Graphentheoretische Konzepte und Algorithmen

Thema 05
Flüsse

Julia Padberg

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Flussprobleme

Grundannahmen

- Modellierung von Transport von Gütern (Strom, Container etc.) entlang der Kanten
- Modellierung mit **schwach zusammenhängenden, schlichten gerichteten Graphen** $G(V, E)$ **mit** $|V| = n$ **Knoten**
- $c(e_{ij}) = c_{ij}$ Kapazität einer Kante $v_i v_j = e_{ij}$ ist die Menge dieses Gutes, die entlang dieser Kante transportiert werden kann, wobei
- alle c_{ij} sind rationale Zahlen sind.
- Solche Graphenwerden oft als **Netzwerke** bezeichnet, z.B.:
  - die Kanten als Nonstop-Flugverbindungen mit Kapazitäten als Passagiere pro Zeiteinheit
  - die Kanten als Telefonleitungen mit Kapazitäten als Anzahl gleichzeitiger Telefongespräche

Begriffe

Definition

Für einen Knoten v_i ist $O(v_i) = \{e_{ij} \in E | s(e_{ij}) = v_i\}$ der **output** dieses Knoten und $I(v_i) = \{e_{ji} \in E | t(e_{ji}) = v_i\}$ der **input** dieses Knoten.
Es gilt $|O(v_i)| = d_-(v_i)$ und $|I(v_i)| = d_+(v_i)$.

Definition

Sei $G = (V, E)$ ein schwach zusammenhängender, schlichter gerichteter Graph mit $|V| = n$ Knoten, dann werden in G zwei Knoten besonders hervorgehoben:
eine Quelle $q := v_1$ mit $d_+(q) = 0$ und
eine Senke $s := v_n$ mit $d_-(s) = 0$.

Fluss

Definition

Ein *Fluss* in G von der Quelle $q = v_1$ zu der Senke $s = v_n$ ist eine Funktion f, die jeder Kante $e_{ij} \in E$ eine nichtnegative rationale Zahl zuordnet, so dass

1. für jede Kante $e_{ij} : f(e_{ij}) \leq c(e_{ij})$ gilt (Kapazitätsbeschränkung),
2. der gesamte Fluss, der von der Quelle v_1 wegtransportiert wird, in vollem Umfang an der Senke v_n eintrifft, $\sum_{e_{1j} \in O(q)} f(e_{1j}) = \sum_{e_{in} \in I(s)} f(e_{in})$ und
3. für jeden übrigen Knoten, den sogenannten *inneren Knoten*, werden eintreffende Mengen des Gutes verlustlos weitergeleitet, d.h. es gilt die **Flusserhaltung**: $\forall j \in \{1, \ldots n\} : \sum_{e_{ij} \in O(v_i), e_{ji} \in I(v_i)} (f(e_{ij}) - f(e_{ji})) = 0$

Wert des Flusses

Der Wert des Flusses f ist dann

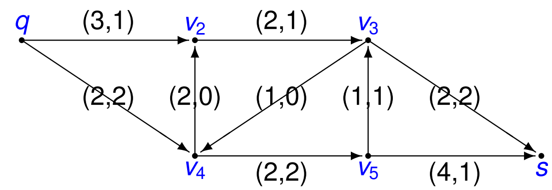
$$d = \sum_{e_{1j} \in O(q)} f(e_{1j}) = \sum_{e_{in} \in I(s)} f(e_{in})$$

Es ist noch zu bemerken, dass in jedem Graphen es nur genau einen Knoten mit $d_+(q) = 0$ und es nur genau einen Knoten mit $d_-(s) = 0$ gibt, d.h. Quelle und Senke sind eindeutig. Dies liegt an der dritten Forderung für den Fluss f.

BSP

BSP:

In dem folgenden Graphen ist jede Kante $e_{ij} = v_i v_j$ mit dem Wertepaar $(c(e_{ij}), f(e_{ij}))$ beschriftet:



Aus der Quelle fließen 3 Mengeneinheiten ab, in der Senke treffen 3 Mengeneinheiten ein, folglich ist der Wert des Flusses 3. In allen anderen Knoten halten sich eintreffende und abfließende Mengen die Waage, und durch jede Kante fließen höchstens soviele Mengeneinheiten, wie ihre Kapazität es zulässt.

Aufgabenstellung:

Maximale Menge, die von der Quelle zur Senke transportiert werden kann

Definition

Es sei f der Fluss eines Graphen $G = (V, E)$, und für jede echte Untermenge X der Knotenmenge V von G bezeichne $\overline{X}$ das **Komplement** von X in V, d.h. $\overline{X} = V \setminus X$.

Wenn X die Quelle q aber nicht die Senke s enthält, sollte intuitiv der Fluss von den Knoten in X zu den Knoten in $\overline{X}$ dem Wert d des gesamten Flusses entsprechen.

Schnitt

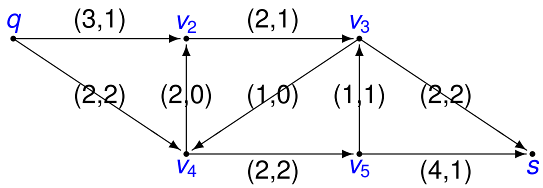
Definition

- Wenn X und Y beliebige Untermengen von Knoten eines Graphen G sind, bezeichnet
  - $A(X, Y)$ die Menge der Kanten, die Knoten aus X mit Knoten aus Y verbinden.
  - $A^+(X, Y)$ ist die Menge der Kanten, ausgehend von Knoten aus X, diese mit Knoten aus Y verbinden.
  - $A^-(X, Y)$ ist die Menge der Kanten, ausgehend von Knoten aus Y, diese mit Knoten aus X verbinden.
- Sei g eine beliebige Funktion, die den Kanten eines Graphen G nichtnegative rationale Zahlen zuordnet, dann ist für zwei beliebige Knotenmengen X, Y von G: $g(X, Y) = \sum_{e \in A^+(X,Y)} g(e)$.
- Ein **Schnitt** ist eine Menge von Kanten $A(X, \overline{X})$, wobei $q \in X$ und $s \in \overline{X}$.

