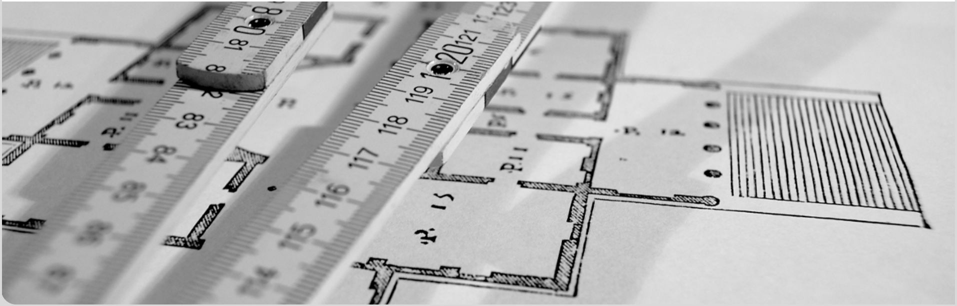
Karlsruher Institut für Technologie

ICPC

Graphen 3

Tobias, Julian, Jakob, Tobias | 7. Juni 2018

ITI WAGNER, IPD TICHY



Outline/Gliederung

Karlsruher Institut für Technologie

1. Einfürung
   - Idee
   - Definition
2. Max-Flow Algorithmen
   - Ford-Fulkerson
   - Edmonds-Karp
3. Min-Cut
4. Sonderfälle
5. Max-Flow Modellierung
6. Bipartite Matching

Idee: Flussnetzwerk

- Transport von Material von einer Quelle zu einer Quelle

Idee: Flussnetzwerk

- Transport von Material von einer Quelle zu einer Quelle
- Materialfluss durch Kanäle

Idee: Flussnetzwerk

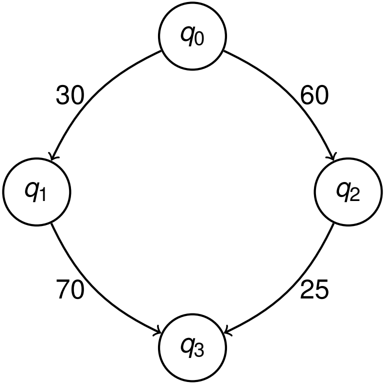
- Transport von Material von einer Quelle zu einer Quelle
- Materialfluss durch Kanäle
- Mehrere Kanäle mit verschiedenen Kapazitäten

Idee: Flussnetzwerk

Karlsruher Institut für Technologie

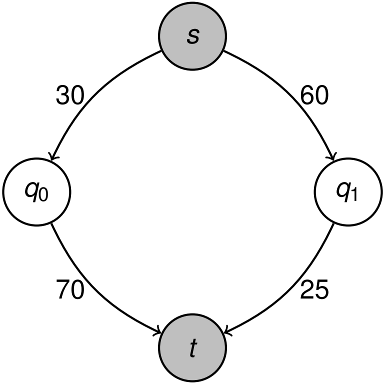
- Transport von Material von einer Quelle zu einer Quelle
- Materialfluss durch Kanäle
- Mehrere Kanäle mit verschiedenen Kapazitäten
- Kanten können sich verzweigen und zusammenfügen

