2. Klausur zur Vorlesung Algorithmentechnik Wintersemester 2005/2006

18. April 2006

Hier Aufkleber mit Name und Matrikelnr. anbringen

Vorname: ________________________

Nachname: ________________________

Matrikelnummer: ________________________

Beachten Sie:

- Bringen Sie den Aufkleber mit Ihrem Namen und Matrikelnummer auf diesem Deckblatt an und beschriften Sie jedes Aufgabenblatt mit Ihren Namen und Matrikelnummer.
- Schreiben Sie die Lösungen auf die Aufgabenblätter und Rückseiten. Zusätzliches Papier erhalten Sie bei Bedarf von der Aufsicht.
- Zum Bestehen der Klausur sind **20** der möglichen **60** Punkte hinreichend.
- Es sind keine Hilfsmittel zugelassen.

Aufgabe	1	2	3	4	5	6	7	8	9	10
a	3	4	1	1	5	1	4	2	3	–
b	5	–	1	3	–	3	3	3	–	–
c	–	–	1	2	–	3	–	–	–	–
$\sum$	8	4	3	6	5	7	7	5	3	12
$\sum$	60									
a										–
b		–			–				–	–
c	–	–			–		–	–	–	–
$\sum$										
$\sum$										

Aufgabe 1: $3 + 5 = 8$ Punkte

(a) Für zwei Algorithmen $\mathcal{A}$ und $\mathcal{B}$ gelten die Rekursionsabschätzungen

$$T_{\mathcal{A}}(n) = T_{\mathcal{A}}(n/3) + T_{\mathcal{A}}(n/4) + n^2$$

bzw.

$$T_{\mathcal{B}}(n) = 4 \cdot T_{\mathcal{B}}(n/4) + n^{\alpha} \quad .$$

Für welche Werte von α ist $\mathcal{A}$ asymptotisch schneller als $\mathcal{B}$? Begründen Sie Ihre Aussage.

(b) Eine Warteschlange (FIFO) soll mit Hilfe von zwei STACKS (LIFO) S_{in} und S_{out} implementiert werden. Dabei darf kein zusätzlicher Speicher verwendet werden. Auf den STACKS seien jeweils nur die Operationen S.PUSH(x), $x \leftarrow S$.POP() und S.ISEMPTY () erlaubt. Die POP-Operation soll dabei das oberste Element x vom Stack nehmen und ausgeben.

- Geben Sie eine möglichst effiziente Implementation der Warteschlangenoperationen ENQUEUE (x) (Einstellen des Elements x in die Warteschlange) und DEQUEUE (Ausgabe und Entfernen des Elements, dass am längsten in der Warteschlange ist) an.
- Analysieren Sie die amortisierte Anzahl von PUSH- und POP-Operationen, die im schlechtesten Fall für eine beliebige Folge von n DEQUEUE- oder ENQUEUE-Operationen benötigt werden.

[bitte wenden]

Algorithmus 1 : ENQUEUE (x)

Eingabe : Element x

Datenstrukturen : Stacks S_{in}, S_{out}

Algorithmus 2 : DEQUEUE

Datenstrukturen : Stacks S_{in} und S_{out}

Ausgabe : Element x, das am längsten in der Warteschlange war

Laufzeitanalyse:

Aufgabe 2: 4 Punkte

In der Datenstruktur UNION-FIND (mit den aus der Vorlesung bekannten Modifikationen zur Beschleunigung) wird bei einer FIND-Operation eine Pfadkompression durchgeführt.

(a) Schreiben Sie in Pseudocode eine möglichst effiziente alternative Prozedur FIND, deren Pfadkompression folgende **Seiteneffekte** garantiert:

- Der Suchpfad von x nach r wird bei der Operation FIND(x) nur einmal durchlaufen (nicht zweimal, wie in der Vorlesung).
- Für jeden Knoten w auf dem Suchpfad von x zur Wurzel r ist der Abstand von r zu w nach der Operation FIND(x) höchstens 2.
- Es wird nur konstant viel zusätzlicher Speicher verbraucht.

