1. Klausur zur Vorlesung Algorithmentechnik Wintersemester 2008/2009

Hier Aufkleber mit Name und Matrikelnummer 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	Mögliche Punkte					Erreichte Punkte				
	a	b	c	d	Σ	a	b	c	d	Σ
1	4	-	-	-	4		-	-	-	
2	2	1	4	1	8					
3	2	2	2	-	6				-	
4	3	1	3	-	7				-	
5	2	3	2	-	7				-	
6	1	2	2	-	5				-	
7	1	4	-	-	5			-	-	
8	2	1	3	-	6				-	
9	12x1				12					
Σ					60					

Problem 1: Ternärzähler — 4 Punkte

Ein k-Ternärzähler ist ein Array $A[0 \dots k-1]$ mit $A[i] \in \{0,1,2\}$ für $i = 0, \dots, k-1$. Zu Beginn sei $A[i] = 0$ für $i = 0, \dots, k-1$. Wir interpretieren den Inhalt von $A[]$ als eine im Dreiersystem dargestellte Zahl z, d.h.

$$z := \sum_{i=0}^{k-1} A[i] \cdot 3^i$$

Die einzig erlaubte Operation für den Ternärzähler $A[]$ ist die Operation ERHÖHE, die die dargestellte Zahl um 1 erhöht. Jede Änderung eines Eintrages in $A[]$ verursacht Kosten von 1. Zeigen oder widerlegen Sie: Die amortisierten Kosten für n ERHÖHE-Operationen liegen in $O(n)$.

Problem 2: Union-Find / Baumpartition — $2 + 1 + 4 + 1 = 8$ Punkte

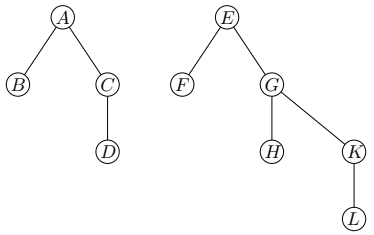
(a) Betrachten Sie die unten abgebildete Union-Find-Datenstruktur. Es sollen nun nacheinander die Operationen

(i) UNION(FIND(B), FIND(G)) und

(ii) FIND(D)

ausgeführt werden.

Zeichnen Sie den Zustand der Datenstruktur jeweils nach Ausführung der Operation (i) und (ii). Die Datenstruktur sei mit Weighted-Union und Pfadkompression implementiert.



Eine *Baum-Partition* eines gewichteten Graphen $G = (V, E)$ mit Gewichtsfunktion $w : E \to \mathbb{R}^+$ und Parameter $\gamma \in \mathbb{R}^+$ ist eine Menge von Kanten $F \subseteq E$ mit folgenden Eigenschaften:

(i) F induziert einen Wald in G.

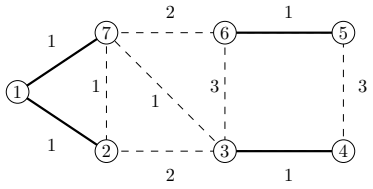
(ii) Für jede Menge $F' \subseteq F$, die einen Baum induziert, gilt

$$w(F') := \sum_{e \in F'} w(e) \leq \gamma \, .$$

Eine Baum-Partition mit Parameter γ heißt *nicht-erweiterbar*, wenn gilt

- Für alle $e \in E \setminus F$ gilt: $F \cup \{e\}$ ist keine Baumpartition mit Parameter γ

(b) Zeigen oder widerlegen Sie, dass die Menge der fett markierten Kanten im folgenden Graphen eine nicht-erweiterbare Baum-Partition des Graphen mit $\gamma = 4$ ist. Die Zahlen, mit denen die Kanten annotiert sind, entsprechen den Gewichten der Kanten.



(c) Formulieren Sie mithilfe der Union-Find-Datenstruktur aus der Vorlesung einen Algorithmus, der eine nicht-erweiterbare Baum-Partition zu einem gegebenen gewichteten Graphen $G = (V, E)$ mit Gewichtsfunktion $w : E \to \mathbb{R}^+$ und einem gegebenen Parameter $\gamma \in \mathbb{R}^+$ berechnet.

Die Ausgabe des Algorithmus sei die Union-Find-Datenstruktur, so dass der Ausdruck FIND(u) = FIND(v) genau dann wahr ist, wenn die Knoten u und v im gleichen Baum der Baum-Partition liegen.

(d) Geben Sie eine möglichst gute obere Schranke für die asymptotische Laufzeit Ihres Algorithmus an. Begründen Sie Ihre Antwort.

