WS2010/2011
BAI3-GKA
Klausurspicker

R. C. Ladiges

26. Januar 2011

s t — **adja/inzi** — **schlicht Multi parallel einfach Isomorph Teil/Unter** — **d, Δ, δ** — **Kn** — **bipartit, planar** — **L**

$G=(V,E)$ Vertex, Edges | $s\_t: E \to P(V)$ $s\_t(e)=\{v,u\}$ | $s,t: E \to V$ source, target | Schlinge $s(e)=t(e)$

adjazent: 2 verbundene V | inzident: V mit E verbunden | Mehrfachkanten: $s(e_1)=s(e_2) \wedge t(e_1)=t(e_2)$

parallele Kante: $s\_t(e_1)=s\_t(e_2)$ | Multigraph: mit mehrfach und parallel | schlichter Graph: mit parallel ohne mehrfach ohne Schlingen

einfacher Graph: schlicht ohne parallel | $H=(W\subseteq V, F\subseteq E)$ | Isomorph: $G \cong H$ bijektive Abbild. $V \to W$ und $F \to E$

Teilgraph: $H \subseteq G$ V/E beliebig entfernen | Untergraph: $H \sqsubseteq G$ nur V entfernen | Knotengrad: $d(v)$ verbundene Kanten (Schlingen doppelt)

$d_-(v)$ von v | $d_+(v)$ nach v | Maximalgrad: $\Delta(G)=max(d(v))$ | Minimalgrad: $\delta(G)=min(d(v))$ | $G_\emptyset$ leerer Graph $E=\emptyset$

K-regulär: $\forall v\in V: d(v)=K$ | $\sum_{v\in V} d(v) = 2\cdot|E|$ | vollständiger Graph: K_n alle V mit V | $K_n \to (|E|=n(n-1)/2$

bipartit: $K_{n,m}$ mit $n=|X\subseteq V|$, $m=|Y\subseteq V|$ X, Y disjunkt 2-färbbar | planar: Kreuzungsfrei zeichenbar

gewichteter Graph: Kantenbewertung/Länge $L: E\to \mathbb{R}$ | erreichbar: v_1 von v_2 erreichbar, wenn $v_1=v_2$ oder ∃ Kantenfolge

A(G) — **M(G)** — **N_G**

Adjazenzmatrix $|V|\times|V|$ $A(G)=(a_{ij})$ | unger: $a_{ij}=|\{e|s\_t(e)=\{i,j\}\}|$ | ger: $a_{ij}=|\{e|s(e)=i \wedge t(e)=j\}|$

Inzidenzmatrix $|V|\times|E|$ $M(G)=(m_{ij})$ | unger: 0 unverbunden, 1 inzident, 2 Schlinge | ger: 0 -"-, -1 von v, +1 nach v

$\sum_j^{|E|} |m_{ij}| = \sum_j^{|V|} a_{ij} = d(i)$ | $\sum_i^{|V|} |m_{ij}| = 2$ $(\forall j)$ | Nachbarschaft: $N_G(W\subseteq V)$ die mit w verbundenen V | abgeschlossene -"-: $cN_G(W)=N_G(W)\cup W$

Kantenfolge Weg, Kreis — **G^+** — **Schnitt**

unger./ger. Nachbarschaftsliste: Nachbarn eines v / $\{u|\exists e: s(e)=v \wedge t(e)=u\}$ | Kantenfolge: $v0e1v1\dots ekvk$ mit $0<K$ $i=1,\dots,K$ $s\_t(e_i)=\{v_{i-1},v_i\}$

Weg: Kantenfolge mit v und e verschieden | Kreis: Weg mit $v_0=v_K$ 1,…,K verschieden ← geschlossene Kantenfolge: $v0=vK$

G hat Kreis, wenn $\forall v\in V: d_-(v)>0$ oder $\forall v\in V: d_+(v)>0$ | $A(G)^K$: a_{ij} = Anzahl Kantenfolgen von $v_i \to v_j$ mit Länge K

transitive Hülle: $G^+=(V,E^+)$ | Zusammenhang: stark/schwach = $Weg_{ij} \wedge Weg_{ji}$ / $Weg_{ij} \vee Weg_{ji}$ | Komponenten: $\sqsubseteq G$ entsprechend

G ist zusammenhängend, wenn G^+ vollständig (K_n) | Schnittknoten | Schnittkante | ← ohne nicht mehr zusammenhängend