Algorithmus 3 : Alternatives FIND

Eingabe : UNION-FIND Datenstruktur, Suchknoten x
Ausgabe : Menge in der x enthalten ist
Seiteneffekte : [siehe oben]

Aufgabe 3: $1 + 1 + 1 = 3$ Punkte

(a) Geben Sie die HEAP-Eigenschaft an:

(b) Folgern Sie daraus, dass in jedem HEAP $A[1]$ das maximale Element ist.

(c) Zeigen oder widerlegen: Die drei größten Elemente eines HEAPs befinden sich in $A[1]$, $A[2]$ und $A[3]$.

Aufgabe 4: $1 + 3 + 2 = 6$ Punkte

Sei $G = (V, E)$ ein zusammenhängender Graph, c eine positive Kantengewichtsfunktion, T ein aufspannender Baum minimalen Gewichts für G, und u und v zwei beliebige Knoten in G.
Die *Länge* eines Weges $u = v_0, \ldots, v_k = v$ in G zwischen u und v sei $\sum_{i=1}^{k} c(\{v_{i-1}, v_i\})$.
Die *Schrittweite* eines Weges $u = v_0, \ldots, v_k = v$ in G zwischen u und v sei $\max_{i=1\ldots k} c(\{v_{i-1}, v_i\})$.

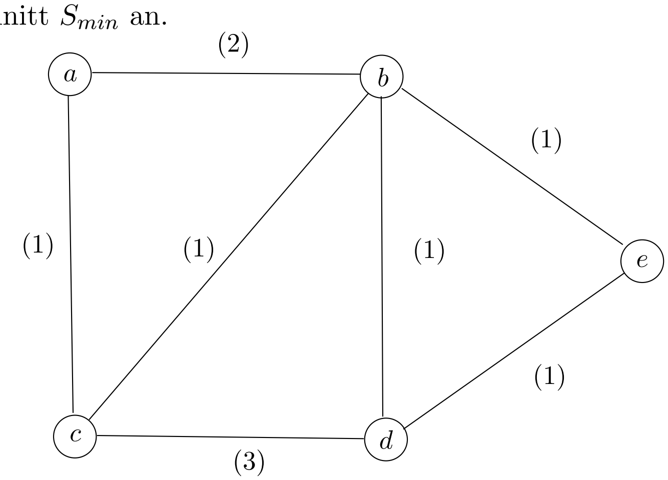
(a) Widerlegen Sie: Der Weg P von u nach v in T ist ein (in G) kürzester Weg von u nach v (das heißt, seine *Länge* ist minimal unter allen Wegen in G zwischen u und v).

(b) Zeigen Sie: Der Weg P von u nach v in T ist ein Weg mit der (in G) kürzesten Schrittweite von u nach v (das heißt, seine *Schrittweite* ist minimal unter allen Wegen in G zwischen u und v).

(c) Zeigen Sie: 2^X (die Menge aller Teilmengen von X), ist ein Matroid über X.

Aufgabe 5: 5 Punkte

Wenden Sie auf den unten abgebildeten Graphen (Kantengewichte in Klammern) den Algorithmus von Stoer & Wagner an. Geben Sie nach jeder Phase die Knoten s und t, den Schnitt der Phase und dessen Gewicht an und zeichnen Sie den nach dem Verschmelzen resultierenden Graphen (mit Kantengewichten). Der Startknoten sei a. Geben Sie zum Schluss den minimalen Schnitt S_{min} an.



minimaler Schnitt $S_{\text{min}} =$

Phase 1

$s =$ $\quad$ $, t =$

Schnitt der Phase $S_1 =$

$c(S_1, V \setminus S_1) =$

Graph:

Phase 2

$s =$ $\quad$, $t =$

Schnitt der Phase $S_2 =$

$c(S_2, V \setminus S_2) =$

Graph:

Phase 3

$s =$ $\quad$, $t =$

Schnitt der Phase $S_3 =$

$c(S_3, V \setminus S_3) =$

Graph:

Phase 4

$s =$ $\quad$, $t =$

Schnitt der Phase $S_4 =$

$c(S_4, V \setminus S_4) =$

Aufgabe 6: $1 + 3 + 3 = 7$ Punkte

Gegeben sei ein Netzwerk D mit ganzzahligen Kantenkapazitäten, auf dem schon ein maximaler, ganzzahliger s-t-Fluss f gegeben ist. Nun wird auf einer Kante e der Kapazitätswert um 1 erhöht.