Definition



Gegeben gerichteter Graph

Definition

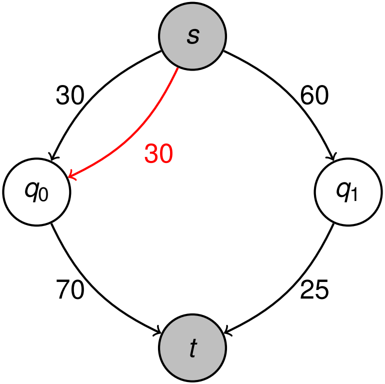


Gegeben gerichteter Graph

s: source

t: sink

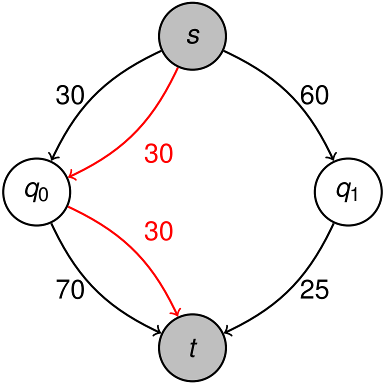
Definition



Fluss

Kapazitätskonfirmität

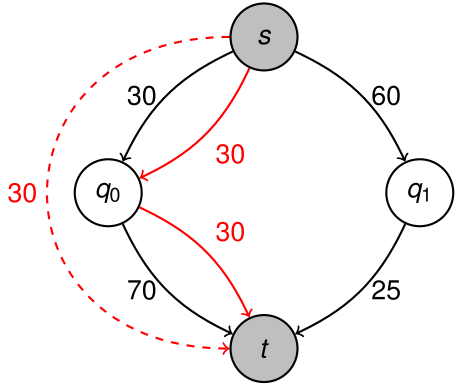
Definition



Fluss

Flusserhaltung

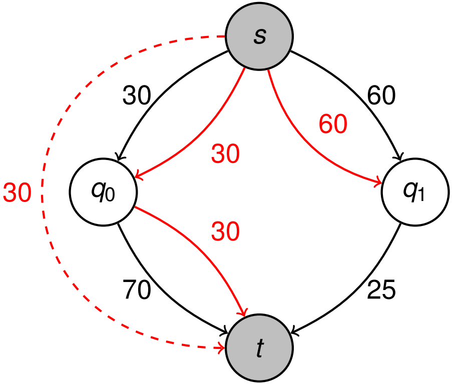
Definition



Fluss

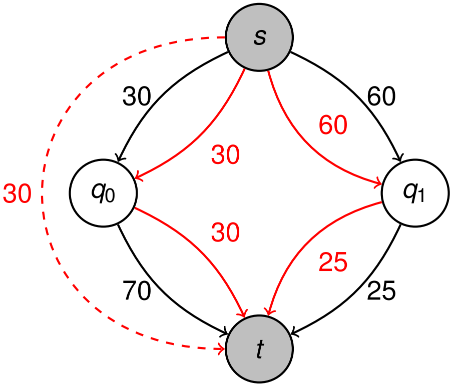
Wert eines s-t-Flusses

Definition



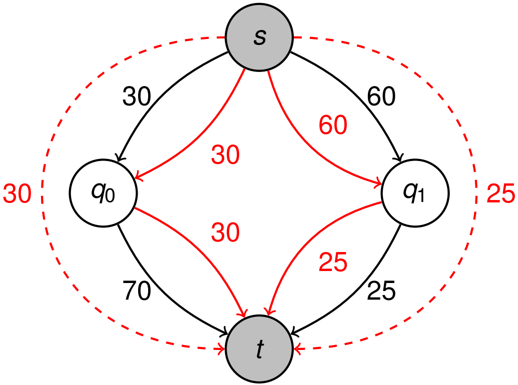
Fluss

Definition



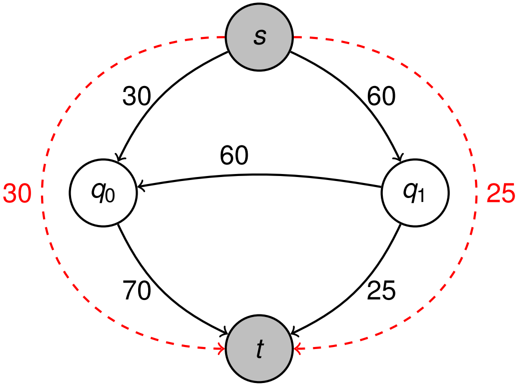
Fluss

Definition



Fluss

Definition



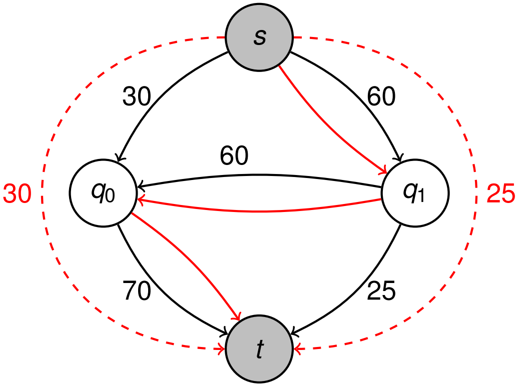
Fluss

Kapazitätskonfirmität

Flusserhaltung

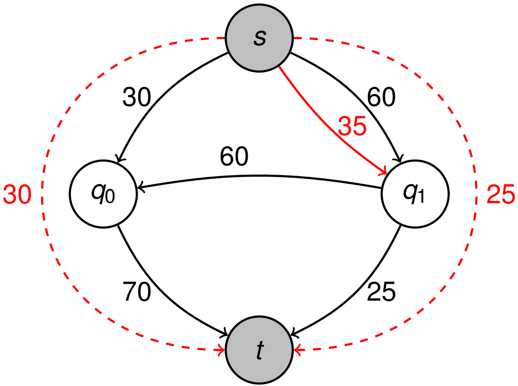
Wert eines s-t-Flusses

Definition



Fluss

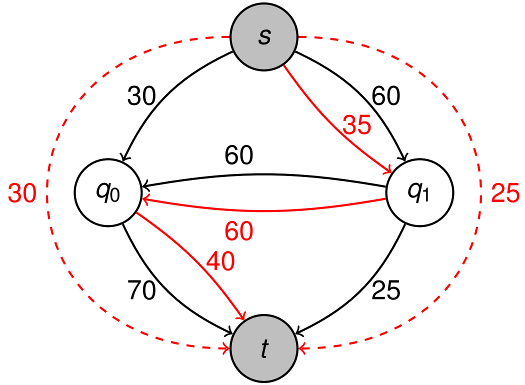
Definition



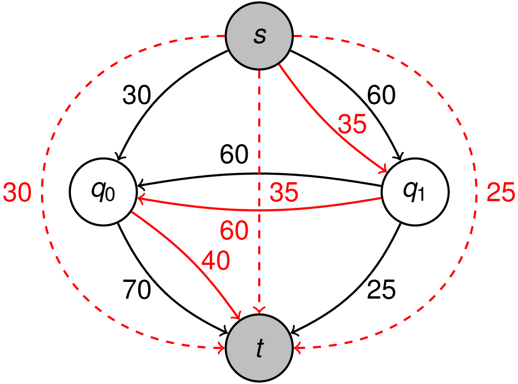
Fluss

Exzess: Werte entsprechend der Kantenkapazität abzüglich bereits vorhandener Flüsse

Definition



Definition



Bestimmung des maximalen Flusses

Karlsruher Institut für Technologie

Idee:

- Starte mit dem leeren Fluss
- Bestimme erweiternden Pfad (augmenting path) P
  - $\Rightarrow$ Ein erweiternder Pfad ist ein einfacher Pfad, der nur Kanten mit positiver Kapazität enthält
- Erweitere die Lösung um P
- Wiederhole so oft, wie es einen passenden Pfad P gibt

Frage: Wie kann P gefunden werden?

Bestimmung des maximalen Flusses

Idee:

- Starte mit dem leeren Fluss
- Bestimme erweiternden Pfad (augmenting path) P
  - $\Rightarrow$ Ein erweiternder Pfad ist ein einfacher Pfad, der nur Kanten mit positiver Kapazität enthält
- Erweitere die Lösung um P
- Wiederhole so oft, wie es einen passenden Pfad P gibt

Frage: Wie kann P gefunden werden?

