



Technische
Universität
Braunschweig



Algorithmen und Datenstrukturen 2 – Übung #0

Matthias Konitzny

21.04.2021

Übersicht

- Organisation
- Wiederholung AuD 1
- Greedy für Fractional Knapsack

Organisation

Homepage

<https://www.ibr.cs.tu-bs.de/courses/ss21/aud2/>

Dort findet ihr

- Allgemeine Informationen
- Aktuelle Hinweise

The screenshot shows the course page for 'Algorithmen und Datenstrukturen 2' at TU Braunschweig. The page features a navigation bar with links for 'Impressum', 'Datenschutz', 'English', 'Login', and 'Schnellsuche'. A sidebar on the left contains a menu with categories like 'News', 'Wir über uns', 'Connected and Mobile Systems', 'Algorithmik', 'Lehrveranstaltungen', 'Abschlussarbeiten', 'Projekte', 'Veröffentlichungen', 'Verteilte Systeme', 'Mikroprozessorlabor', 'Studium', 'Sommersemester 2021', 'Wintersemester 2020/2021', 'Abschlussarbeiten', 'Service', 'Spin-Offs', and 'Forschungsverbände'. The main content area displays course details for 'Algorithmen und Datenstrukturen 2', including the semester (Sommersemester 2021), module number (INF-ALG-23), and a list of lecturers and assistants with their contact information. The page also includes a list of lecture times and locations, and a section for prerequisites and grading.

Technische Universität Braunschweig | Carl-Friedrich-Gauß-Fakultät | Department Informatik | Institut für Betriebssysteme und Rechnerverbund

Impressum | Datenschutz | English | Login | Schnellsuche

Algorithmen und Datenstrukturen 2

Semester: Sommersemester 2021 •
Modulnummer: INF-ALG-23
Veranstaltungsnr.: INF-ALG-042, INF-ALG-043, INF-ALG-044
Studiengänge: Bachelor Wirtschaftsinformatik, Bachelor Informations-Systemtechnik, Bachelor Informatik
IBR Gruppe: ALG (Prof. Fekete)
Art: Vorlesung/Übung

Dozent:  **Prof. Dr. Sándor P. Fekete**
Abteilungsleiter
s.fekete@tu-bs.de
+49 531 3913111
Raum 335

Assistent:  **Matthias Konitzny**
Wissenschaftlicher Mitarbeiter
konitzny@ibr.cs.tu-bs.de
+49 531 3913113
Raum 333

LP: 5
SWS: 2+1+1
Ort & Zeit: Vorlesung: Dienstags, 9-15 Uhr, Raum: Online
Große Übung: Mittwochs, 15:00 Uhr, Raum: Online
Kleine Übung: TBA
Für eine detaillierte Semesterplanung gibt es einen [Semesterplan](#).

Beginn: Erste Vorlesung: 20.04.21
Erste große Übung: 21.04.21
Erste kleine Übung: 03-07.05.21

Voraussetzungen: Keine
Sprache: Deutsch
Scheinerwerb: Studienleistung: Mindestens 50 Prozent der Hausaufgabenpunkte
Prüfungsleistung: Klausur am Ende des Semesters

Organisation

Materialseite

<https://aud2.ibr.cs.tu-bs.de>

Dort findet ihr

- Aktuelle Informationen
- Vorlesungsvideos und Links zu den Live-Veranstaltungen
- Hausaufgaben
- Skript (bei Fehlern bitte Mail an mich)
- Literaturempfehlungen
- Weiterführende Links

The screenshot shows the homepage of the course website. The header is dark red with the course title 'Algorithmen und Datenstrukturen 2' and 'Sommersemester 2021'. A navigation menu includes 'Startseite', 'Vorlesungen', 'Organisation', 'Kapitel', 'Kontakt', and 'Archiv'. The main content area has a white background with a dark red sidebar on the right. The main content features a title 'Algorithmen und Datenstrukturen 2', a welcome message, a description of the course, a list of topics, and a 'Literatur' section. The sidebar contains 'Neuigkeiten' and 'Letzte Vorlesungen' sections.

Algorithmen und Datenstrukturen 2
Sommersemester 2021

Startseite Vorlesungen Organisation Kapitel Kontakt Archiv

Startseite Startseite / Algorithmen und Datenstrukturen 2

Algorithmen und Datenstrukturen 2

Die Vorlesung **Algorithmen und Datenstrukturen 2** ist eine Wahlpflichtveranstaltung für Studierende der Informatik, Wirtschaftsinformatik, Informations- und Systemtechnik; außerdem ist sie offen für interessierte Studierende anderer Studiengänge.