BSP

BSP:

Sei $X = \{q = v_1, v_2, v_3\}$ beliebig und somit $\overline{X} = \{v_4, v_5, s = v_6\}$.



$A(X, \overline{X}) = \{e_{14}, e_{34}, e_{36}, e_{42}, e_{53}\}$ ist die Menge der Kanten, zwischen X und $\overline{X}$.

$A^+(X, \overline{X}) = \{e_{14}, e_{34}, e_{36}\}$ ist die Menge der Kanten aus X

$A^-(X, \overline{X}) = \{e_{42}, e_{53}\}$ ist die Menge der Kanten in X hinein.

Der Fluss von den Knoten in X zu den Knoten in $\overline{X}$:

$\sum_{e_{ij} \in A^+(X,\overline{X})} f(e_{ij}) - \sum_{e_{ij} \in A^-(X,\overline{X})} f(e_{ij}) = (2 + 0 + 2) - (0 + 1) = 3$ Interessant ist auch die Kapazität von X nach $\overline{X}$:

$$c(X, \overline{X}) = \sum_{e_{ij} \in A^+(X,\overline{X})} c(e_{ij}) = 2 + 1 + 2 = 5$$

Erster Fluss-Satz

Satz

Es sei f ein Fluss in einem Graphen $G = (V, E)$, und es sei d der Wert des Flusses. Wenn $A(X, \overline{X})$ ein Schnitt in G ist, dann gilt $d = f(X, \overline{X}) - f(\overline{X}, X)$ und $d \leq c(X, \overline{X})$.

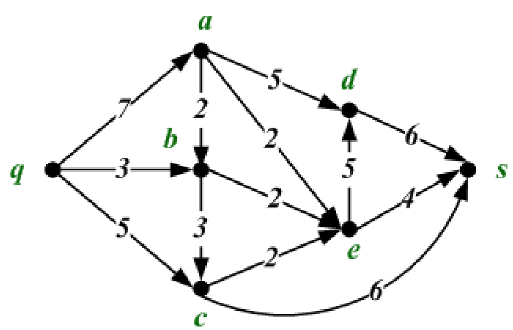
Also

- ist der gesamte von X herauslaufende Fluss minus dem gesamten in X hineinlaufenden Fluss gleich dem gesamten Fluss d
- und dieser überschreitet nie die gesamte Kapazität der Kanten von X nach $\overline{X}$
- und der Wert *jedes beliebigen* Flusses ist kleiner oder gleich der Kapazität der Kanten von X nach $\overline{X}$ für *jeden beliebigen* Schnitt $A(X, \overline{X})$.

Aufgabe 1:

Gegeben ist das Netzwerk N

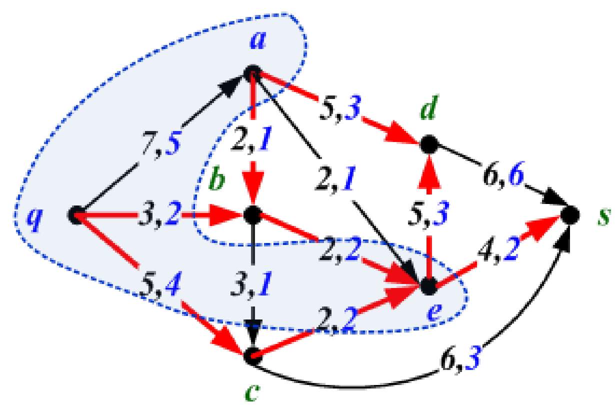
(also ein schlichter, gerichteter, schwach zusammenhängender Graph mit den Kapazitäten)



1. Geben Sie bitte zwei Flüsse f_1 und f_2 an und bestimmen Sie jeweils d_1 und d_2.
2. Geben Sie bitte zwei Schnitte $A(X_1, \overline{X_1})$ und $A(X_2, \overline{X_2})$ an.
3. Berechnen Sie bitte $f_i(X_i, \overline{X_i}) - f_i(\overline{X_i}, X_i)$ und $c(X_i, \overline{X_i})$ für $i = 1, 2$.

Lösung von Aufgabe 1

Fluss f_1 mit $d_1 = 11$



$X_1 = \{q, a, e\}$ und

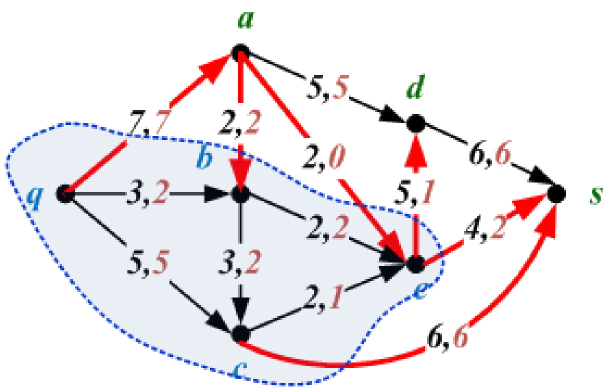
$A(X_1, \overline{X_1}) = \{qb, qc, ab, ad, ed, es, be, ce\}$

$f_1(X_1, \overline{X_1}) - f_1(\overline{X_1}, X_1) = 2 + 4 + 1 + 3 + 3 + 2 - (2 + 2) = 15 - 4 = 11$