Ford-Fulkerson Algorithmus

- Greedy-Algorithmus, veröffentlicht 1956 von L. R. Ford, Jr. und D. R. Fulkerson
- Verwendet Tiefensuche, um den erweiternden Pfad P zu bestimmen
- Die Lösung wird um P erweitert, indem
  - die geringste Kapazität f der Kanten in P bestimmt wird
  - die Kapazitäten aller Kanten in P um f verringert werden
  - die Kapazitäten aller Gegenkanten um f erhöht werden
  - der maximale Fluss um f erhöht wird

Ford-Fulkerson Algorithmus

- Greedy-Algorithmus, veröffentlicht 1956 von L. R. Ford, Jr. und D. R. Fulkerson
- Verwendet Tiefensuche, um den erweiternden Pfad P zu bestimmen
- Die Lösung wird um P erweitert, indem
  - die geringste Kapazität f der Kanten in P bestimmt wird
  - die Kapazitäten aller Kanten in P um f verringert werden
  - die Kapazitäten aller Gegenkanten um f erhöht werden
  - der maximale Fluss um f erhöht wird

Ford-Fulkerson Algorithmus

- Greedy-Algorithmus, veröffentlicht 1956 von L. R. Ford, Jr. und D. R. Fulkerson
- Verwendet Tiefensuche, um den erweiternden Pfad P zu bestimmen
- Die Lösung wird um P erweitert, indem
  - die geringste Kapazität f der Kanten in P bestimmt wird
  - die Kapazitäten aller Kanten in P um f verringert werden
  - die Kapazitäten aller Gegenkanten um f erhöht werden
  - der maximale Fluss um f erhöht wird

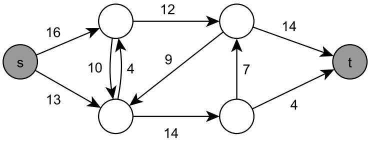
Ford-Fulkerson Algorithmus

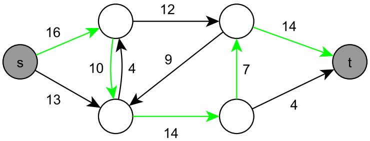
- Greedy-Algorithmus, veröffentlicht 1956 von L. R. Ford, Jr. und D. R. Fulkerson
- Verwendet Tiefensuche, um den erweiternden Pfad P zu bestimmen
- Die Lösung wird um P erweitert, indem
  - die geringste Kapazität f der Kanten in P bestimmt wird
  - die Kapazitäten aller Kanten in P um f verringert werden
  - die Kapazitäten aller Gegenkanten um f erhöht werden
  - der maximale Fluss um f erhöht wird

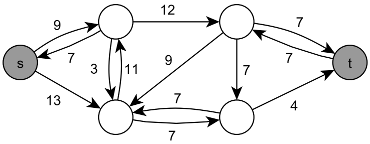
Ford-Fulkerson Algorithmus

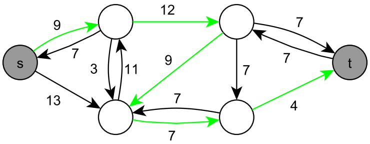
- Greedy-Algorithmus, veröffentlicht 1956 von L. R. Ford, Jr. und D. R. Fulkerson
- Verwendet Tiefensuche, um den erweiternden Pfad P zu bestimmen
- Die Lösung wird um P erweitert, indem
  - die geringste Kapazität f der Kanten in P bestimmt wird
  - die Kapazitäten aller Kanten in P um f verringert werden
  - die Kapazitäten aller Gegenkanten um f erhöht werden
  - der maximale Fluss um f erhöht wird

Ford-Fulkerson Algorithmus

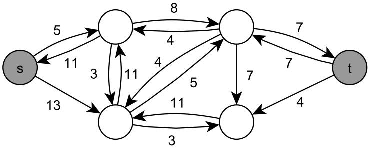


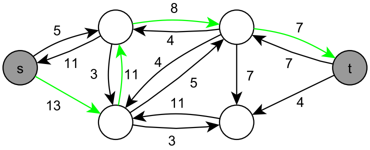


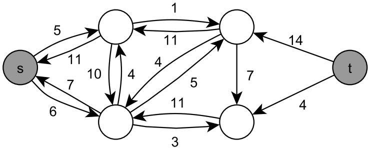


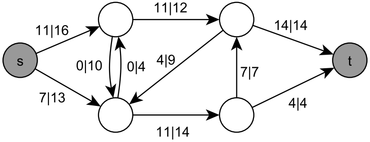


Ford-Fulkerson Algorithmus

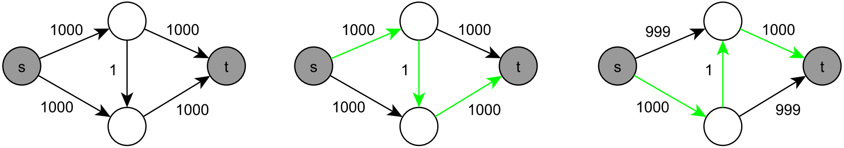








Ford-Fulkerson Algorithmus



- Im Worst-Case wird der maximale Fluss pro Iteration nur um 1 erhöht

$\Rightarrow$ Laufzeit in $\mathcal{O}(|f^*| \cdot |E|)$, wobei $|f^*|$ den Wert des maximalen Flusses beschreibt

- Deshalb **nicht** für ICPC-Aufgaben geeignet!

Edmonds-Karp Algorithmus

- 1972 von J. Edmonds und R. M. Karp veröffentlicht
- Verwendet Breitensuche, um den kürzesten erweiternden Pfad P zu bestimmen
- Erweiterung der Lösung um P analog zu Ford-Fulkerson
- Die Länge des erweiternden Pfades ist monoton steigend
- Es sind maximal $|V| \cdot |E|$ Iterationen notwendig

$\Rightarrow$ Laufzeit in $\mathcal{O}(|V| \cdot |E|^2)$

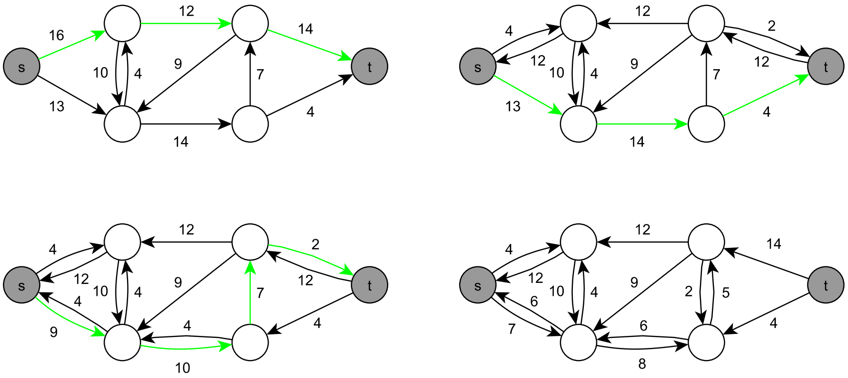
Edmonds-Karp Implementierung

Algorithm 1: Edmonds-Karp

```
Function Max-Flow (G = (V, E), s, t ∈ V, c : E → ℝ⁺)
    maxFlow = 0
    do
        find augmenting path P using BFS
        f = min{c(u, v)|(u, v) ∈ P}
        foreach (u, v) ∈ P do
            c(u,v) -= f
            c(v,u) += f
        end
        maxFlow += f
    while P exists
    return maxFlow
```

