

Ergänzungen und  
Korrekturen zu  
Theoretischem Informatik I.

Gerad im Skript zu  
ergänzen

# Inhaltsverzeichnis

1. Grundlagen: Gerichtete Graphen
2. Grundlagen: Ungerichtete Graphen
3. Zeit und Platz
4. Tiefensuche in gerichteten Graphen
5. Anwendung Tiefensuche: Starke Zusammenhangskomponenten.
6. Tiefensuche in ungerichteten Graphen: Zweifache Zusammenhangskomponenten
7. Minimales Spannbauwerk und Datenstrukturen
8. Kürzester Weg
9. Flüsse in Netzwerken
10. Kombinatorische Suche und Rekonstruktionsalgorithmen.

11. Divide-and-conquer und

Rekurrenzgleichungen

12. Weitere Beispiele zum dynamischen

Programmieren

13. Partitionen (eine unteere Schranke)

Ergänzungen und Korrekturen.

In 8.1.14:

Letzte Zeile: Die Größe  
des Arrays ist nicht unbedingt  
gleich der Knotenzahl, aber  
eindeutig  $\leq |E| + |V|$ . Hier  
ist  $A[1..6]$  falsch, was haben  
wir Beispiel  $A[1..7]$ . Die  
Größe des Arrays ist dann  
 $|E| + \# \text{ Knoten von } A\text{-Graph } \mathcal{G}$ .

Hinte:  $|E| \geq \frac{n}{2}$  statt

$|E| \geq n-1$ . Bei  $|E| < \frac{n}{2}$  haben

wir <sup>einige</sup> isolierte <sup>Knoten</sup> Knoten, was  
i. a. nicht sinnvoll ist.

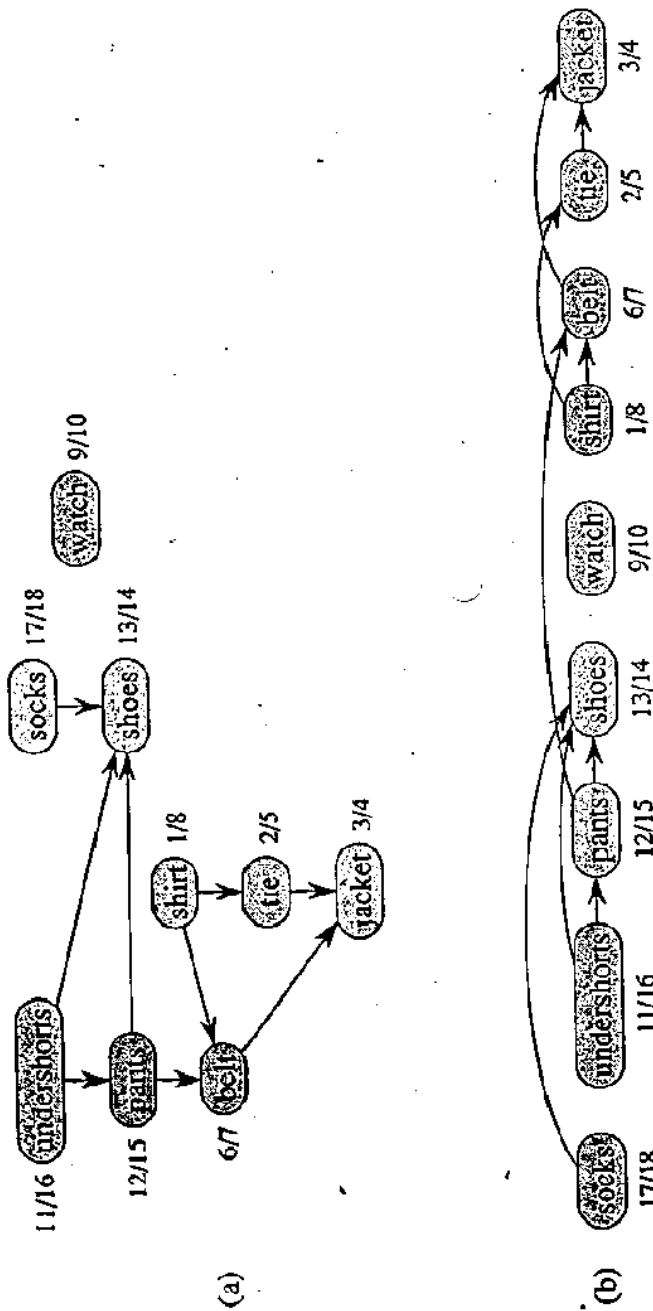
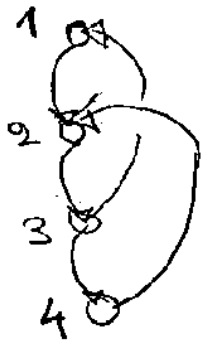


Figure 23.7 (a) Professor Bumstead topologically sorts his clothing when getting dressed. Each directed edge  $(u, v)$  means that garment  $u$  must be put on before garment  $v$ . The discovery and finishing times from a depth-first search are shown next to each vertex. (b) The same graph shown topologically sorted. Its vertices are arranged from left to right in order of decreasing finishing time. Note that all directed edges go from left to right.

Zu §. 4.27: In dem Komplex  
 $\mathcal{K}$  zweimal weiß durch  
 grau zu ersetzen.

Zu §. 5.4. Man beachte auch noch  
 folgendes Beispiel.



