Algorithmen und Datenstrukturen 2 **SoSe 2020**
Prof. Dr. Sándor Fekete **Abgabe:** 18.06.20
Arne Schmidt **Rückgabe:** ab 24.06.20

Übungsblatt 3

Abgabe der Lösungen muss bis zum 18.06.20 um 23:59 Uhr erfolgen. Lösungen müssen per Mail mit einer pdf-Datei (Name der Datei „`blatt_[nr]_[grp]_[team].pdf`“) an den jeweiligen Tutor geschickt werden. Die Email-Adressen sind unter `https://www.ibr.cs.tu-bs.de/alg/index.html` zu finden.

Hausaufgabe 1 (Branch-And-Bound): **(6+4 Punkte)**
In dieser Aufgabe betrachten wir den Branch-and-Bound-Algorithmus für Maximum Knapsack aus der Vorlesung. Wir benutzen Greedy_0 (siehe Blatt 1) als untere Schranke und den (abgerundeten Wert des) Greedy-Algorithmus für Fractional Knapsack als obere Schranke.

a) Wende den Branch-and-Bound-Algorithmus auf folgende, nach $\frac{z_i}{p_i}$ aufsteigend sortierte Instanz für Maximum Knapsack an.

i	1	2	3	4
z_i	4	7	5	7
p_i	5	8	5	6

und $Z = 14$

Beachte folgende Punkte:
- Benutze den Enumerationsbaum aus Abbildung 1.[1]
- Beschrifte die Kanten mit der Auswahl, die getroffen wurde.
- Beschrifte die Knoten mit den aktuell besten Schranken (obere und untere).
- Ist eine Auswahl unzulässig, beschrifte den jeweiligen Knoten mit *unzulässig*.
- Sollten Kanten nicht benutzt werden, streiche sie durch.
- Halte in einer Tabelle fest, welche Objekte eine neue, beste Lösung liefern.

b) Zeige: Für jedes $n \geq 1$ gibt es eine Instanz mit n Objekten, bei der der Branch-and-Bound-Algorithmus keine rekursiven Aufrufe startet.

Hausaufgabe 2 (Integer Knapsack): **(4+3+3 Punkte)**
Betrachte das Problem Integer Knapsack, d.h. Objekte dürfen beliebig oft ganzzahlig verwendet werden. Wir wollen in dieser Aufgabe einen Branch-and-Bound-Algorithmus für dieses Problem entwickeln.

a) Beschreibe kurz einen Greedy-Algorithmus, der die fraktionale Variante (also $x_i \in \mathbb{R}_{\geq 0}$ für alle $i \in \{1, \ldots, n\}$) in $O(n)$ Zeit löst. Begründe außerdem kurz die Korrektheit.

b) Beschreibe kurz einen Greedy-Algorithmus, der in $O(n \log n)$ Zeit eine ganzzahlige, zulässige Lösung zurückgibt, die nicht erweitert werden kann, d.h. das Hinzufügen eines beliebigen Elements macht die Lösung unzulässig.

c) Wandle den Pseudocode des Branch-and-Bound-Algorithmus für Maximum Knapsack so ab, dass er für Integer Knapsack funktioniert.

[1] Unter `https://www.ibr.cs.tu-bs.de/courses/ss20/aud2/webpages/index.html#ueb` gibt es den Baum als .ipe und .svg, um die Daten direkt einzutragen.

