Algorithmen & Wahrscheinlichkeit

Übungsstunde 13

Michelle Honegger - ML F34

Programm

- Semester Recap
- Kahoot!
- Prüfung/Fragen

Semester Recap

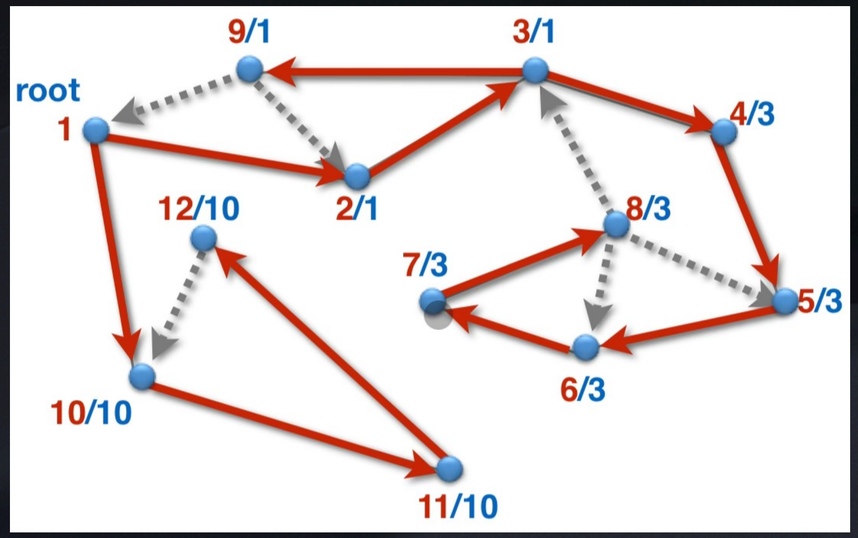
- Graphentheorie
  - Zusammenhang
  - Kreise
  - TSP
  - Matchings
  - Färbungen

Zusammenhang

- Ein Graph ist k-zusammenhängend, wenn gilt:
  - $|V| \geq k+1$ und $\forall X \subseteq V$ ist $G[V \backslash X]$ zusammenhängend
  - Man muss mindestens k Knoten löschen um den Zusammenhang zu zerstören
- Ein Graph ist k-Kanten-zusammenhöngend, wenn gilt:
  - $\forall X \subseteq E$ mit $|X| < k$ ist $(V, E \backslash X)$ zusammenhängend
  - **Knotenzusammenhang ≤ Kantenzusammenhang ≤ minimaler Grad**
  - G ist k-zusammenhängend gdw. $\forall u, v \in V, u \neq v$ gibt es k intern-knotendisjunkte u-v-Pfade

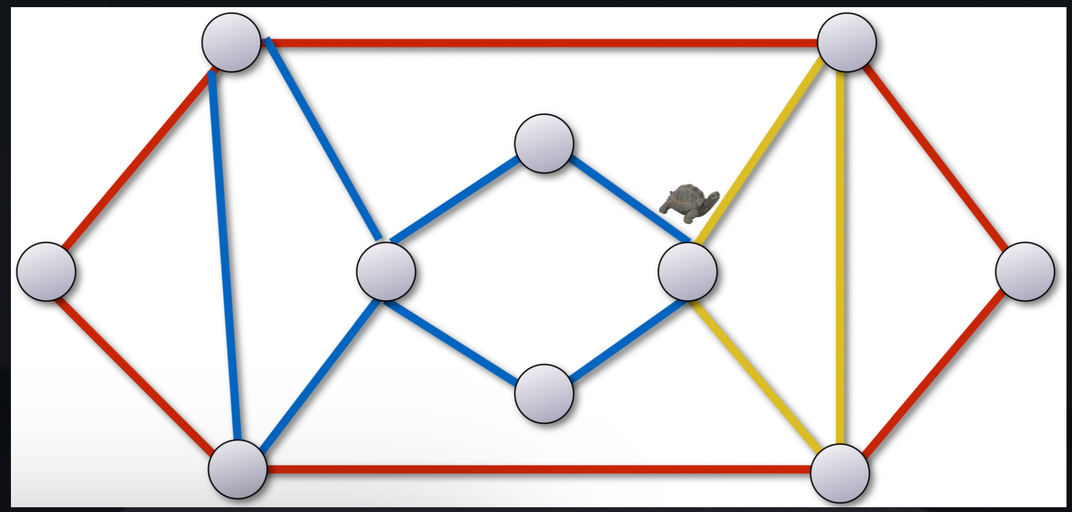
DFS mit Low-Werten

- low[v]:= kleinste dfs-Nummer, die man von v aus durch einen gerichteten Pfad aus Baumkanten und maximal einer Restkante erreichen kann
- v ist Artikulationsknoten gdw. v ≠ root und v hat ein Kind u im DFS-Baum mit low[u] ≥ dfs[v] oder v = root und v hat mindestens zwei Kinder im DFS-Baum
- Eine Baumkante ist eine Brücke gdw. low[w] > dfs[v]
- Läuft in O(m)



Eulertouren

- Jede Kante genau ein Mal - gleicher Start- und Endknoten
- Existiert gdw. Grad aller Knoten gerade ist
- Kann in O(|E|) gefunden werden



- Wenn alle Knotengrade bis auf 2 gerade sind, enthält der Graph einen Eulerweg

Hamiltonkreise

Spezialfälle für einfachere Bestimmung

- n x m Grid **enthält** Hamiltonkreis gdw. **n*m gerade** (für n, m ≥ 2)
- Ikosaeder **enthält** Hamiltonkreis
- d-dimensionale Hyperwürfel **enthält** Hamiltonkreis
- Jeder Graph G = (V, E) mit |V| ≥ 3 und Minimalgrad δ(G) ≥ |V|/2 **enthält** Hamiltonkreis (Dirac)

- Petersengraph **enthält keinen** Hamiltonkreis

Metrisches TSP

2-Approximation

- Funktion l erfüllt Dreiecksungleichung: $l(x,z) \leq l(x,y) + l(y,z) \qquad \forall x,y,z \in [n]$
  - Bestimme MST T es gilt: $l(T) \leq opt(K_n, l)$
  - verdopple alle Kanten von T: $2l(T) \leq 2opt(K_n, l)$
  - bestimme Eulertour W: $l(W) = 2l(T) \leq 2opt(K_n, l)$
  - durchlaufe W mit Abkürzungen, so dass jeder Knoten nur einmal besucht wird => Hamiltonkreis C: $l(C) \leq l(W) = 2l(T) \leq 2opt(K_n, l)$