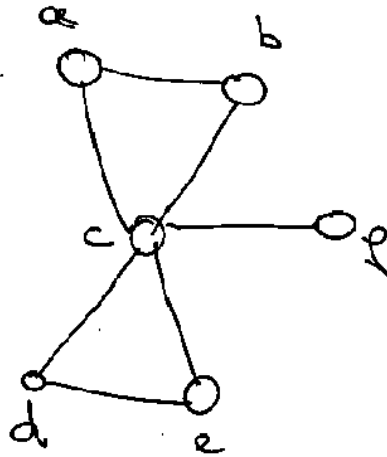
Sei stark zshg. Die Wege

$1 \rightarrow 4$  und  $4 \rightarrow 1$  sind

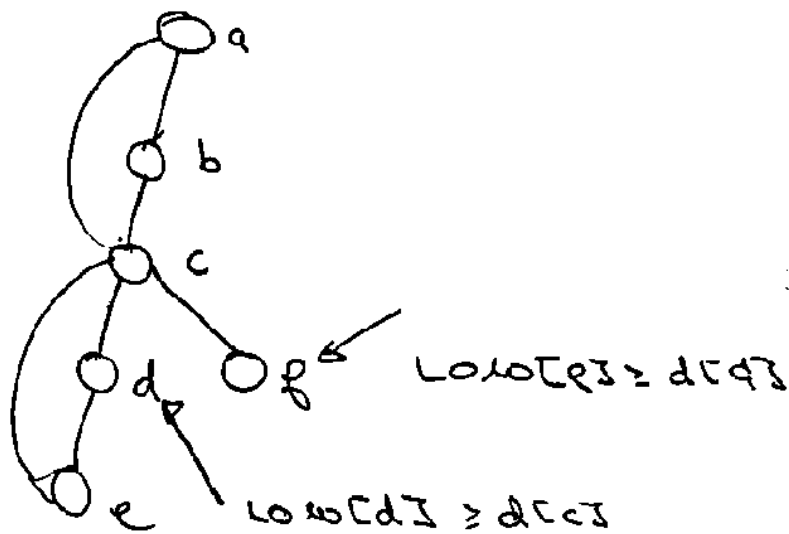
nicht kommutativ. In beiden  
 kommt  $2 \rightarrow 3$  vor.

WIE FINDET MAN DIE

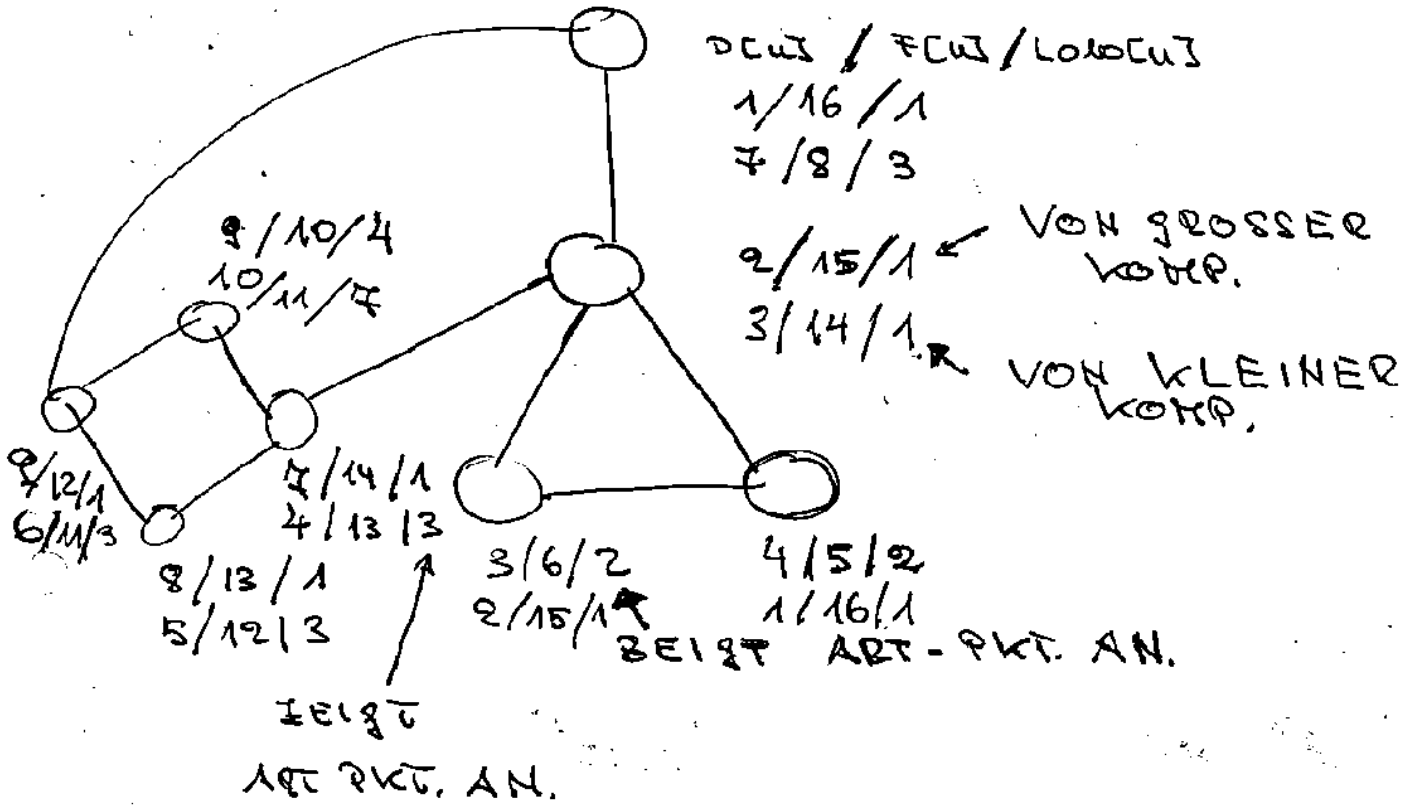
ZWEIFACHEN KOMPONENTEN



MÖGLICHER BAUM:

DFS-VISIT (f) FERTIG  $\Rightarrow$  1. KOMPO.DFS-VISIT (d) FERTIG  $\Rightarrow$  2. KOMPO.DFS-VISIT (a) FERTIG  $\Rightarrow$  3. KOMPO.