$c(X_1, \overline{X_1}) = 3 + 5 + 2 + 5 + 5 + 4 = 24$

Lösung von Aufgabe 1

Fluss f_2 mit $d_2 = 14$



$X_2 = \{q, b, c, e\}$ und
$A(X_2, \overline{X_2}) = \{qa, cs, ed, es, ab, ae\}$
$f_2(X_2, \overline{X_2}) - f_2(\overline{X_2}, X_2) = 7 + 6 + 1 + 2 - (2 + 0) = 16 - 2 = 14$
$c(X_2, \overline{X_2}) = 7 + 6 + 5 + 4 = 22$

Aufgabe 2:

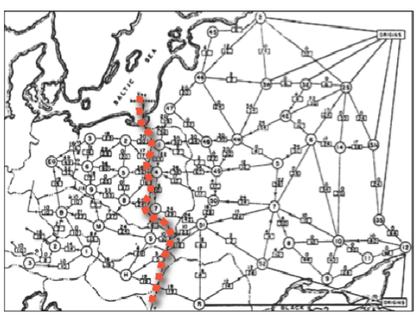
Die inneren Knoten eines Netzwerkes N sind die Knoten $V' := V \setminus \{q, s\}$.
Wieviele Schnitte hat N mit n innere Knoten?

Lösung

Ein Graph G hat 2^n mögliche Schnitte, wenn es n innere Knoten in G gibt.
Denn es gibt diese Schnitte $\{q\} \cup W$, wobei $W \in \mathcal{P}(V')$, und also sind es $2^{|V'|} = 2^n$ verschiedene Schnitte.

Transportkapazität auf Schienennetzen

- Harris-Ross-Report (1954), geheimgehalten bis 1999
- Schienennetz mit 44 Knoten und 105 Kanten
- Problem des Warschauer Pakts: Transport von maximal vielen Gütern nach Westen (max-flow)
- Problem der NATO: eben dieses zu verhindern (min-cut)



Maximalen Flüsse

Ein besonderes Interesse gilt den maximalen Flüssen.

Definition

Ein Fluss f, dessen Wert d maximal ist,
also für alle Flüsse f' gilt $d \geq d'$
heißt **maximaler Fluss**.

Satz

Ein Fluss, dessen Wert

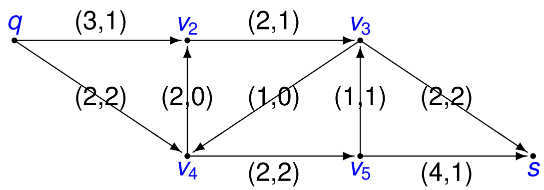
$$\min\{c(X, \overline{X}) | A(X, \overline{X}) \text{ist ein beliebiger Schnitt}\}$$

entspricht, ist ein maximaler Fluss.

BSP

BSP:

Ist $A(X, \overline{X})$ ein Schnitt, dann muss $q \in X$ und $s \in \overline{X}$ sein. Jeder der vier inneren Knoten v_2, v_3, v_4, v_5 ist entweder in X oder $\overline{X}$.



Also gibt es $2^4 = 16$ mögliche Schnitte.
Der Schnitt $A(\{q = v_1, v_2, v_3, v_4\}, \{v_5, s = v_6\})$ hat die kleinstmögliche Kapazität von 4.
Also kann jeder Fluss des Graphen einen Wert von höchstens 4 haben.

Vergrößernder Weg

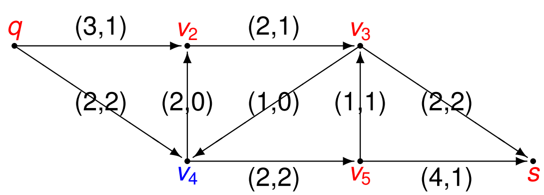
Definition

Ein ungerichteter Weg von der Quelle q zur Senke s heißt ein **vergrößernder Weg**, wenn gilt:

- Für jede Kante e_{ij}, die auf dem Weg entsprechend ihrer Richtung durchlaufen wird (sie wird als *Vorwärtskante* bezeichnet), ist $f(e_{ij}) < c(e_{ij})$.
- Für jede Kante e_{ij}, die auf dem Weg entgegen ihrer Richtung durchlaufen wird (sie wird als *Rückwärtskante* bezeichnet), ist $f(e_{ij}) > 0$.

BSP

In diesem Graphen ist $q = v_1,\ v_2,\ v_3,\ v_5,\ s = v_6$ ein vergrößernder Weg:

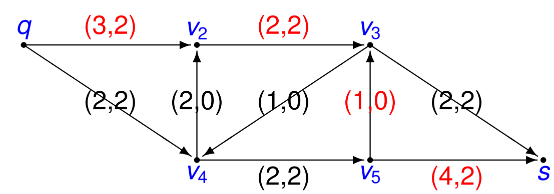


Durch einen vergrößernden Weg kann die Flussstärke d von q nach s folgendermaßen steigen:

- Sei $\delta > 0$ die minimale Differenz zwischen $c(e_{ij})$ und $f(e_{ij})$ für alle Vorwärtskanten des Weges,
  sowie zwischen $f(e_{ij})$ und 0 für alle Rückwärtskanten des Weges.
- Dann wird $f(e_{ij})$ für jede Vorwärtskante auf dem Weg um δ erhöht
- und für jede Rückwärtskante um δ vermindert.