Metrisches TSP

1.5-Approximation

- Funktion l erfüllt Dreiecksungleichung: $l(x,z) \leq l(x,y) + l(y,z) \qquad \forall x,y,z \in [n]$
  - Bestimme MST T es gilt: $l(T) \leq opt(K_n, l)$
  - X:= Knoten mit ungeradem Grad in T; Finde minimales Matching M für X $l(M) \leq \frac{1}{2} opt(K_n, l)$
  - bestimme Eulertour W: $l(W) = l(T) + l(M) \leq \frac{3}{2} opt(K_n, l)$
  - durchlaufe W mit Abkürzungen, so dass jeder Knoten nur einmal besucht wird => Hamiltonkreis C: $l(C) \leq l(W) = l(T) + l(M) \leq \frac{3}{2} opt(K_n, l)$

Matchings

- **inklusionsmaximal**: Falls es kein Matching M‘ gibt mit $M \subseteq M'$ und $|M'| > |M|$
- **kardinalitätsmaximal**: Falls es kein Matching M‘ gibt mit $|M'| > |M|$
- Jedes Matching, das nicht (kardinalitäts-) maximal ist, besitzt einen augmentierenden Pfad.

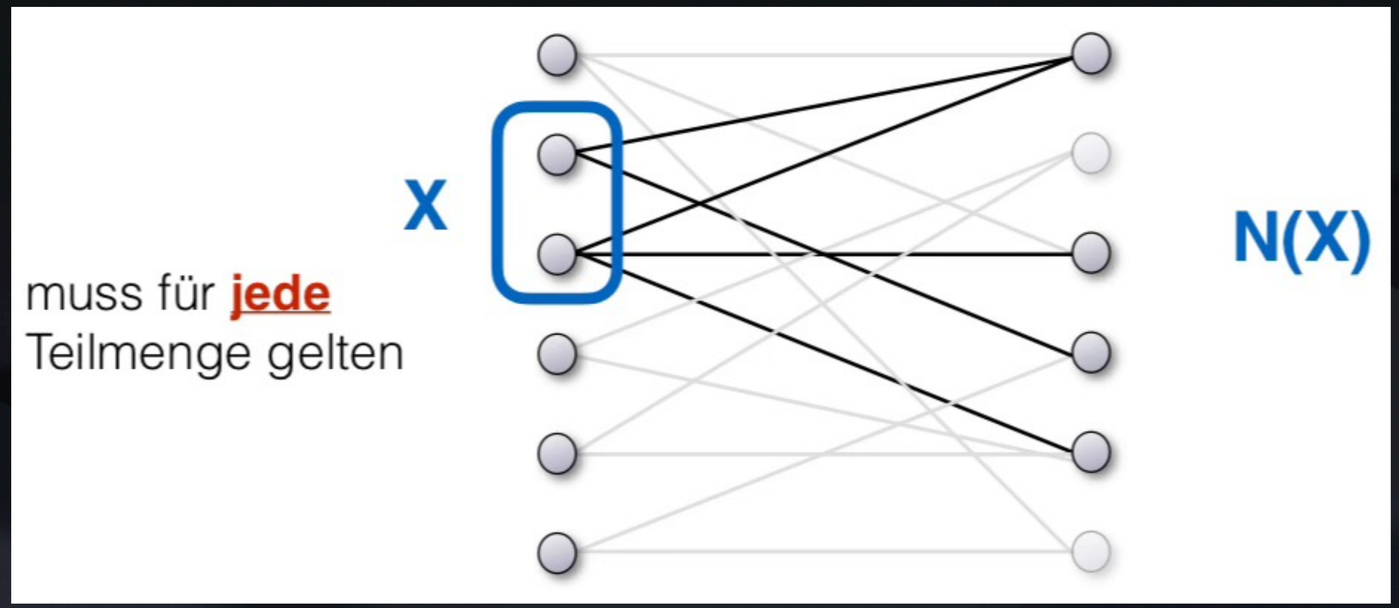
for i = 1 **to** n
  if i ungerade **then**
    L_i := {unbesuchte Nachbarn von L_{i-1} via Kanten in $E \setminus M$}
  else (falls i gerade)
    L_i := {unbesuchte Nachbarn von L_{i-1} via Kanten in M}
  Markiere Knoten aus L_i als besucht.

  if ein Knoten v in L_i ist nicht überdeckt
    then return Pfad zu v (backtracking)

Satz von Hall (Heiratssatz)

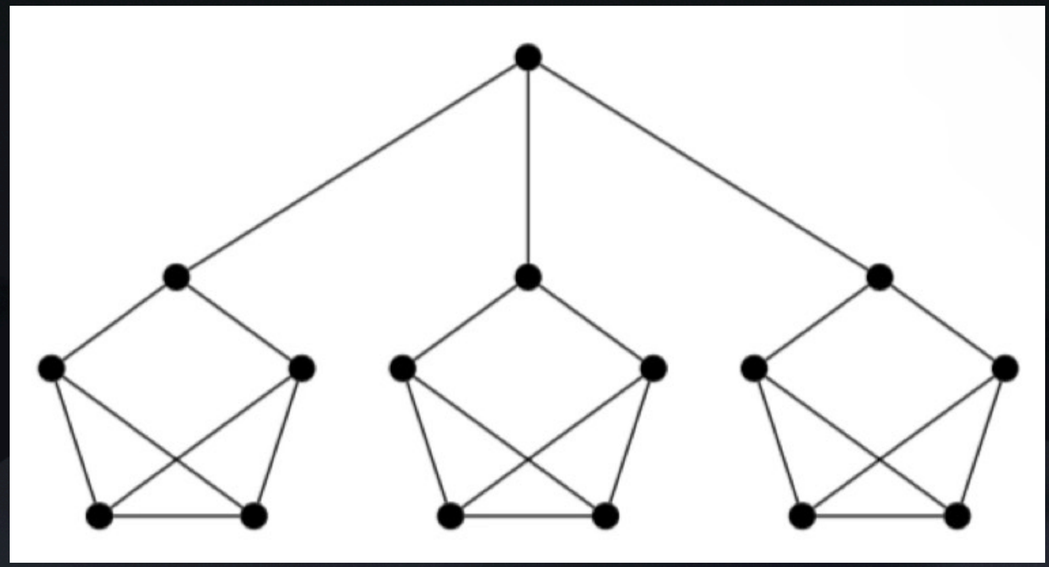
- Ein bipartiter Graph G=(A⊎B,E) enthält ein Matching M der Kardinalität |M|=|A| gdw.

