1. Klausur zur Vorlesung
Algorithmen II
Wintersemester 2012/2013

Hier Aufkleber mit Name und Matrikelnummer anbringen

Vorname: ______________________

Nachname: ______________________

Matrikelnummer: ______________________

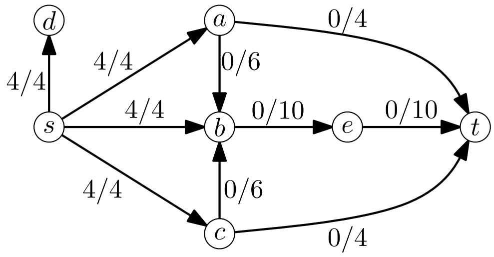
Beachten Sie:

- Bringen Sie den Aufkleber mit Ihrem Namen und Matrikelnummer auf diesem Deckblatt an und beschriften Sie jedes Aufgabenblatt mit Ihrem 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	Σ	a	b	c	Σ
1	1	5	–	6			–	
2	2,5	3	1,5	7				
3	1	1	3	5				
4	4	2	1	7				
5	1,5	3,5	3	8				
6	3	2	4	9				
7	2	1	2	5				
8	2	2	2	6				
9	7x1			7				
Σ				60				

Problem 1: Algorithmus von Goldberg & Tarjan 1 + 5 = 6 Punkte

Betrachten Sie folgendes Flussnetzwerk mit Quelle s und Senke t, das den Zustand des Goldberg-Tarjan-Algorithmus direkt nach der Initialisierung zeigt. Jede Kante e ist mit dem aktuellen Präfluss $f(e)$ sowie der Kapazität $c(e)$ im Format $f(e)/c(e)$ beschriftet.



(a) Geben Sie die zulässige Markierung $\text{dist}(v)$ sowie den Flussüberschuss $e(v)$ für jeden Knoten v nach der Initialisierung an und zählen Sie alle aktiven Knoten auf.

(b) Geben Sie eine Sequenz bestehend aus RELABEL- und PUSH-Operationen an, sodass folgendes gilt:

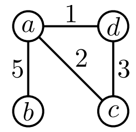
- Die Sequenz darf beliebig viele RELABEL-Operationen enthalten.
- Die Sequenz enthält genau zwei nicht-saturierende PUSH-Operationen.
- Die Sequenz enthält genau drei saturierende PUSH-Operationen.

Geben Sie außerdem für jede Operation an, wie sich die zulässige Markierung $\text{dist}(\cdot)$, sowie die Flussüberschüsse $e(\cdot)$ ändern.

Hinweis: Führen Sie die zwei nicht-saturierenden vor den drei saturierenden PUSH-Operationen aus.

Problem 2: Global minimaler Schnitt $2{,}5 + 3 + 1{,}5 = 7$ Punkte

(a) Wenden Sie den Stoer-Wagner Algorithmus auf den nebenstehenden Graphen an. Die Kantengewichte sind an den Kanten notiert. Geben Sie für jede Phase i die Knoten s_i und t_i, den Schnitt der Phase und dessen Gewicht an. Zeichnen Sie den nach dem Verschmelzen resultierenden Graphen mit Kantengewichten. Der Startknoten in jeder Phase enthalte a. Geben Sie zum Schluss den minimalen Schnitt $S_{\min}$ an.



Phase 1:

$s_1 =$ $t_1 =$ Graph nach Verschmelzen:

Schnitt der Phase $S_1 =$

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

Phase 2:

$s_2 =$ $t_2 =$ Graph nach Verschmelzen:

Schnitt der Phase $S_2 =$

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

Phase 3:

$s_3 =$ $t_3 =$ Graph nach Verschmelzen:

Schnitt der Phase $S_3 =$

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

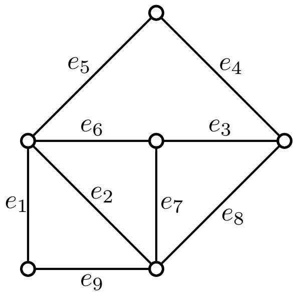
Minimaler Schnitt $S_{\min} =$

(b) Beweisen Sie, dass der Schnitt der Phase mit minimalem Gewicht ein global minimaler Schnitt ist. Sie dürfen verwenden, dass der Schnitt jeder Phase ein minimaler *st*-Schnitt ist.

