuche wird in q gestartet. Von dort geht man jeweils zu den benachbarten Ecken und markiert diese. Von der zuerst markierten Ecke sucht man sich wiederum die nächste, noch nicht markierte Ecke und markiert diese ebenfalls. Dies wird fortgesetzt, bis ein Erweiterungsweg mit minimaler Anzahl von Kanten s erreicht. Endet die Breitensuche bevor s erreicht ist, so ist der Fluss bereits maximal.

4. Der Algorithmus von Edmonds und Karp

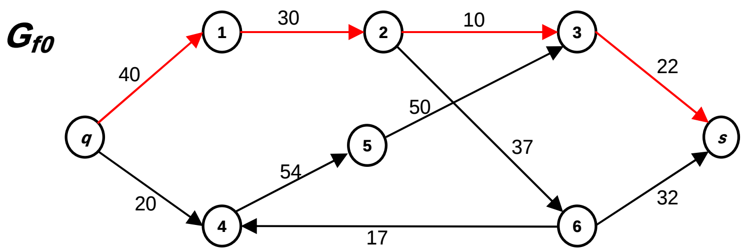
Auch Edmonds und Karp beschäftigten sich mit dem Problem des Flusses.
Sie knüpfen dabei an die Grundlagen von Ford und Fulkerson an.

Für den Graphen G mit Fluss f wird, wie im vorigen Kapitel beschrieben, das entsprechende Restnetzwerk G_f konstruiert und mit Hilfe der Breitensuche ein Erweiterungsweg auf G_f bestimmt.

Auf diesem Erweiterungsweg wird der Fluss um $f\Delta$ so verändert, dass der Wert auf den Vorwärtskanten erhöht und auf den Rückwärtskanten reduziert wird. Damit erhöht sich der Wert des Flusses um $f\Delta$. Ausgehend von dem nun entstandenen Graphen wird erneut das Restnetzwerk erzeugt und ein neuer Erweiterungsweg bestimmt, auf welchem der Fluss erhöht werden kann. Dieser Vorgang wird so lange wiederholt, bis kein Erweiterungsweg von q nach s gefunden werden kann und damit der maximale Fluss erreicht ist.
Zur graphischen Verdeutlichung wird folgend schrittweise die Vergrößerung des Flusses dargestellt.