(a) Zeigen Sie: Der maximale Flusswert erhöht sich dadurch um höchstens 1.

(b) Beschreiben Sie die Arbeitsweise eines Algorithmus, der in Linearzeit einen neuen maximalen s-t-Fluss f' berechnet (falls es einen gibt), nachdem auf einer Kante e der Kapazitätswert um 1 erhöht wurde. [Eine exakte Formulierung in Pseudocode ist nicht erforderlich.] Begründen Sie die Korrektheit.

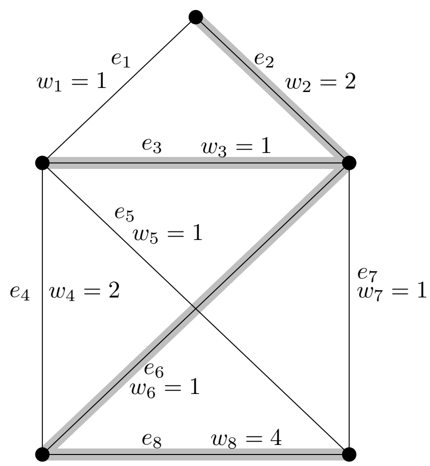
[bitte wenden]

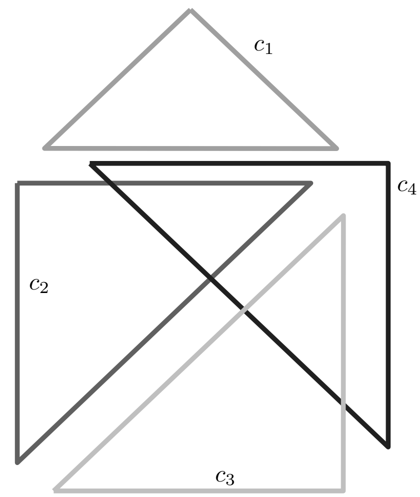
(c) Gegeben sei ein gerichteter Graph $D = (V, E)$ ohne Kantengewichte und zwei Knoten $s, t \in V$. Die Kantenzusammenhangszahl $\kappa(s, t)$ von s und t sei die maximale Anzahl kantendisjunkter s-t-Wege in D. (Eine Menge von Wegen ist kantendisjunkt, wenn keine zwei Wege eine Kante gemeinsam haben.)

Beschreiben Sie die Arbeitsweise eines Algorithmus, der mit Hilfe eines maximalen Flusses die maximale Anzahl kantendisjunkter s-t-Wege ($\kappa(s, t)$) berechnet (Sie können annehmen, dass ein Flussalgorithmus zur Verfügung steht).
Begründen Sie die Korrektheit Ihres Ansatzes knapp.

Aufgabe 7: $4 + 3 = 7$ Punkte

(a) Die folgende, linke Abbildung zeigt den Graphen G_{Nikolaus} mit Kantengewichten. Grau unterlegt ist zudem ein aufspannender Baum T (dick, grau) eingezeichnet (die rechte Abbildung illustriert Teilaufgabe (a)(iii)).





(i) Geben Sie die Fundamentalbasis $\mathcal{B} := \{b_1, b_2, b_3, b_4\}$ zu dem Baum T an.

$$\begin{aligned} \mathcal{B} := \{b_1 &= \\ b_2 &= \\ b_3 &= \\ b_4 &= \qquad\qquad\qquad\qquad \} \end{aligned}$$

(ii) Geben Sie das Gewicht von $\mathcal{B}$ an: $w(\mathcal{B}) =$

(iii) Betrachten Sie die folgende Kreisbasis $\mathcal{C}$ des Graphen, wie in der rechten, obigen Abbildung illustriert:

$$\begin{aligned} \mathcal{C} := \{c_1 &= \{e_1, e_2, e_3\}, \\ c_2 &= \{e_4, e_3, e_6\}, \\ c_3 &= \{e_6, e_7, e_8\}, \\ c_4 &= \{e_5, e_3, e_7\}\} \end{aligned}$$

