Algorithmen und Datenstrukturen 2

WiSe 2021/22 · Prof. Dr. Dell
22. Februar 2022
Klausurnummer: **0000**

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Nachname (Druckschrift)**:** _____

Vorname (Druckschrift)**:** _____

Matrikelnummer: _____

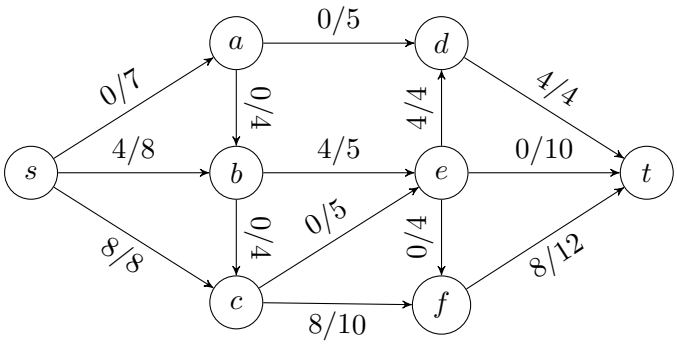
Studiengang: _____

Bitte Hinweise beachten:

- Die Klausur dauert **180 Minuten**.
- Name/Matrikelnummer **nur auf diesem Titelblatt!**
- **Kein Zusatzpapier** abgeben! Am Ende der Klausur finden Sie leere Seiten.
- **Alle elektronischen Geräte sind untersagt!** Insbesondere also Handys, Smartwatches, Taschenrechner. Ausschalten und wegpacken! Nichtbeachtung stellt einen Täuschungsversuch dar und führt zum Nichtbestehen der Klausur.
- Zugelassene Hilfsmittel:
  - Ein DIN A4 Blatt mit handschriftlichen Notizen (beidseitig).
  - Dokumentenechte Stifte in den Farben blau und schwarz. (Füller, Tipp-Ex, Tintenlöscher sind untersagt.)
- Bewertung:
  - **Immer nur eine Lösung** angeben, sonst gilt die Aufgabe als nicht gelöst.
  - Begründungen sind nur notwendig, wenn die Aufgabenformulierung dies verlangt.
  - In allen Multiple-Choice-Fragen sind **genau zwei von fünf Antworten** richtig. Wenn man x richtige Kreuze setzt und y falsche, erhält man $\max\{x-y,0\}$ Punkte, also entweder 0, 1, oder 2 Punkte. Zum Beispiel:

richtige Kreuze	x	2	2	1	1	0
falsche Kreuze	y	0	1	0	1	2
Punkte	$\max\{x-y,0\}$	2	1	1	0	0

Betrachten Sie den s-t-Fluss f in dem folgenden Flussnetzwerk (G, s, t, c):

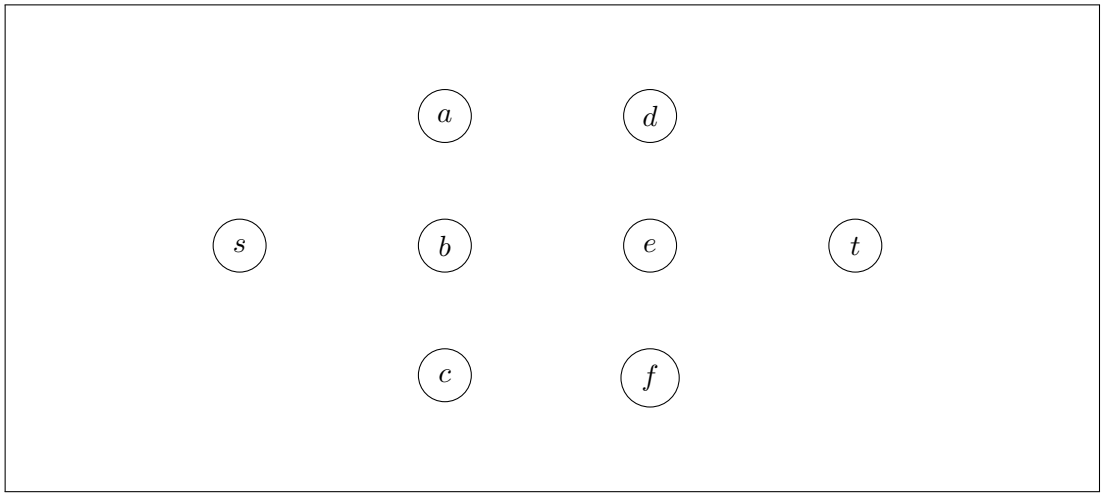


Jede Kante uv ist hierbei mit $f(uv)/c(uv)$ beschriftet, zum Beispiel trägt die Kante be den Fluss $f(be) = 4$ und hat die Kapazität $c(be) = 5$.

a) Der s-t-Fluss f hat den Wert _____.