Kürzester Weg — **BFS**

BFS - Breadth First Search: ① $\forall v\in V$ do$\{\lambda(v)=-1;\}$; $\lambda(s)=0$; int $i=0$; (markierung) ② $\forall v\in V: \lambda(v)==-1 \wedge v\in N_G(\{u|\lambda(u)==i\})$ do$\{\lambda(v)=i+1;\}$;
③ if$(\lambda(t)==-1)$ {goto 4;} else{i++; goto 2;} ④ |kürzester Weg| == i+1

Dijkstra

Dijkstra (schlicht, zusammenhä., gerichtet) Entf(i) = bisher min Entfernung $v_0 \to v_i$ Vorg(i) = Vorgänger Ok(i) = min bekannt
(init) $\forall v\in V$ do{ Entf(v)=∞; Vorg(v) = null; Ok(v)=false;}; Entf(v_0)=0; Vorg(v_0)=v_0;
(do) while $(\exists v\in V: \neg Ok(v))$ {① suche vh mit ¬Ok(v) und min(Entf(v_0)); ② Ok(vh)=true;
③ $\forall v_j \in V: \neg Ok(v_j) \wedge e_{hj}\in E \wedge Entf(v_j) > Entf(vh) + L(e_{hj})$ do {Entf(v_j)=Entf(vh)+L(e_{hj}); Vorg(v_j)=vh;}}

planar Minor — **Polyederformel**

$K_{3,3}$ und K_5 sind nicht planar. | Minoren von planaren G sind planar | G planar → nicht K_5 oder $K_{3,3}$ als Minor

Minoren: Edge verschmelzen ① rausnehmen ② s und t vereinigen (kontrahieren, Mehrfachkanten raus.

Eulersche Polyederformel (für planar): |V| − |E| + |Flächen| = 1 + |Komponenten| für zusammenhängend: = 2

Färbung χ(G) — **K4=** **C4=** **W4=**

Färbung (coloring): $c: V\to \mathbb{N}_0$ | v adjazent u → $c(v)\neq c(u)$ | K-färbbarer G: $\exists c: V\to\{1,2,\dots,K\}$ für G

chromatische Zahl: χ(G) minimale Anzahl benötigter Farben | $\chi(G)\leq\Delta(G)+1$ | G planar → $\chi(G)\leq 4$ (≤ 5)

$\chi(G_\emptyset)=1$ | $H\sqsubseteq G \to \chi(H)\leq\chi(G)$ | G bipartit ↔ $\chi(G)=2$ ↔ G hat keine Kreise ungerader Länge

$\chi(K_n)=n$ | $\chi(C_n)=2+n\%2$ (Kreis mit |V|=n) | $\chi(W_n)=3+n\%2$ (Wagenrad |V|=n+1, 1…n=Kreis, n+1=mitte)

Konfliktgraph: χ(G) = min(|Gruppierungen|) damit keine Konflikte auftreten

Greedy: $v_1, v_2, \dots, v_n$ links nach rechts: $c(v_i)=min(\mathbb{N}_0 \setminus \{c(v_j)|v_j\in N_G(v_i)\})$ | BFS Breitensuche Färbung

Color First: int c = 0; while$(\exists v\in V: \neg c(v))$ {c++; $\forall v\in V: \neg c(v) \wedge \nexists u\in N_G(v): c(u)=c$ do{c(v)=c;};}

Baum — **a(v,v) L(G) H(G)**

DAG: gerichteter kreisloser (azyklisch) Graph | Baum: kreislos zusammenhängend |E|=|V|−1 nur Schnittkanten

Wald: Bäume als Komponenten | Blatt/Leaf: $d_-(v)=1 \wedge d_+(v)=0$ G/blatt bleibt Baum | $\forall v\in$ Baum: $d_-(v)\leq 1$

Abstand $a(v_1,v_2)$ Länge des Weges $v_1\to v_2$ | Länge: $L(G)=max(a(root, v\in V))$ | Höhe: $H(G)=L(G)-1$ H(root)=0

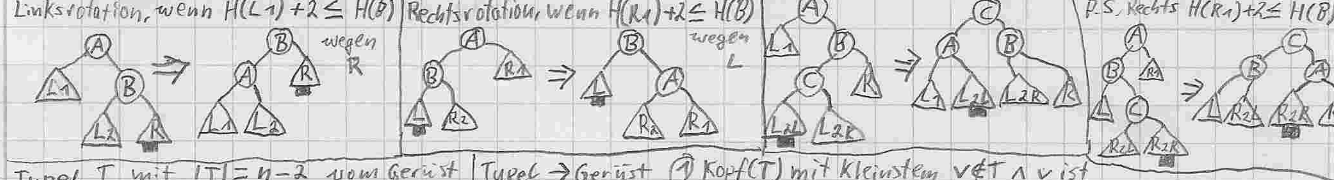
gerichteter Wurzelbaum: alle V von root erreichbar | balanciert: a(root, beliebiges Blatt)=konstant (alle Knoten genau 2 children)