Algorithmen sind das methodische Herz der theoretischen und praktischen Informatik; Datenstrukturen ermöglichen die effiziente Umsetzung von Algorithmen und den effizienten Zugriff auf Input- und Outputdaten. In dieser weiterführenden Vorlesung werden die folgenden grundlegenden Begriffe erarbeitet:

- Elementare Aspekte zu Heuristiken
- Exakte Verfahren: Dynamic Programming, Branch-and-Bound
- Approximationsalgorithmen
- Komplexitätsaspekte
- Hashing

Literatur

- **Skript:** Zu dieser Vorlesung gibt es ein [MANUSKRIFT](#). (Stand: 30.06.20)

Neuigkeiten

Herzlich Willkommen bei Algorithmen und Datenstrukturen 2. Wir starten am 20.04.2021 ins Sommersemester 2021.

Voraussichtlich werden alle Veranstaltungen online stattfinden.

Bitte meldet Euch auf der Mailingliste an! Wir nutzen diese, um kurzfristig Informationen zu versenden. Bitte nutzt, soweit möglich, E-Mail-Adressen der TU Braunschweig. ([mail-liste](#))

Letzte Vorlesungen

- ✓ Herzlich Willkommen bei AuD2!

Mailingliste

Mailingliste

Es gibt eine **Mailingliste** zu dieser Vorlesung. Bitte meldet euch mit eurer tu-bs-Email-Adresse dort an, denn wir werden sie nutzen, um kurzfristig Informationen zu verteilen. Bei technischen Schwierigkeiten wendet euch bitte an **Matthias Konitzny**.
Wichtig: Um Spam zu vermeiden, werden nur noch Mailadressen der TU Braunschweig freigegeben.

Anmeldung über Homepage

Information zur Vorlesung/Übung/Klausur/etc.

Bietet Möglichkeit Fragen zu stellen

- Wir (Prof. Fekete, Hiwis und ich) stehen auf der Liste und können Fragen beantworten
- Ihr könnt euch untereinander Fragen beantworten!

Subscribing to Aud2

Subscribe to Aud2 by filling out the following form. You will be sent email requesting confirmation, to prevent others from gratuitously subscribing you. Once confirmation is received, your request will be held for approval by the list moderator. You will be notified of the moderator's decision by email. This is also a private list, which means that the list of members is not available to non-members.

Your email address:

Your name (optional):

You may enter a privacy password below. This provides only mild security, but should prevent others from messaging with your subscription. Do not use a **valuable password** as it will occasionally be emailed back to you in cleartext.

If you choose not to enter a password, one will be automatically generated for you, and it will be sent to you once you've confirmed your subscription. You can always request a mail-back of your password when you edit your personal options.

Pick a password:

Reenter password to confirm:

Which language do you prefer to display your messages? English (USA)

Would you like to receive list mail batched in a daily digest? No Yes

Anmeldung

Anmeldung zu kleinen Übungen auf der Homepage

Gruppe	Termin (14-täglich)	Tutor
01	Mittwoch: 09:45 – 11:15	Pia Meier
02	Mittwoch: 13:15 – 14:45	David Meyer
03	Freitag: 9:45 – 11:30	Anna Ronsdorf
04	Freitag: 13:15 – 14:45	Dennis Luck

