2. Klausur zur Vorlesung
Algorithmen II
Wintersemester 2012/2013

Hier Aufkleber mit Name und Matrikelnummer anbringen

Vorname: ______________________________

Nachname: ______________________________

Matrikelnummer: ______________________________

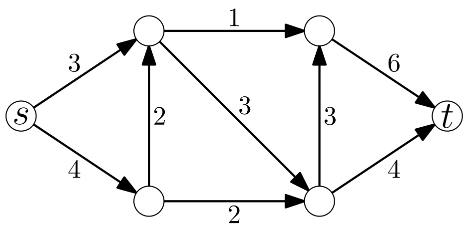
Beachten Sie:

- 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	d	Σ	a	b	c	d	Σ
1	1	1	3	2	7					
2	3,5	2,5	-	-	6			-	-	
3	1,5	3	1,5	-	6				-	
4	1,5	2	1,5	-	5				-	
5	1,5	5	3,5	-	10				-	
6	1,5	2	2,5	-	6				-	
7	2	1	2	-	5				-	
8	1,5	1	2,5	3	8					
9	7x1				7					
Σ					60					

Problem 1: Flüsse und lineare Programmierung $\quad 1 + 1 + 3 + 2 = 7$ Punkte

(a) Betrachten Sie das unten stehende Flussnetzwerk, in dem jede Kante mit ihrer Kapazität beschriftet ist. Geben Sie einen maximalen st-Fluss an, indem Sie die Flusswerte an die Kanten schreiben.



(b) Zeichnen Sie in das Netzwerk aus Aufgabenteil (a) einen minimalen st-Schnitt ein. Begründen Sie, warum Ihr Schnitt minimal ist.

(c) Sei (D, s, t, c) ein Flussnetzwerk bestehend aus einem gerichteten Graph $D = (V, E)$, der Quelle s, der Senke t, sowie der Kapazitätsfunktion $c\colon E \longrightarrow \mathbb{N}$. Formulieren Sie die Berechnung eines maximalen st-Flusses als lineares Programm (LP).

Hinweis: Benutzen Sie für jede Kante $(u, v) \in E$ eine Variable $x_{u,v}$, die angibt, wie viel Fluss auf der Kante (u, v) fließt.

(d) Sei zusätzlich zu Aufgabenteil (c) eine Kostenfunktion $w\colon E \longrightarrow \mathbb{R}$ gegeben, mit der Bedeutung, dass ein Fluss $f(e)$ auf der Kante e die Kosten $f(e) \cdot w(e)$ verursacht. Geben Sie ein LP an, das einen st-Fluss mit festem Wert b und minimalen Kosten berechnet.

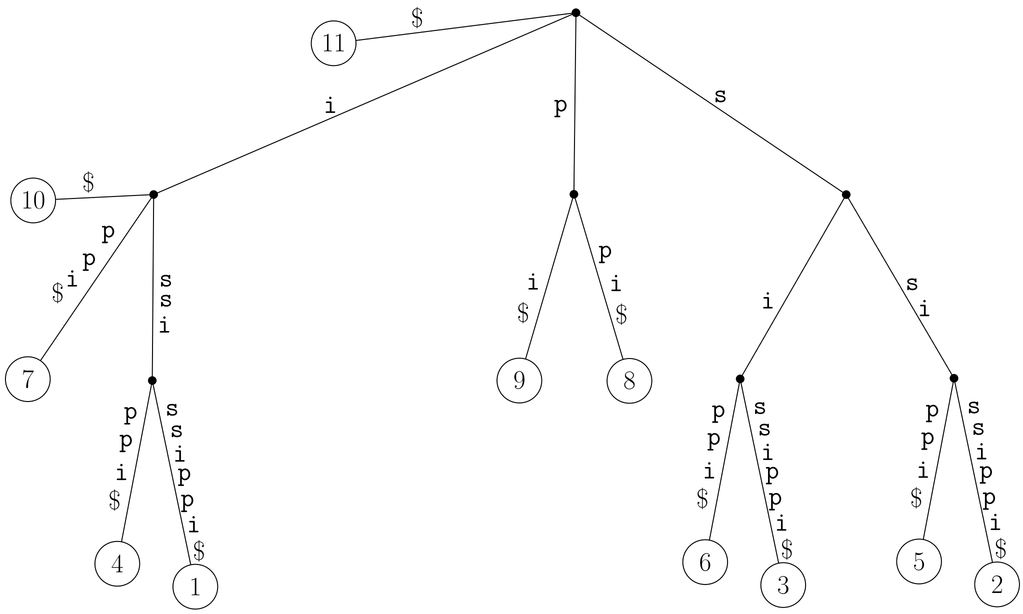
Problem 2: Suffixbäume 3,5 + 2,5 = 6 Punkte

(a) Erstellen Sie den Suffixbaum der Zeichenkette $T = \texttt{TATTA}$. Verwenden Sie das iterative Verfahren aus der Vorlesung, bei dem die Suffixe $S_1, S_2, \ldots$ nacheinander in den Baum einfügt werden und zeichnen Sie für **jeden Schritt einen eigenen Baum**. Stellen Sie sicher, dass jedes Suffix von einem Pfad zwischen einem Blatt und der Wurzel repräsentiert wird.

