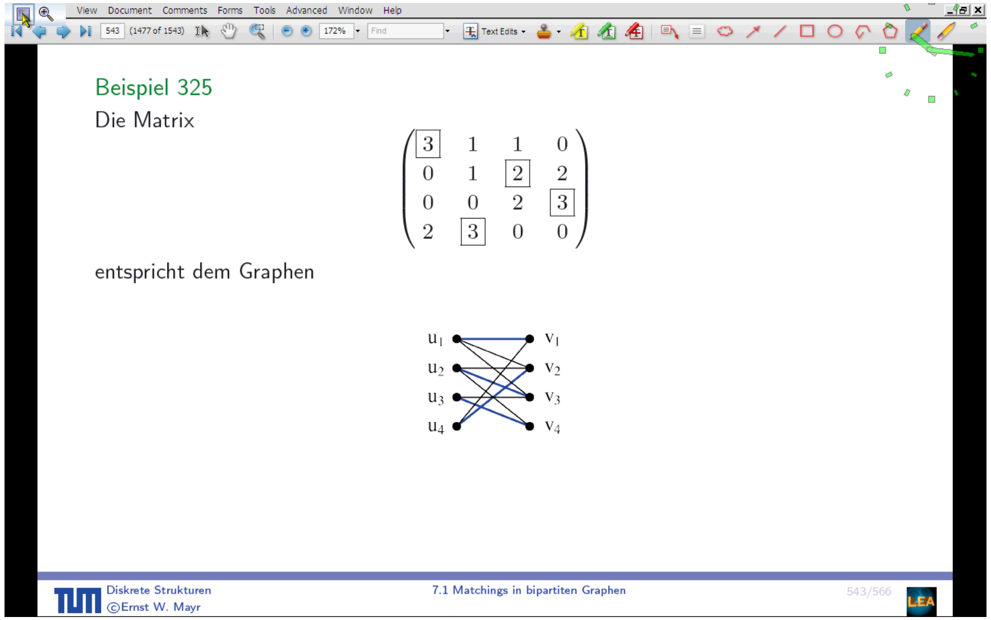
Script generated by TTT

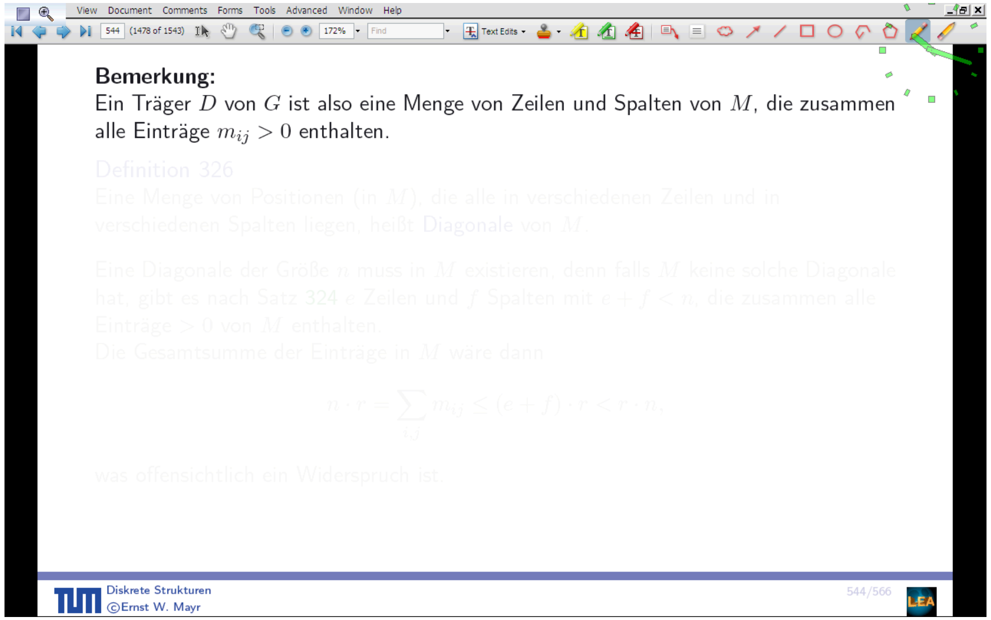
Title: Mayr: 2011 ds (09.02.2012)