BSP

Damit steigt der Fluss von q nach s um δ unter Einhaltung aller Nebenbedingungen:



Inkrement

Definition

Ist eine Kantenfolge $W = v_1 v_2 \dots v_n$ in dem unterliegenden Graphen G' des Graphen G gegeben, dann sind die zugehörigen Kanten in G entweder in der Form $e_{(i-1)i}$, genannt *Vorwärtskante* von W, oder $e_{i(i-1)}$, genannt *Rückwärtskante* von W.
Wenn f ein Fluss in G ist, wird einer Kantenfolge W eine nichtnegative Zahl $i(W)$, das **Inkrement** von W, zugeordnet, wobei

$$i(W) = \min\{i(e_{ij}) | e_{ij} \text{ist eine Kante der Kantenfolge} W\}$$

$$i(e_{ij}) = \begin{cases} c(e_{ij}) - f(e_{ij}) & \text{falls } e_{ij} \text{ eine Vorwärtskante von } W \\ f(e_{ij}) & \text{falls } e_{ij} \text{ eine Rückwärtskante von } W \end{cases}$$

Zweiter Fluss-Satz

Satz

Wenn in einem Graphen G ein Fluss der Stärke d von der Quelle q zur Senke s fließt, gilt genau eine der beiden Aussagen:

1. Es gibt einen vergrößernden Weg.
2. Es gibt einen Schnitt $A(X, \overline{X})$ mit $c(X, \overline{X}) = d$.

Beweis

ZZ genau eine Aussage ist wahr

- Die Ausagen schließen sich gegenseitig aus.
- Von q ausgehend werden andere Knoten erreicht, wobei für Vorwärtskanten $f(e_{ij}) < c(e_{ij})$ oder für Rückwärtskanten $f(e_{ji}) > 0$ gelten muss:
  - Entweder wird so der Knoten s erreicht, dann ist das ein vergrößernder Weg.
  - Oder in Q seien alle von q aus erreichbaren Knoten und $A(Q, \overline{Q})$ der zugehörige Schnitt.
    Für jede Vorwärtskante aus $A(Q, \overline{Q})$ gilt $f(e_{ij}) = c(e_{ij})$ und für jede Rückwärtskante aus $A(Q, \overline{Q})$ gilt $f(e_{ij}) = 0$. Sonst wäre der in $\overline{Q}$ gelegene Endknoten (bzw. der in $\overline{Q}$ gelegene Anfangsknoten) von q aus über einen ungerichteten Weg zu erreichen.
    Wegen des ersten Fluss-satzes ist dann
    $d = f(Q, \overline{Q}) - f(\overline{Q}, Q) = c(Q, \overline{Q}) - 0 = c(Q, \overline{Q})$

q.e.d.

Max-flow-Min-cut Theorem von Ford und Fulkerson

Satz

In einem schwach zusammenhängendem, schlichten Digraphen G mit genau einer Quelle q und genau einer Senke s sowie der Kapazitätsfunktion c und dem Fluss f ist das Minimum der Kapazität eines q und s trennenden Schnitts gleich der Stärke eines maximalen Flusses von q nach s.

Ganzzahligkeitseigenschaft

Bemerkung

Bei ganzzahligen Werten $f(e_{ij})$ und $c(e_{ij})$ muss die zu einem vergrößernden Weg gehörende Größe δ ebenfalls ganzzahlig sein. Wenn wir also bei ganzzahligen Kapazitäten $c(e_{ij})$ ausgehend von $f(e_{ij}) = 0$ für alle $i, j = 1, \ldots, n$ den Maximalwert für d durch wiederholtes Suchen vergrößernder Wege bestimmen, werden alle dabei auftretende Werte für $f(e_{ij})$ und d ganzzahlig sein.

Der Algorithmus von Ford und Fulkerson

Algorithmus

Gegeben sei ein schwach zusammenhängender, schlichter Digraph $G = (V, E)$, eine Kapazitätsfunktion c und ein Fluss f.

- (1) **Initialisierung** Jede Kante wird einem Wert $f(e_{ij})$ initialisiert, der die Nebenbedingungen erfüllt. Markiere q mit $(undef, \infty)$.
- (2) **Inspektion und Markierung**
  - Wähle einen beliebigen markierten, aber noch nicht inspizierten Knoten v_i und inspiziere ihn wie folgt (Berechnung des Inkrements)
    - Für jede Kante $e_{ij} \in O(v_i)$ mit unmarkierter Knoten v_j und $f(e_{ij}) < c(e_{ij})$ markiere v_j mit $(+v_i, \delta_j)$, wobei δ_j die kleinere der beiden Zahlen $c(e_{ij}) - f(e_{ij})$ und δ_i ist.
    - Für jede Kante $e_{ji} \in I(v_i)$ mit unmarkiertem Knoten v_j und $f(e_{ji}) > 0$ markiere v_j mit $(-v_i, \delta_j)$, wobei δ_j die kleinere der beiden Zahlen $f(e_{ij})$ und δ_i ist.
  - Falls alle markierten Knoten inspiziert wurden, gehe nach 4.
  - Falls s markiert ist, gehe zu 3, sonst zu 2.

Der Algorithmus von Ford und Fulkerson

Algorithmus

- (3) **Vergrößerung der Flussstärke** Bei s beginnend lässt sich anhand der Markierungen der gefundene vergrößernde Weg bis zum Knoten q rückwärts durchlaufen. Für jede Vorwärtskante wird $f(e_{ij})$ um δ_s erhöht, und für jede Rückwärtskante wird $f(e_{ji})$ um δ_s vermindert. Anschließend werden bei allen Knoten mit Ausnahme von q die Markierungen entfernt. Gehe zu 2.
- (4) Es gibt keinen vergrößernden Weg. Der jetzige Wert von d ist optimal. Ein Schnitt $A(X, \overline{X})$ mit $c(X, \overline{X}) = d$ wird gebildet von genau denjenigen Kanten, bei denen entweder die Anfangsknoten oder die Endknoten markiert ist.