(b) Der *längste wiederholte Teilstring* in einer Zeichenkette T, ist der längste Teilstring von T, der mindestens zwei Mal in T vorkommt.

(i) Geben Sie den längsten wiederholten Teilstring von $T =$ `ississippi` an und markieren Sie in folgendem Suffixbaum den Pfad, der diesen Teilstring repräsentiert.

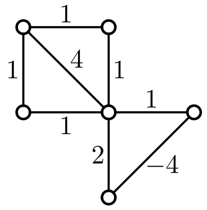
Teilstring: ______________________



(ii) Skizzieren Sie einen Algorithmus, der die Länge des längsten wiederholten Teilstrings in einer Zeichenkette T in Zeit $O(|T|)$ bestimmt. Sie dürfen annehmen, dass der Suffixbaum $S = (V, E)$ von T gegeben ist und dass die String-Tiefe $d(v)$ eines Knotens $v \in V$ in $O(1)$ Zeit bestimmt werden kann.

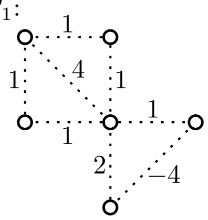
Problem 3: Kreisraum & Kreisbasen 1,5 + 3 + 1,5 = 6 Punkte

Gegeben sei der unten abgebildete Graph $G = (V, E)$. Das Kantengewicht $w(e)$ jeder Kante e ist an der Kante notiert.



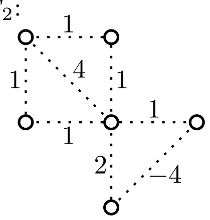
(a) Zeichnen Sie in die folgenden Abbildungen alle (auch nicht einfache) Kreise ein und geben Sie das Gewicht der Kreise an.

C_1:



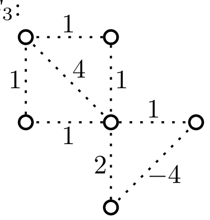
$w(C_1) =$

C_2:



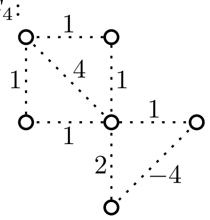
$w(C_2) =$

C_3:



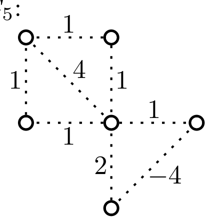
$w(C_3) =$

C_4:



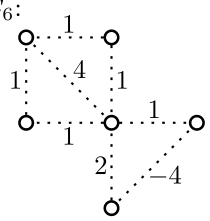
$w(C_4) =$

C_5:



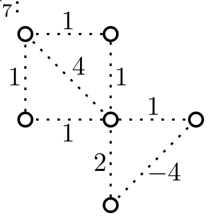
$w(C_5) =$

C_6:



$w(C_6) =$

C_7:



$w(C_7) =$

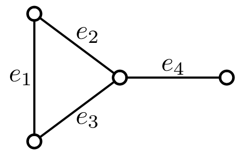
(b) Geben Sie eine minimale Kreisbasis B von G an. Verwenden Sie die Bezeichnungen der Kreise aus Aufgabenteil (a). Zeigen Sie, dass B eine minimale Kreisbasis ist (zu zeigen sind Minimalität **und** Kreisbasis-Eigenschaft).

(c) Geben Sie eine Familie von Graphen G_n (für $n = 1, 2, 3, \dots$) an, sodass G_n $O(n)$ Knoten und $\Omega(2^n)$ Kreise enthält. Begründen Sie Ihre Antwort.

Problem 4: Matroide 1,5 + 2 + 1,5 = 5 Punkte

(a) Sei $G = (V, E)$ ein zusammenhängender Graph mit $|V| \geq 2$. Eine Teilmenge E_u induziert einen unzusammenhängenden Teilgraph, wenn $G_u = (V, E_u)$ unzusammenhängend ist. Zeigen Sie, dass $\mathcal{U}_u = \{E_u \mid E_u \subseteq E \text{ induziert unzusammenhängenden Teilgraph}\}$ ein Unabhängigkeitssystem ist.

(b) Geben Sie für den folgenden Graphen alle Mengen in $\mathcal{U}_u$ explizit an.