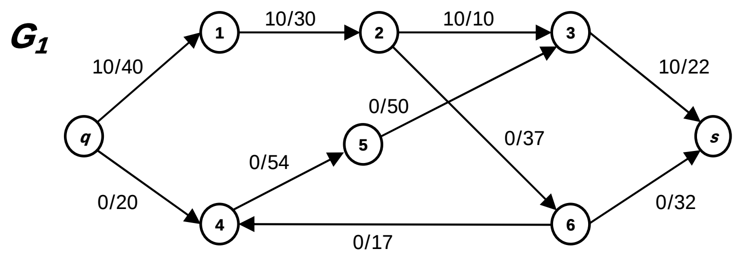
$|f_0| = 0$



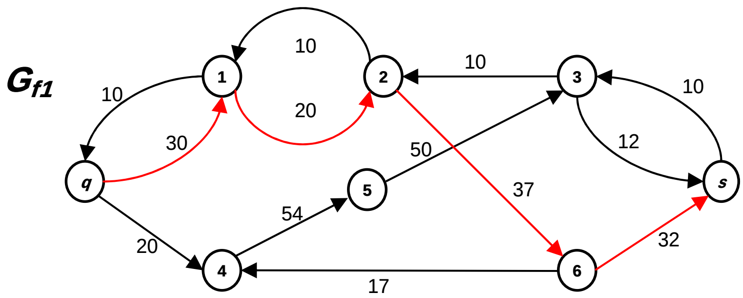
$EW = q,\ 1,\ 2,\ 3,\ s$

$f_v = 10$ $\qquad f_r = \infty$ $\qquad f\Delta = 10$

Abbildung 8 Eine Anwendung des Algorithmus von Edmonds und Karp

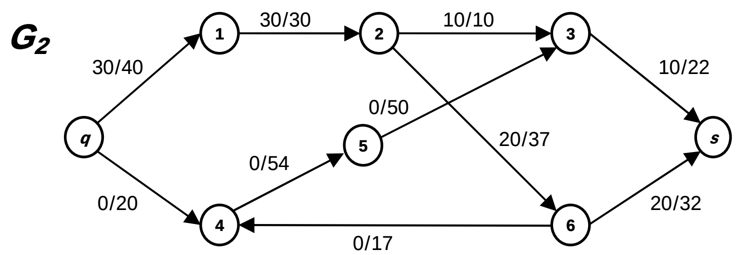


$|f_1| = 10$

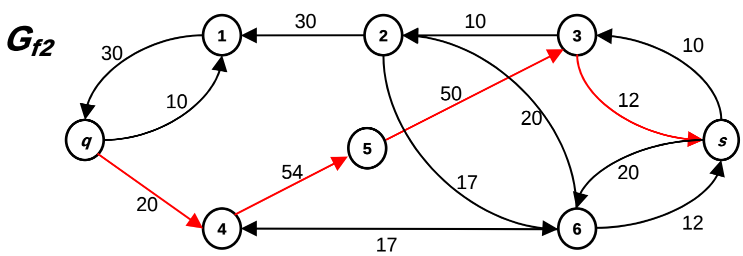


$EW = q,\ 1,\ 2,\ 6,\ s$

$f_v = 20$ $\qquad$ $f_r = \infty$ $\qquad$ $f\Delta = 20$



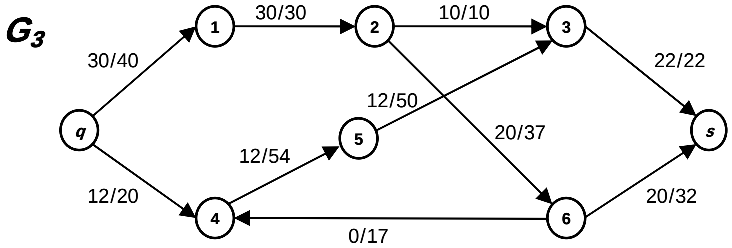
$|f_2| = 30$



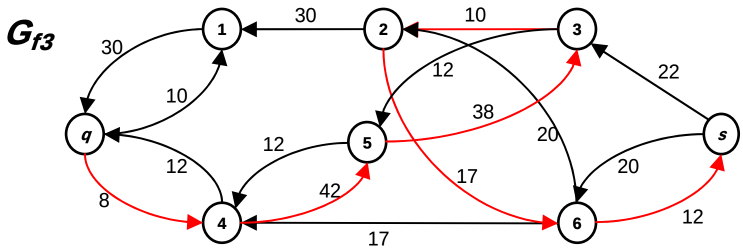
$EW = q,\ 4,\ 5,\ 3,\ s$

$f_v = 12$ $\qquad$ $f_r = \infty$ $\qquad$ $f\Delta = 12$

Abbildung 9 Eine Anwendung des Algorithmus von Edmonds und Karp

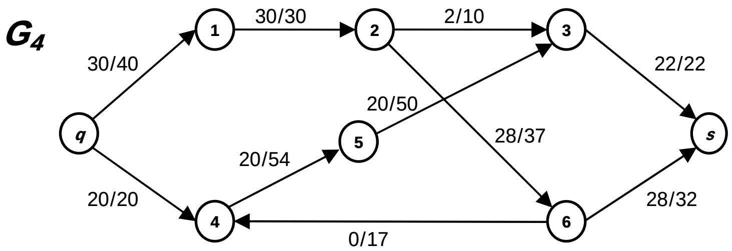


$|f_3| = 42$



$EW = q,\ 4,\ 5,\ 3,\ 2,\ 6,\ s$

$f_v = 8$ $f_r = 10$ $f\Delta = 8$



$|f_4| = 50$

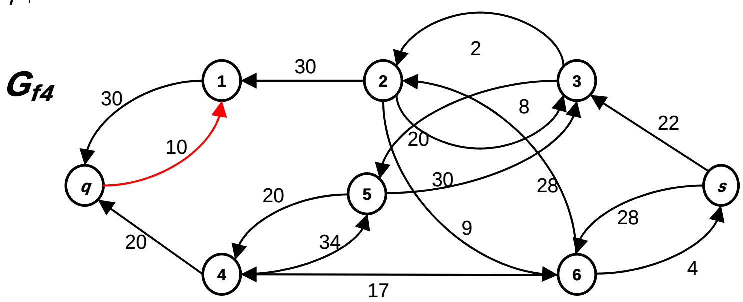


Abbildung 10 Eine Anwendung des Algorithmus von Edmonds und Karp

In G_{f4} kann kein Erweiterungsweg von *q* nach *s* gefunden werden, er endet in 1. Somit ist der maximale Wert des Flusses in G_4 erreicht worden.

Der Algorithmus von Edmonds und Karp bestimmt in maximal *m n/2* Schritten einen maximalen Fluss. Er besitzt eine worst case Komplexität von $O(n\ m^2)$.

5. Der Algorithmus von Dinic

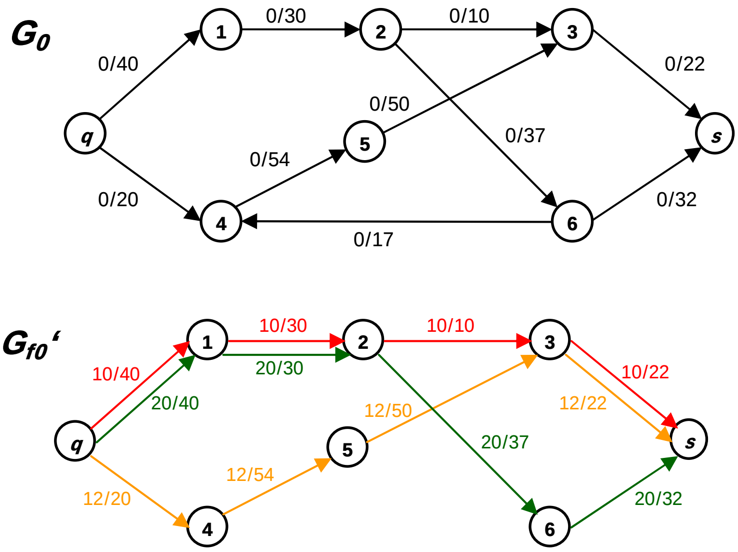