Anmeldung

20. April 2021

Anmeldung Kleine Übungen

in Allgemein

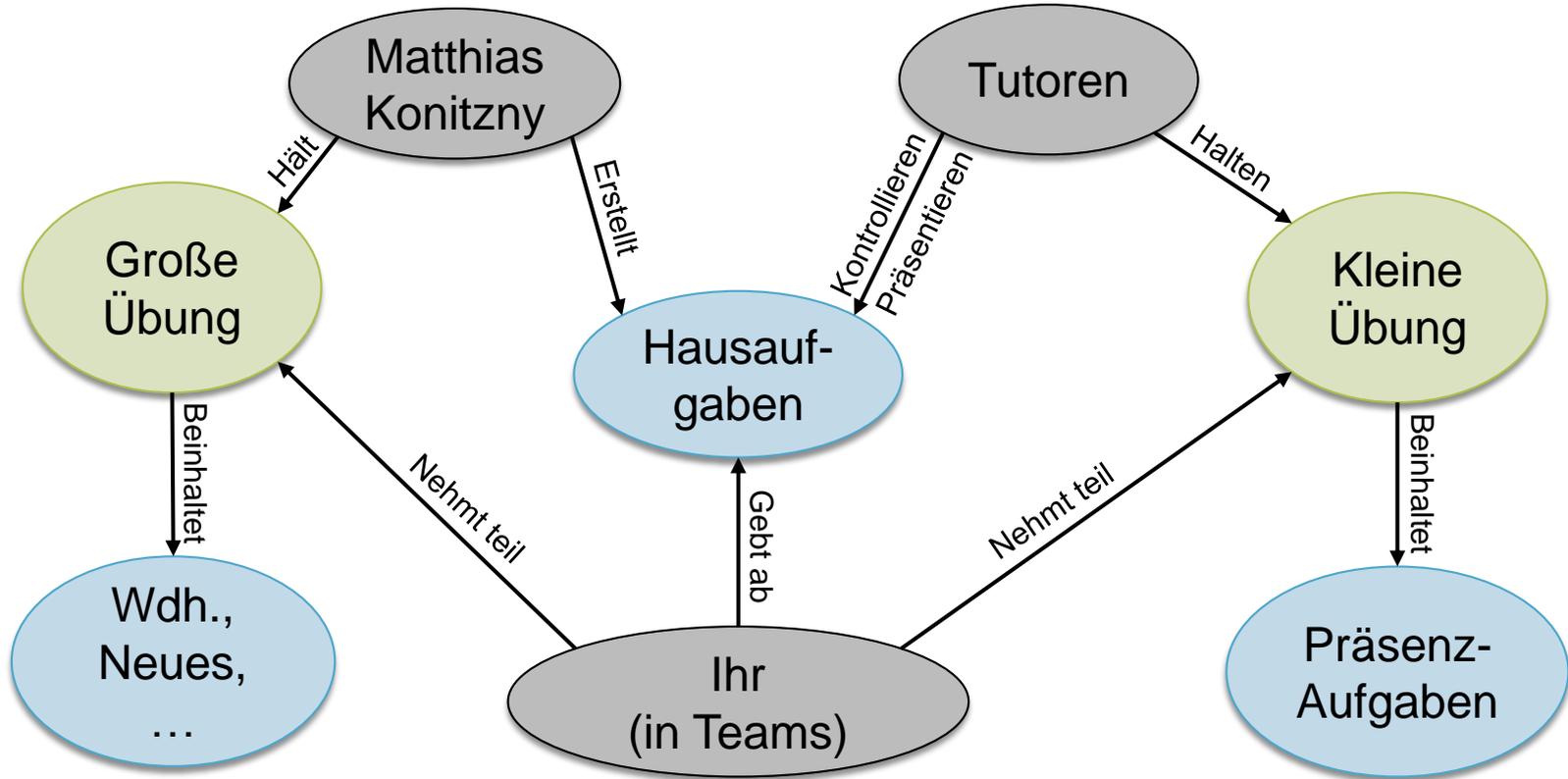
Die Anmeldung für die kleinen Übungen ist ab jetzt bis zum 26.04.21 geöffnet. Anschließend erhalten alle Teilnehmenden eine EMail mit der zugeteilten Übung und Gruppe.

Weitere Informationen

- Die Hausaufgaben sollen in Partnerarbeit erledigt werden. Dazu muss die y-Nummer des/der Wunschpartner*in im Formular angegeben werden.
- Die Wunschpartner*innen werden nur zugeteilt, wenn beide Partner*innen die y-Nummer des jeweils anderen angeben.
- Bitte achtet darauf, dass alle Angaben korrekt sind. Prüft dazu die Bestätigungsemail die ihr nach dem Anmeldevorgang erhaltet.
- Sollten mehrere Anmeldungen derselben Person vorliegen, berücksichtigen wir die Neueste.
- Sollten unvorhergesehene Probleme auftreten, meldet euch bei [Matthias Konitzny](#)

