Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Abteilung Algorithmik

Prof. Dr. Sándor P. Fekete
Arne Schmidt

Sommersemester 2017

Klausur
Algorithmen und Datenstrukturen II
31. 07. 2017

Name: .....

Vorname: .....

Matr.-Nr.: .....

Studiengang: .....

☐ Bachelor ☐ Master ☐ Andere

Mit der Veröffentlichung meines Klausurergebnisses unter meiner Matrikelnummer bin ich einverstanden.

..... *Unterschrift*

Hinweise:

- Bitte das Deckblatt in Druckschrift vollständig ausfüllen.
- Die Klausur besteht aus 14 Blättern, bitte auf Vollständigkeit überprüfen.
- Erlaubte Hilfsmittel: Keine.
- Eigenes Papier ist nicht erlaubt.
- Die Rückseiten der Blätter dürfen beschrieben werden.
- Die Klausur ist mit 50 % der Punkte bestanden.
- Antworten die *nicht* gewertet werden sollen bitte deutlich durchstreichen. Kein Tippex verwenden.
- Mit *Bleistift* oder in *Rot* geschriebene Klausurteile können nicht gewertet werden.
- Werden mehrere Antworten gegeben, werten wir die mit der geringsten Punktzahl.
- Sämtliche Algorithmen, Datenstrukturen, Sätze und Begriffe beziehen sich, sofern nicht explizit anders angegeben, auf die in der Vorlesung vorgestellte Variante.
- Die Bearbeitungszeit für die Klausur beträgt 120 Minuten.

Aufgabe	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	Σ
Punkte	**17**	**6**	**11**	**14**	**15**	**13**	**14**	**10**	**100**
Erreicht									
Note	—	—	—	—	—	—	—	—	

Aufgabe 1: Greedy **(3+7+4+3 Punkte)**

Betrachte folgende Instanz:

Objekt	i	1	2	3	4	5	6	7
Gewicht	z_i	5	20	9	7	4	19	7
Wert	p_i	4	14	6	8	2	19	5

mit $Z = 23$

a) Betrachte die Instanz als eine Instanz von Fractional Knapsack und wende den fraktionalen Greedy-Algorithmus an. Gib in jeder Iteration den aktuellen Gegenstand, zu welchen Teilen er gepackt wird, sowie den Gesamtzustand (Was ist gepackt und wie hoch ist der aktuelle Gesamtwert und -gewicht?) an.

b) Wende $\textsc{Greedy}_0$ auf die Instanz an. Gib in jeder Iteration den aktuellen Gegenstand an, sowie die Menge der bereits gepackten Gegenstände mitsamt ihrem Gesamtgewicht und -wert. Ist die erhaltene Lösung optimal? Begründe Deine Antwort!

c) Zeige: Für jedes $c > 0$ gilt, dass $\textsc{Greedy}_0$ keine c-Approximation ist.

d) Ist der Greedy-Algorithmus für Fractional Knapsack immer optimal? Begründe kurz Deine Antwort!

Aufgabe 2: Dynamic Programming **(6 Punkte)**

Betrachte folgende Instanz:

Objekt i	1	2	3	4	5
Gewicht z_i	8	3	7	7	5

mit $Z = 14$

Wende den Dynamic-Programming-Algorithmus für Subset Sum auf diese Instanz an, indem Du die folgende Tabelle ausfüllst und die Gegenstände in der Reihenfolge ihrer Indizes bearbeitest. (*Hinweis:* Der Eintrag in der Zeile i und der Spalte x entspricht dem Wert $\mathscr{S}(x, i)$.)

$i \backslash x$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0															
1															
2															
3															
4															
5															

Aufgabe 3: Branch and Bound **(2+2+2+5 Punkte)**

Betrachte den Branch-and-Bound-Algorithmus für KNAPSACK aus der Vorlesung mit dem Berechnen von GREEDY_0 als untere Schranke.

a) Welche Bedingung muss gelten, damit für ein $l \geq 1$ die bisherige Auswahl an Objekten zulässig ist?

b) Wie wird geprüft, ob man sich derzeit in einem Blatt befindet?

c) Welche Bedingung muss erfüllt sein, damit weiter verzweigt wird?

d) Gib eine Instanz mit n Objekten an, bei der der Branch-and-Bound-Algorithmus keine rekursiven Aufrufe startet. Zeige, dass Deine Instanz diese Forderung erfüllt!

Aufgabe 4: Modellierung **(3+4+7 Punkte)**

Betrachte das folgende Problem: Wir besitzen n verschiedene Münztypen, jede beliebig oft. Die Aufgabe ist es, mit so wenigen Münzen wie möglich einen Wunschbetrag P zu erreichen. Damit es immer möglich ist, P zu erreichen, besitzt ein Münztyp den Wert 1. Formal lässt sich das Problem folgendermaßen beschreiben:

Gegeben: n Münzwerte $p_1, \ldots, p_n$ mit $1 = p_1 < p_2 < \cdots < p_n$ und ein Betrag P.

Gesucht: Ganze Zahlen $x_1, \ldots, x_n$, sodass $\sum_{i=1}^{n} x_i p_i = P$ und $\sum_{i=1}^{n} x_i$ kleinstmöglich

Sei beispielsweise $p_1 = 1$, $p_2 = 3$, $p_3 = 7$ und $P = 32$. Eine Lösung wäre $x_1 = 1$, $x_2 = 1$ und $x_3 = 4$, denn $1 + 3 + 4 \cdot 7 = 32$. Dass P nicht mit weniger als sechs Münzen erreicht werden kann, kann schnell verifiziert werden.