(/ 1 Punkte)

b) Zeichnen Sie den zugehörigen Residualgraphen G_f in der folgenden Abbildung ein. Geben Sie dabei die Restkapazitäten durch Beschriftung an jeder Kante von G_f an.



(/ 3 Punkte)

c) Ein maximaler s-t-Fluss in (G, s, t, c) hat den Wert _____.

(/ 1 Punkte)

d) Ein minimaler s-t-Schnitt in (G, s, t, c) hat den Wert _____.

(/ 1 Punkte)

e) Ein Beispiel für einen minimalen s-t-Schnitt (S, T) in (G, s, t, c) ist gegeben durch:

$S = \{s, \_\_\_\_\_\}$

$T = \{t, \_\_\_\_\_\}$

(/ 2 Punkte)

Aufgabe 2: Amortisierte Analyse — $1 + 1 + 1 + 4 = 7$ Punkte

Wir betrachten eine Datenstruktur, die zwei globale Variablen s und L sowie ein Array A aufrechterhält, und die Operationen AllocateNew, Insert und Decimate bereitstellt.

```
s = 0
L = 2
A[1..L] = neues Array der Länge L

function AllocateNew()
    B[1..L] = neues Array der Länge L
    for i from 1 to s do
        B[i] = A[i]
    A = B

function Insert(x)
    if s == L then
        L = (L + 1)^2        (*)
        AllocateNew()
    s = s + 1
    A[s] = x

function Decimate()
    s = ⌊s/10⌋
    for i from 1 to s do
        A[i] = A[i · 10]
    AllocateNew()
```

Wir nehmen an, dass wir ein neues Array beliebiger Länge in Zeit $O(1)$ anlegen können. Anfangs ist die Datenstruktur leer (also $s = 0$). Im Folgenden betrachten wir eine beliebige Sequenz von n Insert und/oder Decimate Operationen und sind an den amortisierten Laufzeiten in Θ-Notation **in Abhängigkeit von n** interessiert.

a) Die amortisierte Laufzeit von Insert ist:

$$\Theta\Big(\underline{\qquad\qquad\qquad\qquad}\Big).$$

(/ 1 Punkte)

b) Die amortisierte Laufzeit von Insert wird $\Theta(n)$, wenn wir die mit `(*)` markierte Zeile wie folgt ändern:

$$L = \underline{\qquad\qquad\qquad\qquad}.$$

(/ 1 Punkte)

c) Angenommen wir ändern die mit `(*)` markierte Zeile zu $L = L + \lceil L^{1/3} \rceil$. Die amortisierte Laufzeit von Insert ist jetzt:

$$\Theta\Big(\underline{\qquad\qquad\qquad\qquad}\Big).$$

(/ 1 Punkte)

d) Wir betrachten jetzt wieder die Datenstruktur mit der ursprünglichen Zeile `(*)`. Die amortisierte Laufzeit von DECIMATE ist $\Theta\Big($_____$\Big)$. Begründen Sie Ihre Antwort, indem Sie eine beliebige Analysemethode benutzen, die aus der Vorlesung bekannt ist. Geben Sie an, welche Methode Sie anwenden, und wenden Sie nur <u>eine</u> Methode an (Aggregationsmethode, Buchhaltemethode oder Potenzialmethode).

(/ 4 Punkte)

a) Welche zwei der folgenden fünf Aussagen sind wahr?

□ Wenn es einen Polynomialzeitalgorithmus gibt, der für einen gegebenen Graphen eine echte 4-Färbung findet, dann ist P = NP.

□ Wenn P = NP gilt, dann gibt es keinen Polynomialzeitalgorithmus, der für einen gegebenen Graphen eine echte 5-Färbung findet.

□ Wenn es einen Polynomialzeitalgorithmus gibt, der für einen gegebenen Graphen eine echte 4-Färbung findet, dann ist P ≠ NP.