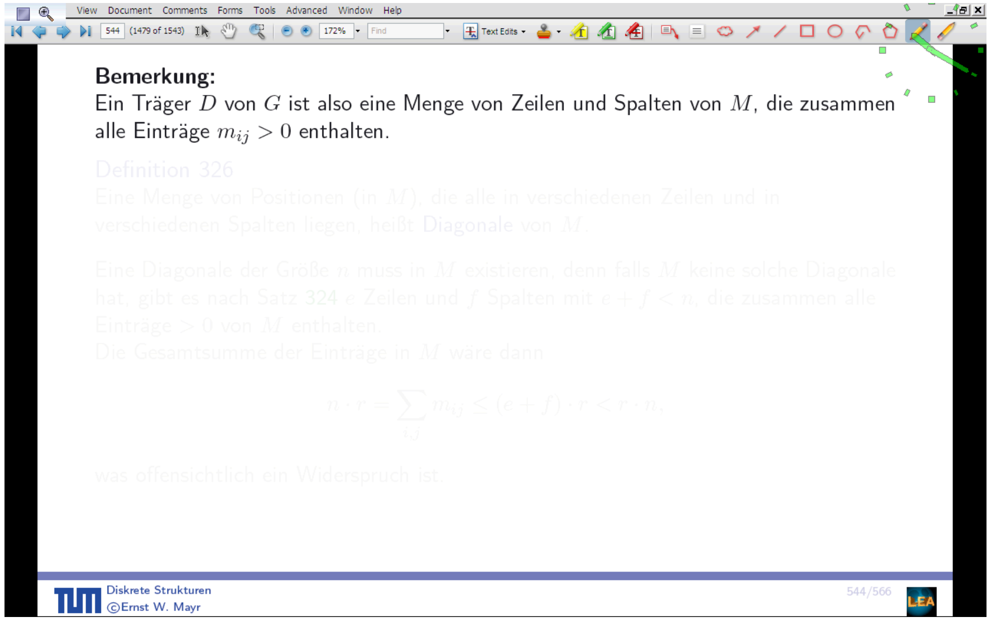
Date: Thu Feb 09 10:16:34 CET 2012

Duration: 110:43 min

Pages: 55







Satz 324

Es gilt:

$$\max\{|M|; M \text{ \textit{Matching}}\} = \min\{|D|; D \text{ \textit{Träger}}\}$$

Bemerkung:
Ein Träger D von G ist also eine Menge von Zeilen und Spalten von M, die zusammen alle Einträge $m_{ij} > 0$ enthalten.

Definition 326

Eine Menge von Positionen (in M), die alle in verschiedenen Zeilen und in verschiedenen Spalten liegen, heißt Diagonale von M.

Eine Diagonale der Größe n muss in M existieren, denn falls M keine solche Diagonale hat, gibt es nach Satz 324 e Zeilen und f Spalten mit $e + f < n$, die zusammen alle Einträge > 0 von M enthalten.

Die Gesamtsumme der Einträge in M wäre dann

$$n \cdot r = \sum_{i,j} m_{ij} \le (e + f) \cdot r < r \cdot n\,,$$

was offensichtlich ein Widerspruch ist.

Sei c_1 der minimale Eintrag > 0 in M, und sei P_1 die zu einer Diagonale der Größe n gehörige Permutationsmatrix (d. h. Einträge $= 1$ an den Positionen der Diagonale, 0 sonst).

Dann gilt:

$$M_1 := M - c_1 P_1$$

ist eine $n \times n$-Matrix mit allen Zeilen- und Spaltensummen $= r - c_1$. Die Matrix M_1 enthält damit mehr Nullen als M.

Damit haben wir gezeigt:

Sei c_1 der minimale Eintrag > 0 in M, und sei P_1 die zu einer Diagonale der Größe n gehörige Permutationsmatrix (d. h. Einträge $= 1$ an den Positionen der Diagonale, 0 sonst).

Dann gilt:

$$M_1 := M - c_1 P_1$$

ist eine $n \times n$-Matrix mit allen Zeilen- und Spaltensummen $= r - c_1$. Die Matrix M_1 enthält damit mehr Nullen als M.

Damit haben wir gezeigt:

Satz 327

Sei M wie oben. Dann gibt es für ein geeignetes k Konstanten $c_i > 0$ und Permutationsmatrizen P_i, $i = 1, \ldots, k$, so dass gilt

$$M = \sum_{i=1}^{k} c_i P_i \qquad \sum_{i=1}^{k} c_i = r\,.$$

Satz 327

Sei M wie oben. Dann gibt es für ein geeignetes k Konstanten $c_i > 0$ und Permutationsmatrizen P_i, $i = 1, \ldots, k$, so dass gilt

$$M = \sum_{i=1}^{k} c_i P_i \qquad \sum_{i=1}^{k} c_i = r\,.$$

Diskrete Strukturen
©Ernst W. Mayr
7.1 Matchings in bipartiten Graphen
546/566

7.2 Konstruktion optimaler Matchings

Satz 328

Ein Matching M ist genau dann Maximum, wenn es dazu keinen augmentierenden Pfad gibt.

Beweis:

7.2 Konstruktion optimaler Matchings

Satz 328

Ein Matching M ist genau dann Maximum, wenn es dazu keinen augmentierenden Pfad gibt.

Beweis:

„$\Rightarrow$“ Offensichtlich.

„$\Leftarrow$“ Sei M ein Matching, zu dem es keinen augmentierenden Pfad gibt. Annahme, M sei kein Maximum Matching, es existiere ein Maximum Matching M'. Betrachte nun $M \Delta M'$. Die Zusammenhangskomponenten dieses Graphen sind alternierende Pfade und Kreise gerader Länge. Da $|M'| > |M|$ gilt, muss es einen alternierenden Pfad mit ungerader Länge geben, der mit Kanten aus M' beginnt und endet. Dies ist aber ein *augmentierender* Pfad. $\square$

7.2 Konstruktion optimaler Matchings

Satz 328

Ein Matching M ist genau dann Maximum, wenn es dazu keinen augmentierenden Pfad gibt.

