1. Klausur zur Vorlesung
Algorithmentechnik
Wintersemester 2006/2007
1. März 2007

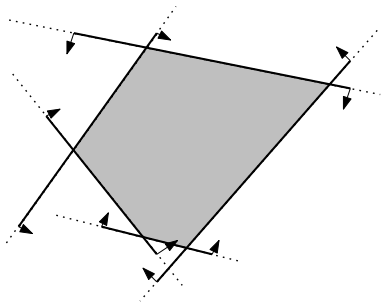
Lösung!

Beachten Sie, dass die enthaltenen Musterlösungen aus didaktischen Gründen umfangreicher formuliert sind als für den Erhalt der vollen Punktzahl erforderlich.

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

Problem 1: Grundlagen $1 + 3 + 2 = 6$ Punkte

Gegeben seien n Halbebenen in der Ebene (siehe Bild rechts). Der folgende rekursive Algorithmus 1 berechnet den Schnitt dieser Halbebenen (graues Polyeder im Bild).
Hinweis: Die Routine SCHNEIDEKONVEXEREGIONEN(C_1, C_2) (Zeile 7) berechnet den Schnitt zweier Polyeder in einer Worst-case-Laufzeit von $\Theta(n^2)$.



Algorithmus 1 : HALBEBENENSCHNITT(H)

Eingabe : Menge H von n Halbebenen in der Ebene
Ausgabe : Das konvexe Polyeder $C := \bigcap_{h \in H} h$.

```
1 Wenn |H| = 1
2     C ← eindeutiges h ∈ H
3 sonst
4     Teile H in Mengen H_1 und H_2 mit Kardinalitäten ⌈n/2⌉ und ⌊n/2⌋
5     C_1 ← HalbebenenSchnitt(H_1)
6     C_2 ← HalbebenenSchnitt(H_2)
7     C ← SchneideKonvexeRegionen(C_1, C_2)
8 Gib C aus
```

(a) Welchen Sinn erfüllen die Zeilen 1 und 2?
Lösung: Abbruchbedingung der Rekursion □

(b) Geben Sie die Rekursionsgleichung für die Laufzeit von Algorithmus 1 an. Begründen Sie diese knapp.
Lösung: $T(n) = \underbrace{T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil)}_{\text{Rekursion Zeile 5 und 6}} + \underbrace{f(n)}_{\text{Zeile 7, } f(n) \in \Theta(n^2)}$ für $n \geq 2$, $T(1) \in O(1)$ (trivialer Fall).
Die Laufzeit von Zeile 7 dominiert Zeilen 1-4. □

(c) Lösen Sie die Rekursionsgleichung aus Teilaufgabe (b) und geben Sie damit eine möglichst scharfe obere und untere Schranke für die Worst-case-Laufzeit an.
Lösung:
Möglichkeit 1: Nach Master-Theorem (Satz 0.14) gilt bei der Darstellung

$$T(n) = a \cdot T(n/b) + f(n)$$

(dabei steht n/b für $\lfloor n/b \rfloor$ und $\lceil n/b \rceil$) das Folgende:
Awendbarkeit: Es ist $n \geq 0$, $a = 2 \geq 1$, $b = 2 > 1$ und $f(n)$ Funktion in n.
Voraussetzungen für Fall 1: Es gilt $f(n) \in \Theta(n^2) \subset \Omega(n^{\log_2 2+\varepsilon})$ und $af(n/b) = 2f(n/2) \leq cf(n)$ für $c = 3/4$ und $n \geq 1 = n_0$.

$$\underbrace{\Longrightarrow}_{\text{Master-Theorem}} \quad T(n) \in \Theta(f(n)) = \Theta(n^2)$$

