Graphentheoretische Definition von Flüssen in Netzwerken

Ein Flussnetzwerk ist ein gerichteter Graph $G=(V,E)$, mit zwei ausgezeichneten Knoten, einer Quelle s und einer Senke t, bei dem jede Kante $(u,v)\in E$ eine nicht - negative Kapazität $c(u,v)\geq 0$ besitzt.

- Die Kapazitätsfunktion ist $c:E\rightarrow \boldsymbol{R}$
- Ein Fluss ist durch die Funktion $f:V\times V\rightarrow \boldsymbol{R}$ gegeben, welche folgende Bedingungen erfüllen muss:

  (a)*Kapazitätsbeschränkung*: Für alle $(u,v)\in V$ gilt: $f(u,v)\leq c(u,v)$
  Ein Fluss durch einen Pfeil muss dessen Kapazitätsbeschränkung einhalten

  (b)*Schiefe Symmetrie*: Für alle $(u,v)\in V$ gilt $f(u,v)=-f(v,u)$

  (c)*Flusserhaltung*: Für alle $u\in V-\{s,t\}$ ist $\sum_{v\in V} f(u,v)=0$

  Kann man auch schreiben als $\underbrace{\sum_{(v',v)\in E} f(v',v)}_{\text{'fliesst in v hinein'}} - \underbrace{\sum_{(v,v'')\in E} f(v,v'')=0}_{\text{'fliesst aus v heraus'}}$

  An jedem Knoten, außer an der Quelle und an der Senke, muss der Fluss erhalten bleiben also gleichviel hinein- wie herausfließen.

- Die Größe f(u,v) ist der Fluss von Knoten u zu Knoten v.
- Der Wert eines Flusses ist $|f|=\sum_{v\in V} f(s,v)$, also die Summe der Flusswerte aller s verlassenden Kanten
- Der Nettofluss in einem Knoten ist die Summe aller den Knoten verlassenden Flüsse minus aller in den Knoten hinein fließenden Flüsse.

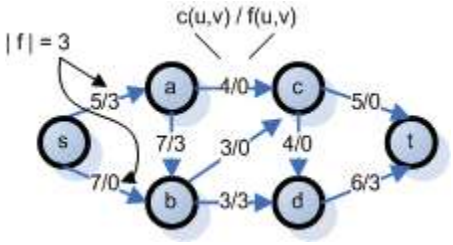

Abbildung 1 Fluss im Netzwerk

Maximale Flüsse

Ein maximaler Fluss f in G ist ein Fluss f mit maximalem Wert f unter allen Flüssen in G.

• *Wie groß kann ein maximaler Fluss überhaupt sein ?*

➔ In einem Netzwerk kann nicht mehr fließen, als aus der Quelle hinaus fließt oder in die Senke hineinfließt.

➔ Der Wert eines jeden Flusses f in einem Flussnetzwerk G wird durch die Kapazität eines beliebigen Schnittes (S,T) begrenzt.

Schnitte in Flussnetzwerken

Ein Schnitt ist eine Zerlegung der Knotenmenge V in 2 disjunkte Teilmengen S und T, so dass $s \in S$ und $t \in T$.
Der Nettofluss in einem Schnitt ist die Summe des positiven Flusses von S nach T minus dem von T nach S, also der gesamte Fluss aus der Menge S abzüglich dem Fluss, welcher aus T wieder in S zurückfließt.
Die Kapazität $c(S,T)$ eines Schnittes S,T ist die Summe der Kapazitäten aller Kanten die von S nach T führen.
Ein minimaler Schnitt ist ein Schnitt mit der kleinsten Kapazität unter allen möglichen Schnitten.
Der Nettofluss ist über jeden Schnitt gleich und $f(S,T) = |f|$
Ist also f ein Fluss in einem Flussnetzwerk G und (S,T) ein Schnitt von G. Dann ist der Nettofluss über (S,T) gleich dem Wert des Flusses f. Dies ist für jeden Fluss und Schnitt gültig, da bei jedem Schnitt von S nach T genauso viel fließen muss wie der Wert des Flusses ist.