Beweis:

„$\Rightarrow$" Offensichtlich.

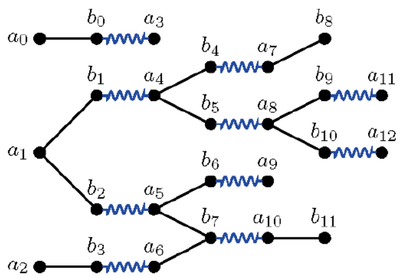
„$\Leftarrow$" Sei M ein Matching, zu dem es keinen augmentierenden Pfad gibt. Annahme, M sei kein Maximum Matching, es existiere ein Maximum Matching M'. Betrachte nun $M \Delta M'$. Die Zusammenhangskomponenten dieses Graphen sind alternierende Pfade und Kreise gerader Länge. Da $|M'| > |M|$ gilt, muss es einen alternierenden Pfad mit ungerader Länge geben, der mit Kanten aus M' beginnt und endet. Dies ist aber ein *augmentierender* Pfad. □

Der Algorithmus zur Konstruktion optimaler Matchings ist eine parallele (simultane) alternierende Breitensuche.

Beispiel 329 (Konstruktion im bipartiten Graph)

Der Algorithmus zur Konstruktion optimaler Matchings ist eine parallele (simultane) alternierende Breitensuche.

Beispiel 329 (Konstruktion im bipartiten Graph)



Ergebnisse und Erweiterungen:

	bipartit	allgemein
ungewichtet	$O(\sqrt{\lvert V\rvert}\cdot\lvert E\rvert)$	$O(\sqrt{\lvert V\rvert}\cdot\lvert E\rvert)$
gewichtet	$O(\lvert V\rvert\cdot(\lvert E\rvert+\lvert V\rvert\cdot\log(\lvert V\rvert)))$	$O(\lvert V\rvert\cdot\lvert E\rvert\cdot\log(\lvert V\rvert))$

Siehe auch:
Zvi Galil: Efficient algorithms for finding maximum matchings in graphs, ACM Computing Surveys 18 (1986), pp. 23–38

Ergebnisse und Erweiterungen:

	bipartit	allgemein
ungewichtet	$O(\sqrt{\lvert V\rvert}\cdot\lvert E\rvert)$	$O(\sqrt{\lvert V\rvert}\cdot\lvert E\rvert)$
gewichtet	$O(\lvert V\rvert\cdot(\lvert E\rvert+\lvert V\rvert\cdot\log(\lvert V\rvert)))$	$O(\lvert V\rvert\cdot\lvert E\rvert\cdot\log(\lvert V\rvert))$

Siehe auch:
Zvi Galil: Efficient algorithms for finding maximum matchings in graphs, ACM Computing Surveys 18 (1986), pp. 23–38

7.3 Reguläre bipartite Graphen

Lemma 330
Sei $G=(U,V,E)$ ein k-regulärer bipartiter Graph ($k\in\mathbb{N}$). Dann hat G ein perfektes Matching.

Beweis:
Sei $A\subseteq U$ und $B=N(A)\subseteq V$. Dann ist $|A|\le|B|$, da ja alle von A ausgehenden Kanten in B enden und, falls $|B|<|A|$, es in B damit einen Knoten mit Grad $>k$ geben müsste. □

Korollar 331
Sei $G=(U,V,E)$ ein k-regulärer bipartiter Graph ($k\in\mathbb{N}$). Dann lässt sich E als disjunkte Vereinigung von k perfekten Matchings darstellen.

7.4 Transversalen

Definition 332
Sei $G=(U,V,E)$ ein bipartiter Graph, M ein Matching in G, und $A\subseteq U$ die in M gematchte Teilmenge der Knotenmenge U. Dann heißt A eine Transversale in G.

Satz 333
Sei $G=(U,V,E)$ ein bipartiter Graph, $T\subseteq 2^U$ die Menge der Transversalen in G. Dann ist (U,T) ein Matroid.

Beweis:
Die ersten beiden Bedingungen für ein Matroid sind klarerweise erfüllt.

Beweis (Forts.):

Seien nun A und A' Transversalen mit den zugehörigen Matchings M und M', und sei $|A'| = |A| + 1$, also auch $|M'| = |M| + 1$. Betrachte $M' \Delta M$.
Dann muss $M' \Delta M$ (mindestens) einen Pfad ungerader Länge enthalten, der mit einer Kante in M' beginnt und mit einer Kante in M' endet (und dazwischen abwechselnd Kanten in M bzw. M' enthält). Dieser Pfad ist ein augmentierender Pfad bzgl. M, und einer der beiden Endpunkte liegt in $A' \setminus A$, kann also zu A hinzugenommen werden. □

Beweis (Forts.):

Seien nun A und A' Transversalen mit den zugehörigen Matchings M und M', und sei $|A'| = |A| + 1$, also auch $|M'| = |M| + 1$. Betrachte $M' \Delta M$.
Dann muss $M' \Delta M$ (mindestens) einen Pfad ungerader Länge enthalten, der mit einer Kante in M' beginnt und mit einer Kante in M' endet (und dazwischen abwechselnd Kanten in M bzw. M' enthält). Dieser Pfad ist ein augmentierender Pfad bzgl. M, und einer der beiden Endpunkte liegt in $A' \setminus A$, kann also zu A hinzugenommen werden. □