  $\forall X \subseteq A : |X| \leq |N(X)|$



Frobenius

- Für alle k gilt: jeder k-reguläre bipartite Graph enthält ein perfektes Matching.
- s:= |(Kanten von X nach N(X)|
  - $s = k*|X| \Rightarrow s \leq k*|N(X)| \Rightarrow |X| \leq |N(X)|$



Matchings

Gabow Algorithmus

- In 2^k-regulären bipartiten Graphen kann man in Zeit O(|E|) ein perfektes Matching bestimmen.

- Da Graph 2^k-regulär ist, sind alle Knotengrade gerade ⇒ Graph enthält Eulertour welche man in O(|E|) finden kann.
  - Davon löschen wir jede 2. Kante und iterieren bis wir das Matching haben

- Kann für k-regulätre bipartite Graphen verallgemeinert werden. (Cole, Ost, Schirra)

Augmentierende Pfade

Hopcroft und Karp Algorithmus

- Laufzeit: 1 Durchlauf O(|V|+|E|)
- Laufzeit: alle Pfade O($\sqrt{|V|}$*|E|)

MAXIMAL_MATCHING $(G = (A \oplus B, E))$ (Hopcroft und Karp)

1: $M := \{e\}$ für irgendeine Kante $e \in E$.
2: **while** es gibt noch augmentierende Pfade **do**
3: $k :=$ Länge eines kürzesten augmentierenden Pfades
4: Finde eine inklusionsmaximale Menge S von paarweise disjunkten augmentierenden Pfaden der Länge k.
5: **for all** P aus S **do**
6: $M := M \oplus P$. // *augmentiere entlang der Pfade aus S*
7: **return** M

- Blossom-Algorithmus für allgemeine Graphen in O(|V|*|E|)

Färbungen

- Eine Knotenfärbung eines Graphen G = (V,E) mit k Farben ist eine Abbildung $c : V \rightarrow [k]$, so dass gilt $c(u) \neq v(u)$ für alle Kanten $\{u, v\} \in E$
- Die **chromatische Zahl** $\chi(G)$ ist die minimale Anzahl Farben, die für eine Knotenfärbung von G benötigt wird.
- bipartit lässt sich in O(|E|) mit DFS/BFS rausfinden
- Für jedes $k \geq 3$ ist das Problem: Gegeben ein Graph G = (V, E), gilt $\chi(G) \leq k$? NP-vollstängdig

Färbungen

Greedy Algorithmus

- Für jede Reihenfolge benötigt der Greedy-Algorithmus höchstens Δ(G)+1 viele Farben

GREEDY-FÄRBUNG (G)

1: wähle eine beliebige Reihenfolge der Knoten: $V = \{v_1, \ldots, v_n\}$
2: $c[v_1] \leftarrow 1$
3: **for** $i = 2$ **to** $i = n$ **do**
4: $\quad c[v_i] \leftarrow \min\{k \in \mathbb{N} \mid k \neq c(u)$ **für alle** $u \in N(v_i) \cap \{v_1, \ldots, v_{i-1}\}\}$

- Δ(G) := maximaler Grad in G
- Es gibt eine Reihenfolge für die der Greedy-Algorithmus χ(G) viele Farben benötigt
- es gibt bipartite Graphen und eine Reihenfolge für die der Greedy-Algorothmus |V|/2 viele Farben benötigt

Satz von Brooks

- Ist G = (V,E) ein zusammenhängender Graph, der weder vollständig noch ein ungerader Kreis ist, so gilt $\chi(G) \leq \Delta(G)$ und es gibt einen Algorithmus, der die Knoten in Zeit O(|E|) mit $\Delta(G)$ Farben färbt.
  - G hat Knoten v mit deg(v) < $\Delta(G)$ -> DFS von v Knoten-Reihenfolge rückwärts
  - G ist k-regulär und hat Artikulationsknoten a -> a finden und von dort Case 1 ev. tauschen von Farben sodass Gleicher Knoten in versch. ZHKs gleiche Farbe bekommt
  - G ist k-regulär und hat keinen Artikulationsknoten -> nächste Slide

Färbungen

Spezialfälle und Zusatzinformation

- Gerade Kreise sind 2 färbbar, ungerade Kreise sind 3 färbbar
- Ein Graph ist genau dann bipartit wenn er keinen Kreis ungerader Länge als Teilgraphen enthält
- **Mycielski Konstruktion**: Für alle $k \geq 2$ gibt es einen dreiecksfreien Graphen G_k mit $\chi(G_k) \geq k$
- Jeden 3-Färbbaren Graphen kann man in Zeit $O(|E|)$ mit $O(\sqrt{|V|})$ Farben färben

3-Färbbarkeit

- Wähle Knoten mit Grad ≥ √n
- dies Kann man maximal √n mal machen
- Danach bleiben maximal √n Knoten übrig
- Wir brauchen 4√n Farben

Färbungen

Spezialfälle und Zusatzinformation

- Jede Landkarte lässt sich mit vier Farben färben (Planare Graphen)
- Ein **planarer Graph** ist ein Graph ohne Kanten Überkreuzungen
- Planare Graphen haben immer mindestens 1 Knoten mit Grad ≤ 5
- Heuristik findet Färbung mit ≤ 6 Farben

Semester Recap Teil 2

Wahrscheinlichkeitstheorie

- Grundbegriffe und Lemmas
- Bedingte W'keiten
- Unabhängigkeit
- Zufallsvariablen
  - Erwartungswert und Varianz
- Mehrere Zufallsvariablen
- Abschätzen von W'keiten

Kombinatorik

	Mit Zurücklegen	Ohne Zurücklegen
Mit Reihenfolge	n^k	$\frac{n!}{(n-k)!}$
Ohne Reihenfolge	$\binom{n+k-1}{k}$	$\binom{n}{k} = \frac{n!}{(n-k)!k!}$

Grundbegriffe

- Für Ereignisse A, B gilt:
  - $\Pr[\emptyset] = 0$, $\Pr[\Omega] = 1$
  - $0 \leq \Pr[A] \leq 1$
  - $\Pr[\overline{A}] = 1 - \Pr[A]$
  - Wenn $A \subseteq B$, so folgt $\Pr[A] \leq \Pr[B]$

Definition 2.1. Ein *diskreter Wahrscheinlichkeitsraum* ist bestimmt durch eine *Ergebnismenge* $\Omega = \{\omega_1, \omega_2, \ldots\}$ von *Elementarereignissen*. Jedem Elementarereignis ω_i ist eine *(Elementar-)Wahrscheinlichkeit* $\Pr[\omega_i]$ zugeordnet, wobei wir fordern, dass $0 \leq \Pr[\omega_i] \leq 1$ und

$$\sum_{\omega \in \Omega} \Pr[\omega] = 1.$$

Eine Menge $E \subseteq \Omega$ heisst *Ereignis*. Die Wahrscheinlichkeit $\Pr[E]$ eines Ereignisses ist definiert durch

$$\Pr[E] := \sum_{\omega \in E} \Pr[\omega].$$

Ist E ein Ereignis, so bezeichnen wir mit $\bar{E} := \Omega \setminus E$ das *Komplementärereignis* zu E.