Für dichte Netzwerke (*m = O(n2)*) wurde ein effizienterer Algorithmus von E.A. Dinic entwickelt. Während der Algorithmus von Edmonds und Karp eine worst case Laufzeit von *O(n5)* besitzt, gelingt Dinic dies mit einer worst case Laufzeit von *O(n3)*.

Die Erhöhung des Flusses erfolgt dabei auf dem gesamten Netzwerk und nicht mehr nur entlang einzelner Wege. Des Weiteren werden Hilfsnetzwerke, im Folgenden als Niveaunetzwerke G_f' bezeichnet, verwendet.
Auf diesen Niveaunetzwerken werden alle möglichen Erweiterungswege geschichtet und damit ein Fluss konstruiert, der nicht maximal sein muss, für den es jedoch keinen Erweiterungsweg gibt, der nur aus Vorwärtskanten besteht. Ein solcher Fluss ist ein *blockierender Fluss*.

Wie bereits in den vorigen Kapiteln beschrieben wurde, wird aus dem Graphen *G* zunächst wieder das Restnetzwerk G_f gebildet.
Aus G_f wird nun das Niveaunetzwerk G_f' konstruiert, wobei in diesem „überflüssige Kanten" entfernt werden. Dies geschieht, in dem das Niveau jeder Ecke festgestellt wird. Das Niveau einer Ecke *Niv(e)* ist die minimale Anzahl von Kanten, mit denen diese Ecke von *q* aus erreicht werden kann. Dies bedeutet, dass G_f' nur die Kanten $k = (e, f)$ enthält, für die gilt $Niv(e) + 1 = Niv(f)$.
Des Weiteren gilt in G_f' $g^-(q) = g^+(s) = 0$.