Möglichkeit 2: Nach Master-Theorem (Satz 0.15) gilt bei der Darstellung

$$T(n) = \sum_{i=1}^{m} T(\alpha_i n) + f(n)$$

das Folgende:
Awendbarkeit: Es ist $0 < \alpha_i = 2 < 1, m = 2 \geq 1$ und $f(n) \in \Theta(n^k), k = 2 \geq 0$.
Voraussetzungen für Fall 1: Es gilt $\sum_{i=1}^{m} \alpha_i^k = \sum_{i=1}^{2} (1/2)^2 = 1/2 < 1$.
$\underbrace{\Longrightarrow}_{\text{Master-Theorem}}$ $T(n) \in \Theta(n^k) = \Theta(n^2)$ □

Problem 2: UNION-FIND $4 + 2 = 6$ Punkte

Das Problem ZUSAMMENHANGSKOMPONENTEN lautet wie folgt:
Gegeben: Ein ungerichteter, ungewichteter, einfacher Graph $G = (V, E)$.
Gesucht: Die Zusammenhangskomponenten von G.

(a) Schreiben Sie einen Algorithmus in Pseudocode zur Lösung des Problems ZUSAMMENHANGSKOMPONENTEN. Benutzen Sie dabei die aus der Vorlesung bekannte Datenstruktur UNION-FIND.

Lösung:

Algorithmus 2 : FINDE-ZUSAMMENHANG

Eingabe : Graph $G = (V, E)$ (ungerichtet, ungewichtet, einfach)
Ausgabe : UNION-FIND-Datenstruktur in der Mengen Zusammenhang darstellen

```
1 Initialisiere UNION-FIND Datenstruktur
2 Für jeden Knoten v ∈ V
3 |  MAKESET(v)
4 Für jede Kante {u, v} ∈ E
5 |  a ← FIND(u)
6 |  b ← FIND(v)
7 |  Wenn a ≠ b
8 |  |  UNION(a, b)
9 Gib UNION-FIND Datenstruktur aus
```

□

(b) Geben Sie eine möglichst scharfe asymptotische obere Schranke für die Worst-case-Laufzeit Ihres Algorithmus an und begründen Sie diese.
Lösung: Durchgeführte Operationen:

- $|V| \in \Theta(|V|)$ Operationen MAKESET
- $2|E| \in \Theta(|E|)$ Operationen FIND
- $O(|V|)$ Operationen UNION (nach $|V| - 1$ UNION: vollst. Zusammenhang)

Daraus folgt, dass $O(|E| + |V|)$ Operationen vom Typ MAKESET, FIND oder UNION durchgeführt werden. Da weder $|E|$ in $|V|$ linear nach oben abgeschätzt werden kann, noch andersherum, muss man diese Formulierung wählen. Für die UNION-FIND Datenstruktur aus der Vorlesung gilt somit eine obere Schranke für die asymptotische Worstcase-Laufzeit von $O((|E| + |V|) \cdot G(|V|))$ für FINDE-ZUSAMMENHANG (beachte, dass gilt: $G(|E|) \in O(G(|V|))$). □

Eine schärfere Analyse liefert, dass die Anzahl Schleifendurchläufe in Zeilen 4-8 sowohl durch $|V| - 1$ als auch durch $|E|$ beschränkt ist. Dies erlaubt eine Abschätzung von $O(|E| + |V| \cdot G(|V|))$, da lediglich die in konstanter Zeit laufenden MAKESET-Operationen nur durch $|V|$ abgeschätzt werden können.

Problem 3: Minimaler Spannbaum $2 + 2 + 4 = 8$ Punkte

(a) Schreiben Sie (umgangssprachlich) die GRÜNE REGEL und die ROTE REGEL der Färbungsmethode von Tarjan auf (kein Pseudocode).

- GRÜNE REGEL:
  Lösung: Wähle einen *Schnitt* in G, der von keiner grünen Kante gekreuzt wird. Unter allen ungefärbten Kanten, die diesen Schnitt kreuzen, wähle eine Kante *minimalen Gewichts* und färbe sie grün. □
- ROTE REGEL:
  Lösung: Wähle einen *Kreis*, der keine rote Kante enthält. Unter allen ungefärbten Kanten, die auf diesem Kreis liegen, wähle eine Kante *maximalen Gewichts* und färbe sie rot. □

Gegeben sei der folgende Algorithmus 3.