Edmonds-Karp Algorithmus

Karlsruher Institut für Technologie



Edmonds-Karp Implementierungsdetails

- In Adjazenzliste neben Zielknoten auch Kapazität und Verweis auf die Rückkante speichern
- Nicht vorhandene Rückkanten mit 0 initialisieren und dem Graphen hinzufügen
- Bei der Breitensuche nur Kanten mit positiver Kapazität berücksichtigen
- Breitensuche abbrechen, sobald t erreicht wurde

Min-Cut

Min-Cut

- Definiere Schnitt $C = (S, T)$ als Partition von V, wobei $s \in S$ und $t \in T$
- $c : E \to \mathbb{R}^+$ eine Kostenfunktion.
- Weiter sei die Schnittmenge $cs = \{(u, v) \in E | u \in S \wedge v \in T\}$
- Minimiere nun:

$$\sum_{e \in cs} c(e)$$

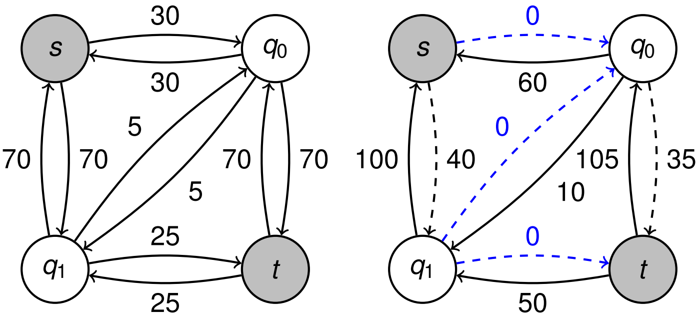
Max-Flow-Min-Cut-Theorem

Max-Flow-Min-Cut-Theorem

- Ein maximaler Fluss hat genau den Wert eines minimalen Schnitts.

Max-Flow-Min-Cut

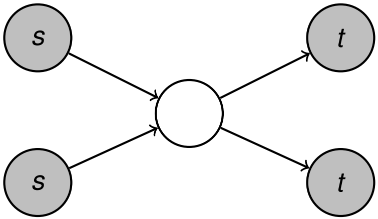
- Bsp.:



- Hier
  - $C = (\{s, q_1\}, \{t, q_0\})$
  - $c = \{(s, q_0), (q_1, q_0), (q_1, t)\}$

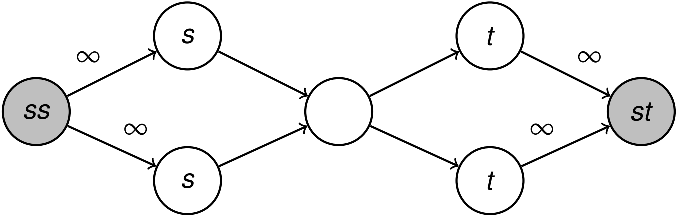
Multi-Quelle/Multi-Abfluss

- Gegeben sei folgende Situation:



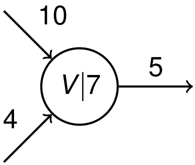
- Problem: Max-Flow Algorithmus kann nur mit einer Quelle und einer Senke arbeiten.
- Lösung: Ertelle Super-Quelle und Super-Senke und verbinde alle Quellen und Senken mit Kantengewicht ∞

Multi-Quelle/Multi-Abfluss Lösung



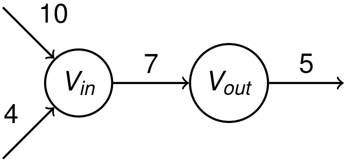
Knotenkapazität

- Gegeben sind Knoten mit Kapazität.
- Bsp.:



Knotenkapazität

- Gegeben sind Knoten mit Kapazität.
- Bsp.:



Modellierung

- Erkennen eines Netzwerkfluss-Problems nicht immer einfach
- Was hilft?
  - Übung
  - Übung
  - ...

Modellierung

Karlsruher Institut für Technologie

- Erkennen eines Netzwerkfluss-Problems nicht immer einfach
- Was hilft?
  - Übung
  - Übung
  - ...

Modellierung

- Erkennen eines Netzwerkfluss-Problems nicht immer einfach
- Was hilft?
  - Übung
  - Übung
  - ...

Modellierung - Beispiel

- Situation: Die Titanic ist gesunken. Es soll ermittelt werden, wie viele Menschen gerettet werden können.
- Eingabe: X, Y, P mit X, Y Dimension der Fläche ($1 \leq X, Y \leq 30$) und P ($P \leq 10$) die Anzahl von Personen, welche gleichzeitig auf ein Holzbrett können.

Symbol	Bedeutung
*	Menschen auf Treibeis
~	Eiskaltes Wasser
.	Treibeis
@	Großer Eisberg
#	Großes Holzbrett

Modellierung - Beispiel

- Gegeben sei nun folgende Eingabe:

*	~	~	#
.	.	.	@
.	~	.	*

- Wandle in Graphen um...

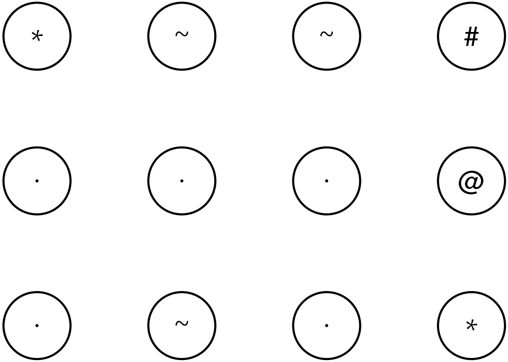
Modellierung - Beispiel

- Gegeben sei nun folgende Eingabe:

*	~	~	#
.	.	.	@
.	~	.	*

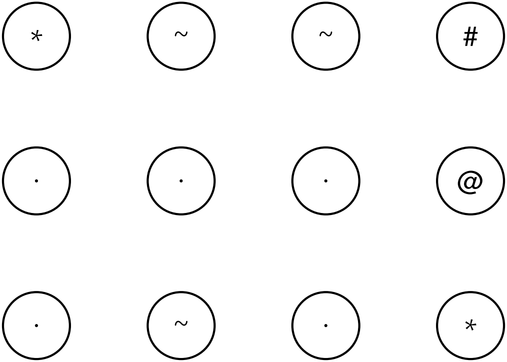
- Wandle in Graphen um...