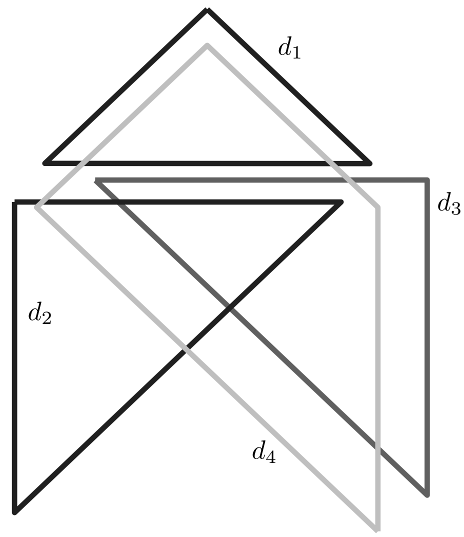
Stellen Sie die Elemente ihrer Fundamentalbasis $\mathcal{B}$ als Linearkombinationen von Elementen aus $\mathcal{C}$ dar:

$$\begin{aligned} b_1 &= \\ b_2 &= \\ b_3 &= \\ b_4 &= \end{aligned}$$

(iv) Sei $\mathcal{D}$ die folgende Menge von Kreisen:

$$\begin{aligned}\mathcal{D} := \{d_1 &= \{e_1, e_2, e_3\},\\ d_2 &= \{e_4, e_3, e_6\},\\ d_3 &= \{e_3, e_5, e_7\},\\ d_4 &= \{e_1, e_2, e_5, e_7\}\}\ .\end{aligned}$$

Es ist $w(\mathcal{D}) = 16$.

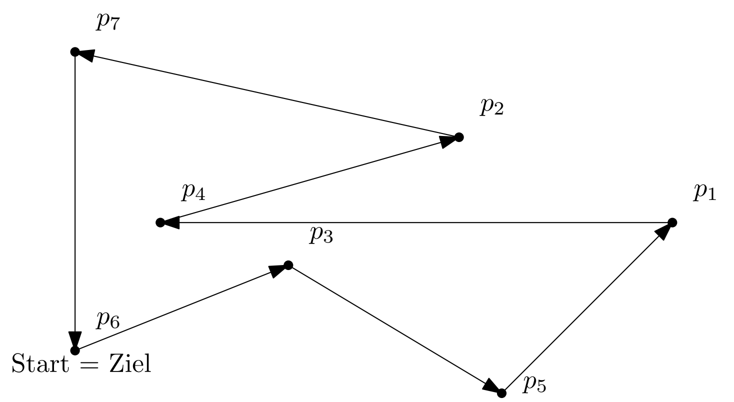


Eine der Mengen $\mathcal{B}$, $\mathcal{C}$ oder $\mathcal{D}$ ist eine minimale Kreisbasis. Welche?

(b) Sei $\mathcal{M}$ die Vereinigung aller Fundamentalbasen eines gegebenen, zusammenhängenden (ungerichteten) Graphen.
Zeigen oder widerlegen Sie: Für jede minimale Kreisbasis $\mathcal{B}$ gilt $\mathcal{B} \subset \mathcal{M}$.

Aufgabe 8: $2 + 3 = 5$ Punkte

Das euklidische *Travelling Salesman Problem* (TSP) lautet wie folgt:
Gegeben: Eine Menge $P = \{p_1, \ldots, p_n\}$ von n verschiedenen Punkten in der Ebene.
Gesucht: Eine möglichst kurze Rundtour durch alle Punkte, also eine Folge $T = p_{i_1}, \ldots, p_{i_{n+1}}$ mit $p_{i_1} = p_{i_{n+1}}$, in der jeder Punkt aus P genau einmal vorkommt (bis auf p_1, der genau zweimal vorkommt) und für die gilt: $\sum_{j=1}^{n} d(p_{i_j}, p_{i_{j+1}}) =$ minimal. Dabei ist $d(a, b)$ der euklidische Abstand von a und b in der Ebene.