Algorithmus 3 : SLOWMST

Eingabe : Graph $G = (V, E)$ (ungewichtet, ungerichtet, zusammenhängend, einfach), injektive Kantengewichtsfunktion $w : E \to \mathbb{N}$
Ausgabe : Kantenmenge T

```
1  T ← E
2  F ← ∅
3  solange |T| > n − 1 tue
4      C ← Kreis in H = (V, T)
5      e ← beliebige Kante auf C
6      T ← T \ {e}
7      F ← F ∪ {e}
8  solange F ≠ ∅ tue
9      e ← beliebige Kante aus F
10     F ← F \ {e}
11     T ← T ∪ {e}
12     C ← Kreis in T
13     Wenn e Kante auf C mit maximalem Gewicht
14         T ← T \ {e}
15     sonst
16         e′ ← beliebige Kante auf C mit w(e′) > w(e)
17         T ← T \ {e′}
18         F ← F ∪ {e′}
19 Gib T aus
```

(b) Was genau induziert T unmittelbar vor der ersten Ausführung von Zeile 8?
Lösung: Die Kantenmenge induziert einen Spannbaum. □

(c) Begründen Sie, dass Algorithmus 3 einen MST ausgibt (kein vollständiger Beweis erforderlich).
Lösung: Zeilen 8 - 18 wandeln den Spannbaum T in einen MST um. In jeder Iteration wird in Zeile 9 - 12 ein Kreis C identifiziert, in dem eine Kante aus F zu T hinzugefügt wird.

Fall 1: Liefert Zeile 13 wahr, so entspricht dies der roten Regel, da die schwerste Kante auf C endgültig entfernt wird.

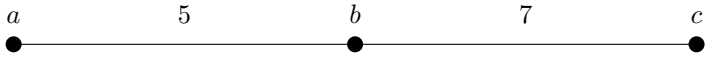
Fall 2: Sonst wird eine Kante größeren Gewichts aus T entfernt und zu F hinzugefügt.

Beachte zunächst die Invariante, dass T in beiden Fällen, und somit in jeder Iteration, stets einen Spannbaum induziert. Es wird in jeder Iteration eine Kante aus F gelöscht oder gegen eine schwerere Kante aus T getauscht. Durch dieses Tauschen (Fall 2) steigt das Gesamtgewicht der Kanten in F echt an. Da dieser Wert jedoch nach oben beschränkt ist, muss irgendwann (nach endlich vielen Tauschen) wieder eine Kante gewählt werden, die maximal auf dem induzierten Kreis ist (Fall 1), und somit gelöscht (rot gefärbt) wird, und somit nicht mehr betrachtet wird.

Somit sinkt $|F|$ nach endlich vielen Schritten auf 0, und zwar nur durch korrekte Rotfärbungen. Die Kanten von T sind am Ende implizit grün gefärbt. Da nun alle Kanten gefärbt sind, ist T nach der Färbungsinvariante ein MST. □

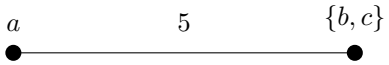
Problem 4: Minimaler Schnitt $2 + 3 = 5$ Punkte

(a) Führen Sie den Algorithmus von Stoer und Wagner auf untenstehendem Graphen durch. Startknoten sei immer Knoten a. Geben Sie die geforderten Informationen an und zeichnen Sie den Graphen nach jeder Phase (also nach dem Verschmelzen).



Lösung:

Phase 1
$s = b, t = c$
Schnitt der Phase: $(\{a, b\}, \{c\})$
Wert des Schnittes aus Phase 1: 7
Graph nach Phase 1:

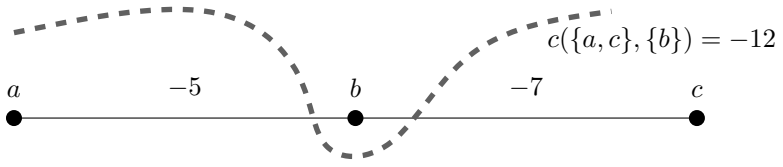


Phase 2
$s = a, t = \{b, c\}$
Schnitt der Phase: $(\{a\}, \{b, c\})$
Wert des Schnittes aus Phase 2: 5
Graph nach Phase 2:

Ergebnis
Resultierender minimaler Schnitt $(\{a\}, \{b, c\})$,
mit Wert: 5 □

(b) Zeigen Sie: In einem (zusammenhängenden, einfachen, ungerichteten) Graphen $G = (V, E)$ mit Kantengewichtsfunktion $w : E \to \mathbb{Z}$ (negative Kantengewichte zugelassen) arbeitet der Algorithmus von Stoer und Wagner nicht notwendigerweise korrekt.
Hinweis: Schauen Sie sich Teilaufgabe (a) scharf an.

Lösung: Auf folgendem Graph ist der Ablauf der Phasen von Stoer und Wagner analog zu Teilaufgabe (a), bei festem Startknoten a. In den Phasen werden Schnitte mit Werten -5 und -7 gefunden, dementsprechend liefert Stoer und Wagner einen minimalen Schnitt mit Wert -7. Der Schnitt $(\{a, c\}, \{b\})$ hat jedoch den Wert -12.



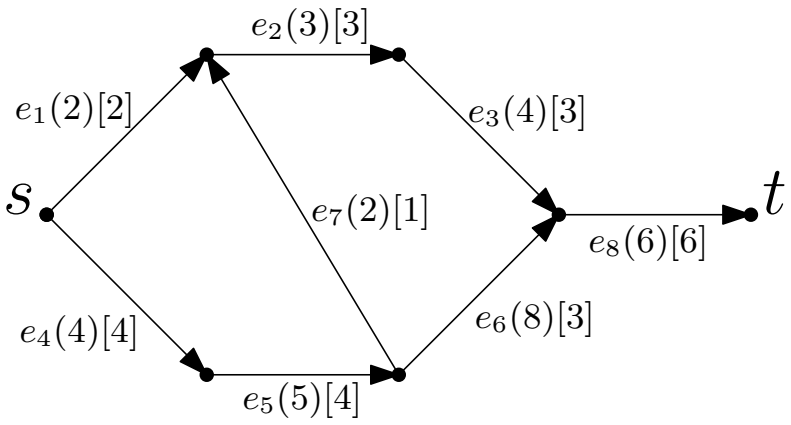
□

Problem 5: Maximaler Fluss $1 + 1 + 4 = 6$ Punkte

Gegeben sei das folgende Netzwerk D mit ganzzahligen Kantenkapazitäten (in Klammern) wie angegeben.

(a) Zeichnen Sie den maximalen Fluss von s nach t ein, indem Sie den Wert neben die entsprechende Kante in der Abbildung schreiben. Dabei soll die Kapazität der Kante e_2 voll ausgenutzt werden.

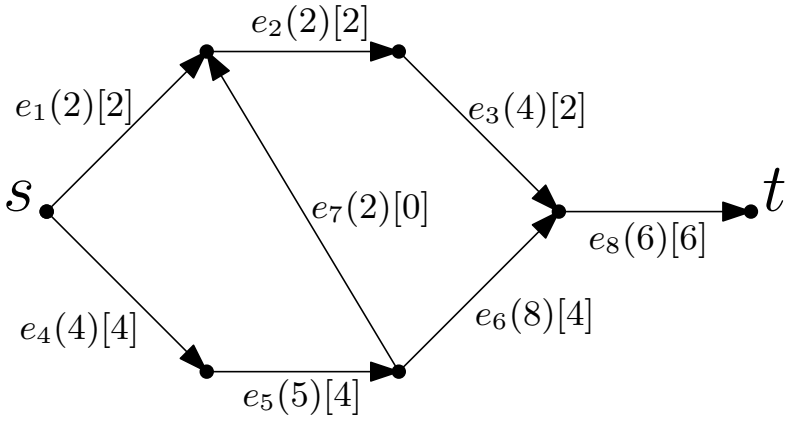
Lösung:



□

(b) Die Kapazität der Kante e_2 verringere sich nun um 1 auf die neue Kapazität 2. Korrigieren Sie Ihren Fluss aus Teilaufgabe (a), so dass wieder ein gültiger maximaler Fluss entsteht, und zeichnen Sie diesen im untenstehenden Netzwerk ein.