Auch dieser Algorithmus wird durch ein graphisches Beispiel verständlich gemacht. Auf die Abbildungen der jeweiligen Restnetzwerke wird verzichtet, da sie zuvor schon detailliert beschrieben und dargestellt wurden. Folgend also jeweils der Graph und das Niveaunetzwerk der einzelnen Schritte.

Dafür wird wie im vorigen Kapitel mit demselben Beispiel und dem trivialen Fluss begonnen.



Die Kante $(6,4)$ ist weggefallen, da $Niv(e_4) = 1$ und damit kleiner als $Niv(e_6) = 3$.

Das Niveau von s ist 4, da s mit minimal 4 Kanten erreicht werden kann.

Alle Erweiterungswege, die s mit dem Niveau von 4 erreichen werden nun in G_{f0}' geschichtet und schließlich wird der Fluss auf den Erweiterungswegen erhöht.

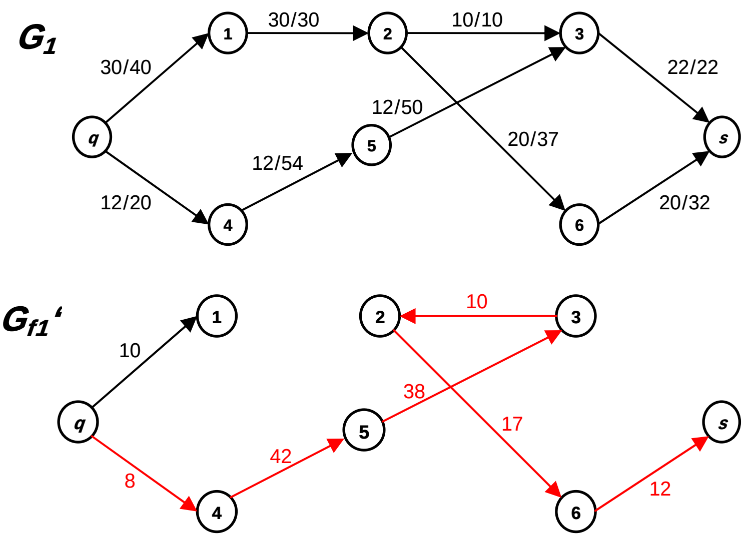


Abbildung 11 Eine Anwendung des Algorithmus von Dinic

Aufgrund der Eckenniveaus sind in G_{f1}' die Kanten (*2,3*) und (*s,3*) entfernt worden. Die Senke wird im Niveau 6 durch den einzig möglichen Erweiterungsweg erreicht. Entlang des EW wird der Fluss um 8 verändert und $|f|$ damit um 8 erhöht.

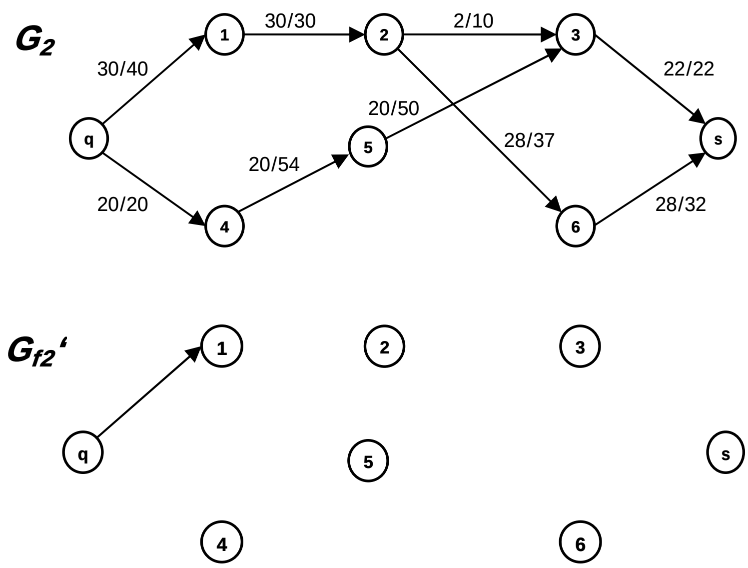


Abbildung 12 Eine Anwendung des Algorithmus von Dinic

Im Niveaunetzwerk G_{f2}' gibt es nur noch einen Weg von q nach 1, nicht jedoch nach s. Der maximale Fluss wird für das Beispiel in G_2 mit $|f| = 50$ erreicht.

