Datenstrukturen & Algorithmen 13. Mai 2015

Handout 11

Sebastian Millius, Sandro Feuz, Daniel Graf

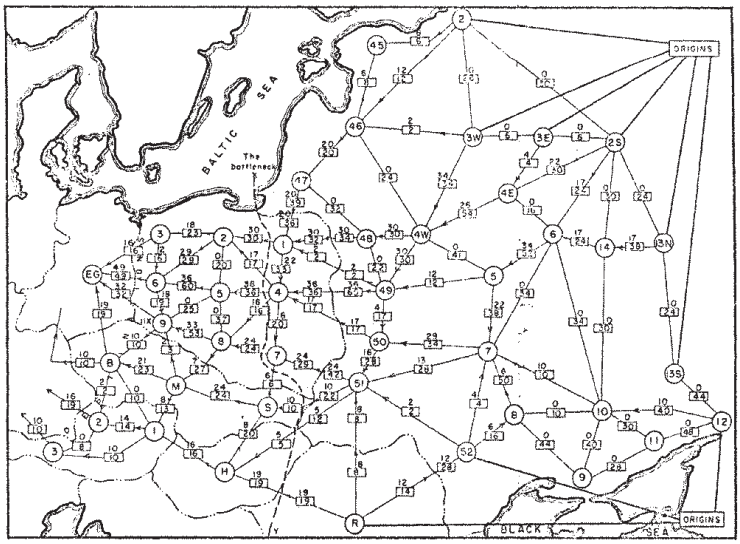
Thema: Network Flow, Min Cut, Matching in bipartiten Graphen

Links

- Eine ausführliche Darstellung mit vielen Übungen findet sich in Kapitel 7 in *Algorithm Design* von Kleinberg und Tardos. Das Buch ist in der Informatikbibliothek erhältlich.
- Lecture Notes
  http://www.cs.washington.edu/education/courses/cse521/10wi/Flow.pdf
  (http://goo.gl/5rfip)
- Ausführliche Darstellung
  http://compgeom.cs.uiuc.edu/~jeffe/teaching/algorithms/notes/21-maxflow.pdf
  (http://goo.gl/uB2N9)
- Anwendungen/Beispiele
  http://compgeom.cs.uiuc.edu/~jeffe/teaching/algorithms/notes/23-maxflowapps.pdf
  (http://goo.gl/lLMAO)
- Für Interessierte: Ein paar Erweiterungen: Edge Lower Bounds, Node Supplies und Demands
  http://compgeom.cs.uiuc.edu/~jeffe/teaching/algorithms/notes/24-maxflowext.pdf
  (http://goo.gl/UpBy9)

Netzwerk Flüsse

Mitte der 1950er Jahre veröffentlichten die Air Force Forscher T.E. Harris und F.S. Ross einen Bericht, der das Schienennetz zwischen der Sowjetunion und ihren Satellitenstaaten in Osteuropa analysierte. Das Netzwerk wurde als ein Graph modelliert mit 44 Knoten, die die geografischen Regionen, und 105 Kanten, die die Eisenbahnverbindungen zwischen diesen Regionen repräsentierten. Jeder Kante wurde ein Gewicht gegeben, dass die Rate repräsentiert mit der Material von der einen Region zur anderen transportiert werden kann.



Harris and Ross [1955]: Schematisches Diagram des Eisenbahnnetzes der Sovietunion, mit einem maximalen Fluss von 163,000 Tonnen von Russland nach Osteuropa und einem minimalen Cut mit einer Käpazität von 163,000 Tonnen (in der Graphik notiert als "The bottleneck")

Bestimmt wurden in diesem Bericht die maximale Gütermenge die aus Russland nach Europa transportiert werden kann, sowie die "günstigste" Möglichkeit das Netzwerk durch Entfernen von Kanten (oder weniger abstrakt: durch Sprengen von Gleisen) zu stören/lahmlegen.

Dies ist eine der ersten dokumentierten Anwendung von Netzwerk Fluss und Minimaler Schnitt Problemen.