Binärbaum $max.(|V|)=2^{H(G)}-1$ | Spannbaum/Gerüst $G\sqsupseteq H=(W=V, F\subseteq E)$ | T(G) = |verschiedene Gerüste|

AVL

AVL-Bäume (balancierten Suchbaum aufbauen) | Prob. Situ. Links $H(L_1)+2\leq H(B)$ | $T(K_n)=n^{n-2}$

Linksrotation, wenn $H(L_1)+2\leq H(B)$ | Rechtsrotation, wenn $H(R_1)+2\leq H(B)$ | P.S. Rechts $H(R_1)+2\leq H(B)$



Gerüst Tupel

Tupel T mit |T| = n−2 vom Gerüst | Tupel → Gerüst ① Kopf(T) mit kleinstem v∉T ∧ v ist unverwendet verbinden; ② Kopf(T) aus T entfernen; ③ if(|T|>0) goto 1; ④ unverwendete verbinden;

Gerüst → Tupel ① v_i mit min(i) und $d(v_i)=1$ ② v_i entfernen; den mit v_i verbundenen Vertex ins Tupel ③ if(|T| < n−2) goto 1;

$T(G)=det(\tilde{M}\cdot\tilde{M}^T)$ | $\tilde{M}$ = beliebige Zeile von M(G') weglassen G' = gerichteter G

determinante: det(Matrix) = ↘ mult, addition ↙ mult, subtraktion | $det\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix} = {+(a_{11}\cdot a_{22}) + (a_{12}\cdot a_{21}) \atop -(a_{12}\cdot a_{21}) - (a_{11}\cdot a_{22})} = 4+6-6-4=0$

Kruskal: MST (Minimum Spanning Tree) finden ① F=∅; E nach Länge sortieren; ② ∀e∈E: F∪{e} ist Baum do {F=F∪{e};};

Flüsse O(v) I(v) — **Schnitt**

q=Quelle, s=Senke mit $d_+(q)=0=d_-(s)$ | Output: $O(v_i)=\{e_{ij}|s(e_{ij})=v_i\}$ | Input: $I(v_j)=\{e_{ij}|t(e_{ij})=v_j\}$

$|O(v_i)|=d_-(v_i)$ $|I(v_i)|=d_+(v_i)$ | A(X,Y) = E zwischen X und Y ⊆ V | $A^+(X,Y)$ = E von X → Y | $A^-(X,Y)=A^+(Y,X)$

Schnitt: $A(X, V\setminus X)\subseteq E$ mit $q\in X \wedge s\in V\setminus X$ | 2^{n-1} Schnitte möglich | Kapazität: $c(e_{ij})=c_{ij}: E\to\mathbb{R}^+$ | Kapazitäts-

$c(X, V\setminus X)=\sum[e\in A^+(X,V\setminus X)]c(e)$ | Fluss: von q nach s Funktion $f: E\to\mathbb{R}^+$ | $\forall e\in E: f(e)\leq c(e)$ beschränkung

$\sum_{e\in O(q)} f(e) = \sum_{e\in I(s)} f(e) = d$ = Flusswert $= f(X,V\setminus X) - f(V\setminus X, X) \leq c(X, V\setminus X)$ | $\forall v\neq q\neq s: \sum_{e\in I(v)} f(e) = \sum_{e\in O(v)} f(e)$ Flusserhaltung

Maximaler Fluss = d maximal

vergrößernder Weg: von q nach s aus Vor- und Rückkanten $f(vor) < c(vor)$ $f(rück) > 0$ | Fluss Vergrößern: ① $\delta > 0$ kleinste differenz $c(vor) - f(vor)$, $f(rück) - 0$ ② $\forall vor$ do $f(vor) += \delta$; ③ $\forall rück$ do $f(rück) -= \delta$ **Flüsse**

Inkrement zu Fluss: $i(W \subseteq V) = \min(\{i(e) \mid e \in W\})$ $i(vor) = c(vor) - f(vor)$, $i(rück) = f(rück)$ W: Kantenfolge

Ford-Fulkerson mark = (vorgänger, δ) ① $\forall e \in E$ do$\{f(e) = 0;\}$ mark(q) = (undef, ∞); **Ford-Fulkerson**