Problem 3: Matroide $2 + 2 + 2 = 6$ Punkte

(a) Sei M eine Menge mit m Elementen. Zu $r \in \mathbb{N}$ mit $r \leq m$ sei $\mathcal{U}_r := \{S \subseteq M \ : \ |S| \leq r\}$. Zeigen Sie, dass für alle $r \leq m$ das Tupel $(M,\mathcal{U}_r)$ ein Matroid ist.

(b) Sei $\mathcal{M} = (S,\mathcal{U})$ ein Matroid und sei $\mathcal{B} \subseteq \mathcal{U}$ die Menge der Basen von $\mathcal{M}$, d.h. die Menge der maximalen unabhängigen Mengen in $\mathcal{U}$. Seien $B_1, B_2 \in \mathcal{B}$ mit $B_1 \neq B_2$ und sei $x \in B_1 \setminus B_2$. Zeigen Sie, dass es ein $y \in B_2 \setminus B_1$ gibt, so dass $(B_1 \setminus \{x\}) \cup \{y\} \in \mathcal{B}$ gilt.

(c) Sei $G = (V, E)$ ein zusammenhängender Graph und seien T_1, T_2 zwei aufspannende Bäume in G. Zu T_1, T_2 bezeichne $E(T_1)$ und $E(T_2)$ jeweils die Menge der Kanten. Sei e_1 eine beliebige Kante in $E(T_1)$. Zeigen Sie: Dann gibt es eine Kante e_2 in $E(T_2)$, so dass $(E(T_1) \setminus \{e_1\}) \cup \{e_2\}$ einen aufspannenden Baum in G induziert.

Anmerkung: Aus der Vorlesung ist bekannt, dass das System aller Teilmengen $E' \subseteq E$, die einen Wald in G induzieren, ein Matroid ist.

Problem 4: Zulässige Gradsequenz $3 + 1 + 3 = 7$ Punkte

(a) Sei $\mathcal{N}$ ein Flussnetzwerk bestehend aus einem einfachen, gerichteten Graphen $D = (V, E)$ mit ausgezeichneten Knoten s (Quelle) und t (Senke) sowie ganzzahligen Kantenkapazitäten $c : E \to \mathbb{N}$.

Zeigen Sie: Der Algorithmus von Goldberg-Tarjan berechnet einen ganzzahligen Maximalfluss.

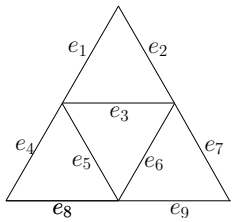
(b) Gegeben sei eine Menge von Knoten $V = \{v_1, \ldots, v_n\}$ sowie Eingangsgrade $e_1, \ldots, e_n \in \{0, \ldots, n-1\}$ und Ausgangsgrade $a_1, \ldots, a_n \in \{0, \ldots, n-1\}$. Das Problem ZULÄSSIGE GRADSEQUENZ besteht nun darin zu entscheiden, ob es einen gerichteten Graphen $G = (V, E)$ ohne Schleifen und Mehrfachkanten gibt, so dass Knoten v_j Eingangsgrad e_j und Ausgangsgrad a_j für alle $j = 1, \ldots, n$ hat.

Gegeben seien die Eingangsgrade $e_1 = 1, e_2 = 1, e_3 = 1$ und Ausgangsgrade $a_1 = 0, a_2 = 2, a_3 = 0$. Zeigen oder widerlegen Sie, dass es einen gerichteten Graphen ohne Schleifen und Mehrfachkanten mit den gewünschten Graden gibt.

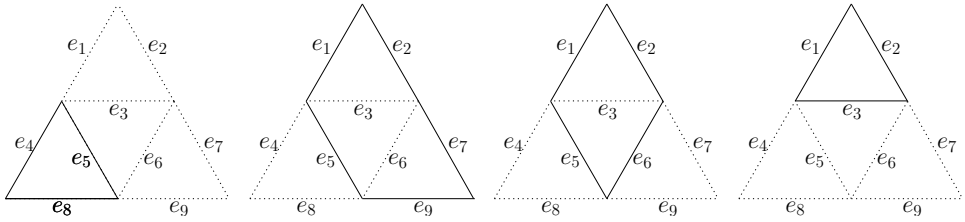
(c) Modellieren Sie das Problem ZULÄSSIGE GRADSEQUENZ als Flussproblem. Die Menge der Knoten des Flussnetzwerkes sei $W := \{s\} \cup \{\overrightarrow{v_i}, \overleftarrow{v_i} \mid 1 \leq i \leq n\} \cup \{t\}$. Geben Sie das Flussnetzwerk formal an. Welche Bedingung muss ein maximaler Fluss f in Ihrem Netzwerk erfüllen, damit ein Graph mit den gewünschten Eigenschaften existiert?