LOW-WERT UND DURCHLAUFREIHENFOLGE



Zu §. 8.21

Die Rechnung in der Vorlesung muß  
folgendermaßen lauten:

$$\frac{2^{O(m)}}{2^{\Omega(m \cdot \log m)}} = \frac{2^{\frac{1}{2} \Omega(m \cdot \log m) - O(m)}}{2^{\frac{1}{2} \Omega(m \cdot \log m)}} \geq \frac{1}{2^{\frac{1}{2} \Omega(m \cdot \log m) - O(m)}} \geq \frac{1}{2^{\frac{1}{2} \Omega(m \cdot \log m)}} \geq \frac{1}{2^{c \cdot m}}$$

$\geq \frac{1}{2^{\frac{1}{2} \Omega(m \cdot \log m) - O(m)}}$   
 $\geq \frac{1}{2^{\frac{1}{2} \Omega(m \cdot \log m)}}$   
 $\geq \frac{1}{2^{c \cdot m}}$  (mit  $c$  = andere Konstante.)

also: # verschiedener Aufrufe ist  
max ein verschwindend kleiner Anteil  
an allen Aufrufen!

Das

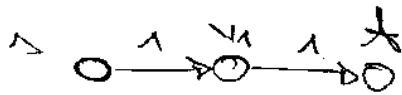
$$\frac{2^{O(m)}}{2^{\Omega(m \cdot \log m)}} = \frac{1}{2^{\Omega(\log m)}} \text{ ist falsch!} \blacktriangleright$$

Seite 9.5 insgesamt neu.

9.5

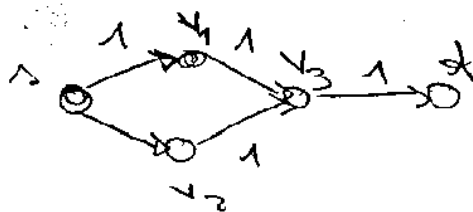
3.  $\epsilon$ -höhe den Fluß entlang des Weges  
soweit es geht. Danach bei 2. weiter.

Dieses Problem ist vom ganz anderen  
Charakter als bisher, da die Anzahl  
des (zunächst) zulässigen Flüsse unendlich  
ist. Ganz einfach zu



Set.  $f(s, v_1) = f(v_1, t) < 1$  so haben

wir einen erlaubten Fluß. Aber sogar  
maximale Flüsse gibt es unendlich  
viele:



Jeder Fluß mit  $f(s, v_1) + f(s, v_2) = 1$   
ist maximal.

Zu 8.811

Erste Zeile: Es ist  $g_f$  mit der

Kapazität  $\kappa_f$  ein Flussnetzwerk.

Rechte  $\kappa_f(u,v)$  ist für alle Paare  
 $(u,v) \in V \times V$  definiert.

Zu 8.9.21

Termination zunächst nur

bei ganzzahligen Kapazitäten.

Erhöhung jedesmal um eine

ganze Zahl  $\geq 1$ . Also ist irgendwann

ein maximaler Fluss erreicht!

9.15

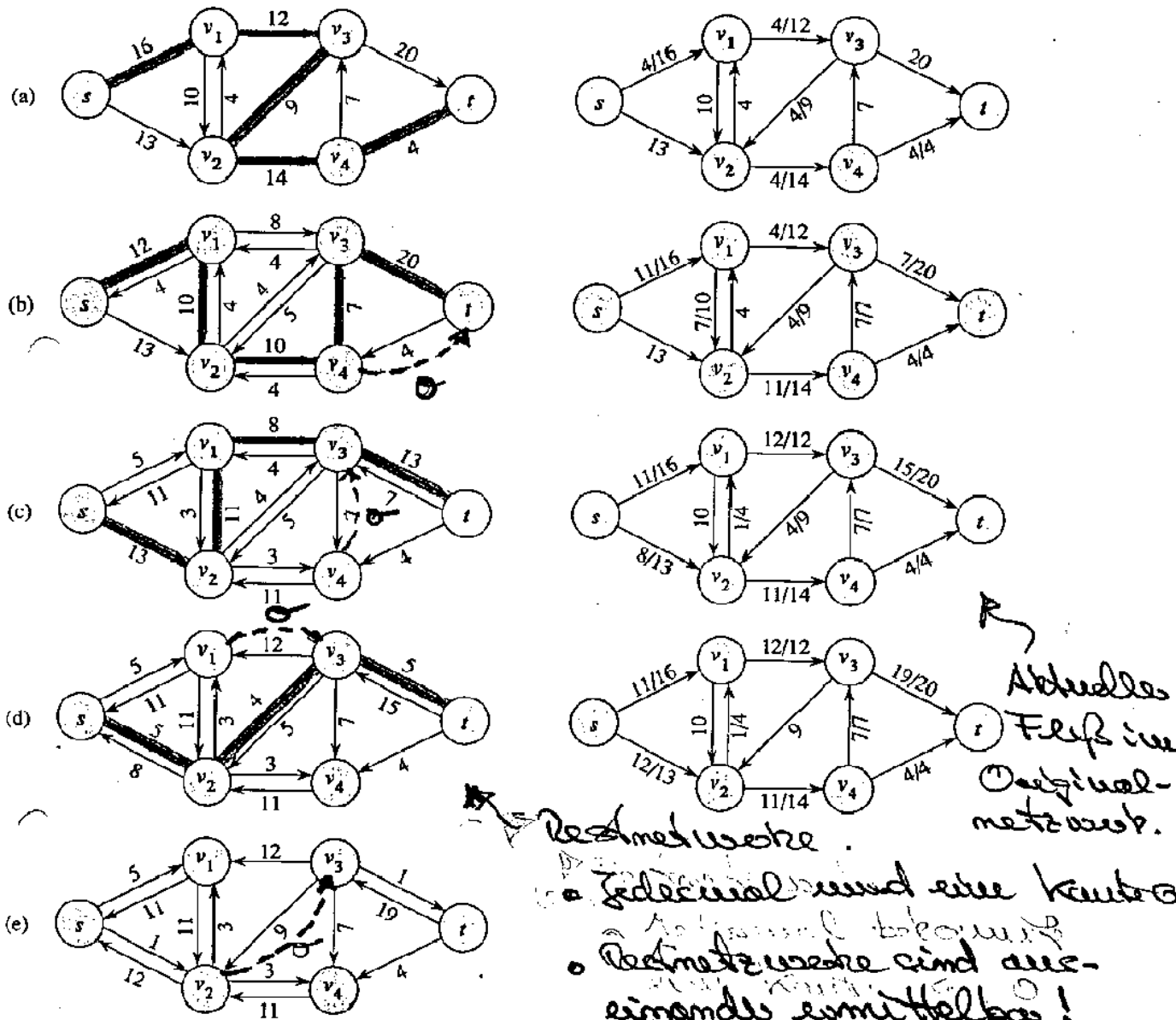
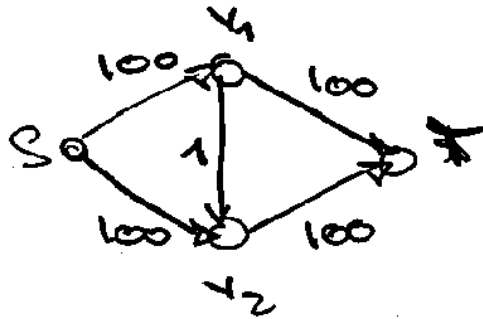