② $\exists v_i \in V: \neg mark(v_i) \wedge \neg insiz(v_i)$ do$\{inspiz(v_i) = true;$ 2a; 2b;$\}$ 2c; 2d; 3; 4;

(2a) $\forall e \in O(v_i): v_i \neq v_j \in s\_t(e) \wedge \neg mark(v_j) \wedge f(e) < c(e)$ do$\{\delta_j = \min(\delta_i, c(e) - f(e));$ mark$(v_j) = (+v_i, \delta_j);\}$

(2b) $\forall e \in I(v_i): v_i \neq v_j \in s\_t(e) \wedge \neg mark(v_j) \wedge f(e) > 0$ do$\{\delta_j = \min(\delta_i, f(e));$ mark$(v_j) = (-v_i, \delta_j);\}$

(2c) if(alle markierten V wurden inspiziert)$\{$goto 4;$\}$ (2d) if($\neg$inspiz(s))$\{$goto 2;$\}$else$\{$goto 3$\}$

③ Anhand markierung. vorgänger von s nach q durchlaufen und: $f(vor) += \delta s$; $f(rück) -= \delta s$; marks weg; goto 2;

④ kein weiterer vergrößernder Weg. d ist Optimal

Paarung/Matching: $M \subseteq E$ mit $\forall e_1, e_2 \in M: s\_t(e_1) \cap s\_t(e_2) = \emptyset$ | maximale Paarung: $M \cup \{$irgendein $e \in E\}$ keine Paarung mehr **Paarung/Matching**

größte Paarung: $\nexists M': |M'| > |M|$ | Paarungszahl: $|M|$ | perfekte Paarung: $V = \{v \mid v \in s\_t(e \in M)\}$ | größte gewichtete Paarung: $\max(\sum_{e \in M} L(e))$

größte Paarung finden (bipartit): $V' = V \cup \{q, s\}$; $E' = E \cup \{e_{qx} \mid x \in X \subseteq V\} \cup \{e_{ys} \mid y \in V \setminus X\}$; $\forall e \in E'$ do$\{$nach s richten; $c(e) = 1$; $f(e) = 0\}$; Ford Fulkerson;

Euler: alle Edges | Hammilton: alle Vertexe | Eulertour: Kreis mit allen Edges | $\exists$ Eulertour gdw. $\forall v \in V: d(v) \% 2 = 0$

Eulerpfad: Weg mit allen Edges | $\exists$ Eulerpfad gdw. $\exists! v_1, v_2 \in V: d(v_1) \% 2 \neq 0 \wedge d(v_2) \% 2 \neq 0$ | $\hookrightarrow$ G ist eulersch

Fleury (Eulertour finden) ① $W_0 = v_0$ (beliebig); $i = 0$; ② e_{i+1} wählen mit $e_{i+1} \notin W_i = v_0 e_1 v_1 \dots e_i v_i \wedge v_i \in s\_t(e_{i+1})$; **Fleury**

$W_{i+1} = W_i, e_{i+1}, v_{i+1}$; $i{+}{+}$; ③ Backtracking (TS) $i{-}{-}$, beende wenn W_i alle Edges enthält

Edmonds Algo (Briefträger-Problem): ① $U = \{v \in V \mid d(v)$ ungerade$\}$; alle in U verbinden: $L(e_{ij}) = L(\min(Weg_{ij}))$;

② konstruiere Graph $K = (U, P(U))$ $L_K(e_{ij}) = \max - L_G(e_{ij})$; max = hinreichend große Zahl.

③ Paarung M mit $\max(\sum_{e \in E} L(e))$ in K finden; ④ $\forall e \in M: e \in E_G$ verdoppeln ⑤ G jetzt Eulersch

MST-Heuristik (Hamiltonkreis finden): ① min. Gerüst berechnen ② Edges von Gerüst verdoppeln ③ Eulertour übers Gerüst

③ wenn in Eulertour Vertex doppelt besucht $\rightarrow$ abkürzen direkte Edge

markierter Graph $G = (V, E, s, t, l, m)$ $C_E \xleftarrow{m} E \overset{s}{\underset{t}{\rightrightarrows}} V \xrightarrow{l} C_V$ | Graphmorphismus $f: G \to H = \langle f_V, f_E \rangle$

$$\begin{tikzcd} & E_G \arrow[dl, "m_G"'] \arrow[r, shift left, "s_G"] \arrow[r, shift right, "t_G"'] & V_G \arrow[dr, "l_G"] & \\ C_E & & & C_V \\ & E_H \arrow[ul, "m_H"] \arrow[r, shift left, "s_H"] \arrow[r, shift right, "t_H"'] & V_H \arrow[ur, "l_H"'] & \end{tikzcd}$$