Anwendung: gewichtetes Zuweisungsproblem, Variante 1

n Nutzer wollen jeweils auf eine aus einer nutzerspezifischen Teilmenge von insgesamt m Ressourcen zugreifen. Jede Ressource kann aber nur von höchstens einem Nutzer in Anspruch genommen werden. Der Wert einer Zuweisung von Ressourcen zu (interessierten) Nutzern ergibt sich als die Summe

$$\sum_{i \in A} w_i \; ,$$

wobei die Zuweisung einem Matching in dem durch Nutzer, Ressourcen und Zugriffswünsche gegebenen Graphen entspricht, $w_i \in \mathbb{R}^+$ ein Gewicht für jeden Nutzer $i \in \{1, \ldots, n\}$ ist, und A die durch die Zuweisung (das Matching) bedachte Teilmenge der Nutzer ist.

7.5 Gewichtetes Matching in bipartiten Graphen

Wir betrachten nun eine zweite Variante eines Zuweisungsproblems, das durch bipartite Graphen $G = (U, V, E)$ mit einer Gewichtsfunktion $w : E \to \mathbb{R}^+$ gegeben ist. Das Gewicht eines Matchings $M \subseteq E$ ist dann

$$\sum_{e \in M} w(e) \; .$$

Wir können o.B.d.A. annehmen, dass $|U| = |V|$ $(= n)$ und G vollständig bipartit (also $G = K_{n,n}$) ist, indem wir zunächst die kleinere der beiden Mengen U und V mit zusätzlichen Knoten auffüllen und dann die fehlenden Kanten durch Kanten mit Gewicht 0 ersetzen.

7.5 Gewichtetes Matching in bipartiten Graphen

Wir betrachten nun eine zweite Variante eines Zuweisungsproblems, das durch bipartite Graphen $G = (U, V, E)$ mit einer Gewichtsfunktion $w : E \to \mathbb{R}^+$ gegeben ist. Das Gewicht eines Matchings $M \subseteq E$ ist dann

$$\sum_{e \in M} w(e) \,.$$

Wir können o.B.d.A. annehmen, dass $|U| = |V|$ $(= n)$ und G vollständig bipartit (also $G = K_{n,n}$) ist, indem wir zunächst die kleinere der beiden Mengen U und V mit zusätzlichen Knoten auffüllen und dann die fehlenden Kanten durch Kanten mit Gewicht 0 ersetzen.

Damit suchen wir in G *optimale perfekte* Matchings. Wir können das Problem, ein perfektes Matching maximalen Gewichts zu finden, reduzieren auf das Problem, ein perfektes Matching minimalen Gewichts zu bestimmen, indem wir jedes Gewicht $w(e)$ durch

$$\max_{e \in E} w(e) \; - \; w(e)$$

ersetzen.

Wir nehmen daher an, dass wir o.B.d.A. ein perfektes Matching minimalen Gewichts in (G, w) suchen.

Für die folgende Diskussion nehmen wir zur Vereinfachung weiter an, dass alle Gewichte $\in \mathbb{N}_0$ sind.

Sei

$$W = (w_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le n}}$$

die zu (G, w) gehörige Gewichtsmatrix, und sei

$$P = (p_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le n}}$$

eine Permutationsmatrix (d.h., jede Zeile und jede Spalte von P enthält genau eine 1 und ansonsten nur Einträge 0).

Die Permutationsmatrix P entspricht einem perfekten Matching M in G mit Gewicht

$$\sum_{i,j} p_{ij} w_{ij} \,.$$

Sei

$$W = (w_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le n}}$$

die zu (G, w) gehörige Gewichtsmatrix, und sei

$$P = (p_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le n}}$$

eine Permutationsmatrix (d.h., jede Zeile und jede Spalte von P enthält genau eine 1 und ansonsten nur Einträge 0).

Die Permutationsmatrix P entspricht einem perfekten Matching M in G mit Gewicht

$$\sum_{i,j} p_{ij} w_{ij} \,.$$

Beobachtung:
Wenn wir von jedem Element einer Zeile (oder Spalte) in W einen festen Betrag p subtrahieren, verringert sich das Gewicht eines jeden perfekten Matchings M um diesen Betrag p, die relative Ordnung (nach Gewicht) unter den perfekten Matchings bleibt bestehen, insbesondere gehen optimale Matchings wieder in optimale Matchings über.

Wir führen nun solche Zeilen- und Spaltenumformungen durch, um eine Diagonale mit möglichst vielen Einträgen $= 0$ zu erhalten.