Figure 27.6 The execution of the basic Ford-Fulkerson algorithm. (a)-(d) Successive iterations of the while loop. The left side of each part shows the residual network  $G_f$  from line 4 with a shaded augmenting path  $p$ . The right side of each part shows the new flow  $f$  that results from adding  $f_p$  to  $f$ . The residual network in (a) is the input network  $G$ . (e) The residual network at the last while loop test. It has no augmenting paths, and the flow  $f$  shown in (d) is therefore a maximum flow.

# BEISPIEL LANGE LAUFZEIT

9.23



$F$  MAXIMAL  $\Leftrightarrow |F| = 200$ .

ERWEITERUNGSWEGE:

$$(S, v_1, v_2, T) + 1$$

BESSER:

$$(S, v_2, v_1, T) + 1$$

$$(S, v_1, T) + 100$$

$$(S, v_1, v_2, T) + 1$$

$$(S, v_2, T) + 100.$$

⋮

200 EINZELNE ERHÖHUNGEN

BEACHTEDAS RESTNETZWERK

"OSZILLIEREN" VON  $v_1 \rightarrow v_2$ .

Man sieht, daß Laufzeit  
 der Größe  $\Omega(|E| \cdot p^*)$  durchaus  
 aufbauen können. Es lassen sich  
 die Kapazitäten so wählen, daß  
 $|Q^*|$  exponentiell in der Länge  
 (in Bits) von  $Q$  und der Kapazitäts-  
 funktion ist. Abhilfe schafft  
 hier: Es wird ein Restgraph  
 immer ein kürzester Erweiterungspfad,  
 d.h. minimale Knotenausschl  
 gewählt.

9.24a

# Algorithmus (Edmonds Kap)

1.  $f(u, v) = 0$
2. while  $x$  in  $\mathcal{F}_f$  von  $s$  erreichbar  $\{$
3.  $\mathcal{B} = \mathcal{B}(G_f, s)$  // Breitensuche
4.  $v_{i-1} = x, v_i = \mathcal{P}[v_{i-1}], \dots$   
 $v_1 = \mathcal{P}[v_2], v_2 = v_0 = \mathcal{P}[v_1]$  //  $\mathcal{P}$  Breitensuche  
 // konstruieren
5.  $W := (x = v_0, v_1, \dots, v_k = x)$  //  $W$  ist eine  
 // k. W. bezogen
7.  $g(v_1, v_2, \dots) := \mathcal{M}_g(W)$  // auf die Kanten  
 // Kantenzahl
8.  $f := f + g$ ;  $\mathcal{F}_f$  berechnen  
 $\}$

2. bezogene von  $f$  als maximaler Fluß.

Beachte nach Algorithmus:

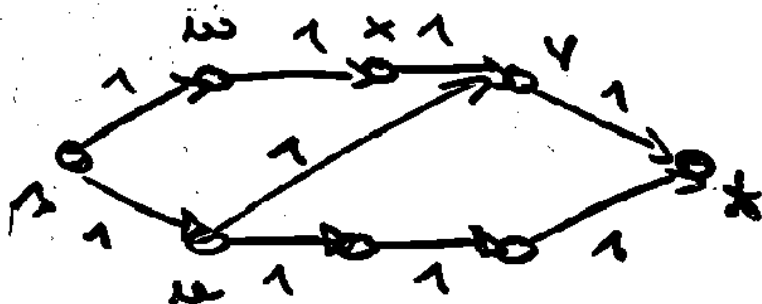
Erweiterungswege = Weg  $s \rightarrow \dots \rightarrow t$  mit  
 minimales Kantenzahl. Vgl. Bsp. p. 9.23,  
 dort gerade mit minimales Kantenzahl 15.

9.25

ZU EDMONDS KARP

6 NEGATIVER FLUSS

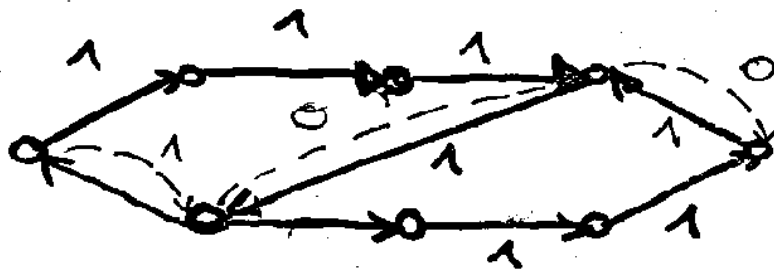
(↪ FLUSS UMLENKEN)



FLUSSNETZ-  
WERK.

KÜRZESTER E-WEG

(s, u, v, t) + 1



RESTNETZ-  
WERK

E-WEG

(s, w, x, y, u, ... t) + 1,

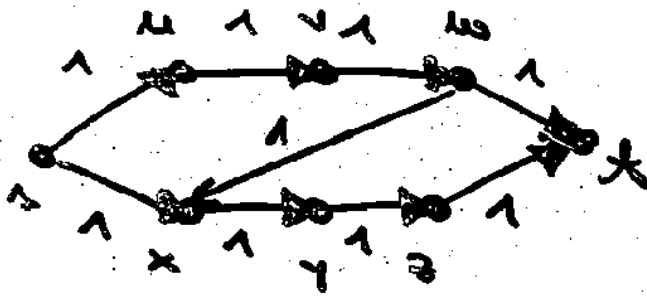
MAX. FLUSS GEFUNDEN.