(c) Zeigen Sie, dass $\mathcal{U}_u$ im allgemeinen **kein** Matroid ist.

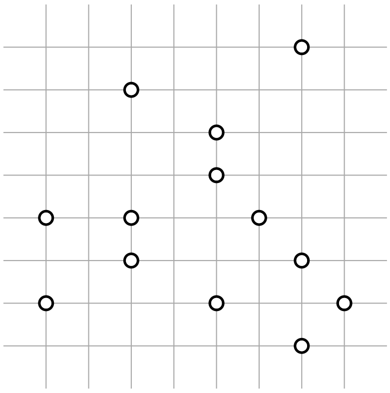
Problem 5: Parametrisierte Algorithmen 1,5 + 5 + 3,5 = 10 Punkte

Das Problem PUNKTÜBERDECKUNG ist wie folgt definiert.

Gegeben: Eine Menge von Punkten $P = \{p_1, \ldots, p_n\}$ in der Ebene, sowie ein Parameter k.

Gesucht: Eine Menge von maximal k Geraden $G = \{g_1, \ldots, g_k\}$, sodass es für jeden Punkt $p_i \in P$ eine Gerade $g_j \in G$ mit $p_i \in g_j$ gibt.

(a) Gegeben sei eine Instanz von PUNKTÜBERDECKUNG bestehend aus der unten abgebildeten Punktmenge, sowie dem Parameter $k = 3$. Zeichnen Sie 3 Geraden ein, die diese Instanz von PUNKTÜBERDECKUNG lösen.



(b) Geben Sie einen FPT-Algorithmus für PUNKTÜBERDECKUNG an. Begründen Sie, warum ihr Algorithmus korrekt ist.

Hinweis: Benutzen Sie die Technik der Kernbildung. Zeigen Sie dazu zunächst, dass eine Gerade, die mehr als k Punkte aus P enthält, in jeder Lösung enthalten sein muss.

Fortsetzung Aufgabenteil (b)

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

Problem 6: Approximationsalgorithmus 1,5 + 2 + 2,5 = 6 Punkte

Ein gerichteter Graph heißt *azyklisch*, falls er keinen gerichteten Kreis enthält.

Sei $G = (V, E)$ ein gerichteter Graph und sei $G_1 = (V, E_1 \subseteq E)$ ein inklusionsmaximaler azyklischer Teilgraph von G. Des Weiteren sei $G_2 = (V, E_2 = E \setminus E_1)$ das Komplement zu G_1.

(a) Zeigen Sie: Für jede Kante $(u, v) \in E_2$ gibt es in G_1 einen gerichteten Pfad von v nach u.

(b) Zeigen Sie: G_2 ist azyklisch.

(c) Betrachten Sie das Problem MAXIMUM ACYCLIC GRAPH:

Gegeben: Gerichteter Graph $G = (V, E)$.

Gesucht: Kardinalitätsmaximaler azyklischer Teilgraph von G.

Skizzieren Sie einen Approximationsalgorithmus für MAXIMUM ACYCLIC GRAPH mit relativer Gütegarantie 2. Beweisen Sie diese Gütegarantie.

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

Ein gerichteter Graph $G = (V, E)$ ist *d-regulär*, wenn jeder Knoten genau d ausgehende und genau d eingehende Kanten hat.

Algorithmus 1 : Randomisierter Algorithmus

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

$S \leftarrow \emptyset$
Für *jedes* $(u, v) \in E$
    $x_{(u,v)} \leftarrow$ blau
    $x_{(u,v)} \leftarrow$ rot mit Wahrscheinlichkeit p
Für *jedes* $(u, v) \in E$
    Wenn $x_{(u,v)} =$ *rot* **oder** $x_{(u,w)} =$ *blau für alle von u ausgehenden Kanten* $(u, w) \in E$
        $S \leftarrow S \cup \{(u, v)\}$

(a) Sei S das Ergebnis von Algorithmus 1. Zeigen Sie: In dem von S induzierten Graph $G' = (V, S)$ hat jeder Knoten mindestens eine ausgehende Kante.

(b) Geben Sie die Wahrscheinlichkeit $P(e \in S)$ dafür an, dass eine gegebene Kante $e \in E$ in S liegt.