Grundbegriffe

Additionssatz

- Wenn die Ereignisse $A_1, \ldots, A_n$ paarweise disjunkt sind (also wenn für alle Paare i ≠ j gilt, dass $A_i \cap A_j = \varnothing$) so gilt:

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] = \sum_{i=1}^{n} \Pr[A_i]$$

- Für eine unendliche Menge von disjunkten Ereignisen $A_1, A_2, \ldots$ gilt analog:

$$\Pr\left[\bigcup_{i=1}^{\infty} A_i\right] = \sum_{i=1}^{\infty} \Pr[A_i]$$

Siebformel

vgl. Boolesche Ungleichung:

$$\Pr\left[\bigcup_{i=1}^{n} A_i\right] \leq \sum_{i=1}^{n} \Pr[A_i]$$

Satz 1.35. *(Siebformel, Prinzip der Inklusion/Exklusion)* Für endliche Mengen $A_1, \ldots, A_n$ $(n \geq 2)$ gilt:

$$\begin{aligned}\left|\bigcup_{i=1}^{n} A_i\right| &= \sum_{l=1}^{n} (-1)^{l+1} \sum_{1 \leq i_1 < \cdots < i_l \leq n} |A_{i_1} \cap \cdots \cap A_{i_l}| \\ &= \sum_{i=1}^{n} |A_i| - \sum_{1 \leq i_1 < i_2 \leq n} |A_{i_1} \cap A_{i_2}| + \sum_{1 \leq i_1 < i_2 < i_3 \leq n} |A_{i_1} \cap A_{i_2} \cap A_{i_3}| \\ &\quad - \cdots + (-1)^{n+1} \cdot |A_1 \cap \cdots \cap A_n|.\end{aligned}$$

- Für n=2: $|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|$
- Für n=3: $|A_1 \cup A_2 \cup A_3| = |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3| + |A_1 \cap A_2 \cap A_3|$

Jeweils $2^n - 1$ Terme

Bedingte Wahrscheinlichkeit

- A und B seien Ereignisse mit $Pr[B] > 0$. Die bedingte W'keit $Pr[A|B]$ von A gegeben B ist definiert durch

$$Pr[A|B] := \frac{Pr[A \cap B]}{Pr[B]}$$

- $Pr[B|B] = 1$, $Pr[B|\overline{B}] = 0$; $Pr[A|\Omega] = Pr[A]$; $Pr[A|B] \sim Pr[A \cap B]$

- Kinderproblem

Multiplikationssatz

- Seien die Ereignisse $A_1, \ldots, A_n$ gegeben. Falls:

$$Pr[A_1 \cap \ldots \cap A_n] = Pr[A_1] \cdot Pr[A_2 \,|\, A_1] \cdot Pr[A_3 \,|\, A_1 \cap A_2] \cdot \cdots Pr[A_n \,|\, A_1 \cap \cdots \cap A_{n-1}]$$

- $Pr[A_i \,|\, A_1 \cap \ldots \cap A_{i-1}] = \dfrac{n - (i-1)}{n}$

- Geburtstagsproblem/Balls into bins

Satz der totalen Wahrscheinlichkeit

- Die Ereignisse $A_1, \ldots, A_n$ seien paarweise disjunkt und es gelte $B \subseteq A_1 \cup \ldots \cup A_n$ Dann folgt:

$$\sum_{i=1}^{n} Pr[B \mid A_i] \cdot Pr[B \cap A_i]$$

- Ziegenproblem

Satz von Bayes

- Die Ereignisse $A_1, \ldots, A_n$ seien paarweise disjunkt. Ferner seit $B \subseteq A_1 \cup \ldots \cup A_n$ ein Ereignis mit Pr[B] > 0. Dann gilt für ein beliebiges i = 1,...,n

$$Pr[A_i \mid B] = \frac{Pr[A_i \cap B]}{Pr[B]} = \frac{Pr[B \mid A_i] \cdot Pr[A_i]}{\sum_{j=1}^{n} Pr[B \mid A_j] \cdot Pr[A_j]}$$

- Krankheitenbsp.
- Beweis: Zähler Bedingte Wahrscheinlichkeit, Nenner: Satz der totalen W'keit

Unabhängigkeit

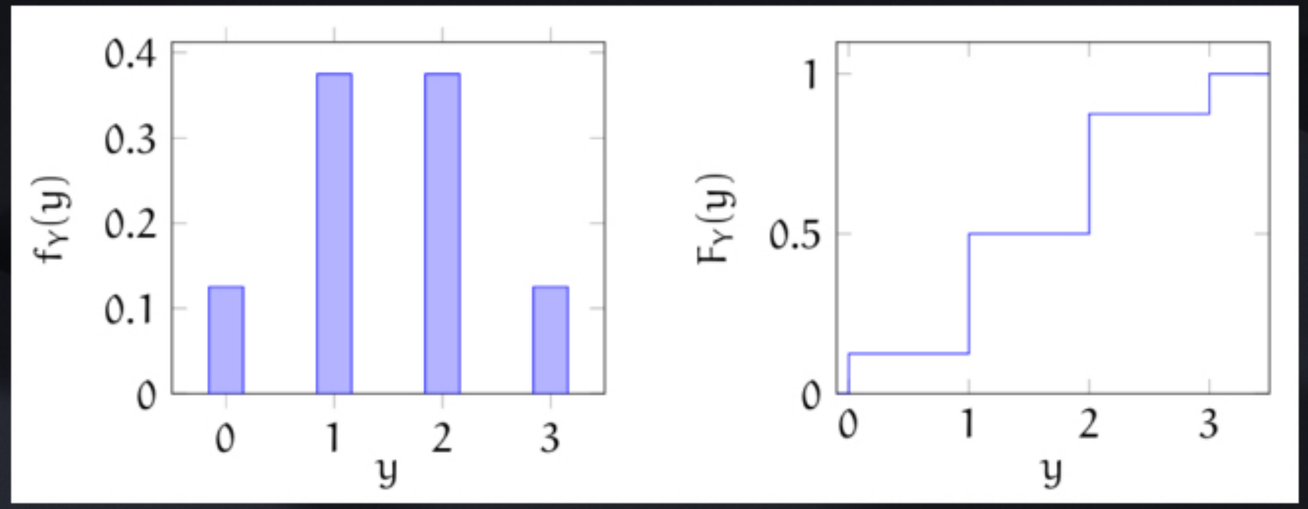
- Ereignisse A und B sind unabhängig dgw. Eintritt von B beainflusst nicht ob A eintritt:
  $Pr[A \cap B] = Pr[A] \cdot Pr[B]$