Modellierung - Beispiel



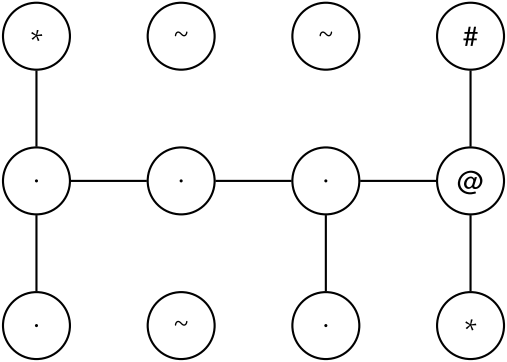
- Verbinde alle Knoten, über die ein Weg möglich ist...

Modellierung - Beispiel



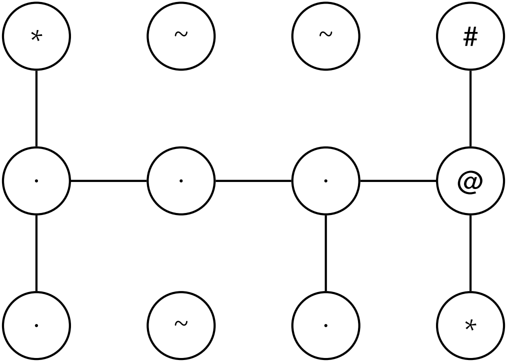
- Verbinde alle Knoten, über die ein Weg möglich ist...

Modellierung - Beispiel



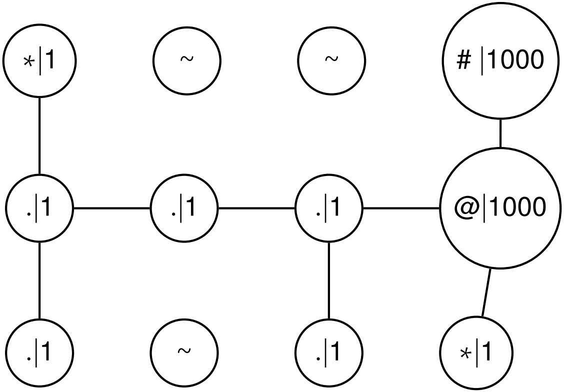
- Füge Knotengewichte hinzu...

Modellierung - Beispiel



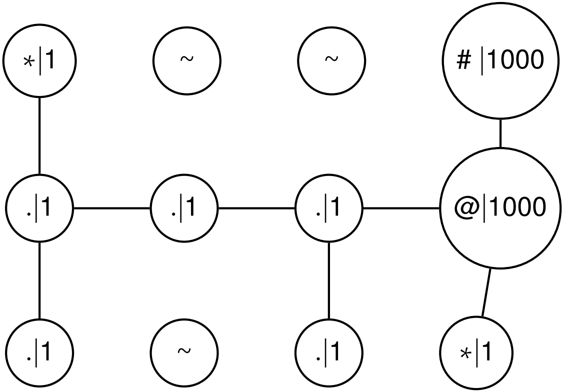
- Füge Knotengewichte hinzu...

Modellierung - Beispiel



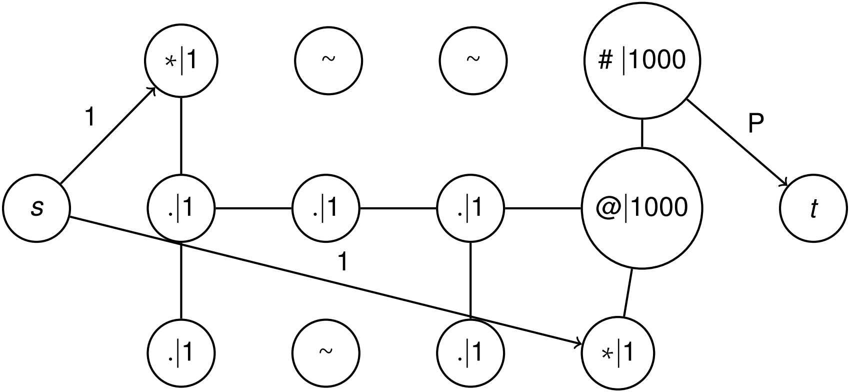
- Verbinde alle Menschen mit s und alle Holzbretter mit t...

Modellierung - Beispiel



- Verbinde alle Menschen mit s und alle Holzbretter mit t...

Modellierung - Beispiel



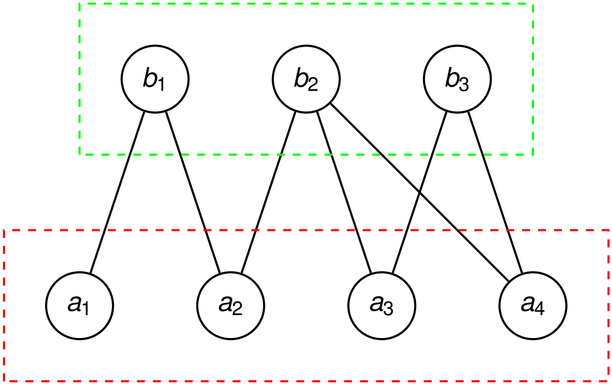
- Bem.: Knotengewichte müssen noch aufgelöst werden

Bipartiter Graph

Bipartiter Graph

- Ein Graph G = (V, E) heißt **bipartit**, wenn sich $V = A \mathbin{\dot\cup} B$ in 2 disjunkte Knotenmengen A und B aufteilen lässt, sodass zwischen den Knoten innerhalb der Teilmengen keine Kanten existieren.