9.26

# 0 VERKÜRZUNG KÜRZESTER

## WEGE FALLS NICHT (!) E.K.

EDMONDS  
KARP



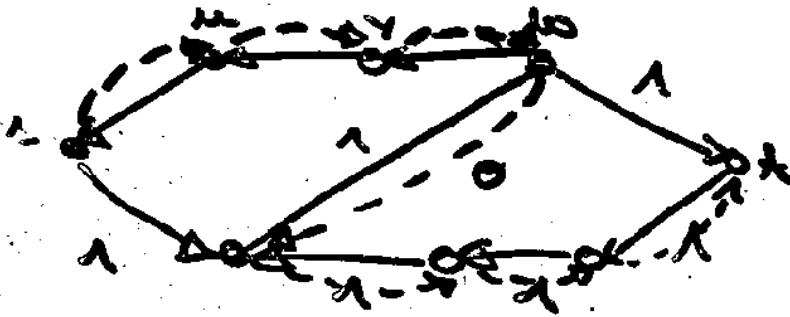
= 9

$$\text{DIST}(s, w) = 3$$

$$\text{DIST}(s, x) = 4$$

$$\text{E-WEG}(s, u, v, w, x, \dots, x) + 1$$

NICHT E.K.



$$\text{DIST}(s, w) = 2$$

$$\text{DIST}(s, x) = 3$$

$$\text{E-WEG}(s, x, w, x) + 1$$

Wir zeigen im restlichen Verlauf  
dieses Abschnitts folgenden

Satz

Sei  $w_1, w_2, w_3, \dots, w_n$

sei von Edmonds' Graph betrachteter

$\epsilon$ -Wege, dann gilt

# Knoten des betrachteten Flussnetzes

(a)  $L_\epsilon(w_1) \leq L_\epsilon(w_2) \leq \dots \leq L_\epsilon(w_n) \leq |V|$

(b)  $L_\epsilon(w_1) \leq L_\epsilon(w_{\lfloor |E|+1}) \leq L_\epsilon(w_{2\lfloor |E|+1})$

$\leq L_\epsilon(w_{3\lfloor |E|+1}) \leq \dots \leq$

Aus diesem Satz folgt

Heißt: Maximal  $|E|$   $\epsilon$ -Wege von gleicher Länge.

Folgerung

Bei Edmonds' Graph gilt:

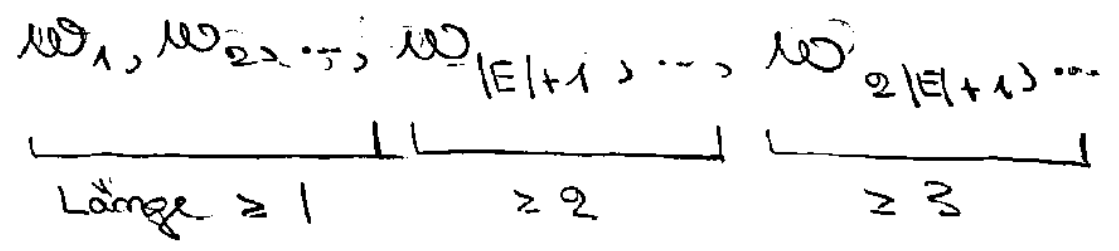
(a)  $|E| \cdot |V|$  Runden der Schläufe

(b)  $O(|E|^2 \cdot |V|)$  Zeit, sofern Kapazitäten

in  $O(n)$  zu bearbeiten sind

Beweis (des Folgerung)

(a) Mit dem letzten Satz ist für



also sicher ist, beim Index  $(|V|-1)|E|+1$  die Länge  $|V|$  erreicht. Wir haben maximal  $|E|$  Wege der Länge  $|V|$  und bei  $|E| \cdot |V|$  ist ganz sicher der letzte Weg erreicht und die Schleife hat maximal  $|E| \cdot |V|$  Durchläufe.

(b) Ein Durchlauf durch die Schleife braucht eine Zeit von

$O(|V| + |E|)$ . Und für  $|E| \geq |V|$  gilt die Behauptung.

$\mathbb{Z}^2$  ist ein  $\mathbb{Z}$ -Modul  $\mathbb{Z}^2$  (2.29)  
 mit dem Skalarprodukt  $\langle \cdot, \cdot \rangle$  ( $\mathbb{Z}^2$ ),  
 wobei  $\langle a, b \rangle = a_1 b_1 + a_2 b_2$   
 $\forall a = (a_1, a_2), b = (b_1, b_2) \in \mathbb{Z}^2$

Betrachten uns einmal einen  
 Lauf der Schleife, zunächst  
 vom allgemeinen Ford Fulkerson.  
 Sei dazu

$f$  = Fluß vor dem Lauf der  
 Schleife

$f'$  = Fluß nach dem Lauf der  
 Schleife

und  $f, f'$  sind die zugehörigen

Restnetzwerke. Es gilt,  $\mathcal{F}_p$  zerlegt  
aus  $\mathcal{F}_q$  indem eine (oder mehrere)

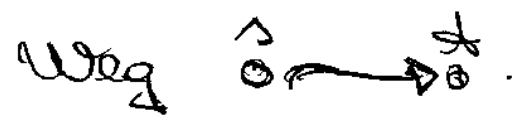
Kanten  $u \rightarrow v$  gelöscht werden  
und durch  $v \rightarrow u$  ersetzt werden.

Dabei gilt für  $u \rightarrow v$ :

- Die Kanten  $u \rightarrow v$  liegen auf  
dem gewählten E-Weg.
- Es ist möglich, daß eine  
Umkehrkante  $v \rightarrow u$  bereits in  
 $\mathcal{F}_q$  vorhanden ist. (Nach mitte!)

Bei Edwards Kamp gilt zusätzlich:

- Die Kanten  $u \rightarrow v$  wie oben  
liegen auf einem Kreisbogen (!)



Wir betrachten einmal alles  
ganz allgemein.

Lemma

Sei  $\mathcal{G}$  ein gerichtetes Graph  
mit Knoten  $s$  und  $t$ .