- Die Ereignisse $A_1, \ldots, A_n$ heissen unabhängig, wenn für alle Teilmengen $I \subseteq \{1,...,n\}$ mit $I = \{i_1, \ldots, i_k\}$ gilt, dass
  $Pr[A_{i_1} \cap \cdots \cap A_{i_k}] = Pr[A_{i_1}] \cdots Pr[A_{i_k}]$.
  Eine unendliche Familie von Ereignissen A_i mit $i \in \mathbb{N}$ heisst unabhängig für jede endliche Teilmenge $I \subseteq \mathbb{N}$ erfüllt ist

- Seien A, b und C unabhängige Ereignisse. Dann sind auch A ∩ B und C bzw. A ∪ B und C unabhängig

Dichte und Verteilung

- $f_x : \mathbb{R} \to [0,1], x \mapsto Pr[X = x]$ Dichte
- $F_X : \mathbb{R} \to [0,1], x \mapsto Pr[X \leq x] = \sum_{x' \in W_x : x' \leq x} Pr[X = x']$ Verteilung



Erwartungswert

- Zu einer Zufallsvariablen X definieren wir den Erwartungswert $\mathbb{E}[X]$ durch:
  $$\mathbb{E}[X] := \sum_{x \in W_X} x \cdot Pr[X = x],$$
  sofern die Summe absolut konvergiert. Ansonsten sagen wir, dass der Erwartungswert undefiniert ist. (Wichtig für unendliche W'keitsräume)
- Ist X eine Zufallsvariable, so gilt: $\mathbb{E}[X] = \sum_{w \in \Omega} X(\omega) \cdot Pr[\omega]$
- Sei X eine Zufallsvariable mit $W_X \subseteq \mathbb{N}_0$. Dann gilt

Linearität des Erwartungswerts

- Für Zufallsvariablen $X_1, \ldots, X_n$ und $X := a_1X_1 + \ldots + a_nX_n + b$ mit $a_1, \ldots, a_n, b \in \mathbb{R}$ gilt:
  $\mathbb{E}[X] = a_1\mathbb{E}[X_1] + \ldots + a_n\mathbb{E}[X_n] + b$

Wir betrachten zwei Zufallsvariablen:

$$X : \Omega \to \mathbb{R}, \qquad Y : \Omega \to \mathbb{R}$$

Summe:

$$X + Y : \quad \Omega \to \mathbb{R}, \quad \omega \mapsto X(\omega) + Y(\omega)$$

$$\begin{aligned}
\mathbb{E}[X + Y] &\overset{\text{Def.}}{=} \sum_{\omega \in \Omega} (X + Y)(\omega) \cdot \Pr[\omega] \\
&= \sum_{\omega \in \Omega} (X(\omega) + Y(\omega)) \cdot \Pr[\omega] \\
&= \sum_{\omega \in \Omega} X(\omega) \cdot \Pr[\omega] + \sum_{\omega \in \Omega} Y(\omega) \cdot \Pr[\omega] \\
&= \mathbb{E}[X] + \mathbb{E}[Y]
\end{aligned}$$

Verteilungen

Bernoulli	Bernoulli(p)	$\{0,1\}$	$f_X(i) = \begin{cases} p & \text{für } i = 1, \\ 1-p & \text{für } i = 0. \end{cases}$	p	$p(1-p)$

- Zufallsexperimente mit Indikatorvariable. Bsp: Münzwurf: Indikator für Zahl.

Binomial	Bin(n,p)	$\{0,1,\ldots,n\}$	$f_X(i) = \binom{n}{i} p^i (1-p)^{n-i}$	np	$np(1-p)$

- Anzahl Bälle in Korb i
- W'keit k Bälle in Korb i

Verteilungen

Poisson	$\mathrm{Po}(\lambda)$	$\mathbb{N}_0$	$f_X(i) = \frac{e^{-\lambda}\lambda^i}{i!}$	λ	λ

- Modelliert seltene Ereignisse
- Anzahl Experimente sehr gross, und Erfolgsw'keit λ/#Versuche

Geometrisch	$\mathrm{Geo}(p)$	$\mathbb{N}$	$f_X(i) = p(1-p)^{i-1}$	$\frac{1}{p}$	$\frac{1-p}{p^2}$

- Warten auf den ersten Erfolg
- W'keit k mal werfen zu müssen

Coupon Collector

- Problem: n verscheidene Coupons und in jeder Runde erhält man eins gleichverteilt
- X := Anzahl Runden bis wir alle Coupons haben
- Ansatz: Problem in Phasen aufteilen: 1 Phase symbolisiert Zeit, von i-1 bis i Coupons gesammelt
- Insgesamt: $\mathbb{E}[X] = \sum_{i=1}^{n} \frac{n}{n-i+1} = n \cdot \sum_{i=1}^{n} \frac{1}{i} = n \cdot H_n$ $\qquad H_n = ln(n) + O(1)$

- Besser sich am Ende Coupons schenkenzu lassen, als am Anfang

Varianz

Definition Für eine Zufallsvariable X mit $\mu = \mathbb{E}[X]$ definieren wir die *Varianz* $\mathrm{Var}[X]$ durch

$$\mathrm{Var}[X] := \mathbb{E}[(X-\mu)^2] = \sum_{x \in W_X} (x-\mu)^2 \cdot \Pr[X = x].$$

Die Grösse $\sigma := \sqrt{\mathrm{Var}[X]}$ heisst *Standardabweichung* von X.

Satz Für eine beliebige Zufallsvariable X gilt

$$\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2.$$

Rechenregeln für Momente

- $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ Stimmt für alle X und Y
- $\mathbb{E}[X \cdot Y] = \mathbb{E}[X] \cdot \mathbb{E}[Y]$ Stimmt, wenn X und Y unabhängig
- $Var[X + Y] = Var[X] + Var[Y]$ Stimmt, wenn X und Y unabhängig
- $Var[X \cdot Y] = Var[X] \cdot Var[Y]$ Stimmt im allgemeinen nicht

Abschätzen von W'keiten

Satz *(Ungleichung von Markov)* Sei X eine Zufallsvariable, die nur nicht-negative Werte annimmt. Dann gilt für alle $t \in \mathbb{R}$ mit $t > 0$, dass

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.$$

Satz *(Ungleichung von Chebyshev)* Sei X eine Zufallsvariable und $t \in \mathbb{R}$ mit $t > 0$. Dann gilt

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\mathrm{Var}[X]}{t^2}.$$

Abschätzen von W'keiten

Satz (Chernoff-Schranken). Seien $X_1, \ldots, X_n$ unabhängige Bernoulli-verteilte Zufallsvariablen mit $\Pr[X_i = 1] = p_i$ and $\Pr[X_i = 0] = 1 - p_i$. Dann gilt für $X := \sum_{i=1}^{n} X_i$:

(i) $\Pr[X \geq (1+\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{3}\delta^2 \mathbb{E}[X]}$ für alle $0 < \delta \leq 1$,

(ii) $\Pr[X \leq (1-\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{2}\delta^2 \mathbb{E}[X]}$ für alle $0 < \delta \leq 1$,

(iii) $\Pr[X \geq t] \leq 2^{-t}$ für $t \geq 2e\mathbb{E}[X]$.