Restgraphen

Man kann das Abändern von Flüssen in Wegen von s nach t ausdrücken, wenn man nicht nur das Erhöhen eines Flusses entlang seiner Kante mit noch freier Restkapazität in Betracht zieht, sondern auch das Verringern eines Flusses entlang einer Kante, also gewissermaßen das Erhöhen eines Flusses entgegen der Kantenrichtung.
Einen Fluss kann man höchstens um f(u,v) Einheiten verringern, bis f(u,v) = 0

Das führt zum Begriff des Restgraphen $Gf,=(V,E_f)$ welcher zu einem Fluss f gerade alle Flussvergrößerungsmöglichkeiten beschreibt.

Wenn weder (u,v) noch (v,u) im ursprünglichen Netzwerk erscheint, dann gilt:
c(u,v) = c(v,u) = 0, f(u,v) = f(v,u) = 0, c_f(u,v) = c_f(v,u) = 0 und man kann daraus schließen, dass eine Kante (u,v) nur dann im Restgraphen erscheinen kann, wenn mind. (u,v) oder (v,u) im ursprünglichen Netzwerk vorkommen.
Daher gilt $|E_f| \leq 2|E|$

Die Restkapazität einer Kante $(u,v) \in V$ ergibt sich durch $c_f(u,v) = c(u,v) - f(u,v)$
Eine Vorwärtskante im Restgraphen ist eine Kante in Richtung des Flusses f in G. Sie entsteht wenn, wenn für eine Kante in G gilt f(u,v) < c(u,v) , wenn also noch Restkapazität entlang des Flusses vorhanden ist.
Eine Rückwärtskante im Restgraphen ist eine Kante in entgegengesetzter Richtung des Flusses f in G und sie entsteht, wenn die Restkapazität c_f der *Kante in entgegengestzter* Richtung größer 0 ist: $c_f(v,u) = c(u,v) - f(u,v) > 0$

- Den gesamten Fluss kann man höchstens um das Minimum der gefundenen Restkapazitäten erhöhen.

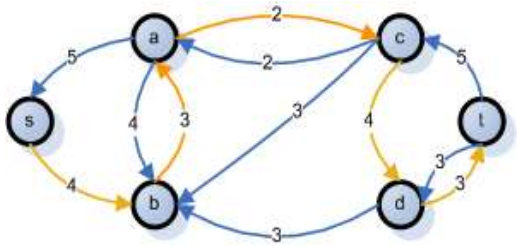

Abbildung 2 Restgraph zu Abb. 1

Zunehmende Wege

Jeder Weg von S nach T im Restgraphen ist ein zunehmender Weg.

Ein Fluss ist genau dann maximal, wenn es für f keinen zunehmenden Weg gibt.
→ Denn wenn es einen zunehmenden Weg für einen Fluss f gibt, dann könnte man den Fluss entlang dieses Weges vergrößern.

Max. Fluss – min. Schnitt Theorem

Sei f ein Fluss in einem Flussnetzwerk G = (V,E) , mit Quelle s und Senke t, dann gelten folgende Bedingungen:

(1)f ist ein maximaler Fluss in G

(2)Der Restgraph G_f enthält keine zunehmenden Wege

(3)Der Wert des Flusses f entspricht der Kapazität eines Schnittes: $|f|=c(S,T)$ für einen Schnitt (S,T) in G

Ford – Fulkerson Methode

Es wird als Methode bezeichnet, da es nicht um eine konkrete Implementierung eines Algorithmus handelt, sondern es verschiedene Implementierungen mit unterschiedlichen Laufzeiten gibt.
Die Methode funktioniert iterativ und nuzt zunehmende Wege in Restgraphen G_f um den Maximalen Fluss f in einem Graphen G zu bestimmen.
Dabei hängt die Laufzeit des Algorithmus maßgeblich davon ab, wie die zunehmenden Wege gefunden werden.

Laufzeit: O(E |f*|)
Dabei ist |f*| der Wert des maximalen Flusses und damit die Anzahl der maximal möglichen Flusserhöhungen.

```
1 for each edge (u,v) ϵ E
2     do f[u,v] ← 0                                        Θ(E)
3        f[v,u] ← 0
4 while es gibt einen Weg p von s nach t im Restgraphen
5     do cf(p) ← min {cf(u,v): (u,v) ist auf p}            O(E)
6           for jede Kante (u,v) in p
7                 do f[u,v] ← f[u,v] + cf(p)
8                    f[v,u] ← - f[v,u]                     O(E)
                                                  f* - Mal
                                                  O(E |f*| )
```