# Kanten  
eines k.w.

Sei  $k = \text{Dist}^{\mathcal{G}}(s, t)$ .  $\mathcal{M} \subseteq \mathcal{G}$

$u \rightarrow v$  Kante auf irgendeinem

k.w.  $s \rightsquigarrow t$ , so sei auch

die Umkehrkante  $v \rightarrow u$  in

$\mathcal{G}$  enthalten.

(a)  $\mathcal{M} \subseteq \mathcal{G}$   $u \rightarrow v$  Kante eines

k.w.  $s \rightsquigarrow t$ , so kommt

mit  $v \rightarrow u$  auf einem k.w.  $s \rightsquigarrow t$

vor.

(b) Enthält eine einfachste Weg  $\omega$

8.99

$\stackrel{1}{\circ} \rightsquigarrow \stackrel{k}{\circ} \rightsquigarrow \stackrel{k+2}{\circ}$  Umkehrkanten, dann

$$L_2(\omega) \geq k + 2l.$$

Beweis

(a) Angenommen auf dem kürzesten Wege kommen irgendwann diese Umkehrkanten vor. Wir betrachten eine  $k$ -W. auf dem eine dieser Umkehrkanten am weitesten vorne steht. Sei

$$\omega = (s = v_0, v_1, \dots, v_k = t)$$

und  $\overset{v_i}{\circ} \rightsquigarrow \overset{v_{i+1}}{\circ}$  sei die erste Umkehrkante

und kein  $k$ -W. benutzt auf seinem

ersten  $i$  Schritten eine Umkehrkante

Also gibt es einen  $k$ -W.

$$u = (s = u_0, \dots, u_i, u_{i+1}, \dots, u_k = t)$$

mit  $u_{j+1} = v_j$  und  $u_j = v_{j+1}$ . Dann

ist  $j \geq i$ , da sonst Umkehrkante

auf  $u$  auf den ersten  $i$  Schritten.

Dann ist aber

$$u' = (\overbrace{s = v_0, v_1, \dots, v_i}^{\text{von } u}, \overbrace{u_{i+1}, \dots, u_k = t}^{\text{von } u})$$

ein Weg mit

$$L_2(u') = i + k - (j+1) \stackrel{j \geq i}{\leq} k - 1,$$

da  $j \geq i$ . Das kann aber nicht

sein, da  $\text{Dist}(s, t) = k$  !

(b) Einige Spezialfälle zunächst:

$l=0$ , v da  $\text{Diel}(s, t) = k$ .

$l=1$ , dann betrachten wir

$$u = (\underbrace{s = u_0, u_1, \dots, u_i}_{M_1}, \underbrace{u_{i+1}, \dots, u_n = t}_{M_2})$$

1 Schritt

und sei  $u_i \rightarrow u_{i+1}$  Mutation

aus einem k. W. Sei also  $W$

k. W. mit

$$W = (s = v_0, v_1, \dots, v_j, v_{j+1}, \dots, v_n = t)$$

mit  $v_j = u_{i+1}, v_{j+1} = u_i$ . Dann ist

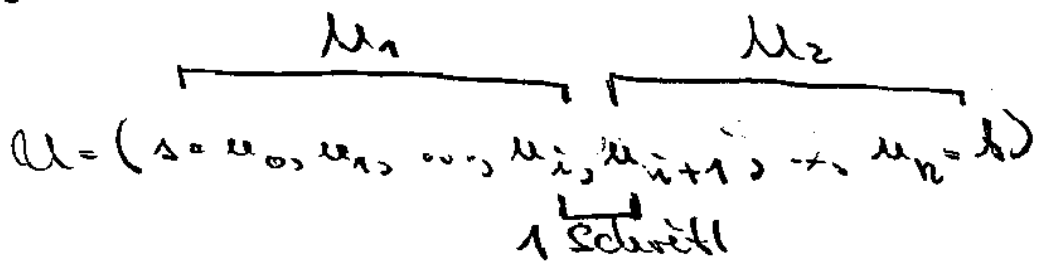
$$L_g(M_1) = i \geq j+1$$

da  $W$  k. W. ist! Ebenso

$$L_g(M_2) = k - i \geq k - j$$

Dann  $L_g(u) \geq (j+1) + 1 + k - j = k + 2$ .

Für beliebiges  $l \geq 2$  gehen wir analog induktiv vor. Jetzt die Behauptung für alle Wege mit  $l-1$  Umkehrkanten. Habe



genau  $l$  Umkehrkanten von  $h$ .  $\omega$ 's.

Bei  $u_i \xrightarrow{u_{i+1}}$  die erste Umkehrung auf  $U$ .

Bei also

$$\omega = (s = v_0, \dots, v_i, v_{i+1}, \dots, v_k = t)$$

$$h\text{-} \omega \text{ mit } v_i = u_{i+1}, v_{i+1} = u_i.$$

$$L_2(\omega) = L_2(\omega) + 1$$

$$L_2(\omega) = l \geq \frac{l}{2} + 1$$

$w$  betrachten

$$w' = ( \overbrace{v_0 = \Delta, \dots, v_j = u_{i+1}}^{\text{Von } u} , \dots , \overbrace{u_{i+1}, \dots, u_{2l} = \Delta}^{\text{Von } u} )$$

$\underbrace{\hspace{10em}}_{\text{Obere Unterteil}} \left\{ \begin{array}{l} l-1 \text{ Un-} \\ \text{terteile} \end{array} \right.$

$\underbrace{\hspace{10em}}_{\text{unq. (a)}}$

Schauen uns die Längen an,  
dann ist nach Ind.-Vor

$$L_2(L_2(w')) \geq l + 2(l-1) \cdot (l-1)$$

Da  $w$   $l$ -mal ist (\*) ist  $\geq$

$$L_2(L_2(w)) \geq i + j + 1$$

Schluss

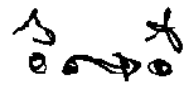
$$\begin{aligned} L_2(w) &= L_2(w') - j + i + 1 \\ L_2(w) &= L_2(w') - j + i + 1 \\ &\geq l + 2(l-1) - j + (j+1) + 1 \\ &= l + 2l \end{aligned}$$

Nun folgt direkt der

Beweis (des Satzes auf §. 9.27).