Während der im vorigen Kapitel erläuterte Algorithmus von Edmonds und Karp den maximalen Fluss erst in G_4 berechnet hat, erreicht man dies mit dem Algorithmus von Dinic bereits in G_2.

6. 0/1 – Netzwerke

Ein 0/1-Netzwerk ist ein Spezialfall von allgemeinen Netzwerken.
In diesen Netzwerken interessiert nicht mehr der maximale Fluss, denn jede Kante besitzt lediglich die Kapazität 1 oder 0. Dies bedeutet, ein Fluss fließt durch eine Kante oder nicht.
Für diese Netzwerke stellt sich die Frage, wie viele Wege gefunden werden können, welche keine gemeinsame Kante benutzen.
Auch für dieses Problem kann der Algorithmus von Dinic zur Lösungsfindung eingesetzt und sogar noch verbessert werden.
Bei einem maximalen Fluss f von G, der auf allen Kanten den Wert 0 oder 1 besitzt, existiert mindestens einen Fluss von q nach s, welcher nur aus Kanten mit dem Fluss 1 besteht. Wird der Fluss auf diesen Kanten auf 0 gesetzt, verringert sich zum einen der Wert des Flusses um 1, und es existiert wiederum ein Weg von q nach s, der nur Kanten mit dem Fluss 1 besitzt.
Dieser Weg hat mit dem vorherigen keine gemeinsam genutzte Kante.
Da jeder neue Weg den Fluss 1 hat, erhöht sich der Wert des Flusses in jedem Schritt um 1. Zum Schluss erhält man so $|f|$ Wege von q nach s, welche paarweise keine Kante gemeinsam haben, also $|f|$ kantendisjunkte Wege.

7. Schlusswort

Schon bei kleineren Netzwerken wird die Suche nach optimalen Lösungen zu maximalen Flüssen unübersichtlich und bei steigender Größe und Komplexität ist diese ohne die Anwendung der entwickelten Verfahren nicht mehr möglich.
Die in dieser Arbeit vorgestellten Algorithmen sind jedoch nicht die einzigen Verfahren, die bis heute entwickelt wurden. Sie bieten jedoch einen guten Einblick in die Möglichkeiten der Lösungsalgorithmen für Flussprobleme.
Der zur heutigen Zeit schnellste Algorithmus zur Bestimmung von maximalen Flüssen wurde 1985 von Goldberg entwickelt und arbeitet vollkommen unabhängig von dem Konzept des Erweiterungsweges.

Aufgrund der Tatsache, dass Kapazitäten immer eine aktuelle obere Schranke besitzen und diese aus wirtschaftlicher Sicht im besten Fall möglichst ausgenutzt werden sollen, ist die Anwendung von Algorithmen zur Berechnung der maximalen Flüsse unbedingt empfehlenswert. Dies trifft ebenfalls für 0/1-Netzwerke zu, um die größte Anzahl an kantendisjunkten Wegen zu finden.
Einen experimentellen Vergleich von Algorithmen zur Bestimmung von maximalen Flüssen wurde bereits von Cherkassky und Goldberg erstellt.

Abbildungsverzeichnis

Literaturverzeichnis

1. Turau, Volker, Algorithmische Graphentheorie, Addison-Wesley 1996

Internetquellen

1. http://www.tfh-berlin.de/~oanhle/
2. http://www.imn.htwk-leipzig.de/~waldmann/edu/ws04/graph/folien/graph/graph.html
3. http://de.wikipedia.org/wiki/Flüsse_und_Schnitte_in_Netzwerken
4. i11www.ira.uka.de/algo/ teaching/scripts/sources/maxflow.pdf