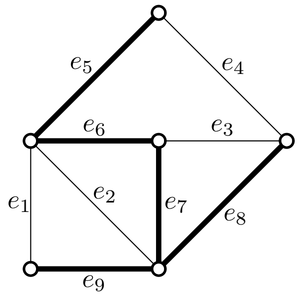
(c) Geben Sie einen Graphen an, bei dem der Stoer-Wagner Algorithmus je nach Ausführung unterschiedliche Schnitte zurückgeben kann. Begründen Sie kurz warum. An welchen Stellen im Algorithmus können Entscheidungen getroffen werden?

Problem 3: Kreisraum & Kreisbasen $\qquad 1 + 1 + 3 = 5$ Punkte

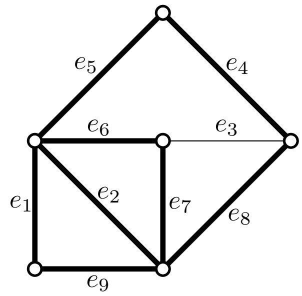
Gegeben sei der unten abgebildete Graph $G = (V, E)$, zusammen mit einem Spannbaum T und einem Kreis C (jeweils fett eingezeichnet).



Graph $G = (V, E)$



Spannbaum T



Kreis C

(a) Geben Sie die Fundamentalbasis B_T des Kreisraums von G bezüglich des Spannbaums T an.

(b) Stellen Sie den Kreis C als Summe von Kreisen aus B_T dar.

(c) Geben Sie eine Kreisbasis B des Kreisraums von G an, die **keine** Fundamentalbasis ist. Begründen Sie warum B eine Basis, aber keine Fundamentalbasis ist.

Problem 4: Matroide $4 + 2 + 1 = 7$ Punkte

(a) Sei $G = (V, E)$ ein zusammenhängender Graph. Eine Teilmenge $E_w \subseteq E$ induziert einen Wald, wenn $G_w = (V, E_w)$ keinen Kreis enthält. Zeigen Sie, dass das Unabhängigkeitssystem $\mathcal{U}_w = \{E_w \mid E_w \subseteq E \text{ induziert einen Wald}\}$ ein Matroid ist.

Hinweis: Sie dürfen verwenden, dass ein Wald $G_w = (V, E_w)$ genau $|V| - k$ Kanten enthält, wobei k die Anzahl der Zusammenhangskomponenten in G_w ist.

(b) Zeigen Sie: Jeder Spannbaum in G ist eine Basis von $\mathcal{U}_w$ und umgekehrt.

(c) Sei $w\colon E \longrightarrow \mathbb{R}$ eine Gewichtsfunktion auf den Kanten von G. Berechnet der folgende Algorithmus einen minimalen Spannbaum in G? Begründen Sie Ihre Antwort.

Algorithmus 1 : MAYBE-MST

Eingabe : Ein zusammenhängender Graph G
Ausgabe : Ein Teilgraph $G' = (V, E')$

Sortiere E aufsteigend nach Gewichtsfunktion w, sei $e_1, \ldots, e_m$ die Sortierung
$E' \leftarrow \emptyset$
Für $i = 1$ **bis** m
    Wenn $E' \cup \{e_i\} \in \mathcal{U}_w$
        $E' \leftarrow E' \cup \{e_i\}$
gib $G' = (V, E')$ zurück

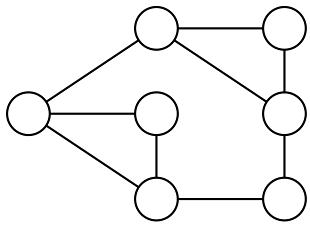
Problem 5: Parametrisierte Algorithmen $1{,}5 + 3{,}5 + 3 = 8$ Punkte

Das Problem DOMINIERENDE MENGE für Graphen mit Maximalgrad d ist wie folgt definiert.

Gegeben: Ein Graph $G = (V, E)$, sodass jeder Knoten maximal d Nachbarn hat, sowie ein Parameter $k = d + \ell$.