Lösung:



□

(c) Gegeben sei ein maximaler Fluss in einem Netzwerk mit ganzzahligen Kantenkapazitäten. Beschreiben Sie ein Verfahren, das diesen Fluss in einen gültigen maximalen Fluss überführt, nachdem sich die Kapazität einer Kante $e = (v, w)$ um 1 verringert hat (kein Code, kein Pseudocode). Nehmen Sie an, dass die Kapazität vor der Verringerung echt positiv war.

Lösung:

Fall 1) Wenn nach der Verringerung gilt $f(e) \leq c(e)$, dann ist nichts zu tun, da der alte Fluss noch immer gültig und maximal ist.

Fall 2) Sonst ist zunächst die Kapazitätsbedingung an e verletzt, und folgende Schritte sind notwendig:

- $f(e) \leftarrow f(e) - 1 \Rightarrow$ Kapazitätsbedingung an e erfüllt, aber Flusserhaltung an v und w nun verletzt.
- Suche erhöhenden Weg von v nach s und erhöhe entlang diesem um 1 (dieser existiert, da zuvor galt $f(e) > 0) \Rightarrow$ Flusserhaltung an v erfüllt.
- Suche erhöhenden Weg von t nach w und erhöhe entlang diesem um 1 (existiert, da zuvor galt $f(e) > 0) \Rightarrow$ Flusserhaltung an w erfüllt..
- Nun ist f wieder ein gültiger Fluss, aber nicht notwendigerweise maximal: Suche einen erhöhenden Weg von s nach t und erhöhe entlang diesem gegebenenfalls.

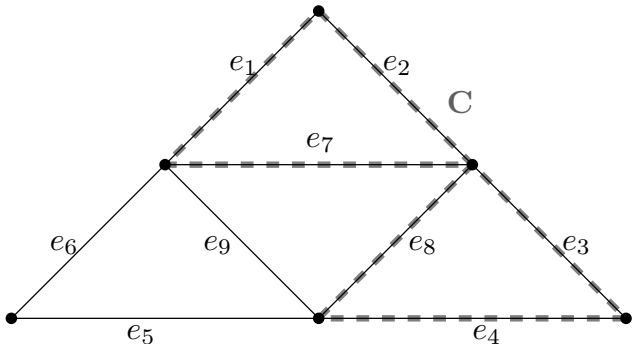
□

Anmerkung: Beachte dass im obigen Fall ein ganzzahliger maximaler Fluss als gegeben angenommen wird, den es in einem Netzwerk mit ganzzahligen Kapazitäten stets gibt. Betrachtet man den allgemeineren Fall, so muss man in Fall 2 noch zusätzlich beachten, dass der Fluss nicht ohne weiteres um 1 korrigiert werden kann, und dass ein einziger "Korrekturweg" jeweils beidseitig nicht notwendigerweise ausreicht. Es ergeben sich also folgende Abweichungen von oben:

- Verringere $f(e)$ um $\Delta = f(e) - c(e)$ statt um 1.
- Suche *solange* erhöhende Korrekturwege, bis insgesamt um Δ verringert wurde (anstatt nur einmal).

Problem 6: Kreisbasen $2 + 1 + 3 = 6$ Punkte

Die folgende Abbildung zeigt den Graphen G_{Tri}. Alle Kantengewichten seien 1. Gestrichelt grau ist zudem ein Kreis C in G_{Tri} eingezeichnet.