- In den Zeilen 1 – 3 wird zunächst jede Kante mit einem Nullwert initialisiert.
  → Θ(E) , wobei E die Anzahl der Kanten ist
- Zeilen 4 – 8 verrichten die eigentliche Arbeit.
  Bei jedem Durchlauf wird der Wert des Flusses entlang des Weges p um die kleinste auf dem Weg p gefundene Restkapazität erhöht.
  Das heißt, solange ein zunehmender Weg im Restgraphen existiert (Zeile 4) wird folgendes gemacht:
  In Zeile 5 wird das Minimum aller Restkapazitäten der auf p liegenden Kanten bestimmt, und wird der Fluss entlang p um dieses Minimum erhöht (Zeile 6 bis 8).
- f* ist der maximale Fluss und |f*| der Wert des maximalen Flusses.
- Die While - Schleife kann höchsten |f*| - Mal ausgeführt werden, da der Flusswert minimal um eins pro Durchlauf der Schleife erhöht wird und damit maximal f* Schleifendurchläufe

stattfinden können.
- Die Auswahl des zunehmenden Wegs ist zwar zufällig, aber um alle zunehmenden Wege zu finden muss man Tiefen oder Breitensuche verwenden, welche O(E‘) Zeit benötigen. E‘ sind die Kanten des Restgraphen welche auf O(E) approximiert werden können.
- Bei einem gegebenen Weg im Restgraphen kann dieser Weg höchstens |E| Kanten lang sein.

Sofern die Kapazitäten und der daraus resultierende Maximalfluss niedrig ist, ist die Laufzeit akzeptabel, jedoch wird es zu einem Problem wenn die Kapazitäten und der resultierende Maximalfluss sehr groß ist.

Edmons – Karp Algorithmus

Dieser Algorithmus isr eine verbesserte Variante der Ford-Fulkerson Methode, welcher die zunehmenden Wege durch Breitensuche findet. Deshalb muss der jeweils gefundene zunehmende Weg auch gleichzeitig ein kürzester Weg von s nach t im Restgraphen sein.
In jedem Knoten wird zusätzlich die Entfernung zur Quelle s gespeichert.
Generell wird bei dieser Implementierung angenommen das die Entfernung des kürzesten zunehmenden Weges von der Quelle zu einem Knoten monoton mit jeder Flusserhöhung wächst.
Wenn der Edmonds-Karp Algorithmus auf einem Flussnetzwerk G = (V,E) mit Quelle s und Senke t läuft, dann ist die Anzahl der durch den Algorithmus durchgeführten Flusserhöhungen O(V E).
Es ergibt sich dann eine Gesamtlaufzeit von $O(V E^2)$

Bipartites Matching

Ein ungerichteter Graph ist ein bipartiter Graph $G=(V,E)$,wenn man die Menge der Knoten V in 2 disjunkte Teilmengen L und R aufteilen kann $V=L\cup R$, und es ausschließlich Kanten (l,r) mit $l\in L$ und $r\in R$ gibt.
Ein Bipartites Matching ist eine Menge $M\subseteq E$, so dass für alle Kanten $v\in V$ maximal eine Kante in M existiert.
Ein maximales Matching ist ein Matching mit der höchsten Anzahl an Kanten, so das $|M|\geq|M'|$ für jedes Matching M‘ für einen Graph G.

Um ein maximales Matching in einem bipartitem Graphen zu finden kann dieser in ein Flussnetzwerk umgewandelt werden, dann kann man mittels der Ford-Fulkerson Methode den Maximalen Fluss in diesem Netz finden.
Dazu muss man nur zunächst durch folgende Schritte den Graphen in ein Flussnetzwerk umwandeln:
1. Hinzufügen der Knoten s und t als Quelle und Senke.
2. Die Kantenmenge des bipartiten Graphen wird übernommen und ihre Richtung in dem gerichtetem Graph ist jeweils aus der Menge L in die Menge R.
3. Zusätzlich werden neue Kanten von s zu allen Knoten in der Menge L und von allen Knoten in der Menge R nach t eingeführt.
4. Nun wird jeder Kante in der entstandenen Menge eine gleich bleibende Kapazität 1 zugewiesen.

Quellen: •Introduction to Algorithms,Cormen, Leiserson, Rivest, Stein •Algorithmen und Datenstrukturen,Ottmann, Widmayer