Beispiel 334

Sei

$$W = \begin{pmatrix} 9 & 11 & 12 & 11 \\ 6 & 3 & 8 & 5 \\ 7 & 6 & 13 & 11 \\ 9 & 10 & 10 & 7 \end{pmatrix}$$

Nachdem wir von jeder Zeile das minimale Gewicht subtrahieren, erhalten wir

$$W' = \begin{pmatrix} 0 & 2 & 3 & 2 \\ 3 & 0 & 5 & 2 \\ 1 & 0 & 7 & 5 \\ 2 & 3 & 3 & 0 \end{pmatrix}$$

Beispiel (Forts.)

$$W' = \begin{pmatrix} 0 & 2 & 3 & 2 \\ 3 & 0 & 5 & 2 \\ 1 & 0 & 7 & 5 \\ 2 & 3 & 3 & 0 \end{pmatrix}$$

Nachdem wir von jeder Spalte das minimale Gewicht subtrahieren, erhalten wir

$$W'' = \begin{pmatrix} 0 & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & 0 & 4 & 5 \\ 2 & 3 & 0 & 0 \end{pmatrix}$$

Beispiel (Forts.)

Diese Matrix enthält eine Diagonale der Größe 3 mit Einträgen $= 0$:

$$W'' = \begin{pmatrix} \boxed{0} & 2 & 0 & 2 \\ 3 & 0 & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ 2 & 3 & \boxed{0} & 0 \end{pmatrix}$$

Aus Satz 324 folgt, dass die maximale Länge einer 0-Diagonale gleich der minimalen Anzahl von Zeilen und Spalten ist, die alle 0en bedecken.
Falls wir noch keine 0-Diagonale der Länge n haben, iterieren wir folgenden Algorithmus:

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten.

Die Gewichte ändern sich also wie folgt:

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0.

Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e + f) + ef .$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned}\Delta w &= (ef)w - (n^2 - n(e + f) + ef)w \\ &= (n(e + f) - n^2)w < 0\end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i, j) nicht überdeckt ist;
2. um 0, falls (i, j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i, j) von einer Zeile *und* einer Spalte überdeckt ist.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0.

Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e+f) + ef\,.$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned}\Delta w &= (ef)w - (n^2 - n(e+f) + ef)w\\ &= (n(e+f) - n^2)w < 0\end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

Beispiel (Forts.)

In unserem Beispiel ergibt sich

$$W'' = \begin{pmatrix} \boxed{0} & \boxed{2} & \boxed{0} & \boxed{2} \\ 3 & \boxed{0} & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ \boxed{2} & \boxed{3} & \boxed{0} & \boxed{0} \end{pmatrix}$$

Der Algorithmus bestimmt $w = 1$:

$$\Rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 \\ 2 & -1 & 1 & 1 \\ 0 & -1 & 3 & 4 \\ 2 & 3 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \boxed{0} & 2 \\ 2 & \boxed{0} & 1 & 1 \\ \boxed{0} & 0 & 3 & 4 \\ 2 & 4 & 0 & \boxed{0} \end{pmatrix}$$

1. finde eine minimale Anzahl von e Zeilen und f Spalten ($e + f < n$), die zusammen alle Einträge $= 0$ enthalten;
2. sei w das Minimum der nicht überdeckten Elemente;
3. subtrahiere w von den $n - e$ nicht überdeckten Zeilen;
4. addiere w zu den f überdeckten Spalten

Die Gewichte ändern sich also wie folgt:

1. um $-w$, falls (i,j) nicht überdeckt ist;
2. um 0, falls (i,j) von einer Zeile *oder* Spalte überdeckt ist, aber nicht beides;
3. um $+w$, falls (i,j) von einer Zeile *und* einer Spalte überdeckt ist.

Beispiel (Forts.)

In unserem Beispiel ergibt sich

$$W'' = \begin{pmatrix} \boxed{0} & \boxed{2} & \boxed{0} & \boxed{2} \\ 3 & \boxed{0} & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ \boxed{2} & \boxed{3} & \boxed{0} & \boxed{0} \end{pmatrix}$$

Der Algorithmus bestimmt $w = 1$:

$$\Rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 \\ 2 & -1 & 1 & 1 \\ 0 & -1 & 3 & 4 \\ 2 & 3 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \boxed{0} & 2 \\ 2 & \boxed{0} & 1 & 1 \\ \boxed{0} & 0 & 3 & 4 \\ 2 & 4 & 0 & \boxed{0} \end{pmatrix}$$

Beispiel (Forts.)

In dem durch die Matrix

$$W = \begin{pmatrix} 9 & 11 & \boxed{12} & 11 \\ 6 & \boxed{3} & 8 & 5 \\ \boxed{7} & 6 & 13 & 11 \\ 9 & 10 & 10 & \boxed{7} \end{pmatrix}$$

gegebenen bipartiten Graphen hat also das durch die markierten Kanten gegebene perfekte Matching minimales Gewicht.

Bemerkung: Bei geeigneter Implementierung ist die Laufzeit des Algorithmus $O(n^3)$.

Insbesondere sind die resultierenden Gewichte wieder ≥ 0.
Die Anzahl der doppelt (von Zeilen *und* Spalten) überdeckten Positionen ist $e \cdot f$, die Anzahl der nicht überdeckten Positionen ist

$$n^2 - n(e+f) + ef\,.$$

Der resultierende Gewichtsunterschied ist daher

$$\begin{aligned} \Delta w &= (ef)w - (n^2 - n(e+f) + ef)w \\ &= (n(e+f) - n^2)w < 0 \end{aligned}$$

Damit muss unsere Iteration enden und wir finden eine 0-Diagonale der Länge n, entsprechend einer optimalen Zuordnung.