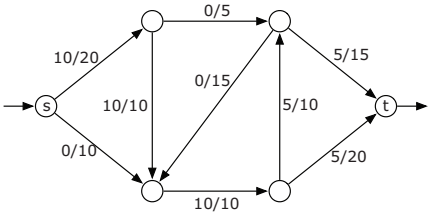
Netzwerk Fluss Probleme sind in der Graphen-Theorie weit verbreitet und haben sehr vielen Anwendungen. Flüsse sind nur auf gerichteteten, gewichteten Graphen $G = (V, E, c)$ mit speziellen Knoten $s, t \in V$[1] definiert. Eine Kante von G kann als ein (gerichtetes) Wasserrohr interpretiert werden. Dabei ist das Gewicht der Kante gerade die Kapazität des Rohrs ("wieviel Wasser kann durchfliessen"). Die zentrale Frage ist nun: wieviel Wasser kann von s ausgehend zu t fliessen, resp. was ist der Wert eines maximalen Flusses. Ein Fluss beschreibt dabei die Belegung der Kanten, also wieviel Wasser in einem bestimmten Rohr fliessen wird. Er ist also eine Funktion von den Kanten auf die reellen Zahlen $f : E \to \mathbb{R}$.

Dabei muss gelten:

- Nichtnegativität: Für alle Kanten e gilt: $f(e) \geqslant 0$
- Flow Conservation: "Was in einen Knoten reingeht, geht auch wieder raus":

$$\forall v \neq s, t : \sum_{u \in V, (u,v) \in E} f(u, v) = \sum_{w \in V, (v,w) \in E} f(v, w)$$

Der Wert des Flusses beschreibt wieviel Wasser von s rausgeht (analog: wieviel bei t reinkommt). Formal ist der Wert des Flusses f: $val(f) = \sum_{u \in V, (s,u) \in E} f_{(s,u)} - \sum_{u \in V, (u,s) \in E} f_{(u,s)}$. Ein Fluss ist maximal, falls er maximalen Wert hat (es gibt also mehrere maximale Flüsse, aber alle haben den gleichen Wert).



Ein (s, t)-Fluss mit Wert 10. Jede Kante ist gekennzeichnet mit Fluss/Kapazität.

Wir kennen drei verschiedene Algorithmen (Ford-Fulkerson, Edmonds-Karp und Dinits) um einen maximalen Fluss zu bestimmen. Alle funktionieren nach dem gleichen Muster:

MaximumFlow

Input: Network $(G = (V, E, c), s, t)$
Output: Maximum Flow f

```
1  f = ∅
2  while
3      Create Residual Network R
4      Find Augmenting Path p in R
5      if p = ∅
6          return f
7      else f = f + p
```

Die Grundidee ist, dass wir solange es geht einen aktuellen Fluss f erhöhen, indem wir einen Weg von s nach t im Residual-Graph (=Restgraph) finden. Falls wir f nicht mehr erhöhen können (resp. es keinen $s - t-$Pfad im Residual-Graphen mehr gibt), sind wir fertig und wissen, dass f maximal ist.

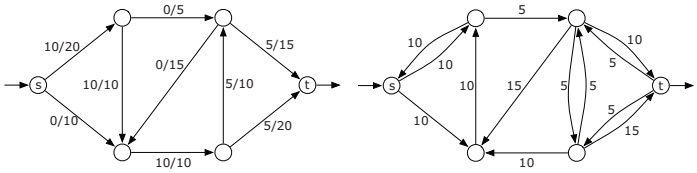
[1] Dabei nennen wir s Quelle (engl. source) und t Senke (engl. sink).

Der Residual-Graph R zu einem Netzwerk G mit Fluss f ist dabei wie folgt definiert:

$$
\begin{aligned}
R &= (V, E', r) \\
E' &= \{(u,v) | (u,v) \in E, f_{(u,v)} < c_{(u,v)}\} \cup \{(v,u) | (u,v) \in E, f_{(u,v)} > 0\} \\
r_{(u,v)} &= \begin{cases} c_{(u,v)} - f_{(u,v)}, & \text{falls } (u,v) \in E \\ f_{(u,v)}, & \text{falls } (v,u) \in E \text{ (sonst)} \end{cases}
\end{aligned}
$$

In Worten hat der Residualgraph zwei Kanten für jede Kante des Netzwerks. Einmal eine "Vortwärtskante" welche als Gewicht die noch verbleidende Restkapazität enthält (bsp: eine Kante e hat Kapazität $c_e = 10$ und der Fluss f schickt 8 Einheiten durch (also $f_e = 8$), dann haben wir im Residualgraphen ebenfalls die Kante $e \in E$ mit Gewicht $r_e = 2$). Desweiteren eine "Rückwärtskante", welche den Fluss selbst als Gewicht hat. Diese sagt in gewissem Sinne aus, wieviel vom Fluss wieder zurückfliessen könnte (bsp: eine Kante $e = (u,v) \in E$ hat Kapazität $c_e = 10$ und der Fluss f schickt 8 Einheiten durch (also $f_e = 8$), dann haben wir im Residualgraphen eine Kante $e^{opp} = (v,u) \in E'$ mit Gewicht $r_{e^{opp}} = 8$).

