Rucksackproblem (Knapsack)

Problemstellung:
- Gegeben sei eine Menge von n Items, die ein bestimmtes Gewicht und einen bestimmten Wert haben
- Ziel ist es eine Auswahl von Items zu finden, die in einen Rucksack mit der Kapazität k passt und einen Maximalen Gesamtwert hat

Praktische Relevanz:
- Logistik - Gewinnbringende Beladung eines LKWs mit maximaler Effizienz
- Problem lässt sich abändern → Gewicht = Zeit (Items als Aufträge)

Exakter Lösungsalgorithmus (M1):
(1) Alle möglichen Kombinationen der Items erzeugen und Gewicht überprüfen
(2) Wert jeder Kombination speichern und die mit dem größten Wert wählen
➡ Art der Problemlösung: Brute force [“rohe Gewalt”]

Laufzeit:
- $\mathcal{O}(2^n)$ - Exponentiell
  - Es gibt 2^n Möglichkeiten, Teilmengen aus einer n-elementigen Menge (Items) zu bilden
- Problemgröße: Anzahl n der Gegenstände
- Kostenmaß: Anzahl der abzuarbeitenden Kombinationen
  - Kostenfunktion: K(n) = 2^n - Exponentielle Komplexität
- Bis heute ist kein polynominaler Algorithmus bekannt
- Beinhaltet Subset-Sum Problem, dass NP-Vollständig ist
➡ Rucksackproblem ist ebenfalls NP-Vollständig

Praktische Näherungslösung (M2):
- Greedy [“gierig”] Algorithmus - Nutzung eines Profitabilitätsindex
(1) Jedes Item Bewerten (Gewicht / Wert)
(2) Items nach Bewertung Sortieren
(3) Rucksack mit best bewerteten Gegenständen füllen

Quellen [Aufgerufen: 15.12.2022]:

- http://www2.math.uni-wuppertal.de/~beisel/Rucksack/mainKnapsack.pdf
- https://www.abenteuer-informatik.de/PDF/ai2020_oa_druckversion_a4.pdf
- https://www.tinohempel.de/info/info/ti/rucksackproblem.html
- https://www.inf-schule.de/algorithmen/komplexitaet/rucksackproblem/station_komplexitaetsbetrachtungen
- https://de.wikipedia.org/wiki/Greedy-Algorithmus