Beispiel (Forts.)

In unserem Beispiel ergibt sich

$$W'' = \begin{pmatrix} \boxed{0} & \boxed{2} & \boxed{0} & \boxed{2} \\ 3 & \boxed{0} & 2 & 2 \\ 1 & \boxed{0} & 4 & 5 \\ \boxed{2} & \boxed{3} & \boxed{0} & \boxed{0} \end{pmatrix}$$

Der Algorithmus bestimmt $w = 1$:

$$\Rightarrow \begin{pmatrix} 0 & 2 & 0 & 2 \\ 2 & -1 & 1 & 1 \\ 0 & -1 & 3 & 4 \\ 2 & 3 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \boxed{0} & 2 \\ 2 & \boxed{0} & 1 & 1 \\ \boxed{0} & 0 & 3 & 4 \\ 2 & 4 & 0 & \boxed{0} \end{pmatrix}$$

Beispiel (Forts.)

In dem durch die Matrix

$$W = \begin{pmatrix} 9 & 11 & \boxed{12} & 11 \\ 6 & \boxed{3} & 8 & 5 \\ \boxed{7} & 6 & 13 & 11 \\ 9 & 10 & 10 & \boxed{7} \end{pmatrix}$$

gegebenen bipartiten Graphen hat also das durch die markierten Kanten gegebene perfekte Matching minimales Gewicht.

Bemerkung: Bei geeigneter Implementierung ist die Laufzeit des Algorithmus $O(n^3)$.

7.6 Das Problem des chinesischen Postboten

Gegeben ist ein zusammenhängender, gewichteter Multigraph $G = (V, E, w)$.
Gesucht ist ein Kreis minimalen Gewichts, der jede Kante mindestens einmal enthält.

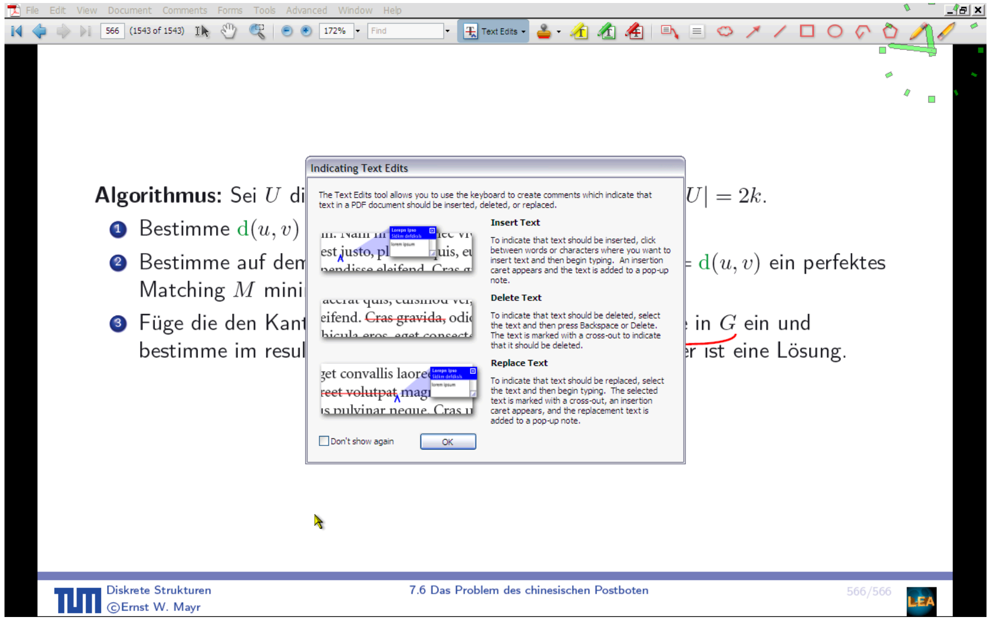
Beispiel 335 (In der optimalen Lösung werden die dickeren grünen Kanten zweimal verwendet)

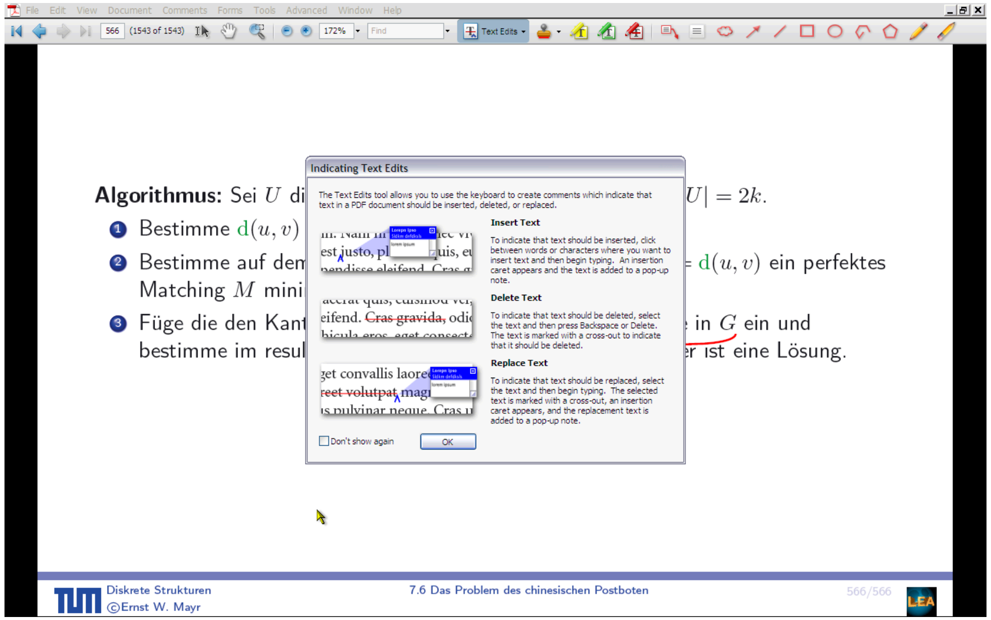
Algorithmus: Sei U die Menge der Knoten ungeraden Grades, $|U| = 2k$.