Problem 5: Kreisbasen $2 + 3 + 2 = 7$ Punkte

Gegeben ist der folgende Graph $G = (V, E)$:



(a) Außerdem ist folgende Fundamentalkreisbasis $\{e_4, e_5, e_8\}, \{e_1, e_2, e_5, e_7, e_9\}, \{e_1, e_2, e_5, e_6\}, \{e_1, e_2, e_3\}$ von G gegeben:



Geben Sie alle Spannbäume von G an, die diese Fundamentalkreisbasis induzieren und argumentieren Sie, dass es keine weiteren gibt.

(b) Alle Kanten in G haben einheitliches Gewicht. Geben Sie eine minimale Kreisbasis B von G an. Zeigen Sie zusätzlich, dass B eine minimale Kreisbasis ist (zu zeigen sind Minimalität UND die Kreisbasen-Eigenschaft).

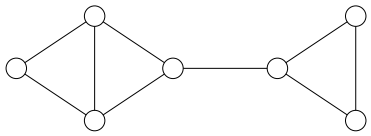
(c) Gegeben seien t ungerichtete, zusammenhängende Graphen $G_1 = (V_1, E_1), \ldots, G_t = (V_t, E_t)$ mit Kreisbasen $B_1, \ldots, B_t$. Diese Graphen werden zu einem Graphen $G = (V_1 \cup \ldots \cup V_t, E_1 \cup \ldots \cup E_t \cup \widetilde{E})$ vereinigt. Die zusätzlichen Kanten $\widetilde{E}$ werden so eingefügt, dass G zusammenhängend ist. Beachten Sie, dass dabei neue Kreise entstehen können.

Geben Sie eine Formel für die Kardinalität einer Kreisbasis von G in Abhängigkeit der gegebenen Kreisbasen $B_1, \ldots, B_t$, der Kantenmenge $\widetilde{E}$ und des Parameters t an.

Problem 6: Unabhängige Menge — $1 + 2 + 2 = 5$ Punkte

Gegeben sei ein Graph $G = (V, E)$. Eine *unabhängige Menge* von G ist eine Menge von Knoten $V' \subseteq V$, so dass für jede Kante in G höchstens ein Endknoten in V' ist. Das Optimierungsproblem UNABHÄNGIGE MENGE besteht darin, eine unabhängige Menge maximaler Kardinalität zu finden.

(a) Zeichnen Sie eine unabhängige Menge maximaler Kardinalität in den folgenden Graphen ein.



(b) Formulieren Sie das Problem UNABHÄNGIGE MENGE als ILP. Erklären Sie die Bedeutung der Variablen und Nebenbedingungen.

(c) Betrachten Sie das folgende Lineare Programm:

$$\min \sum_{\{u,v\}\in E} X_{\{u,v\}} \tag{1}$$

$$X_{\{u,v\}} \in \{0,1\} \qquad \{u,v\} \in E \tag{2}$$

$$\sum_{u:\ \{u,v\}\in E} X_{\{u,v\}} \geq 1 \qquad v \in V \tag{3}$$

Geben Sie jeweils eine sinnvolle Interpretation der Variablen und Nebenbedingungen an und interpretieren Sie die Zielfunktion des Programms entsprechend.

Problem 7: Minimaler Metrischer Steiner-Baum $1 + 4 = 5$ Punkte

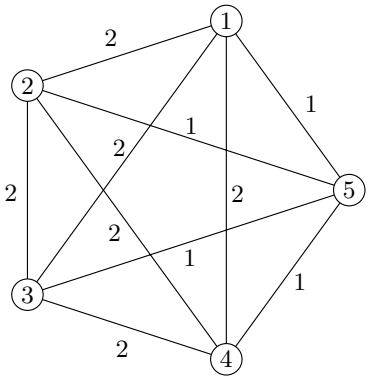
Gegeben sei ein vollständiger Graph $G = (V, E)$ und eine Knotenmenge $S \subset V$, sowie eine Gewichtsfunktion $w : E \to \mathbb{N}$, die die Dreiecksungleichung erfüllt, d.h. für alle Knoten $u, v, x \in V$ gilt $w(u, x) \leq w(u, v) + w(v, x)$.

Gesucht ist ein minimaler Steiner-Baum T, d.h. ein Baum in G, der alle Knoten aus S enthält, so dass das Gewicht

$$w(T) := \sum_{e\in E(T)} w(e)$$

minimal ist. Beachten Sie, dass T nicht alle Knoten aus V enthalten muss.

(a) Gegeben sei der folgende Graph und $S := \{1, 2, 3, 4\}$. Die Gewichte sind direkt an den Kanten annotiert. Zeichnen Sie einen minimalen Steiner-Baum ein.