Randomisierte Algorithmen

- deterministisch: selbe Eingabe, selber Output
- nicht-deterministisch: selbe Eingabe, nicht unbedingt selber Output

- Monte Carlo: Korrektheit als Zufallsvariable (dafür immer die selbe Laufzeit)
- Las Vegas: Laufzeit als Zufallsvariable (bei uns gibt es nach gewisser Zeit ??? aus)

- Für Fehlerreduktion lassen wir den Algorithmus jeweils mehrmals laufen

Quicksort

- sortiert Array in erwarteter Laufzeit O(nlog(n))

```
QuickSort(A, ℓ, r)
1: if ℓ < r then
2:     p ← Uniform({ℓ, ℓ + 1, ..., r})        ▷ wähle Pivotelement zufällig
3:     t ← Partition(A, ℓ, r, p)
4:     QuickSort(A, ℓ, t − 1)
5:     QuickSort(A, t + 1, r)
```

Quickselect

- gibt das k-kleinste Element aus. Erwartete Laufzeit O(n)
-

```
QuickSelect(A, ℓ, r, k)
1: p ← Uniform({ℓ, ℓ + 1, ..., r})          ▷ wähle Pivotelement zufällig
2: t ← Partition(A, ℓ, r, p)
3: if t = ℓ + k − 1 then
4:     return A[t]                           ▷ gesuchtes Element ist gefunden
5: else if t > ℓ + k − 1 then
6:     return QuickSelect(A, ℓ, t − 1, k)    ▷ gesuchtes Element ist links
7: else
8:     return QuickSelect(A, t + 1, r, k − t)  ▷ gesuchtes Element ist rechts
```

Primzahltest

- GGT: Pr[“prim” | nicht prim] = $\frac{|\mathbb{Z}_n^*|}{n-1}$
- Fermat: Alle Elemente haben ord(n-1), falls n prim Pr[Prim | nicht Prim] < 1/2
  Wenn Zahl keine Carmichaelzahl
  Pr[“prim” | nicht prim] = $\frac{|PB_n|}{n-1}$
- Miller Rabin: Pr[“prim” | nicht prim] ≤ 1/4
  n-1 = 2^kd

MILLER-RABIN-PRIMZAHLTEST(n)

1: **if** $n = 2$ **then**
2: **return** ‘Primzahl’
3: **else if** n gerade oder $n = 1$ **then**
4: **return** ‘keine Primzahl’
5: Wähle $a \in \{2, 3, \ldots, n-1\}$ zufällig und
6: berechne $k, d \in \mathbb{Z}$ mit $n - 1 = d2^k$ und d ungerade.
7: $x \leftarrow a^d \pmod n$
8: **if** $x = 1$ or $x = n - 1$ **then**
9: **return** ‘Primzahl’
10: **repeat** $k - 1$ mal
11: $x \leftarrow x^2 \pmod n$
12: **if** $x = 1$ **then**
13: **return** ‘keine Primzahl’
14: **if** $x = n - 1$ **then**
15: **return** ‘Primzahl’
16: **return** ‘keine Primzahl’

Lange Pfade

Bunte Pfade

BUNT(G, i)

1: **for all** $v \in V$ **do**
2: $P_i(v) \leftarrow \emptyset$
3: **for all** $x \in N(v)$ **do**
4: **for all** $R \in P_{i-1}(x)$ **mit** $\gamma(v) \notin R$ **do**
5: $P_i(v) \leftarrow P_i(v) \cup \{R \cup \{\gamma(v)\}\}$

- findet alle bunten Pfade der Länge i, basierend auf bunten Pfaden der Länge i-1.
- $O(\binom{k}{i} \cdot i \cdot m)$

Regenbogen(G, γ)

1: **for all** $v \in V$ **do** $P_0(v) \leftarrow \{\{\gamma(v)\}\}$
2: **for** $i = 1..k-1$ **do** Bunt(G, i)
3: **return** $\bigcup_{v \in V} P_{k-1}(v) \neq \emptyset$

findet, ob es einen bunten Pfad der Länge k-1 gibt.

$O(2^k km)$

Flüsse in Netzwerken

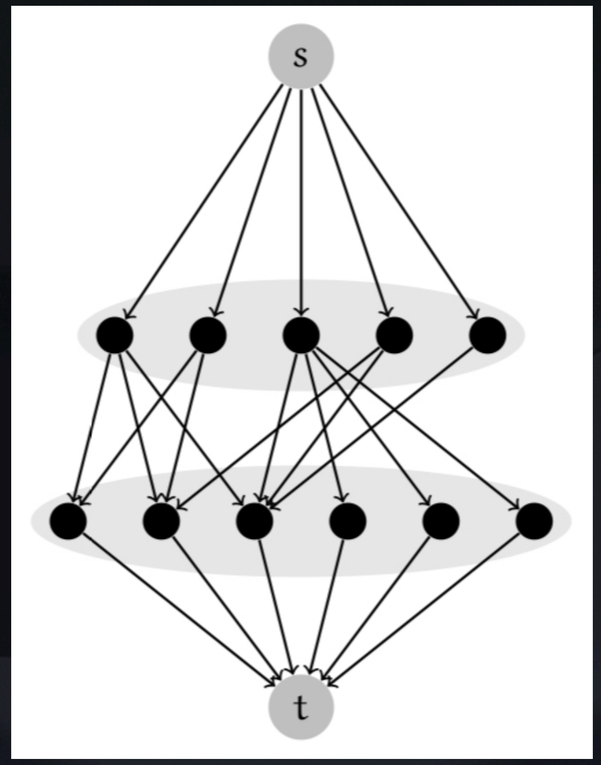
- Sei N = (V,A,c,s,t) euin Netzwerk. Ein Fluss in N ist eine Finktion f : A $\rightarrow \mathbb{R}$ mit den Bedingungen:
  - Zulässigkeit: 0 ≤ f(e) ≤ c(e)
  - Flusserhaltung: Für alle v ∈ V \ {s,t} gilt: $\sum_{u \in V:(u,v) \in A} f(u,v) = \sum_{u \in V:(v,u) \in A} f(v,u)$
- Der Wert eines Flusses f ist definiert als: val(f) := netoutflow(s) := $\sum_{u \in V:(s,u) \in A} f(s,u) - \sum_{u \in V:(u,s) \in A} f(u,s)$