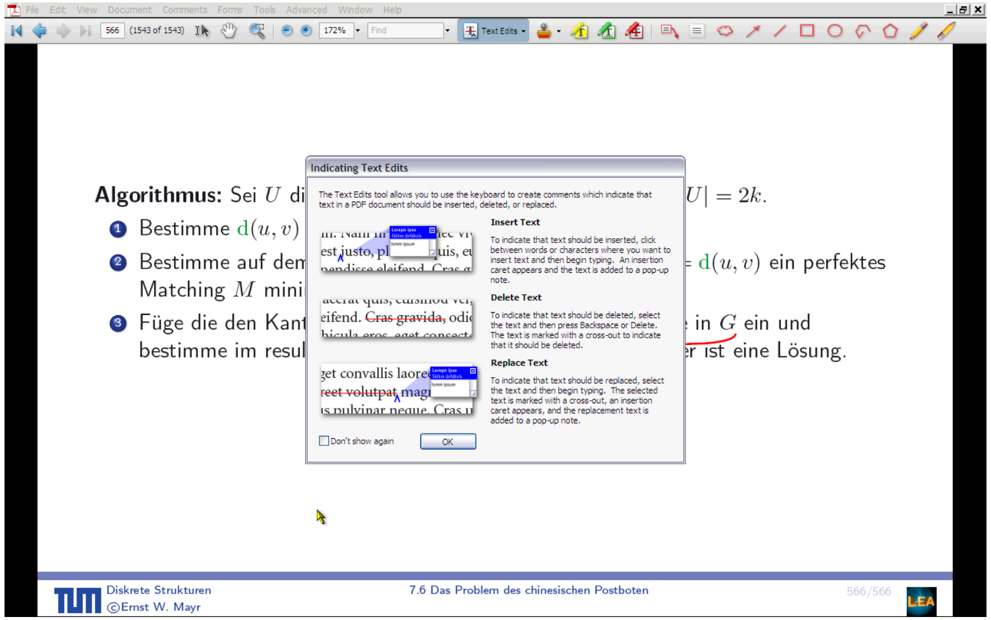
1. Bestimme $d(u, v)$ für alle $u, v \in U$.
2. Bestimme auf dem K_{2k} mit Kantengewichtung $w(\{u, v\}) = d(u, v)$ ein perfektes Matching M minimalen Gewichts.
3. Füge die den Kanten in M entsprechenden kürzesten Pfade in G ein und bestimme im resultierenden Graphen einen Eulerkreis. Dieser ist eine Lösung.

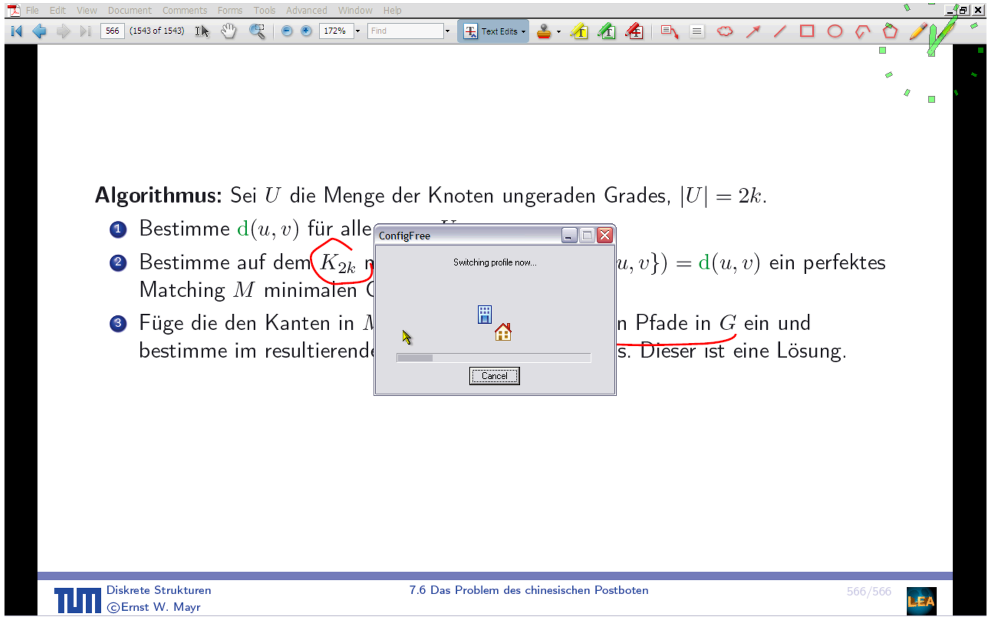
Algorithmus: Sei U die Menge der Knoten ungeraden Grades, $|U| = 2k$.