□ Wenn es keinen Polynomialzeitalgorithmus gibt, der für einen gegebenen Graphen eine echte 4-Färbung findet, dann ist P = NP.

□ Wenn es keinen Polynomialzeitalgorithmus gibt, der für einen gegebenen Graphen eine echte 4-Färbung findet, dann ist P ≠ NP.

(/ 2 Punkte)

b) Seien Π und Π' zwei Entscheidungsprobleme in NP. Welche zwei der folgenden fünf Aussagen sind wahr?

□ Um zu beweisen, dass ein Problem Π' NP-hart ist, genügt es, zu beweisen, dass Π NP-hart ist und dass es eine Polynomialzeitreduktion von Π' nach Π gibt.

□ Wenn Π' NP-hart ist und es eine Polynomialzeitreduktion von Π nach Π' gibt, dann ist Π NP-hart.

□ Wenn P = NP gilt, dann gibt es eine Polynomialzeitreduktion von Π' auf das Problem $\{ G \mid G$ ist ungerichteter Graph mit höchstens 10 Knoten $\}$.

□ Eine typische Polynomialzeitreduktion von Π nach Π' bildet Instanzen von Π' auf Instanzen von Π ab.

□ Um zu beweisen, dass ein Problem Π' NP-hart ist, genügt es, zu beweisen, dass Π NP-hart ist und dass es eine Polynomialzeitreduktion von Π nach Π' gibt.

(/ 2 Punkte)

c) Wir betrachten das folgende Entscheidungsproblem CliqueAndEdge:

Eingabe:	Ungerichteter Graph $G = (V, E)$ und eine natürliche Zahl $k \in \mathbb{N}$.
Frage:	Gibt es $k + 1$ verschiedene Knoten $s_0, s_1, \ldots, s_k \in V$, sodass $\{s_0, s_1\}$ eine Kante und $\{s_1, \ldots, s_k\}$ eine Clique in G sind?

Beweisen Sie, dass das Problem CliqueAndEdge NP-hart ist.

(/ 5 Punkte)

a) Sei $\Sigma = \{0, 1\}$. Welche zwei der folgenden fünf Sprachen L sind **unentscheidbar**?

□ $L = \Sigma^*$

□ $L = \{ \langle M, w \rangle \mid M \text{ hält auf Eingabe } w \text{ nach mindestens } |w|^2 \text{ Schritten} \}$

□ $L = \{ a_1 \dots a_n \in \Sigma^* \mid n \in \mathbb{N} \text{ und } a_1 + \cdots + a_n = 9 \}$

□ $L = \{ \langle M \rangle \mid M(w) = 1 \text{ für alle } w, \text{ die gerade Zahlen kodieren} \}$

□ $L = \{ \langle M, w \rangle \mid M \text{ hält auf Eingabe } w \text{ nach höchstens } |w|^2 \text{ Schritten} \}$

(/ 2 Punkte)

b) Welche zwei der folgenden fünf Aussagen sind wahr?

□ Jede semi-entscheidbare Menge ist entscheidbar.

□ Für alle Sprachen L und L' gilt: Wenn es eine Reduktion von L nach L' gibt, dann ist L unentscheidbar oder L' entscheidbar.

□ Für alle Sprachen L und L' gilt: Wenn $L \subseteq L'$ gilt und L' entscheidbar ist, so ist auch L entscheidbar.

□ Für alle Sprachen L gilt: Wenn L entscheidbar ist, dann ist $\overline{L}$ semi-entscheidbar.

□ Für alle Sprachen L und L' gilt: Wenn es eine Reduktion von L nach L' gibt und L' entscheidbar ist, dann ist L entscheidbar.

(/ 2 Punkte)

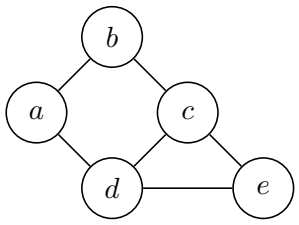
c) Wir betrachten die folgende Sprache:

HaltDiverge $= \{ \langle M, N, w \rangle \mid M$ hält auf Eingabe w und N hält auf Eingabe w nicht $\}$.

Beweisen Sie, dass L unentscheidbar ist.