Dabei sind der Fluss f, die Kapazität c von G und das Gewicht r von R einfach Funktionen von den Kanten auf die reellen Zahlen. Falls eine Kante im Residualgraph Gewicht 0 hätte wird sie weggelassen. Bemerke, dass wir im ursprünglichen Netzwerk G keine Zyklen der Länge zwei haben dürfen (formal: $(u,v) \in E \Rightarrow (v,u) \notin E$), da sonst der Residualgraph nicht mehr sauber definiert wäre (das ist aber keine grosse Restriktion; was könnten wir tun wenn wir so einen Fall bearbeiten möchten? Think about it! (Nein, wir können aus den beiden Kanten nicht einfach eine machen)).

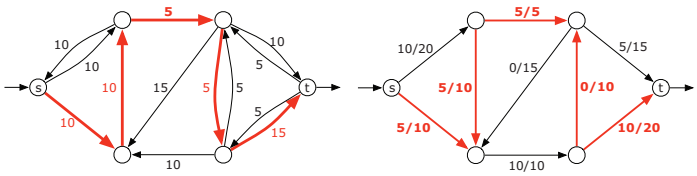


Graph und entsprechender Residualgraph

Die verschiedenen Algorithmen (Ford-Fulkerson, Edmonds-Karp, Dinits) unterscheiden sich im wesentlichen nur dadurch, wie nun in dem Residualgraph ein Pfad von s nach t gesucht wird (resp. welcher Pfad gesucht wird). Ford-Fulkerson nimmt einfach einen beliebigen (beispielsweise mittels DFS auf R). Edmonds-Karp und Dinits nehmen jeweils einen mit möglichst wenig Kanten (Edmonds-Karp findet diesen mittels BFS, Dinits mittels vorberechnen eines Shortest-Path-Graphen). Für alle drei gilt, falls sie keinen Pfad finden ist der aktuelle Fluss maximal. Ansonsten haben wir einen Pfad $p = v_1 v_2 \dots v_i \dots v_k$ von s nach t im Graphen R (also $v_1 = s$ und $v_k = t$). Sei m der kleinste Wert auf dem Pfad (also $m = \min_{i=1\dots k-1} r_{(v_i, v_{i+1})}$), dann können wir den Fluss f wie folgt um m vergrössern:

$$
\begin{aligned}
f_{(v_i, v_{i+1})} &= f_{(v_i, v_{i+1})} + m, \text{falls } (v_i, v_{i+1}) \in E \\
f_{(v_{i+1}, v_i)} &= f_{(v_{i+1}, v_i)} - m, \text{falls } (v_{i+1}, v_i) \in E
\end{aligned}
$$

Der Fluss f wird also auf allen Vorwärtskanten auf dem Pfad p um den Wert m erhöht und auf allen Rückwärtskanten um m verringert. Bemerke, dass sich der Wert des Flusses dabei immer erhöht, obwohl er auf einzelnen Kanten verringert wird. Diese Vergrösserung des Flusses entspricht auch gerade unserem intuitiven Verständnis der Rückwärtskanten: statt explizit Wasser durch eine Rückwärtskante fliessen zu lassen, lassen wir einfach weniger in der Vorwärtsrichtung fliessen.



Pfad mit Wert 5, Graph nach Erhöhung des Flusses

Nach dem Erhöhen des Flusses müssen wir wiederum den Residualgraphen berechnen. Bemerke, dass wir dazu nicht "from scratch" beginnen müssen, sondern lediglich den aktuellen Residualgraphen entlang des Pfades p erneuren müssen. Wenn wir den Graphen als Adjazenzmatrix darstellen ist dies extrem einfach, da wir immer die Assoziation zwischen Vorwärts- und Rückwärtskante haben. Bei Adjazenzlisten ist es etwas schwieriger (think about it). Die Laufzeiten der einzelnen Algorithmen sind:

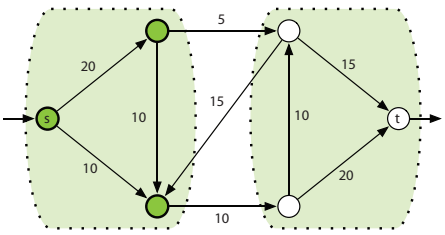
- **Ford Fulkerson** Bei ganzzahligen Kantenkapazitäten wird in jedem Augmentierungsschritt der Wert des Flusses f um mindestens 1 erhöht. Sei U die maximale Kantenkapazität, dann ist der der Wert eines maximal Flusses höchstens $|V| \cdot U$ (think about it). Demnach werden höchstens $|V| \cdot U$ Schritte ausgeführt. Jeder Schritt beinhaltet die Suche nach einem Pfad im Residualgraphen und braucht typischerweise (unter Verwendung von DFS) $O(|V| + |E|)$, was zu einer Gesamtlaufzeit von $O(|V| \cdot |E| \cdot U)$ führt.
- **Edmonds Karp** Es kann gezeigt werden, dass höchstens $|E| \cdot |V|$ Schritte ausgeführt werden müssen, wenn wir immer einen Pfad in R mit minimaler Kantenanzahl wählen (BFS statt DFS). Dies ist nicht trivial, aber auch nicht all zu schwer zu sehen (think about it). In jedem Schritt suchen wir einen Pfad in R mittels BFS, was jeweils $O(|E| + |V|)$ Zeit braucht. Total $O(|E|^2 \cdot |V|)$.
- **Dinits** Der BFS Schritt wird durch Vorberechnen eins Shortest-Path-Graphen auf $O(|V|)$ (statt $O(|E| + |V|)$) reduziert, was die totale Laufzeit auf $O(|E| \cdot |V|^2)$ verringert.

Maximale Flüsse und minimale Schnitte

Ein Schnitt in einem Netzwerk ist eine Partitionierung der Knoten V in zwei Teile S und $V \setminus S$, so dass s und t nicht in der gleichen Partition liegen (also $s \in S$ und $t \in V \setminus S$). Der Wert des Schnittes ist die Summe der Kapazitäten der Kanten, die von S nach $V \setminus S$ gehen, also $val(S) = \sum_{u \in S, v \in V \setminus S, (u,v) \in E} c_{(u,v)}$. Intuitiv gesprochen wollen wir die Quelle von der Senke trennen, indem wir einige Kanten durchschneiden. Der Wert des Schnittes sind die Kapazitäten der Kanten, die in "Vorwärtsrichtung" durchschnitten werden. Ein minimaler Schnitt ist ein Schnitt mit minimalem Wert. Die wichtigste Aussage über Schnitte ist: der Wert eines minimalen Schnittes ist gleich dem Wert eines maximalen Flusses. Formal:

$$\min_{S \subseteq V} val(S) = \max_{\text{Fluss } f} val(f)$$

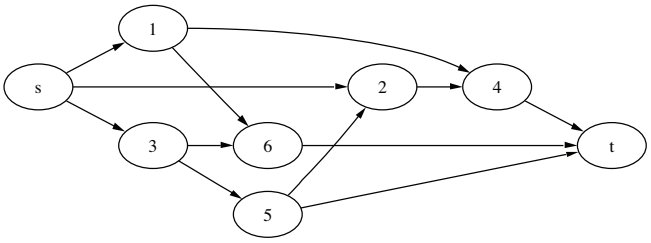
Insbesondere finden wir einen minimalen Schnitt S von G einfach durch das Suchen eines maximalen Flusses f von G und anschliessend markieren aller von s erreichbaren Knoten auf dem Residualgraph. Eine Teilmenge $S \subseteq V$ wird von s noch erreichbar sein, jedoch t sicherlich nicht mehr (sonst gäbe es noch einen Pfad von s nach t in R und f wäre noch nicht maximal). Demnach ist $t \notin S$ und S wirklich ein Schnitt. Zu zeigen, dass solch ein Schnitt minimal ist, ist nicht trivial (Überlegungsaufgabe!).



Ein minimaler Schnitt mit Wert 15. Die Kanten sind mit ihrer Kapazität gekennzeichnet.

Übung

Betrachte das folgende Netzwerk, bei dem alle Kanten Kapazität 1 haben.



Der Edmonds-Karp Algorithmus wählt jeweils den kürzesten (Anzahl Kanten) Pfad. Führe diesen Algorithmus auf dem obigen Netzwerk aus. Insbesondere zeige das Residualnetzwerk nach jedem Erhöhungsschritt. Warum ist der resultierende Fluss maximal?