In jedem Lauf der Schliefe löscht

Edmonds Kopf mindestens eine

Kante auf einem k. W. 

und trägt die Umkehrkante ein.

(a) Fangen wir mit  $W_1$  an,

bei  $L_q(W_1) = k$ . Zunächst

beobachtet E. Kopf alle E-Wege

der Länge k. Folglich es solche

Wege gibt und nach letztem

Lemma keine Umkehrkante

betreuen. Irgendwann sind die

Weges der Länge  $k$  nicht mehr möglich. Dann haben wir nun noch  $k$ . W.'s der Länge  $k' \geq k+1$ . Erstl. mit Umkehrkanten von  $k$ . W.'s der Länge  $k$ . Wir betrachten all diese Wege und machen so weiter.

(b) Seien einmal

$W_1, \dots, W_\ell$   $k$ . W.'s

in einem Restnetzwerk.

In jeder Runde nach Edmonds

Kopf verschwindet mindestens

eine Kante auf den  $k$ . W.'s.

Nach  $|E|$  Runden sind alle

Kanten auf den  $W_1, \dots, W_\ell$  sicherl. 23

Prinzip: Umkehrkanten von  $k$ . W.'s bringen keine zusätzlichen  $k$ . W.'s

9.28

fort. Nach dem Lemma

enthaltet die  $W_i$  untereinander

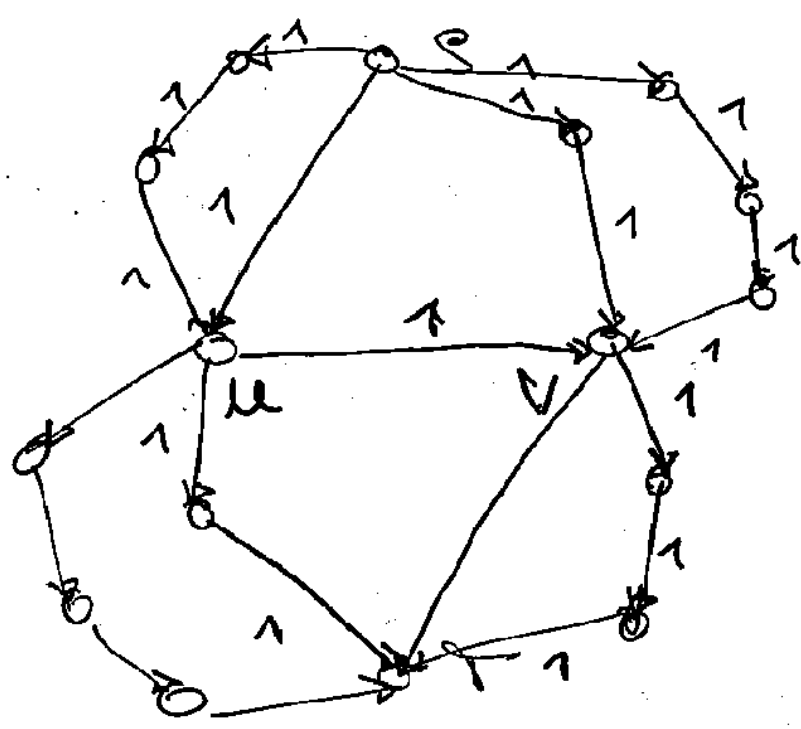
keine Nebenkanten. Also

besteht in  $\widehat{W}_i$  aus  $|E| + 1$  einer

Nebenkante vor. Dann vergrößert

sich die Länge.  $\square$

# OSZILLATION BEI EDMONDS KARP



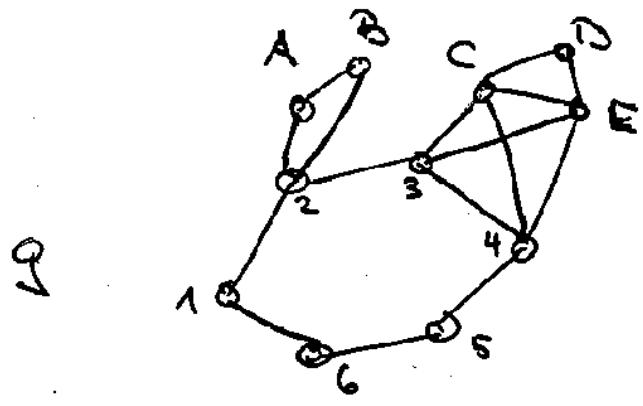
1.  $E\text{-WEG } (s, u, v, t) + 1$
2.  $E\text{-WEG } (s, -, v, u, -, t) + 1$
3.  $E\text{-WEG } (s, -, -, u, v, -, -, t) + 1$
4.  $E\text{-WEG } (s, -, -, v, u, -, -, t) + 1$
- ⋮

VERGLEICHE DABEIEGEN S. 9.23,

# NACHTRAG ZUM BEWEIS

AUF §. 10.22/23

GERADE GRADE  $\Rightarrow$  EULERSCHER KREIS



Q

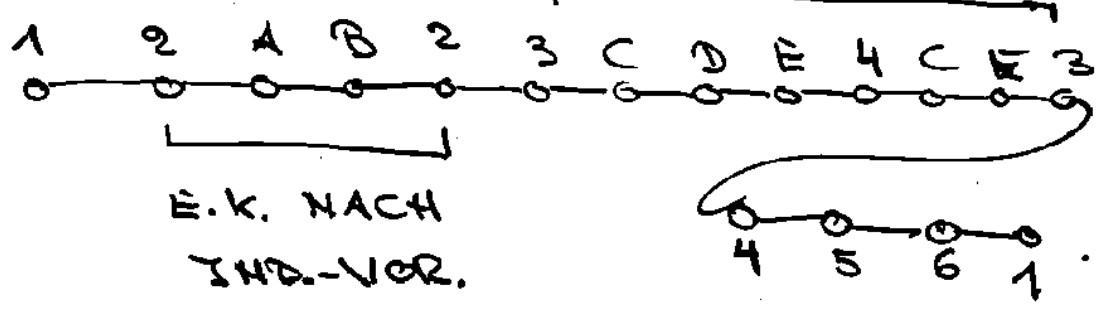
Y.A. KEIN E.K

WEG W ETWA

$$W = (1, 2, 3, 4, 5, 6, 1)$$

DANN E.K.