(/ 4 Punkte)

a) Gegeben sei folgender Graph G:



Geben Sie eine Sequenz von Kanten an, sodass die schrittweise Kontraktion dieser Kanten zu einem minimalen Schnitt von G führt:

_____.

Geben Sie die genaue Wahrscheinlichkeit an, dass der Kontraktionsalgorithmus einen minimalen Schnitt von G ausgibt, wenn als erste Kante $\{c, e\}$ kontrahiert wird:

_____.

(/ 2 Punkte)

b) Wir erinnern uns an den *Select*-Algorithmus: Für ein gegebenes Array $S = [a_1, \ldots, a_n]$ und eine gegebene Zahl $k \in \{1, \ldots, n\}$ liefert $\mathit{Select}(S, k)$ die k-kleinste Zahl im Array S zurück. Beispielsweise liefert $\mathit{Select}(S, 1)$ die kleinste Zahl in S.

Intern muss der Algorithmus vor jedem Rekursionsschritt ein Pivotelement auswählen. Wir implementieren den Algorithmus nun mit der folgenden Pivotregel:

(R): Wähle zufällig das minimale oder maximale Element als nächstes Pivotelement.

Wir rufen $\mathit{Select}(S, k)$ mit Pivotregel (R) auf. Geben Sie die Laufzeit in Θ-Notation abhängig von n an:

$\Theta\Big($_____$\Big)$.

Sei $B = [4, 5, 1, 6, 3, 2]$ ein Array von ganzen Zahlen. Geben Sie eine mögliche Reihenfolge von Pivotelementen an, die für den Aufruf $\mathit{Select}(B, 5)$ mit Pivotregel (R) ausgewählt werden können:

_____.

(/ 2 Punkte)

c) Wir betrachten den *Quicksort*-Algorithmus, um das Array $C = [4, 5, 1, 6, 3, 2]$ zu sortieren. Geben Sie eine Reihenfolge von Pivotelementen an, damit der Aufruf $\mathit{Quicksort}(C)$ die kleinstmögliche Anzahl von Vergleichen benötigt:

_____.

(/ 1 Punkte)

d) Professor Regloh möchte auf seiner Website gerne Porträtfotos von sich und seinen wissenschaftlichen Mitarbeiter:innen veröffentlichen. Dazu beauftragt er seine Sekretärin Aidualc die Namen von m Fotograf:innen aus Frankfurt und ein Beispiel ihrer Arbeit zusammenzustellen. Es ist allerdings allgemein bekannt, dass es nur k gute Fotograf:innen in Frankfurt gibt. Um eine:n gute:n Fotograf:in zu beauftragen, geht er wie folgt vor:

> FindeFotograf:in
> Ziehe einen uniform zufälligen Namen n_i eines:r Fotograf:in. Prüfe dann anhand von n_i's Beispielfotos, ob n_i gut ist. Wenn ja, dann beauftrage n_i damit, Porträtfotos für Professor Regloh zu machen. Wenn das nicht der Fall ist, suche weiter.

Sei n_j ein:e gute:r Fotograf:in. Sei X_j eine Zufallsvariable, die den Wert 1 annimmt, wenn Professor Regloh n_j beauftragt und 0 wenn nicht. Dann gilt:

$P(X_j = 1) =$ _____.

Was ist die erwartete Laufzeit von FindeFotograf:in? Oder in anderen Worten, was ist die erwartete Anzahl von Porträtfotos, die Professor Regloh anschauen muss?

_____

(/ 2 Punkte)

a) Welche zwei der folgenden fünf Aussagen sind wahr?

□ Jede *basic feasible solution* eines linearen Programms ist eine optimale Lösung.

□ Jedes lineare Programm, dessen zugehöriges Polyhedron beschränkt und nicht leer ist, hat eine optimale Lösung.

□ In jedem linearen Programm, das eine optimale Lösung hat, ist die optimale Lösung eindeutig.

□ Die Menge aller optimalen Lösungen eines linearen Programms ist konvex.

□ Jede optimale Lösung in einem linearen Programm ist auch eine *basic feasible solution*.

(/ 2 Punkte)

b) Welche zwei der folgenden fünf Aussagen sind wahr?

□ *Complementary slackness* kann benutzt werden, um von einer optimalen Lösung des dualen Programms eine optimale Lösung des primalen Programms zu berechnen.

□ Es gibt immer eine zulässige Lösung x des primalen Programms und eine zulässige Lösung y des dualen Programms, sodass $c^T x = b^T y$ gilt.