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

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

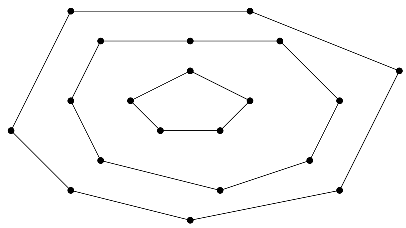
Problem 8: Konvexe Hülle 1,5 + 1 + 2,5 + 3 = 8 Punkte

Gegeben sei eine endliche, nicht-leere Punktmenge $P \subset \mathbb{R}^2$. Sei $H(S)$ die konvexe Hülle einer Teilmenge $S \subseteq P$. Die *Schichten* von P sind rekursiv definiert durch:

Basisfall: $S_1 = H(P)$

Allgemeiner Fall: $S_i = H(P \setminus (S_1 \cup \ldots \cup S_{i-1}))$ für $i > 1$

Betrachten Sie nun das Problem Onion Peeling: Gesucht sind die Schichten $S_1, \ldots, S_k$ von P, sodass $S_k \neq \emptyset$ und $S_{k+1} = \emptyset$ gilt. Folgende Abbildung zeigt eine Punktmenge mit $k = 3$ nicht-leeren Schichten.



(a) Zeigen Sie, dass für zwei nicht-leere Schichten S_{i-1} und S_i mit $i > 1$ gilt: Die Punkte aus S_i liegen innerhalb des konvexen Polygons S_{i-1}.

(b) Zeigen Sie, dass der Algorithmus SimpleOnionPeeling für $n = |P|$ die Laufzeit $O(n^2 \log n)$ hat.

Algorithmus 2 : SimpleOnionPeeling

Eingabe : Endliche Menge $P \subset \mathbb{R}^2$ bestehend aus n Punkten.
Ausgabe : Die Schichten $S_1, \ldots, S_k$ von P, sodass gilt $S_k \neq \emptyset$ und $S_{k+1} = \emptyset$.

$i \leftarrow 1$
$P_1 \leftarrow P$
solange $P_i \neq \emptyset$ **tue**
  $S_i \leftarrow$ Ergebnis von Graham Scan angewendet auf P_i
  $P_{i+1} \leftarrow P_i \setminus S_i$ //Kann in $O(n)$ Zeit berechnet werden.
  $i \leftarrow i + 1$
return $S_1, \ldots, S_{i-1}$

(c) Geben Sie eine Familie von Punktmengen $P_1, P_2, \ldots$ mit $|P_n| \in O(n)$ an, sodass die Laufzeit von SIMPLEONIONPEELING für diese Instanzen in $\Omega(n^2 \log n)$ liegt. Begründen Sie Ihre Antwort.

(d) Skizzieren Sie einen Algorithmus, der das Problem ONION PEELING in $O(n^2)$ Laufzeit löst und beweisen Sie diese Laufzeit.

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

- ☐ $\Theta(k \log k \cdot n^2 + 5^k \cdot n^k)$
- ☐ $\Theta(n)$
- ☐ $\Theta(2^k \cdot n! + \sqrt{n})$
- ☐ $\Theta(2^{k!} \cdot n + n^2 \log n)$

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

Bei einer Menge von n Strecken in der Ebene

- ☐ können alle Schnittpunkte zwischen diesen Strecken in $O(n \log n)$ berechnet werden.
- ☐ kann jede Strecke $n-1$ andere Strecken schneiden.
- ☐ schneidet jede Strecke mindestens eine andere.
- ☐ kann es $\Omega(n^2)$ viele Schnittpunkte zwischen diesen Strecken geben.

Ein minimaler st-Schnitt in einem zusammenhängenden, ungerichteten und ungewichteten Graphen $G = (V, E)$

- ☐ kann in $O(|E|)$ Zeit berechnet werden, wenn ein maximaler st-Fluss gegeben ist.
- ☐ kann Gewicht $|V| - 1$ haben.
- ☐ trennt t von mindestens einem Knoten $v \in V$ mit $v \neq s$.
- ☐ hat Gewicht mindestens 1.

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

- ☐ PAS
- ☐ FPAS
- ☐ AFPAS
- ☐ APAS

Sei G ein Graph und C ein minimales VERTEX COVER der Größe k in G. Dann enthält C

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

Sei $G = (V, E)$ ein Graph. Die Menge $\mathcal{U}$ enthalte die Menge $V' \subseteq V$ genau dann, wenn für alle $\{u, v\} \in E$ gilt: $u \notin V'$ oder $v \notin V'$. Dann

☐ ist $\mathcal{U}$ ein Matroid.
☐ ist $\mathcal{U}$ ein Unabhängigkeitssystem.
☐ gilt $V \in \mathcal{U}$.
☐ ist es $\mathcal{NP}$-Schwer eine kardinalitätsmaximale Menge in $\mathcal{U}$ zu finden.

Beim (h, k)-Paging

☐ darf der Online-Algorithmus maximal k Fehlzugriffe haben.
☐ hat der optimale Offline-Algorithmus immer weniger Fehlzugriffe als der Online-Algorithmus.
☐ hat der Online-Algorithmus einen mindestens genauso großen Cache zur Verfügung wie der optimale Offline-Algorithmus.
☐ sind konservative Online-Algorithmen 2-kompetitiv, wenn $h = \frac{k}{2}$.