Restnetzwerke

- Sei N = (V,A,c,s,t) ein Netzwerk ohne entgegen gerichtete Kanten und sei f ein Fluss in N. Das restnetzwerk N_f := (V,A_f,r_fs,t) ist wie folgt definiert:
  - Ist e ∈ A mit f(e) < c(e), dann ist e auch iene Kante in A_f, mit $r_f(e)$:= c(e) - f(e)
  - Ist e ∈ A mit f(e) > 0, dann ist e^{opp} in A_f, mit $r_f(e^{opp})$ = f(e)
  - Nur oben beschirebene Kanten sind in A_f
- $r_f(e)$, e ∈ A_f, nennen wir die restkapazität der Kante e

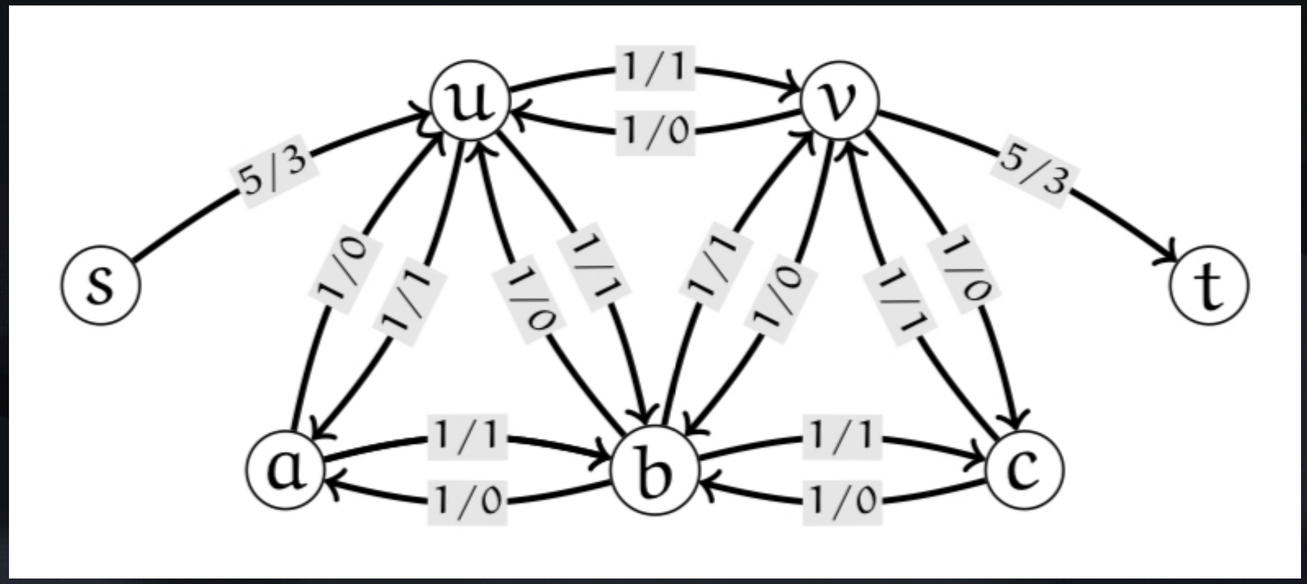
Bipartites Matching

- $A := (\{s\} \times U) \cup \{(u,w) \in U \times W | \{u,w\} \in E \cup (W \times \{t\})$



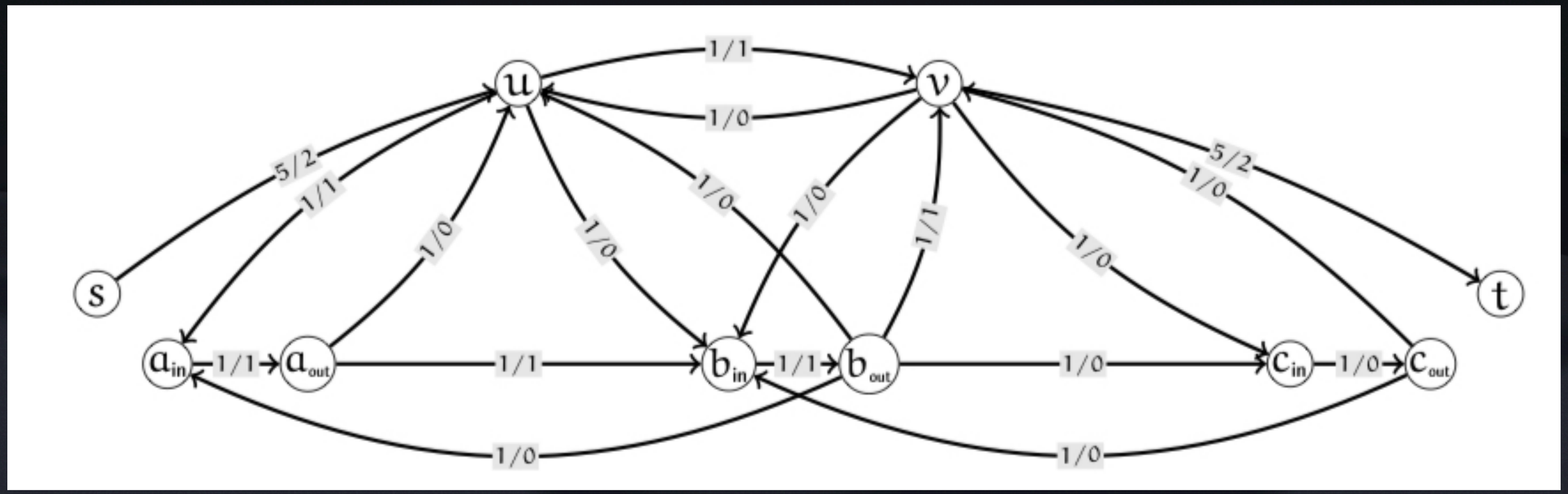
Kantendisjunkte Pfade

- Wir erstetzen jede Kante {x,y} durch zwei gerichtete Kanten (x,y) und (y,x)
- Wir fügen s und t hinzu und verbinden diese mit u und v mit Kapazität = #Knoten