Ein beispielhafte TSP-Tour durch alle Punkte. $p_6, p_3, p_5, p_1, p_4, p_2, p_7, p_6$

(a) Zeigen Sie: Ein minimaler Spannbaum (MST) ist kürzer als die kürzeste TSP-Tour.

(b) Geben Sie einen Faktor-2-Approximationsalgorithmus für das euklidische TSP an. [Hinweis: Benutzen Sie dazu einen MST (Sie brauchen keinen MST-Algorithmus anzugeben).]

Algorithmus 4 : APPROX-TSP

Eingabe : n verschieden Punkte $P = \{p_1, \dots, p_n\}$ in der Ebene

Ausgabe : TSP-Tour, die höchstens doppelt so lang ist wie das Optimum

Aufgabe 9: 3 Punkte

Sei Π ein Problem aus $\mathcal{RP}$ mit zugehörigem Algorithmus A. Algorithmus A ist also ein randomisierter einseitiger Monte Carlo Algorithmus. Geben Sie einen randomisierten Algorithmus B an mit polynomialer Laufzeit, für den gilt:

$$\begin{cases} I \in Y_\Pi \longrightarrow \Pr(B(I) \text{ ist „JA“}) \geq \frac{3}{4} \\ I \notin Y_\Pi \longrightarrow \Pr(B(I) \text{ ist „JA“}) = 0 \end{cases}$$

Begründen Sie, dass B die gewünschten Eigenschaften hat.

Algorithmus 5 : Algorithmus B

Eingabe : Probleminstanz I, Algorithmus A
Ausgabe : „JA“ oder „NEIN“

Aufgabe 10: $12 \times 1 = 12$ Punkte

Kreuzen Sie für folgende Aussagen an, ob diese wahr oder falsch sind.

Hinweis: Für jede richtige Antwort gibt es einen Punkt, für jede falsche Antwort wird ein Punkt abgezogen. Es wird keine negative Gesamtpunktzahl für diese Aufgabe geben.

Aussage		
$\log_e(n^2) \in \Theta(\log_2 n)$.	Wahr	Falsch
Heapsort braucht zum Sortieren einer rückwärts sortierten Folge eine asymptotische Laufzeit in $(\Omega(n \log n) \setminus \Theta(n \log n))$.	Wahr	Falsch
Die Umwandlung einer n-elementigen Eingabefolge in einen Heap ist asymptotisch in Linearzeit möglich.	Wahr	Falsch
Jeder ungerichtete kreisfreie Graph mit n Knoten hat genau $n-1$ Kanten.	Wahr	Falsch
Zwei MST eines Graphen haben das gleiche minimale, das gleiche durchschnittliche und das gleiche maximale Kantengewicht.	Wahr	Falsch
Sei T ein MST von G. Wenn jedes Kantengewicht in G mit einer positiven Zahl $\lambda \in \mathbb{R}$ multipliziert wird, dann bleibt T ein MST.	Wahr	Falsch
Sei G ein ungerichteter Graph mit positiven Kantengewichten. Jeder MinCut von G wird von mindestens einer Kante minimalen Gewichts (in G) gekreuzt.	Wahr	Falsch
Ist der zulässige Bereich eines LP beschränkt, dann hat das LP eine eindeutige Lösung.	Wahr	Falsch
Die Laufzeit von Goldberg-Tarjan ist $O(\|V\|^2\|E\|)$, wenn eine Relabel-Operation in $O(\|V\|)$ und eine PUSH-Operation in $O(1)$ implementiert werden kann.	Wahr	Falsch
Das LP zu einem MaxFlow-Problem (auf einem Graphen mit mindestens einer Kante) hat stets einen beschränkten zulässigen Bereich.	Wahr	Falsch
Das MinCut-Problem mit negativen Kantengewichten ist NP-schwer, wenn der Eingabegraph ein Baum ist.	Wahr	Falsch
Im Algorithmus von de Pina (algebraisch) gilt die Invariante: „Für alle zukünftigen Zeugen $S_{k+1,j}$ (mit $j \geq k+1$) gilt $\langle S_{k+1,j}, C_k \rangle = 0$.“	Wahr	Falsch