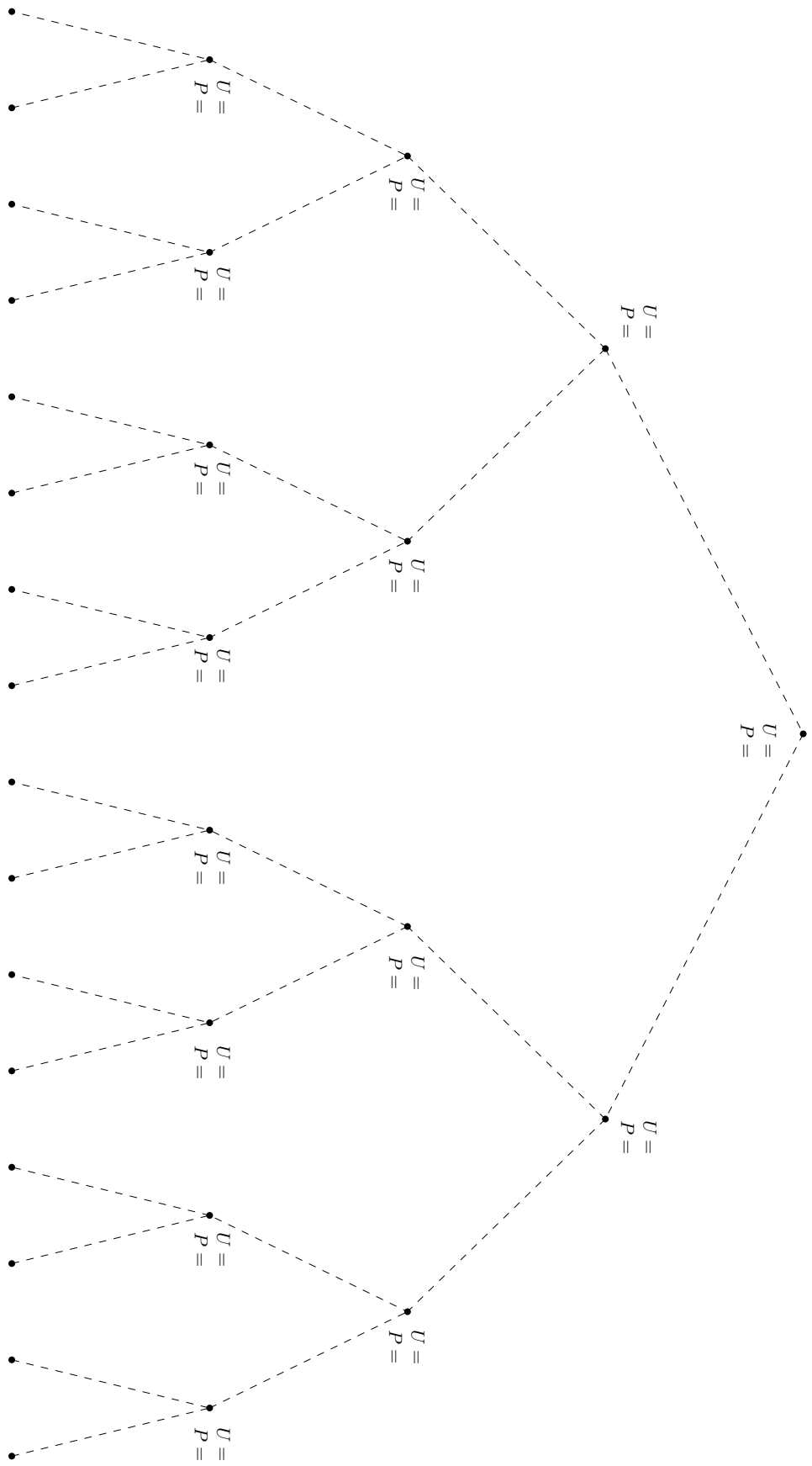

Abbildung 1: Ein Entscheidungsbaum.

Hausaufgabe 3 (Lageroptimierung): **(4+6 Punkte)**

Der Versandhändler Pesos benötigt Lagerhallen für seinen Betrieb Anasom, um Waren zu lagern und dann zu versenden. Allerdings möchte Anasom nur ein festes Sortiment anbieten, sodass die Waren dauerhaft in den Lagerhallen gelagert werden. Nun stellt sich die Frage, welche Waren angeboten werden sollen und welche Lagerhallen dafür benötigt werden, um möglichst viel Profit zu machen. Eine Ware i nimmt Raum $z_i > 0$ in Anspruch und liefert monatlich einen Erlös $p_i > 0$; eine Lagerhalle i liefert zusätzlichen Raum (entsprechend $z_i < 0$), kostet aber monatlich Miete (entsprechend $p_i < 0$).

Formal lässt sich das LAGERHALLEN-PROBLEM so beschreiben:

Gegeben: $z_1, \ldots, z_n, p_1, \ldots, p_n \in \mathbb{Z}$ mit $z_i/p_i \in \mathbb{R}^+$ für alle $i = 1, \ldots, n$.
Objekte mit $z_i < 0$ entsprechen Lagerhallen, Objekte mit $z_i \geq 0$ entsprechen Waren.

Gesucht: Eine Auswahl $S \subset \{1, \ldots, n\}$ mit

- $\sum_{i \in S} z_i \leq 0$
- $\sum_{i \in S} p_i$ maximal

a) Zeige: Es gibt Instanzen dieses Problems, für die der Branch-and-Bound-Algorithmus für MAXIMUM KNAPSACK aus der Vorlesung keine optimale Lösung liefert. Wir nehmen dabei an, dass GREEDY_0 (siehe Blatt 1) als untere Schranke und der Greedy-Algorithmus für FRACTIONAL KNAPSACK als obere Schranke genutzt wird. Der Test der Zulässigkeit (zweite Zeile im Algorithmus 3.1 in der Vorlesung) wird auf

$$\textbf{\textit{if}} \sum_{j=1}^{\ell-1} b_j z_j > 0 \quad \textbf{\textit{then return}}$$

geändert.

b) Jetzt überlegt Anasom-Chef Pesos, einen Informatiker anzustellen, der einen Algorithmus entwirft, der das LAGERHALLEN-PROBLEM löst. Dieser sieht, dass man den Branch-and-Bound-Algorithmus für MAXIMUM KNAPSACK doch noch verwenden kann, indem man den Input anpasst.

Zeige: Jede Instanz I des LAGERHALLEN-PROBLEMS lässt sich in eine Instanz I' von MAXIMUM KNAPSACK in $O(n)$ Zeit transformieren, so dass umgekehrt aus einer optimalen Lösung P'_{OPT} für I' eine optimale Lösung P_{OPT} für I abgelesen werden kann. Begründe außerdem kurz, dass deine Transformation korrekt ist und die Laufzeit einhält.

(Hinweis:
(1) MAXIMUM KNAPSACK benötigt eine Kapazität und keine negativen Gewichte/Werte.
(2) Kapazität und Kosten der Lagerhallen müssen verändert werden.)