Nachname	<input type="text"/>
Vorname	<input type="text"/>
Matrikelnummer	<input type="text"/>
Studiengang	<input type="text" value="Informatik"/>
Semester	<input type="text" value="1"/>
y-Nummer	<input type="text"/>
E-Mail-Adresse	<input type="text"/>
Termine	<input type="checkbox"/> Mittwoch 9:45 Uhr <input type="checkbox"/> Mittwoch 13:15 Uhr <input type="checkbox"/> Freitag 9:45 Uhr <input type="checkbox"/> Freitag 13:15 Uhr
y-Nummer Partner	<input type="text"/>
<input type="button" value="Eingaben Abschieken"/>	

Übungsbetrieb



Hausaufgaben

- Voraussichtlich 5 bewertete Blätter à 20 Punkte → insgesamt 100 Punkte erreichbar.
- Für die Studienleistung müssen 50% der Punkte erreicht werden.
 - **ABER:** Keine Mindestpunktzahl pro Blatt!
- Bearbeitung in Zweiergruppen
 - Vorschlag: Jeder löst die Aufgaben selbstständig – anschließend Diskussion
- Abgabefrist steht auf dem jeweiligen Blatt.
- Zu spät abgegebene Hausaufgaben werden mit 0 Punkten bewertet
- Abgabe als Team per Mail an die Tutor*innen. (<https://www.ibr.cs.tu-bs.de/alg/index.html>)

Hausaufgaben abgeben

Wie werden Hausaufgaben abgegeben?

1. Erstellen der elektronischen Lösung

- Einscannen/Abfotografieren einer auf Papier geschriebenen Lösung (ggf. große Dateien)
- TeX, Word, OpenOffice, etc. (ggf. umständlich für Formeln)
- Konvertierung ins PDF Format (nur eine einzige Datei!)
- Dateiname im Format `blatt[nr]_[name1]_[name2]_[gruppennummer].pdf`

2. Abgabe per Mail

- Eine Person schickt elektronische Lösung per Mail an den Tutor.
- Alle Personen des Teams müssen auf CC stehen.
- Nutzt keine Filehosting-Dienste (Dropbox, Google-Drive, etc)

Zeitlicher Ablauf

Anmeldung

Bis zum 26.04.21
Um 23:59 Uhr

Einteilung
in Gruppen

Am 27.04.21

0. kl.
Übung

05.05./07.05.

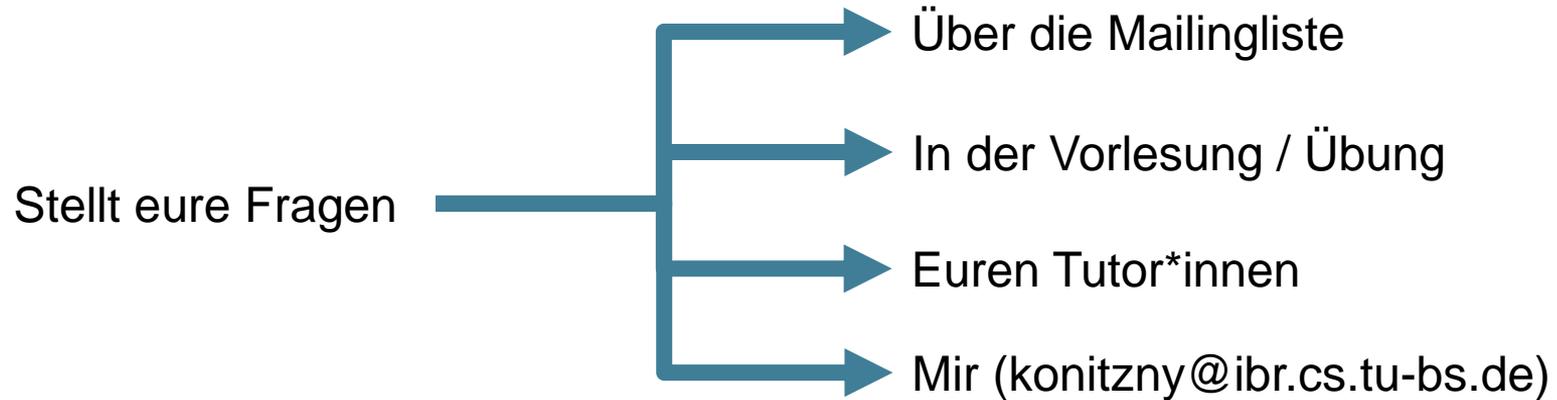
Personen ohne
Team müssen bis
07.05. ein Team
finden.

Einteilung
in Teams

08.05.

Nur für Personen,
die noch keinem
Team zugewiesen
sind.

Mehr Fragen!