(a) Geben Sie eine minimale Kreisbasis $\mathcal{B}$ von G_{Tri} an (mit Scharfblick oder Methode). Geben Sie das Gewicht von $\mathcal{B}$ an.
Lösung:
Scharfblick: $|\mathcal{B}| = 4 \Rightarrow$ wähle 4 Dreiecke, eine leichtere Basis kann es nicht geben:
$B = \{b_1 = \{e_1, e_2, e_7\}, \quad b_2 = \{e_3, e_4, e_8\}, \quad b_3 = \{e_5, e_6, e_9\}, \quad b_4 = \{e_7, e_8, e_9\}\}$,
Gewicht: 12 □

(b) Bilden Sie den Kreis C (gestrichelt, grau) durch eine Linearkombination ihrer Basiselemente aus $\mathcal{B}$.
Lösung: $C = b_1 \oplus b_4$ □

(c) Begründen Sie, dass in G_{Tri} keine minimale Kreisbasis existiert, die eine Fundamentalbasis ist.
Lösung: Da jeder Kreis, der nicht in $\mathcal{B}$ ist, ein Gewicht größer als 3 hat, ist $\mathcal{B}$ die eindeutige minimale Kreisbasis von G_{Tri}. Somit bleibt zu zeigen, dass $\mathcal{B}$ nicht fundamental ist. In einer Fundamentalbasis wird jedes Element durch eine eindeutige Nichtbaumkante identifiziert, und enthält keine andere Nichtbaumkante. Sei o.B.d.A. e_7 diese Kante für b_4, dann kann b_1 nicht mehr durch eine andere Nichtbaumkante definiert werden $\Rightarrow$ Widerspruch. □

Problem 7: Randomisierte Algorithmen $2 + 3 + 1 = 6$ Punkte

Sei $n \in \mathbb{N}$. Gegeben seien drei Polynome $P_1(x), P_2(x), P_3(x)$ aus $\mathbb{R}[x]$, mit $\deg P_1 \leq n$, $\deg P_2 \leq n$ und $\deg P_3 \leq 2n$. Ferner sei $k \in \mathbb{N}$ gegeben. Zu prüfen ist, ob gilt: $P_1(x) \cdot P_2(x) = P_3(x)$.

Algorithmus 4 : PolynomTest$(P_1(x), P_2(x), P_3(x), k)$

Eingabe : $k \in \mathbb{N}$ und drei Polynome aus $\mathbb{R}[x]$: $P_1(x), P_2(x), P_3(x)$, mit $\deg P_1 \leq n$, $\deg P_2 \leq n$ und $\deg P_3 \leq 2n$

Ausgabe : Aussage, ob gilt: $P_1(x) \cdot P_2(x) = P_3(x)$

```
1 S ← beliebige endliche Teilmenge aus ℝ mit |S| ≥ 2n + 1
2 Für i ← 1 bis k
3 |   r ← zufälliges Element aus S
4 |   Wenn P1(r) · P2(r) ≠ P3(r)
5 |   |   Gib Nein aus
6 Gib Ja aus
```

(a) Zeigen Sie: Wenn $P_1(x) \cdot P_2(x) = P_3(x)$ gilt, dann gibt Algorithmus 4 das korrekte Ergebnis aus.
Lösung: Wenn $P_1(x) \cdot P_2(x) = P_3(x)$ für alle $x \in \mathbb{R}$ gilt, so gilt insbesondere $P_1(r) \cdot P_2(r) = P_3(r)$ für jedes $r \in S \subseteq \mathbb{R}$. Somit liefert Zeile 4 stets **falsch**. Also terminiert der Algorithmus nach k Iterationen mit Zeile 6, was korrekt ist. □

(b) Zeigen Sie: Die Wahrscheinlichkeit, dass Algorithmus 4 fälschlicherweise Ja ausgibt, ist kleiner gleich $(2n/|S|)^k$.
Lösung: Das Polynom $Q(x) = P_1(x) \cdot P_2(x) - P_3(x)$ hat maximal $2n$ Nullstellen. Der Fehler tritt auf, wenn Q nicht das Nullpolynom ist und in *jeder* Iteration r Nullstelle von Q ist. Pro Iteration wird mit Wahrscheinlichkeit $p_b \leq 2n/|S|$ eine Nullstelle von Q ausgewählt. Wegen $|S| \geq 2n + 1$ gilt zudem, dass $|S|$ nicht *ausschließlich* Nullstellen von Q enthalten kann, also $p_b < 1$. Für k Iterationen ist die Fehlerwahrscheinlichkeit des Algorithmus somit $(2n/|S|)^k$, da in jeder Iteration ein Fehler gemacht werden muss um ein falsches Endergebnis zu liefern.