Bipartiter Graph



Matching

Matching

- Sei $G = (V, E)$ ein Graph. Ein **Matching** $M \subseteq E$ ist eine Menge paarweise knotendisjunkter Kanten, d.h.
  $\forall e_1 = \{u_1, v_1\}, e_2 = \{u_2, v_2\} \in M, e_1 \neq e_2 : e_1 \cap e_2 = \varnothing$
- Analog für gerichtete Graphen

Maximales Matching

- Ein Matching heißt **maximales Matching**, wenn nicht durch Hinzufügen einer Kante ein größeres Matching erstellt werden kann. (D.h. es gibt keine Kante $e = \{u, v\}$, wobei u und v nicht Teil des Matchings sind.)

Matching

Karlsruher Institut für Technologie

Matching

- Sei $G = (V, E)$ ein Graph. Ein **Matching** $M \subseteq E$ ist eine Menge paarweise knotendisjunkter Kanten, d.h.
  $\forall e_1 = \{u_1, v_1\}, e_2 = \{u_2, v_2\} \in M, e_1 \neq e_2 : e_1 \cap e_2 = \varnothing$
- Analog für gerichtete Graphen

Maximales Matching

- Ein Matching heißt **maximales Matching**, wenn nicht durch Hinzufügen einer Kante ein größeres Matching erstellt werden kann. (D.h. es gibt keine Kante $e = \{u, v\}$, wobei u und v nicht Teil des Matchings sind.)

Mehr Matchings

Kardinalitätsmaximales Matching

- Ein Matching $M \subseteq E$ heißt **kardinalitätsmaximales Matching**, wenn es kein größeres Matching gibt. (D.h. $\forall$ Matchings $M' : |M| \geq |M'|$).

Perfektes Matching

- Ein Matching M heißt **perfekt**, falls $2 * |M| = |V|$, d.h. jeder Knoten $v \in V$ kommt in M vor.

Mehr Matchings

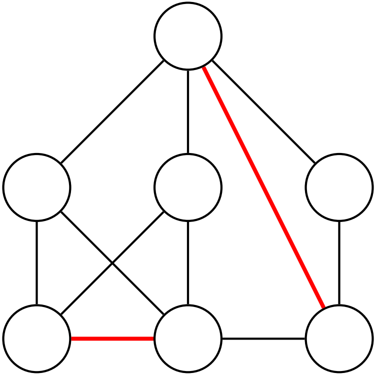
Kardinalitätsmaximales Matching

- Ein Matching $M \subseteq E$ heißt **kardinalitätsmaximales Matching**, wenn es kein größeres Matching gibt. (D.h. $\forall$ Matchings $M' : |M| \geq |M'|$).

Perfektes Matching

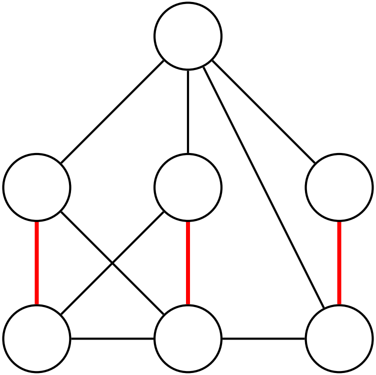
- Ein Matching M heißt **perfekt**, falls $2 * |M| = |V|$, d.h. jeder Knoten $v \in V$ kommt in M vor.

Beispiel: Matching



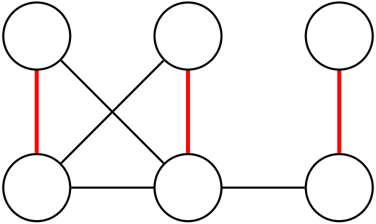
Maximales Matching

Beispiel: Matching



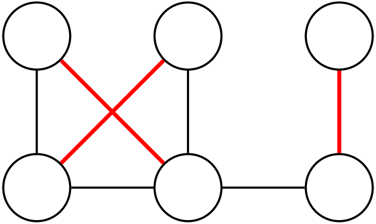
Kardinalitätsmaximales Matching

Beispiel: Matching



Perfektes Matching

Beispiel: Matching



anderes perfektes Matching

Beispiel

- Gegeben: Menge von Aufgaben A, Personen B
- Für jede Person eine Liste von Aufgaben, die diese Person erledigen kann
- Gesucht: Zuteilung von Aufgaben an Personen, sodass möglichst viele Aufgaben erledigt werden
- Jede Aufgabe kann von maximal einer Person zugeteilt werden, jeder Person kann maximal eine Aufgabe zugeteilt werden
- Lösung: Modellierung als *bipartiter Graph* mit Knotenmengen A und B
- Kante zwischen Aufgabe a und Person b, wenn b Aufgabe a lösen kann
- Lösung entspricht *kardinalitätsmaximalem Matching*

Beispiel

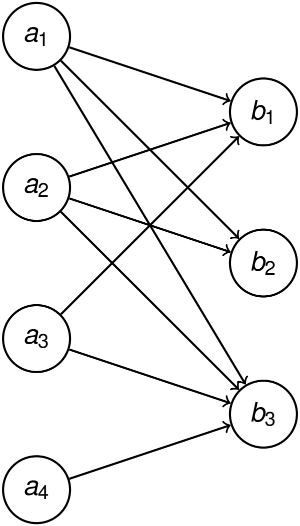
- Gegeben: Menge von Aufgaben *A*, Personen *B*
- Für jede Person eine Liste von Aufgaben, die diese Person erledigen kann
- Gesucht: Zuteilung von Aufgaben an Personen, sodass möglichst viele Aufgaben erledigt werden
- Jede Aufgabe kann von maximal einer Person zugeteilt werden, jeder Person kann maximal eine Aufgabe zugeteilt werden
- Lösung: Modellierung als *bipartiter Graph* mit Knotenmengen *A* und *B*
- Kante zwischen Aufgabe *a* und Person *b*, wenn *b* Aufgabe *a* lösen kann
- Lösung entspricht *kardinalitätsmaximalem Matching*