Gesucht: Eine Knotenmenge $V' \subseteq V$ mit $|V'| \leq \ell$, sodass für jeden Knoten $v \in V$ der Knoten v und/oder einer seiner Nachbarn in V' enthalten ist.

(a) Gegeben sei eine Instanz von DOMINIERENDE MENGE bestehend aus dem unten abgebildeten Graphen mit Maximalgrad $d = 3$, sowie dem Parameter $k = 5$. Markieren Sie $\ell = 2$ Knoten, die diese Instanz von DOMINIERENDE MENGE lösen.



(b) Geben Sie einen FPT-Algorithmus für DOMINIERENDE MENGE an. Begründen Sie, warum Ihr Algorithmus korrekt ist.

Hinweis: Benutzen Sie einen Entscheidungsbaum.

Fortsetzung Aufgabenteil (b)

(c) Zeigen Sie mit Hilfe einer Laufzeitanalyse, dass Ihr Algorithmus aus Aufgabenteil (b) ein FPT-Algorithmus ist.

Problem 6: Approximationsalgorithmen $3 + 2 + 4 = 9$ Punkte

Das Problem INDEPENDENT SQUARES sei wie folgt definiert:

Gegeben: Menge $Q = \{q_1, \dots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene.

Gesucht: Möglichst große unabhängige Menge $S \subseteq Q$. Dabei heißt $S \subseteq Q$ *unabhängig*, falls für alle $q_i, q_j \in S$ mit $i \neq j$ gilt, dass q_i und q_j sich nicht schneiden.

Betrachten Sie den Algorithmus SWEEPLINE, der eine inklusionsmaximale unabhängige Teilmenge $S \subseteq Q$ berechnet.

Algorithmus 2 : SWEEPLINE

Eingabe : Menge $Q = \{q_1, \dots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene mit Mittelpunkten $c_1, \dots, c_n$, sodass für die x-Koordinaten der Mittelpunkte gilt $x(c_1) < \ldots < x(c_n)$.

Ausgabe : Unabhängige Menge $S \subseteq Q$.

$S \leftarrow \emptyset$

Für $i = 1, \dots, n$

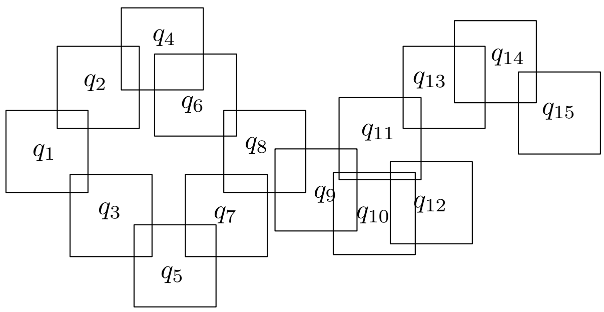
  Wenn $q_i \in Q$

    $S \leftarrow S \cup \{q_i\}$

    $Q \leftarrow Q \setminus (\{q_i\} \cup \{q_j \in Q \mid q_j$ und q_i schneiden sich.$\})$

return S

(a) Geben Sie die unabhängige Menge S an, die SWEEPLINE auf folgender Instanz berechnet. Geben Sie zudem eine kardinalitätsmaximale unabhängige Menge S' derselben Instanz an.



(b) Geben Sie eine Familie $Q_1, Q_2, Q_3, \ldots$ gleichgroßer, achsenparalleler Quadrate an, sodass gilt $|Q_n| \in \Theta(n)$ und $|\textsc{SweepLine}(Q_n)| = \frac{1}{2}|\operatorname{OPT}(Q_n)|$ für alle $n \in \mathbb{N}$. Dabei bezeichnet $\operatorname{OPT}(Q)$ die kardinalitätsmaximale unabhängige Menge von Q. Begründen Sie Ihre Antwort.

(c) Zeigen Sie, dass für jede Instanz Q die Ungleichung $|\textsc{SweepLine}(Q)| \geq \frac{1}{2}|\operatorname{OPT}(Q)|$ gilt. Dabei bezeichnet $\operatorname{OPT}(Q)$ die kardinalitätsmaximale unabhängige Menge von Q.