BSP: Algorithmus von Ford und Fulkerson

Siehe Buch: Christoph Klauck, Christoph Maas, Graphentheorie und Operations Research für Studierende der Informatik,
Wissner Verlag, 1999,
Seiten 94-96

Vereinfachung

- Ford Fulkerson sucht durch das *beliebig* einen beliebigen vergrößernden Weg;
- Edmond und Karp suchen hier den kürzesten vergrößernden Weg, der an dieser Stelle durch Verwendung einer Queue für die markierten Ecken erzielt werden kann.

Vereinfachung für „händische“ Ausführung

Für die „händische“ Ausführung dürfen Sie mit Tiefensuche und einen impliziten A^* vereinfachen.

Vereinfachung

. . . . problematisch für die Implementierung

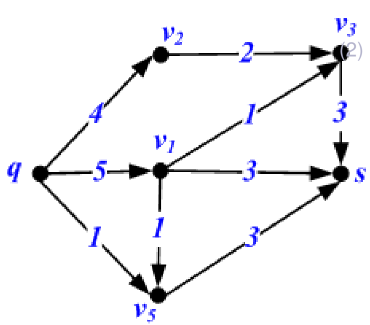
(2) **Inspektion und Markierung**

- Wiederhole bis s erreicht ist:
  Wähle einen **geeigneten Knoten**[1] v_i, der markiert, aber noch nicht inspiziert ist, und inspiziere ihn wie folgt (Berechnung des Inkrements)
  - Für jede Kante $e_{ij} \in O(v_i)$ mit unmarkierter Knoten v_j und $f(e_{ij}) < c(e_{ij})$ markiere v_j mit $(+v_i, \delta_j)$, wobei δ_j die kleinere der beiden Zahlen $c(e_{ij}) - f(e_{ij})$ und δ_i ist.
  - Für jede Kante $e_{ji} \in I(v_i)$ mit unmarkiertem Knoten v_j und $f(e_{ji}) > 0$ markiere v_j mit $(-v_i, \delta_j)$, wobei δ_j die kleinere der beiden Zahlen $f(e_{ij})$ und δ_i ist.
- Falls s markiert ist, gehe zu 3.
- Falls es keinen **geeigneten Knoten**[1] gibt, gehe nach 4.

[1] Hier steckt A^* und Tiefensuche mit drin.

Aufgabe 3:

Berechnen Sie bitte für folgendes Netzwerk mit Hilfe des Algorithmus von Ford und Fulkerson den maximalen Fluss.

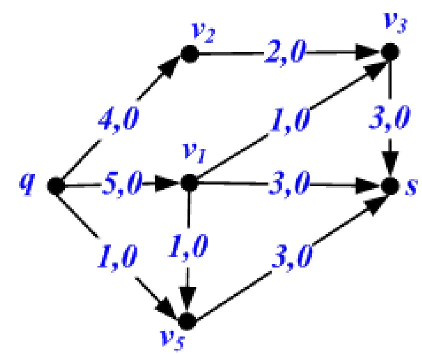


(2) Wiederhole bis s erreicht ist:
Wähle geeigneten Knoten v_i, der markiert, aber nicht inspiziert ist, und inspiziere ihn:

- Für jede Kante $e_{ij} \in O(v_i)$ mit unmarkierter Knoten v_j und $f(e_{ij}) < c(e_{ij})$ markiere v_j mit $(+v_i, \delta_j)$, wobei δ_j die kleinere der beiden Zahlen $c(e_{ij}) - f(e_{ij})$ und δ_i ist.
- Für jede Kante $e_{ji} \in I(v_i)$ mit unmarkiertem Knoten v_j und $f(e_{ji}) > 0$ markiere v_j mit $(-v_i, \delta_j)$, wobei δ_j die kleinere der beiden Zahlen $f(e_{ij})$ und δ_i ist.

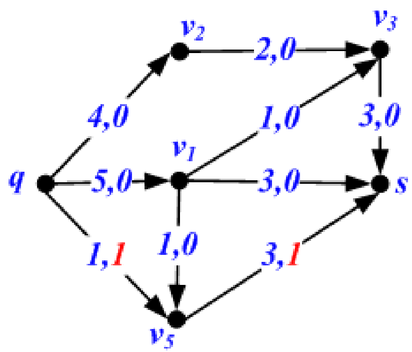
(3) **Vergrößerung der Flussstärke** ... jede Vorwärtskante wird $f(e_{ij})$ um δ_s erhöht,
... jede Rückwärtskante wird $f(e_{ji})$ um δ_s vermindert. Anschließend werden bei allen Knoten mit Ausnahme von q die Markierungen entfernt. Gehe zu 2.

Lösung von Aufgabe 3 (1)



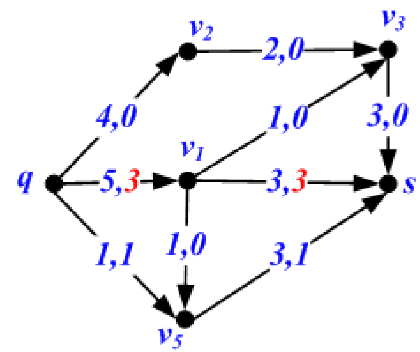
Knoten	q	v_5	s
Kennung	$(\perp, \infty)$	$(+q, 1)$	$(+v_5, 1)$

Lösung von Aufgabe 3 (2)