Reduktionen auf maximale Flüsse/minimale Schnitte

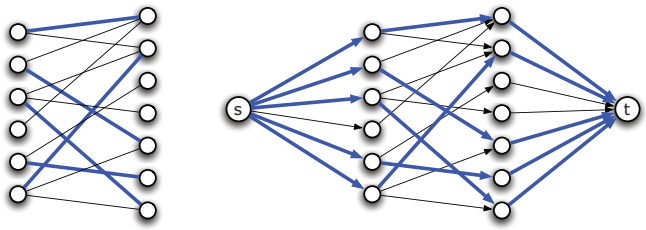
Matching in einem bipartiten Graphen

Eine andere Anwendung von maximalen Flüssen ist das Finden eines maximalen Matchings in einem bipartiten Graphen. Ein *Matching* ist eine Auswahl von Kanten, so dass keine zwei Kanten einen Endpunkt gemeinsam haben. Wir suchen nun ein Matching in einem bipartiten Graphen mit der maximalen Anzahl von Kanten.

Wir können das Problem lösen in dem wir es auf das Maximum Flow Problem reduzieren. Es sei G der gegebene bipartite Graph mit Knotenmenge $L \cup R$, so dass jede Kante einen Knoten in L mit einem Knoten in R verbindet.

Wir erstellen einen neuen gerichteten Graphen G' in der folgenden Art

- Orientiere jede Kante von L nach R
- Füge zwei neue Knoten s und t hinzu
- Verbinde s mit jedem Knoten in L
- Verbinde jeden Knoten in R mit t
- Jede Kante hat Kapazität 1



Reduktion des Maximum Matching Problems auf das Bestimmen eines maximalen Flusses

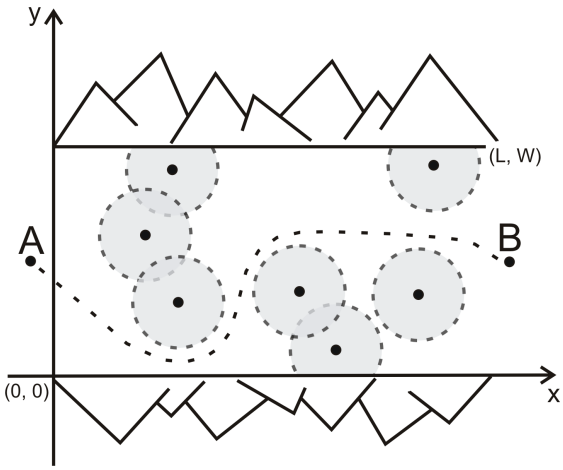
Überlege dir, dass ein maximaler Fluss in G' einem maximalen Matching in G entspricht!

Flussprobleme können nicht nur auftreten wenn wir ein Rohrsystem mit Wasser haben, sondern viele Probleme können darauf reduziert werden. Das schwierige ist dabei oft die Reduktion zu finden. Im folgenden ein paar Beispielaufgaben, die auf ein maximalen Fluss resp. minimaler Schnitt Problem reduziert werden können[2]

[2]Lösungen und Tipps auf Anfrage, ein kleiner Hinweis jedoch vorneweg: Bei einigen der folgenden Probleme könnte es nötig sein, dass wir Kapazitäten auf den Knoten statt auf den Kanten haben wollen. Wie könnte das gehen?

Gefängnisausbruch

Eine Gruppe Gefangener plannt ihren Gefängnisausbruch. Sie wissen schon wie sie die Gefängnismauern überwinden können und müssen es nur noch von dort bis ins nächste Dorf schaffen. Die Gefängnismauern (A im unteren Bild) und das Dorf (B im unteren Bild) sind durch eine Schlucht voll von Wachleuten getrennt. Die Wachleute sind sehr faul und stehen einfach nur rum. Jeder von ihnen hat einen Sichtradius von 100 Metern, das heisst es könnte möglich sein die Schlucht zu durchqueren ohne von einem Wachmann gesehen zu werden (indem man nie näher als 100m an einem vorbei geht).
Gegeben die Breite und Länge der Schlucht sowie die Positionen der Wachmänner, bestimme ob es möglich ist die Schlucht unentdeckt zu durchqueren. Falls nicht, bestimme die minimale Anzahl an Wachmänner, die eliminiert werden müssen, um die Schlucht ungesehen durchqueren zu können. Ein Wachmann kann eliminiert werden auch wenn er von einem anderen Wachmann gesehen wird.