1. Bestimme $d(u, v)$ für alle $u, v \in U$.
2. Bestimme auf dem K_{2k} mit Kantengewichtung $w(\{u, v\}) = d(u, v)$ ein perfektes Matching M minimalen Gewichts.
3. Füge die den Kanten in M entsprechenden kürzesten Pfade in G ein und bestimme im resultierenden Graphen einen Eulerkreis. Dieser ist eine Lösung.


File Edit View Document Comments Forms Tools Advanced Window Help
566 (1543 of 1543)
172%
Find
Text Edits
Algorithmus: Sei U di
$U| = 2k$.
1 Bestimme $\mathrm{d}(u, v)$
2 Bestimme auf den
$= \mathrm{d}(u, v)$ ein perfektes
Matching M mini
3 Füge die den Kant
in G ein und
bestimme im resul
r ist eine Lösung.
Indicating Text Edits
The Text Edits tool allows you to use the keyboard to create comments which indicate that text in a PDF document should be inserted, deleted, or replaced.
Insert Text
To indicate that text should be inserted, click between words or characters where you want to insert text and then begin typing. An insertion caret appears and the text is added to a pop-up note.
Delete Text
To indicate that text should be deleted, select the text and then press Backspace or Delete. The text is marked with a cross-out to indicate that it should be deleted.
Replace Text
To indicate that text should be replaced, select the text and then begin typing. The selected text is marked with a cross-out, an insertion caret appears, and the replacement text is added to a pop-up note.
Don't show again
OK
Diskrete Strukturen
©Ernst W. Mayr
7.6 Das Problem des chinesischen Postboten
566/566
LEA


File Edit View Document Comments Forms Tools Advanced Window Help
566 (1543 of 1543)
172%
Find
Text Edits
Algorithmus: Sei U di
$U| = 2k$.
1 Bestimme $\mathrm{d}(u, v)$
2 Bestimme auf den
$= \mathrm{d}(u, v)$ ein perfektes
Matching M mini
3 Füge die den Kant
in G ein und
bestimme im resul
r ist eine Lösung.
Indicating Text Edits
The Text Edits tool allows you to use the keyboard to create comments which indicate that text in a PDF document should be inserted, deleted, or replaced.
Insert Text
To indicate that text should be inserted, click between words or characters where you want to insert text and then begin typing. An insertion caret appears and the text is added to a pop-up note.
Delete Text
To indicate that text should be deleted, select the text and then press Backspace or Delete. The text is marked with a cross-out to indicate that it should be deleted.
Replace Text
To indicate that text should be replaced, select the text and then begin typing. The selected text is marked with a cross-out, an insertion caret appears, and the replacement text is added to a pop-up note.
Don't show again
OK
Diskrete Strukturen
©Ernst W. Mayr
7.6 Das Problem des chinesischen Postboten
566/566
LEA


File Edit View Document Comments Forms Tools Advanced Window Help
566 (1543 of 1543)
172%
Find
Text Edits
Algorithmus: Sei U di
$U| = 2k$.
1 Bestimme $\mathrm{d}(u, v)$
2 Bestimme auf dem
$= \mathrm{d}(u, v)$ ein perfektes
Matching M mini
3 Füge die den Kant
in G ein und
bestimme im resul
r ist eine Lösung.
Indicating Text Edits
The Text Edits tool allows you to use the keyboard to create comments which indicate that text in a PDF document should be inserted, deleted, or replaced.
Insert Text
To indicate that text should be inserted, click between words or characters where you want to insert text and then begin typing. An insertion caret appears and the text is added to a pop-up note.
Delete Text
To indicate that text should be deleted, select the text and then press Backspace or Delete. The text is marked with a cross-out to indicate that it should be deleted.
Replace Text
To indicate that text should be replaced, select the text and then begin typing. The selected text is marked with a cross-out, an insertion caret appears, and the replacement text is added to a pop-up note.
Don't show again
OK
Diskrete Strukturen
©Ernst W. Mayr
7.6 Das Problem des chinesischen Postboten
566/566
LEA


File Edit View Document Comments Forms Tools Advanced Window Help
566 (1543 of 1543)
172%
Find
Text Edits
Algorithmus: Sei U die Menge der Knoten ungeraden Grades, $|U| = 2k$.
1 Bestimme $\mathrm{d}(u, v)$ für alle
2 Bestimme auf dem K_{2k} m
$u, v\}) = \mathrm{d}(u, v)$ ein perfektes
Matching M minimalen C
3 Füge die den Kanten in M
n Pfade in G ein und
bestimme im resultierende
s. Dieser ist eine Lösung.
ConfigFree
Switching profile now...
Cancel
Diskrete Strukturen
©Ernst W. Mayr
7.6 Das Problem des chinesischen Postboten
566/566
LEA