Was ist mit
der Klausur?

Fragen?



Previously on AuD...

Problem vs Instanz

Problem

Allgemeine Formulierung der Ein- und Ausgabe

Eingabe:

$$z_1, \dots, z_n, Z, p_1, \dots, p_n$$

Ausgabe:

$S \subseteq \{1, \dots, n\}$ mit $\sum_{i \in S} z_i \leq Z$
und $\sum_{i \in S} p_i$ maximal.

Lösung: Angabe eines Algorithmus

Instanz

Konkrete Werte für Ein- und Ausgabe

Eingabe: $Z = 12$ und Objekte

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

Ausgabe:

$S = \{1, 2, 4, 5\}$ denn
 $\sum_{i \in S} z_i = 9 \leq 12$ und
 $\sum_{i \in S} p_i = 11$

Ist das bestmöglich?

Lösung: Angabe konkreter Werte

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

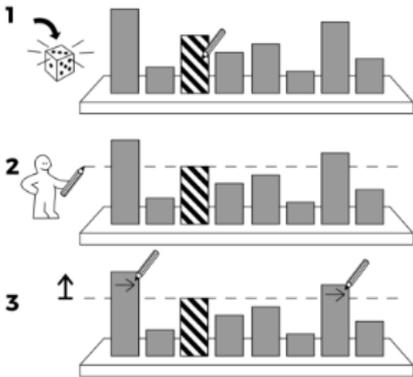
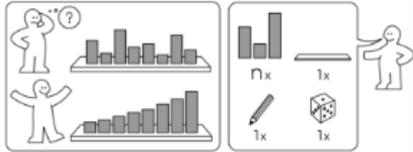
und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

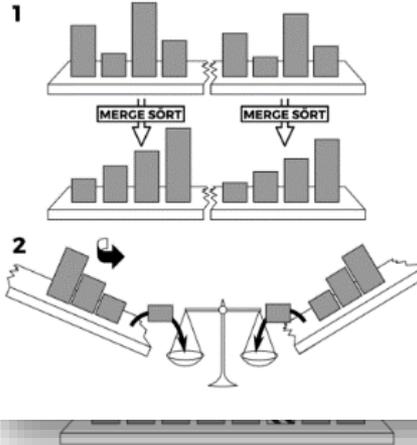
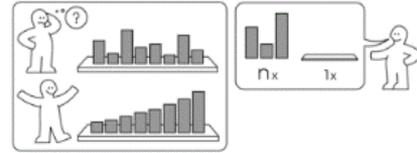
- 1: **Sortiere** $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;
Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.
Setze $j = 1$.
 - 2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**
 - 3: $x_{\pi(j)} := 1$
 - 4: $j := j + 1$
 - 5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$
 - 6: **return**
-

Ein paar Sortierverfahren

KWICK SÖRT

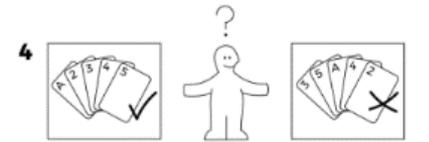
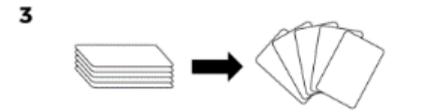
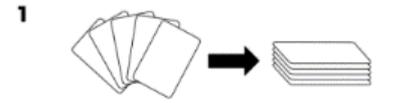
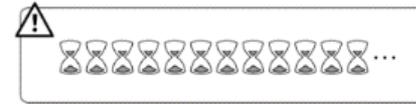
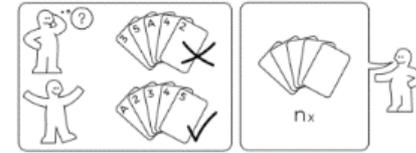


MERGE SÖRT



BOGO SÖRT Nicht ganz ernst gemeint ;-)

idea-instructions.com/bogo-sort/
v1.0. CC by-nc-sa 4.0 **IDEA**



Laufzeit – Sortieren

Wie lange dauern diese Algorithmen?

Algorithmus	Best-Case	Average-Case	Worst-Case
Quicksort	$O(n \log n)$	$O(n \log n)$	$O(n^2)$
Mergesort	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$

Theorem:

Jedes vergleichsbasierte Verfahren benötigt $\Omega(n \log n)$ Schritte.

Laufzeit

Sei $f(n)$ die Laufzeit eines Algorithmus mit Inputgröße n .