Problem 7: Randomisierte Algorithmen $2 + 1 + 2 = 5$ Punkte

Ein Graph $G = (V, E)$ ist *d-regulär*, wenn jeder Knoten genau d Nachbarn hat.

Algorithmus 3 : RANDOMISIERTER ALGORITHMUS

Eingabe : d-regulärer Graph $G = (V, E)$, Wahrscheinlichkeit $p \in [0, 1]$
Ausgabe : Teilmenge $S \subseteq V$

$S \leftarrow \emptyset$
Für *jedes* $v \in V$
  $x_v \leftarrow$ blau
  $x_v \leftarrow$ rot mit Wahrscheinlichkeit p
Für *jedes* $v \in V$
  Wenn $x_v = rot$ **oder** $x_u =$ *blau für alle Nachbarn u von v*
    $S \leftarrow S \cup \{v\}$

(a) Zeigen Sie: Das Ergebnis S von Algorithmus 3 ist eine dominierende Menge, das heißt, für alle $v \in V$ gilt $v \in S$ oder $u \in S$ für einen Nachbar u von v.

(b) Geben Sie die Wahrscheinlichkeit $P(v \in S)$ dafür an, dass ein gegebener Knoten v in S liegt.

(c) Geben Sie den Erwartungswert von $|S|$ in Abhängigkeit von p, d und $n = |V|$ an.

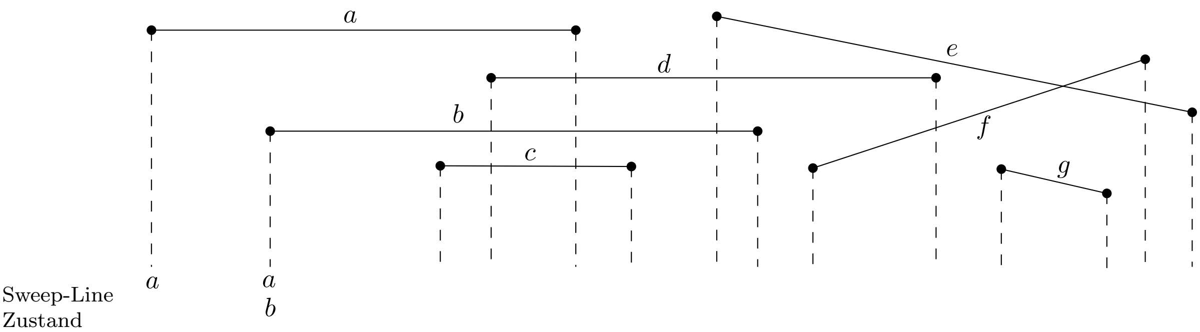
Hinweis: Betrachten Sie die Zufallsvariablen $Y_v = \begin{cases} 1, & \text{wenn } v \in S \\ 0, & \text{sonst.} \end{cases}$

Problem 8: Schnittpunkte von Strecken 2 + 2 + 2 = 6 Punkte

(a) Geben Sie eine Familie von Streckenmengen $S_1, S_2, \ldots$ mit $|S_n| \in O(n)$ an, sodass S_n mindestens $\Omega(n^2)$ sich schneidende Streckenpaare enthält. Begründen Sie Ihre Antwort.

(b) Um zu überprüfen ob zwei Strecken sich schneiden, soll auf die unten abgebildeten Strecken der aus der Vorlesung bekannte Sweep-Line-Algorithmus INTERSECT angewendet werden. Die potentiellen Haltepunkte (Event-Point Schedule) sind durch gestrichelte Linien markiert. Die ersten beiden Haltepunkte sind bereits mit dem Sweep-Line Zustand beschriftet.

Vervollständigen Sie die Abbildung, indem Sie jeden Haltepunkt mit dem zugehörigen Sweep-Line Zustand beschriften. *Achtung:* potentielle Haltepunkte an denen der Algorithmus nicht anhält sollen nicht beschriftet werden.