(b) Betrachten Sie den folgenden Algorithmus:

Algorithmus 1 : Steiner-Baum

Eingabe : vollständiger Graph $G = (V, E)$, $S \subseteq V$, Gewichtsfunktion $w : E \to \mathbb{R}$, die die Dreiecksungleichung erfüllt

Ausgabe : Steiner-Baum T für S

1 $E' \leftarrow \{\{u, v\} \mid \{u, v\} \subseteq S\}$
2 $G' \leftarrow (S, E')$
3 $T \leftarrow$ Spannbaum minimalen Gewichts in G'

Zeigen Sie, dass Algorithmus 1 ein 2-Approximationsalgorithmus ist, d.h. dass

$$\mathcal{A}(I) \leq 2 \cdot OPT(I)$$

gilt. Hierbei sei I eine Instanz des Problems, $\mathcal{A}(I)$ das Gewicht des von Algorithmus 1 zurückgegebenen Baumes sowie $OPT(I)$ das Gewicht einer optimalen Lösung.

Hinweis: Betrachten Sie die Rundtour, die durch das Traversieren eines optimalen Steiner-Baumes induziert wird.

Problem 8: Randomisierte Algorithmen $2 + 1 + 3 = 6$ Punkte

Gegeben sei folgender Las-Vegas Algorithmus UNBEKANNTERALGORITHMUS($A[]$). Die Eingabe des Algorithmus sei ein Array $A[]$ mit Einträgen $A[1], \ldots, A[n] \in \mathbb{N}$. Die Einträge von $A[]$ seien paarweise verschieden.

Algorithmus 2 : UNBEKANNTERALGORITHMUS($A[]$)

Input : Ein Array $A[]$ von n paarweise verschiedenen Zahlen
Output : Eine Zahl

```
1  r ← Random(1,n)
2  m ← A[r]
3  found ← true
4  for i=1 to n do
5  |  if A[i] > m then
6  |  |  found ← false
7  if found=true then
8  |  return m
9  else
10 |  return UnbekannterAlgorithmus(A[])
```

(a) Was berechnet UNBEKANNTERALGORITHMUS($A[]$)?

(b) Geben Sie die Wahrscheinlichkeit an, dass sich UNBEKANNTERALGORITHMUS($A[]$) genau 10 mal selbst rekursiv aufruft (also Zeile 10 genau 10 mal ausgeführt wird).

(c) Was ist die erwartete Anzahl an rekursiven Selbstaufrufen von UNBEKANNTERALGORITHMUS($A[]$)? Begründen Sie Ihr Ergebnis.

Problem 9: $12 \times 1 = 12$ 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.

	Wahr	Falsch
Bei amortisierter Analyse wird die erwartete Laufzeit über alle Eingaben gemittelt.	☐	☐
Der parallele Algorithmus $\mathcal{A}$ benötige zum Sortieren eines Arrays mit n Elementen und m Prozessoren die Laufzeit $O(\frac{n \log n}{\log m})$. Dies ist kostenoptimal.	☐	☐
Sei P ein ILP für ein ganzzahliges Minimierungsproblem und sei R die Relaxierung von P. Dann ist die optimale Lösung von R eine untere Schranke für die optimale Lösung von P.	☐	☐
Der Algorithmus First Fit (FF) ist ein APAS für Bin Packing.	☐	☐
Sei G ein Graph und v ein Knoten in G. Dann ist mindestens entweder v oder alle Nachbarn von v in jedem Vertex Cover von G.	☐	☐
Der Algorithmus von De Pina hat eine Gesamtlaufzeit von $\mathcal{O}(mn \log n)$.	☐	☐
Sei G ein Graph mit n Knoten, m Kanten und k Zusammenhangskomponenten. Dann hat jede Kreisbasis von G mindestens $m-n+k$ Elemente.	☐	☐
Sei G ein Graph und sei C ein beliebiger Kreis in G. Dann gilt: Die leichteste Kante von C ist in jedem minimalen Spannbaum enthalten.	☐	☐
Die average-case Laufzeit eines Algorithmus $\mathcal{A}$ ist eine untere Schranke für die worst-case Laufzeit von $\mathcal{A}$.	☐	☐
Falls es für jedes Problem in $\mathcal{FPT}$ einen Lösungsalgorithmus gibt, der polynomiell in der Eingabegröße ist, dann gilt: $\mathcal{P} = \mathcal{NP}$.	☐	☐
Sei P ein lineares Programm und seien x und y zwei zulässige Lösungen von P. Dann ist $\lambda x + (1-\lambda)y$ für $\lambda \in [\frac{1}{4}, \frac{3}{4}]$ eine zulässige Lösung von P.	☐	☐
Jeder Präfluss ist ein Fluss, aber nicht jeder Fluss ist auch ein Präfluss.	☐	☐