O -Notation:

Können wir garantieren, dass $f(n) \leq c_1 \cdot g(n)$ ab einem n_0 gilt, so schreiben wir

$$f(n) \in O(g(n)).$$

Maximale Laufzeit

Ω -Notation:

Können wir garantieren, dass $f(n) \geq c_2 \cdot g(n)$ ab einem n_0 gilt, so schreiben wir

$$f(n) \in \Omega(g(n)).$$

Mindestlaufzeit

Fractional Knapsack – Greedy Algorithmus

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

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

$Z = 12$ und

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

j	$\pi(j)$	$x_{\pi(j)}$	$\sum_{i=1}^j x_i z_i$	$Z - \sum_{i=1}^j x_i z_i$	$\sum_{i=1}^j x_i p_i$
-----	----------	--------------	------------------------	----------------------------	------------------------

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

j	$\pi(j)$	$x_{\pi(j)}$	$\sum_{i=1}^5 x_i z_i$	$Z - \sum_{i=1}^5 x_i z_i$	$\sum_{i=1}^5 x_i p_i$
1	1	1	2	10	3

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

j	$\pi(j)$	$x_{\pi(j)}$	$\sum_{i=1}^5 x_i z_i$	$Z - \sum_{i=1}^5 x_i z_i$	$\sum_{i=1}^5 x_i p_i$
1	1	1	2	10	3
2	4	1	6	6	8

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

j	$\pi(j)$	$x_{\pi(j)}$	$\sum_{i=1}^5 x_i z_i$	$Z - \sum_{i=1}^5 x_i z_i$	$\sum_{i=1}^5 x_i p_i$
1	1	1	2	10	3
2	4	1	6	6	8
3	2	1	7	5	9

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

j	$\pi(j)$	$x_{\pi(j)}$	$\sum_{i=1}^5 x_i z_i$	$Z - \sum_{i=1}^5 x_i z_i$	$\sum_{i=1}^5 x_i p_i$
1	1	1	2	10	3
2	4	1	6	6	8
3	2	1	7	5	9
4	5	1	9	3	11

Fractional Knapsack – Greedy Algorithmus

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n$

Ausgabe: $x_1, \dots, x_n \in [0, 1]$

mit

$$\sum_{i=1}^n z_i x_i \leq Z$$

und

$$\sum_{i=1}^n p_i x_i = \text{Maximal}$$

1: Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;

Dies ergibt die Permutation $\pi(1), \dots, \pi(n)$.

Setze $j = 1$.

2: **while** $(\sum_{i=1}^j z_{\pi(i)} \leq Z)$ **do**

3: $x_{\pi(j)} := 1$

4: $j := j + 1$

5: Setze $x_{\pi(j)} := \frac{Z - \sum_{i=1}^{j-1} z_{\pi(i)}}{z_{\pi(j)}}$

6: **return**

$Z = 12$ und

i	1	2	3	4	5
z_i	2	1	7	4	2
p_i	3	1	4	5	2
$\frac{z_i}{p_i}$	$\frac{2}{3}$	1	$\frac{7}{4}$	$\frac{4}{5}$	1
RF	1	3	5	2	4

j	$\pi(j)$	$x_{\pi(j)}$	$\sum_{i=1}^5 x_i z_i$	$Z - \sum_{i=1}^5 x_i z_i$	$\sum_{i=1}^5 x_i p_i$
1	1	1	2	10	3
2	4	1	6	6	8
3	2	1	7	5	9
4	5	1	9	3	11
5	3	3/7	12	0	12 + 5/7

Knapsack – Greedy Algorithmen

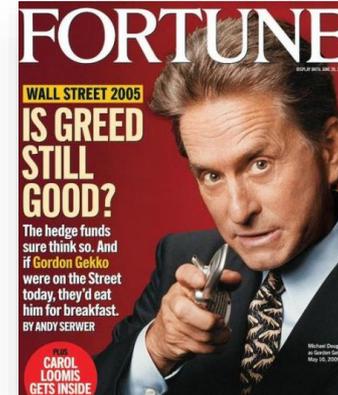
Greedy für fraktionales Knapsack



https://c1.staticflickr.com/9/8246/8491125952_c2c7c854d8.jpg

Greedy ist optimal!

Greedy für ganzzahliges Knapsack



<http://coverjunkie.com/uploads/1315147575.jpg>

Ist Greedy noch optimal?