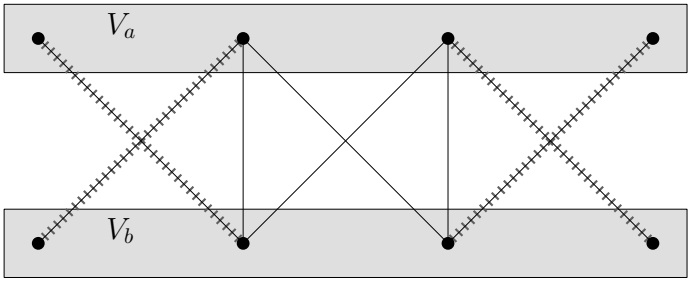
Beachte, dass diese Argumentation nicht voraussetzt, dass S *zufällig* gewählt wird, sondern für *beliebiges* S gilt, also sogar bei einer Implementation, die immer dasselbe S verwendet und einem "Angreifer", der P_1, P_2 und P_3 geschickt wählt. Würde S zufällig aus $\mathbb{R}$ gewählt, wäre die Wahrscheinlichkeit dass S überhaupt Nullstellen von Q enthält, gleich 0. □

(c) Von welchem Typ randomisierter Algorithmen ist Algorithmus 4?
Begründen Sie Ihre Aussage kurz.
Lösung: Der Algorithmus terminiert immer, liefert aber unter Umständen ein falsches Ja. Ein No ist stets korrekt.
⇒ Monte Carlo Algorithmus mit einseitigem Fehler. □

Problem 8: ILP für MAXIMALES BIPARTITES MATCHING $2 + 3 + 2 = 7$ Punkte

Das Problem MAXIMALES BIPARTITES MATCHING lautet wie folgt:
Gegeben: Ein ungerichteter, ungewichteter, bipartiter Graph $G = (V := V_a \dot\cup V_b, E)$ mit der Partition V_a, V_b.
Hinweis: In einem bipartiten Graphen ist $E \subseteq \{\{v, w\} \mid v \in V_a, w \in V_b\}$.
Gesucht: $M \subseteq E$ mit $|M|$ maximal so, dass jeder Knoten $v \in V$ zu maximal einer Kante aus M inzident ist (M ist dann ein *maximales Matching*).

(a) Zeichnen Sie in dem untenstehenden Graphen ein maximales Matching ein.
Lösung:



□

(b) Modellieren Sie das Problem als ILP. Erläutern Sie knapp Zielfunktion, Variablen und Nebenbedingungen. Was besagt der Wert der Zielfunktion einer Lösung?
Lösung: Führe für jede Kante $\{i, j\} \in E$ eine Variable $x_{ij} \in \{0, 1\}$ ein. $x_{ij} = 1$ bedeutet, dass die Kante im Matching ist, $x_{ij} = 0$ sonst

Zielfunktion:

$$(\max) f = \sum_{\{i,j\} \in E} x_{ij} \quad \text{maximiere Anzahl Kanten im Matching}$$

Nebenbedingungen:

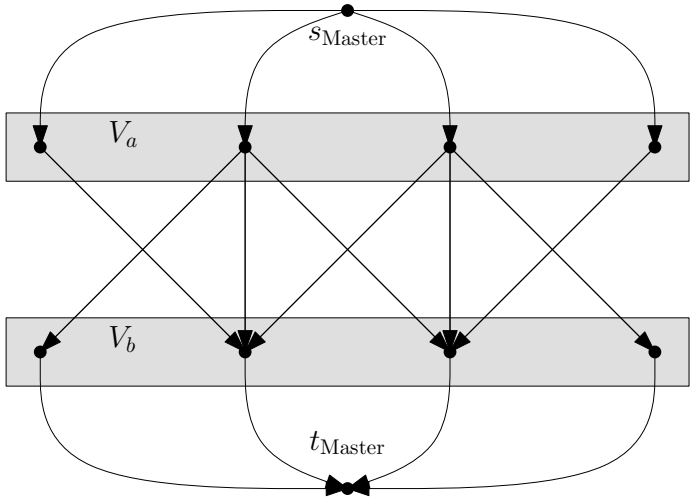
$$\forall i \in V : \sum_{\{i,j\} \in E} x_{ij} \leq 1 \quad \text{und} \quad x_{i,j} \in \{0, 1\}, \forall i, j \in V$$