E.K N.Z.V.



# UNGLEICHE BRÜDER (10.11.2)

## POLYNOMIAL

KÜRZESTER WEG (OHNE NEG. KREISE)

EULER KREIS

2-FÄRBBARKEIT

2-SAT

MIN. SCHNITT

## EXPONENTIELL

KÜRZ. WEG MIT NEG. KREISE IM GRAPH

LÄNGSTER KREISFREIER WEG

HAMILTON KREIS

3-FÄRBBARKEIT

3-SAT

MAX-2-SAT

MAX. SCHNITT

- EXPO.  $\leftrightarrow$  DURCHORDNEN UNVERHEIDBAR.

- ZUEINANDER ÜBERSETZBAR
- KEIN BEWEIS, DASS NUR EXPO (ABER VERMUTET)

# MASTER THEOREM

U. 88

(KORREKTUR)

SEI

$$T(N) = \sum_{j=1}^k T(\alpha_j \cdot N) + \mathcal{O} \cdot N^L$$

$0 \neq \alpha_j \neq 1$ . ALLES KONST.,

AUSSER  $N$ . DANN

$T(N)$	}	$\mathcal{O}(N^L)$	$\sum \alpha_j^L \neq 1$
		$\mathcal{O}(N^L \log^2 N)$	$\sum \alpha_j^L = 1$
		$\mathcal{O}(N^C)$	$\sum \alpha_j^L > 1$

$\sum \alpha_j^C = 1$  BESTIMMT  $C$ .

11.89

# BEISPIELE ZUM MASTER THEOREM

LINEARES SELECT :  $L=1$   $O(N)$ ,  
 $\Omega=2$

MERGESORT :  $L=1$ ,  $\alpha_1^1 + \alpha_2^1 = 1$ ,  $\Omega=2$   
 $O(N \cdot \log N)$

MATRIZENMULT :  $L=2$ ,  $\Omega=7$

$$\alpha_3 = \frac{1}{2} \cdot \sum_j \alpha_j^2 = 1$$

$$\Rightarrow \alpha_3^2 = \frac{1}{2}$$

$$\Rightarrow \Omega^2 = 7 \Rightarrow 2 \cdot \log_2 7$$

$$O(N^{\log_2 7})$$

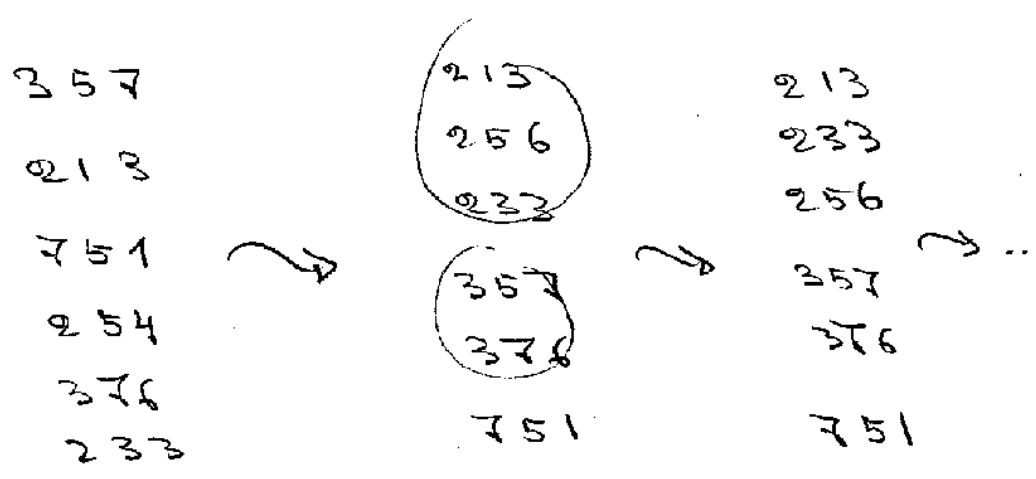
LITERATUR : SCHÖNING

Bei Counting Sort werden die zu sortierenden Zahlen als Indices eines Arrays verwendet.

Das geht bei größeren Zahlen nicht.

Deshalb Radixsort (verwendet Counting Sort auf den Ziffern).

Ziffernummern Sortieren ist bekannt. Also Sortieren von den größten zur kleinsten Stelle:



Man muß die Stücke einzeln

sortieren. Dazu braucht man

# Stellen. # Ziffern viele Punkte

in das zu entfernende away bringen.

Radixsort: Fängt auf der letzten

Stelle an und arbeitet immer

mit immer kleineren Zahlen away.

Dabei wichtig: Stabilität

des Verfahrens für die

Ziffern (Counting Sort)

Also ein Beispiel:

Stapel!

13.20

357

213

751

254

376

233

751

213

233

254

376

357

213

233

751

254

357

376



Stapel

213

233

254

357

376

751



Invariante: Nach i-tem Lauf  
kannst sortiert bzgl. den  
letzten i Ziffern.

# Algorithmus (Radix sort)

Eingabe: array  $A[1..m]$  of subkeys  
und  $d = \#$  Ziffern.

Ausgabe:  $A$  sortiert.

1. for  $i = 1$  to  $d$  } // Letzte Stelle 1.  
// 1. Stelle 1.
2. Sortiere  $A$  auf Stelle  $i$   
mit Counting Sort □

Zeit: Ziffern aus  $0, \dots, k$

$$\text{dann } \mathcal{O}(d \cdot (m+k)) = \mathcal{O}(d \cdot m)$$

bei  $k$  konstant (etwa  $k = 9$ )

Sind die Zahlen groß, also von

der Größe  $\mathcal{O}(m)$ , ist  $d = \mathcal{O}(\log m)$   
und wir haben auch nur  $\mathcal{O}(m \cdot \log m)$ . 39