Sei nun $\textsc{opt}(i, j)$ die minimale Anzahl an Münzen mit den Werten $p_1, \ldots, p_i$, die den Betrag j erreichen.

a) Erstelle eine Rekursionsgleichung, mit der $\textsc{opt}(i, j)$ berechnet werden kann.

b) Berechne mit der Rekursionsgleichung aus Aufgabenteil a) den Wert von $\textsc{opt}(3, P)$ mit den Münztypen $p_1 = 1, p_2 = 2, p_3 = 5$ und $P = 7$

c) Entwirf ein dynamisches Programm, das für gegebene Werte $p_1, \ldots, p_n$ und gegebenes P die minimale Anzahl an Münzen zurück gibt, die benötigt werden, um P zu erreichen.

Aufgabe 5: Approximation **(11+4 Punkte)**

a) Wende $\textsc{Greedy}_k$ auf die folgende Instanz an.

i	1	2	3	4
$p_i = z_i$	9	3	7	20

mit $Z = 30$ und $k = 2$.

Gib dazu die folgenden Mengen bzw. Werte tabellarisch an:

- $\overline{S}$
- $\sum_{i \in \overline{S}} z_i$
- $Z - \sum_{i \in \overline{S}} z_i$
- $A_{\overline{S}} := \textsc{Greedy}_0(\{z_i | i \notin \overline{S}\}, Z - \sum_{i \in \overline{S}} z_i, \{p_i | i \notin \overline{S}\})$
- $\sum_{i \in \overline{S}} p_i + A_{\overline{S}}$
- G_k
- S

Achte darauf, dass $\overline{S}$ mit der kleinsten Menge anfängt und mit der größten endet. Zusätzlich soll $\overline{S}$ lexikographisch sortiert sein, das heißt, für zwei gleichgroße Mengen $\overline{S}_1$ und $\overline{S}_2$ kommt $\overline{S}_1$ vor $\overline{S}_2$, falls das kleinste Element $x \in \overline{S}_1 \setminus \overline{S}_2$ kleiner ist als das kleinste Element $y \in \overline{S}_2 \setminus \overline{S}_1$.

b) Zeige oder widerlege: Für $k = n - 1$ ist $\textsc{Greedy}_k$ immer optimal.

Aufgabe 6: Komplexität **(7+6 Punkte)**

Betrachte folgende 3SAT-Instanz:

$$S = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee x_3)$$

a) Reduziere S auf eine Instanz I von SUBSET SUM, so dass I genau dann eine Lösung vom Wert Z hat, wenn S erfüllbar ist.

b) Gib drei prinzipiell verschiedene Vorgehensweisen für ein NP-vollständiges Problem an, benenne jeweils die Vor- und Nachteile jeder Alternative und führe jeweils ein Beispiel für solch einen Algorithmus an. (*Hinweis*: Es reicht, die Algorithmen zu benennen, sie müssen nicht komplett aufgeschrieben werden.)

Aufgabe 7: Hashing **(9+2+3 Punkte)**

a) Betrachte ein anfangs leeres Array A der Größe 11, es gibt also die Speicherzellen $A[0], \ldots, A[10]$. In diesem Array führen wir offenes Hashing mit der folgenden Hashfunktion durch:

$$t(i, x) = x^2 + 2ix \mod 11$$

Dabei ist x ein einzusetzender Schlüssel und i die Nummer des Versuches, x in eine unbesetzte Speicherzelle des Arrays zu schreiben, beginnend bei $i = 0$. Berechne zu jedem der folgenden Schlüssel die Position, die er in A bekommt:

$$8, \quad 3, \quad 2, \quad 5, \quad 6, \quad 7$$

Dabei sollen die Schlüssel in der gegebenen Reihenfolge eingefügt werden und der Rechenweg soll klar erkennbar sein. Trage die Elemente in das Array in Tabelle 1 ein.

A[0]	**A[1]**	**A[2]**	**A[3]**	**A[4]**	**A[5]**	**A[6]**	**A[7]**	**A[8]**	**A[9]**	**A[10]**

Tabelle 1: Das Array A.

b) Wie ist der Belegungsfaktor β einer Tabelle der Größe m definiert und welchen Belegungsfaktor besitzt die Hashtabelle aus Aufgabenteil a) nach Einfügen aller Schlüssel?

c) Sei $h(x)$ eine Hashfunktion und $Prob(h(x) = j) = \frac{1}{m}$, wobei m die Größe der Hashtabelle ist. Wie hoch ist die Wahrscheinlichkeit, dass 3 Datensätze kollisionsfrei in eine Hashtabelle der Größe $m = 10$ eingefügt werden können? Wie hoch ist die Wahrscheinlichkeit bei 4 Datensätzen? Gib die Wahrscheinlichkeiten in Prozent an!

Aufgabe 8: Kurzfragen **(2+2+2+2+2 Punkte)**

a) Die Überprüfung, ob ein Schlüssel in einer Hashtabelle existiert, funktioniert mit jedem Hashverfahren in konstanter Zeit. ☐ wahr ☐ falsch

b) Lösungen für Fractional Knapsack geben immer obere Schranken für Integer Knapsack. ☐ wahr ☐ falsch

c) Aus $P \neq NP$ folgt, dass Fractional Knapsack $\notin NP$. ☐ wahr ☐ falsch

d) Bei 23 Personen liegt die Wahrscheinlichkeit bei etwa 51%, dass zwei Personen am gleichen Tag Geburtstag haben. ☐ wahr ☐ falsch

e) Die Laufzeit des dynamischen Programms für Knapsack ist nur von der Anzahl der Objekte abhängig. ☐ wahr ☐ falsch

(*Hinweis: Falsche Antworten geben* ***keine*** *Punktabzüge.*)

Viel Erfolg ☺