Knotendisjunkte Pfade

- Wir konstruieren das Netzwerk für Kantendisjunkte Pfade
- Wir ersetzen jeden Knoten x ∈ V\{u,v} durch x_{in} und x_{out} und verbinden Eingangskanten mit x_{in} und Ausgangskanten mit x_{out} und eine Kante mit Kapazität 1 zwischen x_{in} und x_{out}



Bildsegmentierung

Netzwerkkonstruktion

- Wir fügen s und t hinzu
- Wir verbinden s (gerichtet) mit allen Pixeln in P mit Kapazität α_p
- Wir verbinden t (gerichtet) mit allen Pixeln in P mit Kapazität β_p
- Benachbarte Pixel in P verbinden wir mit je 2 gerichteten Kanten mit je Kapazität γ_e

Minimale Schnitte in Graphen

Kantenkontraktion

- Problem: Gegeben ein Multigraph G, bestimme $\mu(G)$
- Idee: bei e = {u,v} verschmelzen wir u und v zu einem Knoten mit $deg_{G/e}x_{u,v} = deg_G(u) + deg_G(v) - 2k$
- $\mu(G/e) \geq \mu(G)$ und falls es in G einen minimalen Schnitt C mit $e \notin C$ gibt, dann gilt $\mu(G/e) = \mu(G)$
- Wir kontrahieren randomisiert bis nur noch 2 Knoten übrig sind und returnen diesen Schnitt
  Laufzeit: O(n^2)

Bootstrapping

- **Idee**: Letzte Schritte von rand. Algorithmus sind am gefährlichsten, also diese deterministisch ausführen
- Laufzeit: O(n(n-t) + z(t))
- Erfolgsw'keit $\geq \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdots \frac{t}{t+2} \cdot \frac{t-1}{t+1} \cdot p*(t) = \frac{t(t-1)}{n(n-1)} \cdot p*(t)$
- wir wählen: z(t) = O(t^4) p3(t) = 1-e^-1 und erhalten für t = n^1/2 eine Laufzeit: O(λn^3)

Kleinster Umschliessender Kreis

Algorithmus

RANDOMISED_PRIMITIVEVERSION(P)

```
1: repeat forever
2:     wähle Q ⊆ P mit |Q| = 3 zufällig und gleichverteilt
3:     bestimme C(Q)
4:     if P ⊆ C•(Q) then
5:         return C(Q)
```

- Laufzeit: T ~ Geo$(1/\binom{n}{3}) \Rightarrow O(n \cdot n^3)$

Kleinster Umschliessender Kreis

Algorithmus

```
RANDOMISED_CLEVERVERSION(P)
1: repeat forever
2:     wähle Q ⊆ P mit |Q| = 11 zufällig und gleichverteil
3:     bestimme C(Q)
4:     if P ⊆ C•(Q) then
5:         return C(Q)
6:     verdoppele alle Punkte von P ausserhalb von C(Q)
```

- Laufzeit: $O(n \cdot T) = O(n \cdot log(n))$

Konvexe Hülle

Grundbegriffe

- **Gegeben**: Punktemenge P $\subset \mathbb{R}^d$
- **Gesucht**: konvexe Hülle von P, conv(P) als Folge $(q_0, q_1, \ldots, q_{h-1}), h \leq n$ Ecken des Polygons beginnend bei beliebiger Ecke, dann gegen Uhrzeigersinn
- **Konvexe Hülle**:= minimale konvexe Menge, die P beinhaltet
- **Randkante**:= geordnetes Paar qr für $q, r \in P$ mit $q \neq r$ s.d. alle Punkte von P links von qr liegen
- **Allgemeine Lage**:= keine drei Punkte auf einer Geraden und keine zwei Punkte die gleiche x-Koordinate

Konvexe Hülle

Jarvis Wrap

- Laufzeit: O(n*h)
- **Spezialfälle**:
  - 2 Punkte mit kleinster x-Koordinate
    Wir wählen den mit kleinerer y-Koordinate.
  - 3 oder mehr Punkte auf einer Linie
    Wir machen $(det(p, q, q_{next}) < 0) \vee (det(p, q, q_{next}) = 0 \wedge |qp| > |qq_{next}|)$

```
JarvisWrap(P)
1: h ← 0
2: p_now ← Punkt in P mit kleinster x-Koordinate
3: repeat
4:     q_h ← p_now
5:     p_now ← FindNext(q_h)
6:     h ← h + 1
7: until p_now = q_0
8: return (q_0, q_1, ..., q_{h-1})
```

Konvexe Hülle

Local repair

- **Verbesserungsschritt**:= falls p,p',p'' aufeinanderfolgende Punkte und p' links von pp'' liegt, entfernen wir p' aus der Folge
- **Laufzeit**: O(n*log(n))

LocalRepair$(p_1, p_2, \ldots, p_n)$

▷ setzt $(p_1, p_2, \ldots, p_n)$, $n \geq 3$, nach x-Koordinate sortiert, voraus

1: $q_0 \leftarrow p_1$
2: $h \leftarrow 0$
3: **for** $i \leftarrow 2$ **to** n **do** ▷ untere konvexe Hülle, links nach rechts
4: **while** $h > 0$ und q_h links von $q_{h-1}p_i$ **do**
5: $h \leftarrow h - 1$
6: $h \leftarrow h + 1$
7: $q_h \leftarrow p_i$ ▷ $(q_0, \ldots, q_h)$ untere konvexe Hülle von $\{p_1, \ldots, p_i\}$
8: $h' \leftarrow h$
9: **for** $i \leftarrow n - 1$ **downto** 1 **do** ▷ obere konvexe Hülle, rechts nach links
10: **while** $h > h'$ und q_h links von $q_{h-1}p_i$ **do**
11: $h \leftarrow h - 1$
12: $h \leftarrow h + 1$
13: $q_h \leftarrow p_i$
14: **return** $(q_0, q_1, \ldots, q_{h-1})$ ▷ Ecken der konvexen Hülle, gg. Uhrzeigersinn

Konvexe Hülle

Ebene Graphen

- **Ebener Graph**:= Kanten schneiden sich höchstens in Endpunkten
- **Triangulierung**:= man kann keine Kante hinzufügen, ohne dass sich Kanten kreuzen
- **#Verbesserungsschritte** = 2n-2-h
  Wir beginnen mit 2n-2 Kanten, in jedem Schritt machen wir aus 2 Katnten 1. h Kanten bleiben
- **#Dreiecke der Triangulierung** = 2n-2-h
  Wir beginnen mit 0 Dreiecken. Mit jedem Schritt entsteht 1
- **#Kanten der Triangulierung** = 3n-3-h
  Am Anfang n-1 Kanten (nicht doppelt gezählt) in jedem Schritt + 1