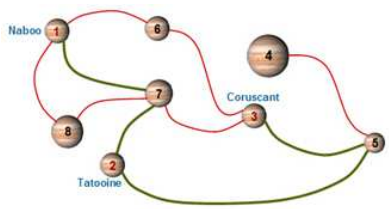


Glückliche Professoren

An der ETH gibt es jeweils einen harten Kampf zwischen den Professoren um die beliebtesten Vorlesungssäle für die einzelnen Vorlesungen. Jeder Professor muss eine oder mehrere Vorlesungen halten und hat einen oder mehrere Lieblingssäle. Er ist nur glücklich, wenn er alle seine Vorlesungen in seinen Lieblingssälen halten kann. Du kannst davon ausgehen, dass für ein ganzes Semester ein Saal nur einer Vorlesung zugeordnet sein kann (und umgekehrt). Gegeben die Menge der Professoren P und die Menge der Säle S sowie die Vorlesungen und Lieblingssäle der Professoren, bestimme ob es möglich ist, alle Professoren glücklich zu machen.

Intergalaktische Reise

Der Jedi Master Qui-Gon Jinn und sein Lehrling Obi-Wan Kenobi wurden von Senatorin Padmé Amidala gebeten Naboo vor einer Invasion durch die Handelsföderation zu bewahren. Sie müssen Naboo sofort zu verlassen und nach Tatooine reisen um Beweise der Invasionspläne der Föderation zu erhalten. Danach müssen sie den Republik Hauptstadt Planeten Coruscant aufsuchen, um die Pläne dem Senat vorzulegen.



Um ihnen bei diesem Unterfangen zu helfen, stellt Senatorin Amidala eine intergalaktische Karte zur Verfügung. Diese Karte zeigt die Verbindungen zwischen den Planeten, die noch nicht von der Handelsföderation blockiert sind. Jedes Paar von Planeten hat höchstens eine Verbindung zwischen ihnen, und alle Verbindungen sind in beide Richtungen befahrbar (also ungerichtet).

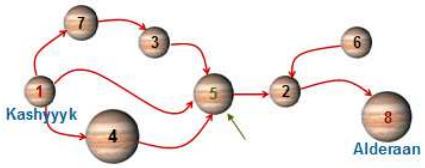
Um zu vermeiden, von feindlichen Spionen entdeckt zu werden, dürfen die Jedi Ritter auf diesem Abenteuer *keinen Planeten mehr als einmal besuchen*. Können Sie ihnen helfen zu bestimmen, ob ein solcher Pfad existiert? Im obigen Beispiel ist dieser Weg grün/fett eingezeichnet.

Hint: Das Problem lässt sich also reduzieren auf: Gegeben ein ungerichteter,ungewichteten Graph $G = (V, E)$ und drei Knoten $u, v, w \in V$. Bestimme ob es einen Pfad von u via v nach w gibt, der keinen Knoten doppelt besucht.

Eine Lösung kann programmiert und auf https://www.spoj.pl/problems/IM/ hochgeladen werden.

In der Falle

Der Separatisten Führer General Grievous, der zweite im Kommando des Count Dooku, hat erfahren, dass Kanzler Palpatines Konvoi, der von Obi-Wan und Anakin eskortiert wird, von Kashyyyk im Middle Rim nach Alderaan fliegt. General Grievous ist sich bewusst, dass es mehrere Wege von Kashyyyk nach Alderaan gibt, die jeweils unterschiedliche Planeten besuchen.



Um seinen Entführungsversuch zu einem Erfolg zu verhelfen, beschliesst er, seine Roboter auf dem Planeten am nächsten zu Kashyyyk (aber nicht auf Kashyyyk selbst) zu senden, der auf *allen möglichen Wegen von Kashyyyk nach Alderaan* liegt. Da Sie Ihre Treue Count Dooku geschworen haben, ist es Ihre Aufgabe diesen Planeten zu identifizieren.

Im obigen Beispiel ist dies der Planet mit der Nummer 5.

Die Planeten-Karte, die Ihnen zu diesem Zweck gegeben ist besteht aus einer Reihe von gerichteten Verbindungen zwischen den Planeten. Sie wissen auch, dass ein Paar von Planeten höchstens eine Verbindung zwischen ihnen in jeder Richtung haben kann und es immer einen Weg von Kashyyyk nach Alderaan gibt.

Eine Lösung kann programmiert und auf https://www.spoj.pl/problems/EN/ hochgeladen werden.

Anzahl Wege

Gegeben ist ein gerichteter Graph. Wie viele Knoten-disjunkte Pfade gibt es von einem Knoten s zu einem anderen Knoten t? (d.h. wie viele Pfade gibt es von s nach t, so dass jedes Paar von Pfaden ausser s und t keinen Knoten gemeinsam haben?)