Bewahrung von: Quelle, Ziel, Markierung **Graphmorphismus**

$f = g$ wenn $\forall v, e: f(v) = g(v) \wedge f(e) = g(e)$ | abstrakter G: $[G] = \{G' \mid G \cong G'\}$ | f inj/sur/bij wenn f_V und f_E $\rightarrow$ Isomorphismus

regelbasierte Graphersetzung: Regel $r = \langle L \supseteq K \subseteq R \rangle$ | zu löschen: $L - K$ | hinzufügen: $R - K$ | erhalten: K **Graphersetzung**

$L \supseteq K \subseteq R$ nur K darf mit Außen verbunden sein. | Direkte Ableitung: $G \xRightarrow{(r,g)} M$ | Ableitung: wenn $G \cong M$ oder Folge direkter Ableitungen **Regeln**

$$\begin{tikzcd} L \arrow[d, "g"] & K \arrow[l, phantom, "\supseteq"] \arrow[d, "d"] & R \arrow[l, phantom, "\subseteq"] \arrow[d, "h"] \\ G & D \arrow[l, phantom, "\supseteq"] & H \arrow[l, phantom, "\subseteq"] \arrow[r] & M \end{tikzcd}$$

① löschen: $D = G - g(L - K)$ ② hinzufügen: $H = D + (R - K)$ **Ableitungen**

Graphersetzungssystem: $\langle C, \mathcal{R} \rangle = \langle \langle C_V, C_E \rangle, \{r_1, r_2, \dots, r_n\} \rangle$ Markierungsalphabet und Regeln

TypGraph TG: bsp.: ⑧ $\rightarrow$ ⓐ | NAC: negative Anwendungsbedingung $\nexists n: L \to N$ **Graphgrammatik**

Graphgrammatik: $\langle C, N, \mathcal{R}, S \rangle$ $N \subseteq C$ = Nonterminale S = Startgraph | Turingmaschine $M = (Q, T, \delta, q_0, \square, F)$ **Turing**

Q = Zustandsmenge | T = Bandalphabet mit $\square \in T$ | $\square$: Endesymbol | F = Endzustände | Prog $\delta \subseteq Q \times T \times Q' \times T' \times p \in \{L, R, S\}$ Bewegung

Regeln sind Umkehrbar: löschen/verkleben invers zueinander | Ableitungen sequentiell Unabhängig wenn: $R_1 \cap L_2 \subseteq K_1 \cap K_2$ **Umkehrbar**

S/T-Netz: $N = (P, T, W, K)$ | P = Stellen ◯ ⦾ Token | Ableitungen parallel Unabhängig wenn: $L_1 \cap L_2 \subseteq K_1 \cap K_2$ **Unabhängig**

T = Transitionen ▭ | $W: P \times T \to \mathbb{N}_0$, $W: T \times P \to \mathbb{N}_0$ Gewichtsfunktion (wieviele zum schalten / wieviele bekommt p durchs schalten) | Kapazitätsfunktion $K: P \to \mathbb{N}_0 \cup \{\infty\}$ wieviele Tokens M kann P "halten" **Petrinetz**

$T \to P_1, P_2, \dots$ alle P bekommen Token **S/T-Netz**

$P \to T_1, T_2, \dots$ nur ein T schaltet nichtdeterministisch | Vorbereich: $\bullet t = \{p \mid W(p, t) > 0\}$ Nachbereich: $t \bullet = \{p \mid W(t, p) > 0\}$ | Markierung: $M: P \to \mathbb{N}_0$ mit $M(p) \leq K(p)$

$t \in T$ ist aktiviert, wenn ① $\forall p \in \bullet t: W(p, t) \leq M(p)$ ② $\forall p \in t \bullet: M(p) + W(t, p) \leq K(p)$

Folgemarkierung: Schaltschritt: $M \to M'$ bzw. $M[t> M'$ $\forall p \in P: M'(p) = M(p) - W(p, t) + W(t, p)$

Informationen zur Signatur

	**Unterzeichner**	EMAILADDRESS=robin.ladiges@haw-hamburg.de, CN=Robin Christopher Ladiges
	Datum/Zeit	Wed Jan 26 18:34:23 CET 2011
	Austeller-Zertifikat	CN=CAcert Class 3 Root, OU=http://www.CAcert.org, O=CAcert Inc.
	Serien-Nr.	44727
	Methode	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signatur)