Beispiel

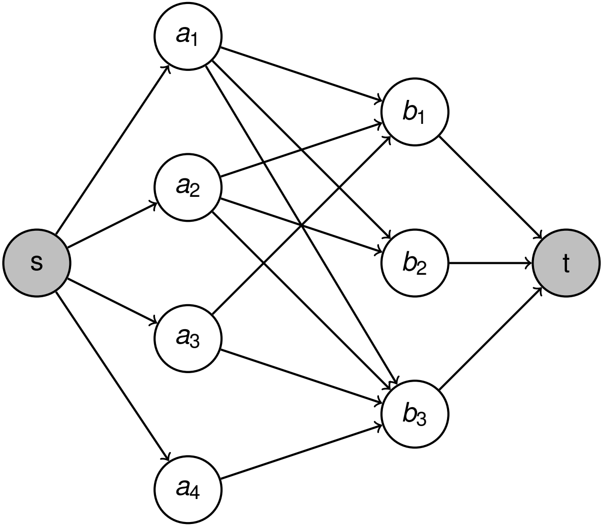
- Gegeben: Menge von Aufgaben A, Personen B
- Für jede Person eine Liste von Aufgaben, die diese Person erledigen kann
- Gesucht: Zuteilung von Aufgaben an Personen, sodass möglichst viele Aufgaben erledigt werden
- Jede Aufgabe kann von maximal einer Person zugeteilt werden, jeder Person kann maximal eine Aufgabe zugeteilt werden
- Lösung: Modellierung als *bipartiter Graph* mit Knotenmengen A und B
- Kante zwischen Aufgabe a und Person b, wenn b Aufgabe a lösen kann
- Lösung entspricht *kardinalitätsmaximalem Matching*

Modellierung

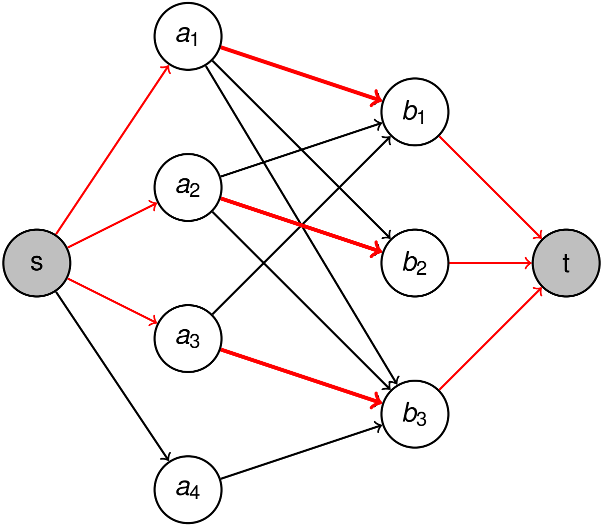
Karlsruher Institut für Technologie



Modellierung



Modellierung



Modellierung

- Finden von kardinalitätsmaximalen Matchings in bipartiten Graphen $G = (V, E = A \cupdot B)$:
  - Einfügen von neuen Knoten s und t
  - Einfügen von Kanten zwischen s und allen Knoten $v_A \in A$, und zwischen allen Knoten $v_B \in B$ und t.
  - Jede Kante im Graph (alte und neu eingefügte) hat Kapazität 1.
  - Berechnen des maximalen Flusses von s nach t.
- Kanten des *maximalen Flusses* zwischen A und B entsprechen *kardinalitätsmaximalen Matching*.
- Laufzeit Edmonds-Karp (in diesem Fall): $\mathcal{O}(|E| * |V|)$

Mehr Graphentheorie

Max Independent Set

- Ein **independent set** $S \subseteq V$ ist eine Menge von Knoten, die paarweise nicht *adjazent* sind, d.h. $\forall u, v \in S : \{u, v\} \notin E$ (zwischen keinen zwei Knoten existiert eine Kante).
- Ein **maximal independent set** ist ein maximal großes *independent set*.
- Das bedeutet: Jeder Knoten ist entweder selbst in S, oder einer seiner Nachbarn ist in S.

Min Vertex Cover

- Ein **vertex cover** $U \subseteq V$ ist eine Menge von Knoten, sodass jede Kante von G zu einem Knoten aus U *inzident* ist, d.h. $\forall \{v, w\} = e \in E : \exists u \in U : u \in e$.
- Ein **minimum vertex cover** ist ein minimal großes *vertex cover*.

Mehr Graphentheorie

Karlsruher Institut für Technologie

Max Independent Set

- Ein **independent set** $S \subseteq V$ ist eine Menge von Knoten, die paarweise nicht *adjazent* sind, d.h. $\forall u, v \in S : \{u, v\} \notin E$ (zwischen keinen zwei Knoten existiert eine Kante).
- Ein **maximal independent set** ist ein maximal großes *independent set*.
- Das bedeutet: Jeder Knoten ist entweder selbst in S, oder einer seiner Nachbarn ist in S.

Min Vertex Cover

- Ein **vertex cover** $U \subseteq V$ ist eine Menge von Knoten, sodass jede Kante von G zu einem Knoten aus U *inzident* ist, d.h. $\forall \{v, w\} = e \in E : \exists u \in U : u \in e$.
- Ein **minimum vertex cover** ist ein minimal großes *vertex cover*.

Noch mehr Graphentheorie

Knigs Theorem

- Sei G ein *bipartiter* Graph, M ein *kadinalitätsmaximales Matching*, U ein *minimum vertex cover*.
- Dann gilt $|M| = |U|$.
- In Worten: Die Anzahl an Kanten eines kardinalitätsmaximalen Matchings entspricht der Anzahl an Knoten in einem minimum vertex cover *in einem bipartiten Graph*.

Andere Erkenntnis

- In einem bipartiten Graph mit *minimum vertex cover* U und *maximalen independent set* S gilt:
- $|U| + |S| = |V|$.