(c) Sei S eine Menge von Strecken mit paarweise disjunkten Endpunkten und seien $a, b, c, d \in S$ vier unterschiedliche Strecken, wobei es in S genau zwei Schnittpunkte zwischen Streckenpaaren gibt, nämlich zwischen a und b sowie zwischen c und d.

Zeigen oder widerlegen Sie: Wenn a und b vor c und d in den Sweep-Line Zustand aufgenommen werden, dann findet der Algorithmus INTERSECT aus der Vorlesung immer den Schnittpunkt zwischen a und b.

Problem 9: Verschiedenes $7 \times 1 = 7$ Punkte

Jeder der folgenden Aussagenblöcke umfasst mehrere Einzelaussagen. Die sich unterscheidenden Aussagenbausteine sind durch Kästchen gekennzeichnet. Kreuzen Sie genau jene Bausteine an, die in einer wahren Einzelaussage enthalten sind. Jeder Aussagenblock enthält mindestens eine wahre Einzelaussage. Unvollständig oder falsch angekreuzte Aussagenblöcke werden mit null Punkten bewertet. Sie erhalten einen Punkt für jeden Aussagenblock, für den Sie genau die richtige Menge an Aussagenbausteinen angekreuzt haben.

Ein Algorithmus mit asymptotischer Laufzeit

- ☐ $\Theta(n^2 + n \log n \cdot 3^{k!})$
- ☐ $\Theta(n \log n + 3^{3k} \cdot n!)$
- ☐ $\Theta(2^k \cdot n^k + \sqrt{k} \cdot 3n)$
- ☐ $\Theta(\log n)$

ist ein FPT-Algorithmus, wobei n die Eingabegröße und k der Parameter ist.

Die konvexe Hülle

- ☐ von n Strecken kann in $O(n \log n)$ Zeit berechnet werden.
- ☐ von n Punkten hat immer die Größe mindestens $\Omega(n)$.
- ☐ eines Dreiecks ist das Dreieck selbst.
- ☐ eines einfachen Polygons ist das einfache Polygon selbst.

Ein maximaler st-Fluss in einem gerichteten Graphen $G = (V, E)$ mit Kapazitäten 1

- ☐ kann in polynomieller Zeit berechnet werden, wenn eine minimale Kreisbasis von G gegeben ist.
- ☐ kann den Wert $|V| - 1$ haben.
- ☐ hat ganzzahligen Wert.
- ☐ kann den Wert 0 haben.

Eine Menge von Algorithmen $\{A_\varepsilon \mid \varepsilon \in \mathbb{R}^+\}$ mit Laufzeit $O(\left(\frac{1}{\varepsilon}\right)^5 \cdot 2^\varepsilon \cdot n^3)$ und relativer Approximationsgüte $1 + \varepsilon$ ist ein

- ☐ PAS
- ☐ FPAS
- ☐ AFPAS
- ☐ APAS

Sei G ein Graph und I ein maximales INDEPENDENT SET der Größe k in G. Dann enthält I

- ☐ alle isolierten Knoten. (Ein Knoten ist isoliert, wenn er Grad 0 hat.)
- ☐ alle Knoten mit Grad maximal k.
- ☐ keinen Knoten mit Grad größer als $n - k$.
- ☐ für jede Kante mindestens einen der beiden Endknoten.

Sei G ein ungewichteter Graph, T ein Spannbaum in G, sowie B_T die Fundamentalbasis des Kreisraums von G bezüglich T. Dann

- ☐ enthält B_T nur einfache Kreise.
- ☐ ist B_T eine minimale Kreisbasis.
- ☐ ist jede Kante von G in maximal zwei Kreisen von B_T enthalten.
- ☐ ist jede Kante von G in mindestens einem Kreis von B_T enthalten.

Beim (h, k)-Paging

- ☐ hat der Online-Algorithmus einen kleineren Cache zur Verfügung als der optimale Offline-Algorithmus.
- ☐ hat der optimale Offline-Algorithmus mindestens $h - k$ Fehlzugriffe.
- ☐ sind konservative Online-Algorithmen k-kompetitiv.
- ☐ hat der optimale Offline-Algorithmus mindestens so viele Fehlzugriffe wie der Online-Algorithmus.