jeder Knoten darf maximal zu einer Kante aus dem Matching inzident sein.

Der Wert von f einer Lösung des ILPs entspricht der Anzahl Kanten im Matching. Jede Kante mit $x_{ij} = 1$ ist im Matching enthalten. □

(c) Ist das Problem MAXIMALES BIPARTITES MATCHING in der Komplexitätsklasse $\mathcal{P}$ enthalten? Begründen Sie Ihre Antwort.
Lösung: Ja, das Problem kann mit einem Flussproblem modelliert werden. Hierbei wird eine Mastersenke s_{Master} und eine Masterquelle t_{Master} eingeführt. Dann ist s_{Master} adjazent zu allen Knoten aus V_a, und t_{Master} zu allen aus V_b. Alle Kanten haben Kapazität 1 und werden nach "unten" gerichtet. Auf diesem Netzwerk wird nun ein maximaler Fluss gesucht. Alle Kanten mit Flusswert 1 sind dann im maximalen Matching enthalten. Der Wert des maximalen Flusses entspricht dann der Kardinalität des maximalen Matchings.

Sowohl diese Konstruktion, als auch das Flussproblem ist in $\mathcal{P}$.



□

Problem 9: $10 \times 1 = 10$ 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, nicht angekreuzte Aufgaben werden nicht gewertet. Es wird keine negative Gesamtpunktzahl für diese Aufgabe geben.

	Wahr	Falsch
Zur Bestimmung eines MST ist auf Graphen mit $\|E\| \in O(\|V\|)$ der Algorithmus von Kruskal dem von Prim vorzuziehen (bzgl. Laufzeit), sofern die Kanten nach Gewicht sortiert vorliegen.	X	
Es gilt: $o(n \cdot (\log n)^2) \cap O(n \cdot \log(n^2)) \cap \Omega(n \cdot \log\log n) \neq \emptyset$.	X	
Auf einer CREW-PRAM kann das Produkt von n Zahlen in einer asymptotischen Laufzeit von $O(\log n)$ berechnet werden.	X	
In einem ungerichteten, einfachen, zusammenhängenden Graphen ist der Kreisraum orthogonal zum Schnittraum.	X	
Die Menge aller bipartiten Teilgraphen (induziert durch Kantenmengen) eines Graphen bilden ein Matroid über dem Kantenraum.		X
Wenn bei einer amortisierten Analyse für die Potentialfunktion $\mathbb{C}$ und zwei aufeinanderfolgende Zustände stets gilt: $\mathbb{C}(D_{i+1}) \geq \mathbb{C}(D_i)$, dann sind die amortisierten Kosten eine obere Schranke für die Gesamtkosten.	X	
Das Löschen des maximalen Elements aus einem MAX-HEAP mit n Elementen (mit Wiederherstellung der HEAP-Eigenschaft), hat eine Worstcase-Laufzeit von $\Theta(n)$ (Implementierung nach Vorlesung).		X
Bei der Ausführung des Algorithmus von GOLDBERG-TARJAN werden höchstens $O(\|V\| \cdot \|E\|)$ PUSH-Operationen durchgeführt (Standardimplementierung nach Vorlesung).		X
Die asymptotische Laufzeit eines FPAS ist polynomiell in der Eingabegröße und in $1/\epsilon$ (ϵ wie in Vorlesung).	X	
Jeder parametrisierte Algorithmus für k-VERTEX-COVER hat eine Laufzeit, die exponentiell in $\|V\|$ ist.		X