Noch mehr Graphentheorie

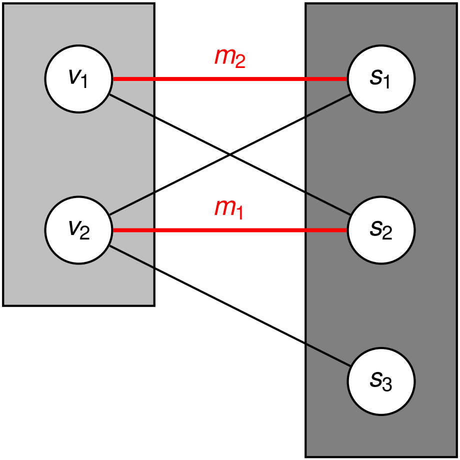
Knigs Theorem

- Sei G ein *bipartiter* Graph, M ein *kadinalitätsmaximales Matching*, U ein *minimum vertex cover*.
- Dann gilt $|M| = |U|$.
- In Worten: Die Anzahl an Kanten eines kardinalitätsmaximalen Matchings entspricht der Anzahl an Knoten in einem minimum vertex cover *in einem bipartiten Graph*.

Andere Erkenntnis

- In einem bipartiten Graph mit *minimum vertex cover* U und *maximalen independent set* S gilt:
- $|U| + |S| = |V|$.

Veranschaulichung



Beispiel: Guardian of Decency

- Gegeben: N Schüler (mit Größe, Geschlecht, Musikgeschmack, Lieblingssport).
- Gesucht: Wie viele können maximal auf Exkursion gehen, sodass je zwei Schüler kein Pärchen werden können, da...
  - ...sich ihre Größe um mehr als 40 cm unterscheidet,
  - ...sie das selbe Geschlecht haben,
  - ...ihr Musikgeschmack unterschiedlich ist,
  - ...oder sie den selben Lieblingssport haben.
- Kante zwischen Personen, wenn sie ein Pärchen werden könnten
- Graph ist bipartit: Männliche und weibliche Schüler
- Aufgabenstellung: Maximum independent set
- Lösung: $|\text{maximum independent set}| = N - |\text{kardinalitätsmaximales Matching}|$

Beispiel: Guardian of Decency

- Gegeben: N Schüler (mit Größe, Geschlecht, Musikgeschmack, Lieblingssport).
- Gesucht: Wie viele können maximal auf Exkursion gehen, sodass je zwei Schüler kein Pärchen werden können, da...
  - ...sich ihre Größe um mehr als 40 cm unterscheidet,
  - ...sie das selbe Geschlecht haben,
  - ...ihr Musikgeschmack unterschiedlich ist,
  - ...oder sie den selben Lieblingssport haben.
- Kante zwischen Personen, wenn sie ein Pärchen werden könnten
- Graph ist bipartit: Männliche und weibliche Schüler
- Aufgabenstellung: Maximum independent set
- Lösung: $|\text{maximum independent set}| = N - |\text{kardinalitätsmaximales Matching}|$

Beispiel: Guardian of Decency

- Gegeben: N Schüler (mit Größe, Geschlecht, Musikgeschmack, Lieblingssport).
- Gesucht: Wie viele können maximal auf Exkursion gehen, sodass je zwei Schüler kein Pärchen werden können, da...
  - ...sich ihre Größe um mehr als 40 cm unterscheidet,
  - ...sie das selbe Geschlecht haben,
  - ...ihr Musikgeschmack unterschiedlich ist,
  - ...oder sie den selben Lieblingssport haben.
- Kante zwischen Personen, wenn sie ein Pärchen werden könnten
- Graph ist bipartit: Männliche und weibliche Schüler
- Aufgabenstellung: Maximum independent set
- Lösung: $|\text{maximum independent set}| = N - |\text{kardinalitätsmaximales Matching}|$

Bonusfolie

Karlsruher Institut für Technologie

- Es gilt: *G bipartit* $\Leftrightarrow$ G enthält keine ungeraden Kreise
  - Damit zum Beispiel bipartit: Jeder Baum
- Sei *G bipartit* mit $V = A \cup B$. Dann: G hat *perfektes Matching* $\Leftrightarrow \forall S \subset A : |N(S)| \geq |S|$ (Wobei die Nachbarn $N(S)$ alle zu einem Knoten $s \in S$ adjazenten Knoten sind).
- Es gibt asymptotisch bessere Algorithmen zum Finden maximaler Flüsse, zum Beispiel *Dinitz* ($\mathcal{O}(|V|^2 * |E|)$ oder Push-Relabel-Algorithmen

Bonusbeispiel

- Definition: *Complete prime pairing*: Teile Liste M von natürlichen Zahlen so vollständig in Paare auf, dass die Summe beider Elemente eines Paares immer eine Primzahl ist.
- Gegeben: Liste N von paarweise unterschiedlichen Zahlen $n_i \in \mathbb{N}$.
- Gesucht: Liste $M \subseteq N$, sodass: $\forall m \in M$:
  - $\{m, n_0\}$ ist ein prime pair, d.h. $m + n_0$ ist prim
  - $N \setminus \{n_0, m\}$ besitzt ein *complete prime pairing*.
- Erkenntnis: Alle Primzahlen müssen als Summe einer geraden und einer ungeraden Zahl zustande kommen.
- Damit: Bipartiter Graph, Kante zwischen a und b, falls $a + b$ eine Primzahl ist.
- Falls beide Mengen unterschiedlich groß sind, gibt es keine Lösung
- Ansonsten: Gehe alle Nachbarn n' von n_0 durch und überprüfe, ob $G - \{n_0, n'\}$ ein *perfektes Matching* hat.

Bonusbeispiel

- Definition: *Complete prime pairing*: Teile Liste M von natürlichen Zahlen so vollständig in Paare auf, dass die Summe beider Elemente eines Paares immer eine Primzahl ist.
- Gegeben: Liste N von paarweise unterschiedlichen Zahlen $n_i \in \mathbb{N}$.
- Gesucht: Liste $M \subseteq N$, sodass: $\forall m \in M$:
  - $\{m, n_0\}$ ist ein prime pair, d.h. $m + n_0$ ist prim
  - $N \setminus \{n_0, m\}$ besitzt ein *complete prime pairing*.
- Erkenntnis: Alle Primzahlen müssen als Summe einer geraden und einer ungeraden Zahl zustande kommen.
- Damit: Bipartiter Graph, Kante zwischen a und b, falls $a + b$ eine Primzahl ist.
- Falls beide Mengen unterschiedlich groß sind, gibt es keine Lösung
- Ansonsten: Gehe alle Nachbarn n' von n_0 durch und überprüfe, ob $G - \{n_0, n'\}$ ein *perfektes Matching* hat.