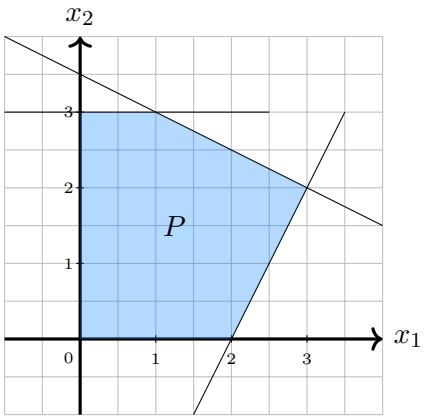
□ Jede zulässige Lösung x des primalen Programms hat einen Wert $c^T x$, der echt kleiner ist als der Wert der optimalen Lösung des dualen Programms.

□ Wenn das primale Programm unbeschränkt ist, dann ist das duale Programm *infeasible*.

□ Jede optimale Lösung des dualen Programms ist eine zulässige Lösung für das primale Programm.

(/ 2 Punkte)

c) Gegeben sei folgendes Polytop P in $\mathbb{R}^2$:



Geben Sie alle zulässigen Basen (*basic feasible solutions*) von P der Form (x_1, x_2) an:

_____.

Betrachten Sie nun das folgende lineare Programm (LP_1):

$\max \begin{pmatrix} 1 & 3 \end{pmatrix}^T \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ unter der Nebenbedingung $(x_1, x_2) \in P$

Geben Sie eine optimale Lösung für (LP_1) mit dem zugehörigen Zielfunktionswert an:

_____.

(/ 2 Punkte)

d) Gegeben sei folgendes lineare Programm (LP_2):

$$\begin{array}{lrcll} \min & x_1 & + & 16x_2 & \\ \text{s.t.} & 2x_1 & & & \leq 64 \\ & 4x_1 & + & 32x_2 & \geq 128 \\ & 8x_1 & & & = 256 \\ & x_1 & & & \geq 0 \\ & & & x_2 & \in \mathbb{R} \end{array}$$

Geben Sie (LP_2) in Standardform an:

(/ 3 Punkte)

Sei $G = (V, E)$ ein ungerichteter Graph. Eine Knotenmenge $I \subseteq V$ heißt *unabhängig*, falls $(v, w) \notin E$ für alle $v, w \in I$ gilt. Wir definieren das Entscheidungsproblem IndependentSet wie folgt:

Problem:	IndependentSet
Eingabe:	Ein Graph G mit n Knoten und eine Zahl $k \in \mathbb{N}$.
Frage:	Enthält G eine unabhängige Knotenmenge mit k Knoten?

a) Sei $N(v)$ die Menge aller Nachbarn von Knoten v. Beweisen Sie: Für alle maximalen unabhängigen Mengen $I \subseteq V$ und alle Knoten $v \in V$ gilt $(N(v) \cup \{v\}) \cap I \neq \emptyset$.

(/ 3 Punkte)

b) Sei Δ der Maximalgrad von G, also die kleinste Zahl Δ, sodass jeder Knoten von G höchstens Δ Nachbarn hat. Überlegen Sie sich einen **rekursiven** *bounded search tree* Algorithmus, der das IndependentSet-Problem in einer Laufzeit von

$$f(k, \Delta) \cdot \text{poly}(n)$$

löst, wobei $\text{poly}(n)$ ein Polynom in n und $f(k, \Delta)$ eine beliebige Funktion ist, die nur von k und Δ abhängt. Beschreiben Sie Ihren Algorithmus **in Pseudocode**.

Hinweis: Aufgabenteil a) könnte nützlich sein.

(/ 5 Punkte)

c) Betrachten Sie folgenden rekursiven Algorithmus:

```
function Rec(n, k)
    if k ≤ 0 then return 0
    else
        for i from 1 to n do
            print "Irgendeine Ausgabe, die nur wegen der Laufzeit wichtig ist."
        return Rec(n − 99, k − 1) + Rec(n − 7, k − 1) + Rec(n − 5, k − 1)
```

Geben Sie jetzt die Laufzeit von Rec in Θ-Notation in Abhängigkeit von n und k an:

$\Theta\Big($_____$\Big)$.

Beweisen Sie, dass Rec tatsächlich die von Ihnen angegebene Laufzeit hat.

(/ 2 Punkte)