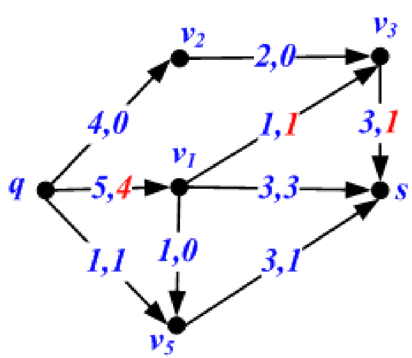
Knoten	q	v_1	s
Kennung	$(\perp, \infty)$	$(+q, 5)$	$(+v_1, 3)$

Lösung von Aufgabe 3 (3)



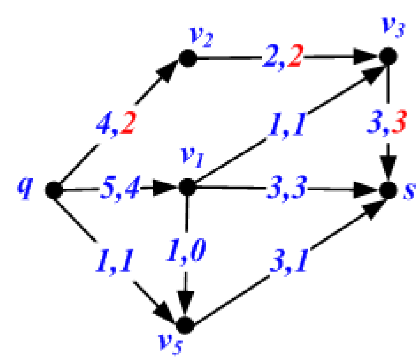
Knoten	q	v_1	v_3	s
Kennung	$(\perp, \infty)$	$(+q, 2)$	$(+v_1, 1)$	$(+v_3, 1)$

Lösung von Aufgabe 3 (4)



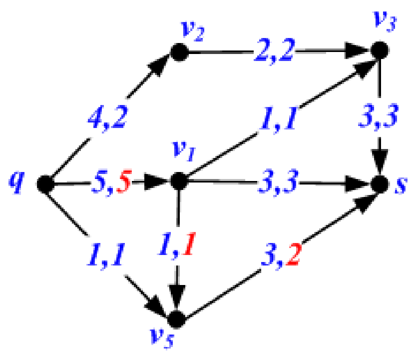
Knoten	q	v_2	v_3	s
Kennung	$(\perp, \infty)$	$(+q, 4)$	$(+v_2, 2)$	$(+v_3, 2)$

Lösung von Aufgabe 3 (5)



Knoten	q	v_1	v_5	s
Kennung	$(\perp, \infty)$	$(+q, 1)$	$(+v_1, 1)$	$(+v_5, 1)$

Lösung von Aufgabe 3 (6)



Knoten	q	v_2
Kennung	$(\perp, \infty)$	$(+q, 2)$

kein weiterer vergrößernder Weg

also maximaler Fluss mit $d = 8$

Graphentheoretische Konzepte und Algorithmen

Paarungen

Julia Padberg

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Paarung

http://de.wikipedia.org/wiki/Paarung_(Graphentheorie)

Definition

Eine **Paarung** (Matching) eines schlichten, ungerichteten Graphen $G = (V, E)$ ist eine Teilmenge der Kanten $M \subseteq E$, wenn je zwei beliebige verschiedene Kanten $e_1, e_2 \in M$ mit verschiedenen Knoten inzident sind[2].

Eine Paarung M ist also eine Kantenmenge M, wobei in M keine zwei Kanten einen gemeinsamen Knoten besitzen.
Algorithmen:

- Ungarische Methode
- Algorithmus von Hopcroft und Karp (für bipartite Graphen)
- Auktionsalgorithmus nach Bertsekas

(besonders für Graphen mit wenigen Kanten)

[2] Man sagt auch e_1 und e_2 sind disjunkt.

Eigenschaften

http://de.wikipedia.org/wiki/Paarung_(Graphentheorie)

Definition

- Eine Paarung M von G ist **maximal**, wenn es keine Kante $e \in E$ gibt, so dass $M \cup e$ eine Paarung ist.
- Gibt es in G keine Paarung, die mehr Elemente als M enthält, so ist M **größte Paarung**.
- Ist jeder Knoten von V zu einer Kante von M inzident, dann ist M eine **perfekte Paarung**.
- Die Anzahl der Elemente einer größten Paarung nennt man **Paarungszahl**.
- Bei kantenbewerteten Graphen definiert man die Größe einer Paarung über die Summe ihrer Kantenbewertung. Die **größte gewichtete Paarung** ist dann eine Paarung, die diesen Wert maximiert.

Beispiel

http://de.wikipedia.org/wiki/Paarung_(Graphentheorie)

Beim Arbeitsamt

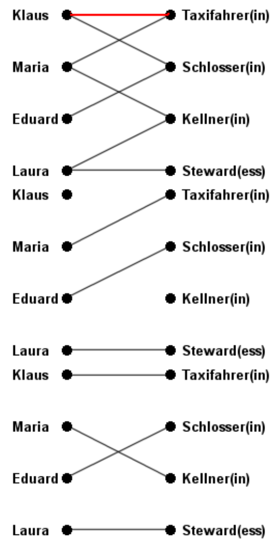
- gibt's vier Stellenangebote und -gesuche;
- aber einige können mehrere Berufe.

Lösung (maximal, aber nicht perfekt)

- unmöglich, allen einen Job zu vermitteln;
- denn für zwei Schlosser gibt's eine Stelle.

Lösung (perfekt, durch neue Berufswahl)

- d.h. für Klaus:
  arbeitslos oder Taxifahrer werden;
- Klaus wird also Taxifahrer.

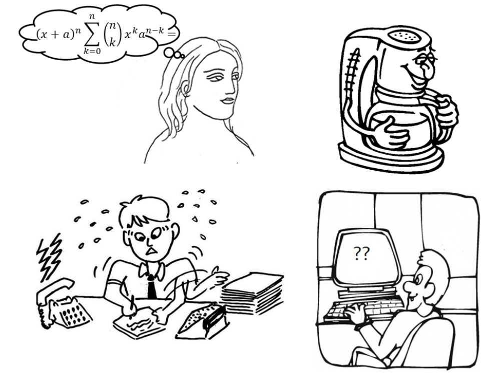


Aufgabe 4:

Es sollen 2er-Arbeitsgruppen gebildet werden, so dass möglichst viele Eigenschaften in möglichst vielen Gruppen vertreten sind.

Die Studierenden haben ein oder zwei der folgenden Eigenschaften:

- T ist in Theorie sehr gut
- P kann toll programmieren
- K holt bereitwillig Kaffee
- O ist ein Organisationstalent



Fortsetzung

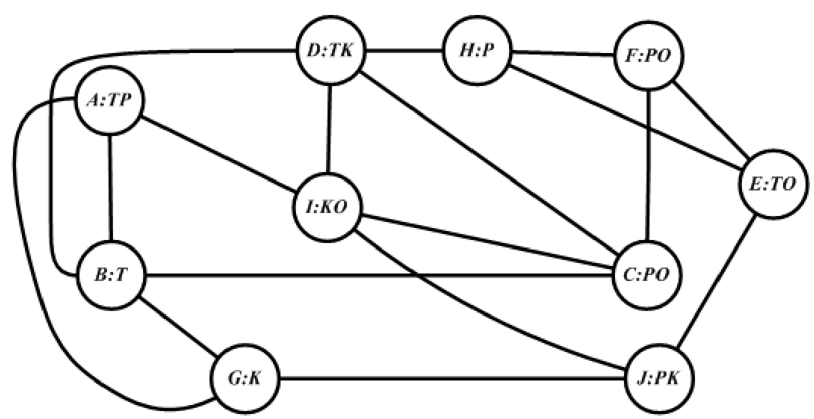
Es sollen 2er-Arbeitsgruppen gebildet werden, so dass möglichst viele Eigenschaften in möglichst vielen Gruppen vertreten sind.

- A ist in Theorie sehr gut, kann toll programmieren und kennt B,I und G
- B ist in Theorie sehr gut und kennt A,D,G und C
- C ist ein Organisationstalent, kann toll programmieren und kennt B, D,I und F
- D ist in Theorie sehr gut, holt bereitwillig Kaffee und kennt B,C, I und H
- E ist in Theorie sehr gut, ist ein Organisationstalent und kennt F, H und J
- F ist ein Organisationstalent, kann toll programmieren und kennt C, E und H
- G holt bereitwillig Kaffee und kennt A,B und J
- H kann toll programmieren und kennt D,E und F
- I ist ein Organisationstalent, holt bereitwillig Kaffee und kennt A,C,D und J
- J kann toll programmieren, holt bereitwillig Kaffee und kennt E, G und I

Lösung von Aufgabe 4

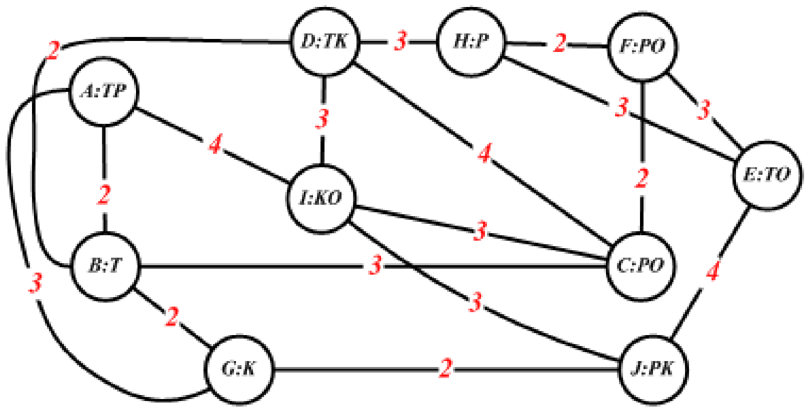
Es sollen 2er-Arbeitsgruppen gebildet werden, so dass möglichst viele Eigenschaften in möglichst vielen Gruppen vertreten sind.

Der Graph modelliert die Fähigkeiten der Studierenden (Knotenattribute) und ihre Bekanntschaft (Kanten) untereinander.



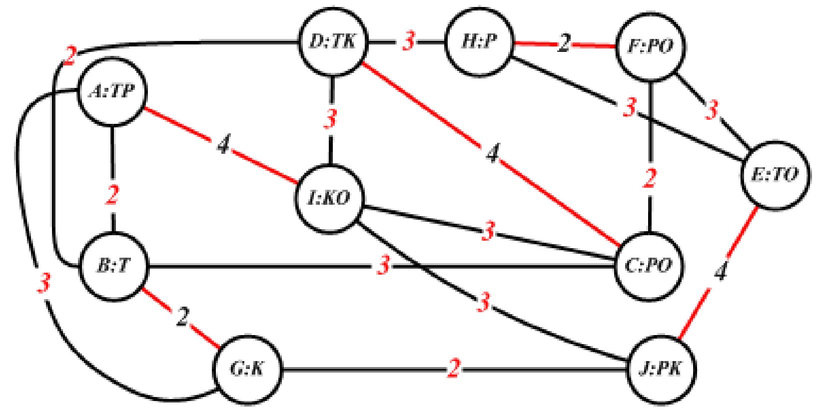
Lösung von Aufgabe 4

Erst werden die Kanten mit der Anzahl der Fähigkeiten der potentiellen AG bewertet.



Lösung von Aufgabe 4

und dann eine Paarung mit maximaler Kantengewichtssumme gesucht.



Heiratssatz

http://de.wikipedia.org/wiki/Paarung_(Graphentheorie)

Definition (Nachbarschaft)

Sei $W \subseteq V$ eines Graphen $G = (V, E)$.
Die Nachbarschaft von W ist dann $N_G(W) = \{v | (v, w) \text{ oder } (w, v) \in E\}$.

Satz (von Hall)

In bipartiten Graphen G mit Bipartition $\{X, Y\}$ existiert eine Paarung, die jeden Knoten aus X überdeckt, genau dann, wenn für jede Teilmenge $S \subseteq X$ gilt, dass ihre Nachbarschaft mindestens so groß ist wie S selbst:

$$|S| = |N_G(S)|$$

Größte Paarung und maximaler Fluss

http://de.wikipedia.org/wiki/Paarung_(Graphentheorie)

- Berechnung einer größten Paarung bipartiter Graphen mit Hilfe der Berechnung eines maximalen Flusses
- Erweiterung zu einem Netzwerk
- Kanten mit Flusswert ungleich Null sind eine größte Paarung des bipartiten Graphen

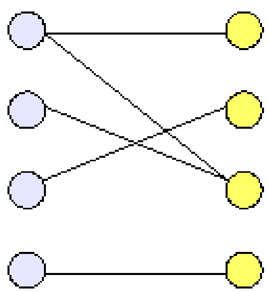
Größte Paarung und maximaler Fluss

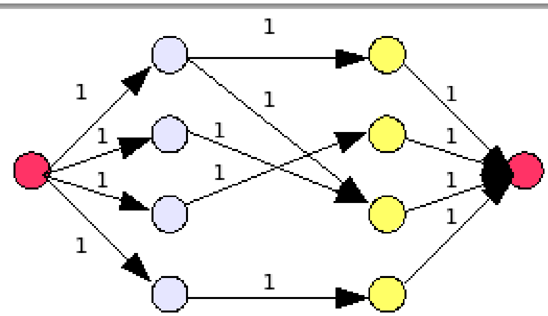
http://de.wikipedia.org/wiki/Paarung_(Graphentheorie)

Algorithmus

Erweiterung des bipartiten Graph $G = (X \uplus Y, E)$:

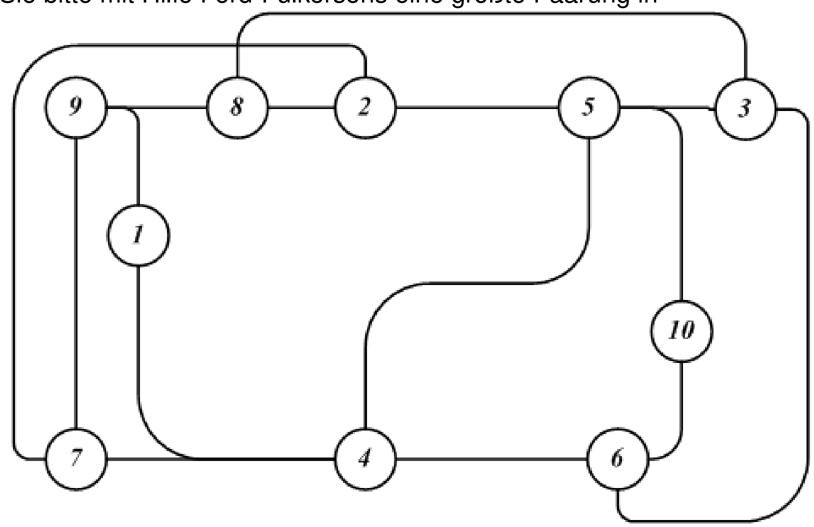
1. Neue Quelle q mit Kanten (q, x) für alle $x \in X$ und
2. neue Senke s mit Kanten (y, s) für alle $y \in Y$.
3. Die Kanten werden in Richtung der Senke ausgerichtet.
4. Die Kapazität jeder Kante des Netzwerks ist 1.



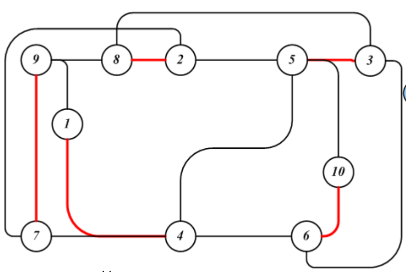


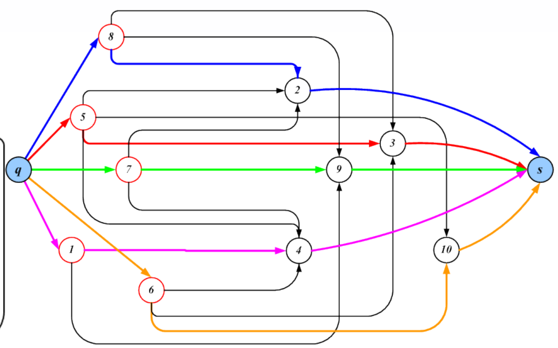
Aufgabe 5:

Finden Sie bitte mit Hilfe Ford-Fulkersons eine größte Paarung in



Lösung von Aufgabe 5





Knoten	q	1	5	6	7	8	3	s
Kenn	$(\perp, \infty)$	(+q,1)	(+q,1)	(+q,1)	(+q,1)	(+q,1)	(+5,1)	(+3,1)
Knoten	q	1	6	7	8	9	s	
Kenn	$(\perp, \infty)$	(+q,1)	(+q,1)	(+q,1)	(+q,1)	(+7,1)	(+9,1)	
Knoten	q	1	6	8	2	s		
Kenn	$(\perp, \infty)$	(+q,1)	(+q,1)	(+q,1)	(+8,1)	(+2,1)		
Knoten	q	1	6	4	s			
Kenn	$(\perp, \infty)$	(+q,1)	(+q,1)	(+6,1)	(+4,1)			
Knoten	q	1	4	6	10	s		
Kenn	$(\perp, \infty)$	(+q,1)	(+1,1)	(-4,1)	(+6,1)	(+10,1)		