Technische Universität Braunschweig

Algorithmen und Datenstrukturen II

6. Vorlesung

Linda Kleist, 29.05.2019

Approximation von Maximum Knapsack

Wie gut approximiert GREEDY$_0$ das Problem MAXIMUM KNAPSACK?

Algorithmus 6 (GREEDY$_0$)

Eingabe: $z_i, ..., z_n, Z, p_i, ..., p_n$

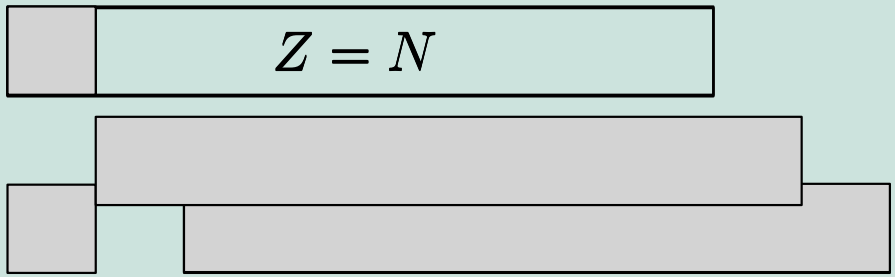
Ausgabe: $[(x_1, ..., x_n); G_0]$ mit $x_i \in \{0, 1\}$, $\sum_{i=1}^{n} x_i z_i \leq Z$, $G_0 := \sum_{i=1}^{n} x_i p_i$

1: Sortiere $\{1, ..., n\}$ nach $\frac{z_i}{p_i}$ aufsteigend; $\rightarrow$ Permutation $\pi(1), ..., \pi(n)$.
2: **for** $j = 1$ to n **do**
3: **if** $\left(\sum_{i=1}^{j-1} z_{\pi(i)} x_{\pi(i)} + z_{\pi(j)} \leq Z \right)$ **then**
4: $x_{\pi(j)} := 1$
5: **return** $[(x_1, ..., x_n); G_0]$

Approximation von Maximum Knapsack

Wie gut approximiert GREEDY$_0$ das Problem MAXIMUM KNAPSACK?

Beispiel



$z_1 = 1 - \varepsilon$ $\quad$ $z_2 = N$

$p_1 = 1$ $\quad$ $p_2 = N$

GREEDY$_0$ liefert $S_0 = \{1\}$ mit einem Gewinn von $G_0 = 1$.
Das Optimum ist $S_{\text{opt}} = \{2\}$ mit einem Gewinn von $G_{\text{opt}} = N$.

Satz 6

Die von Algorithmus 6 (GREEDY$_0$) berechnete Lösung einer Instanz für MAXIMUM KNAPSACK kann beliebig weit vom Optimum entfernt sein.
(GREEDY$_0$ approximiert MAXIMUM KNAPSACK nicht).

Approximation von Maximum Knapsack

Algorithmus 7 (modified GREEDY$_0$)

Eingabe: $z_i, ..., z_n, Z, p_i, ..., p_n$

Ausgabe: $[(x_1, ..., x_n); G_0']$ mit $x_i \in \{0, 1\}$, $\sum_{i=1}^{n} x_i z_i \leq Z$, $G_0' := \sum_{i=1}^{n} x_i p_i$

1: $(X, G_0) = \text{GREEDY}_0(z_i, ..., z_n, Z, p_i, ..., p_n)$ ▷ GREEDY$_0$-Lösung
2: $k = \text{argmax}_i\{p_i \mid z_i < Z, i \in \{1, ..., n\}\}$ ▷ Index des Objekts maximalen Gewinns
3: **if** $G_0 > p_k$ **then** ▷ Vergleich beider Optionen
4: **return** (X, G_0)
5: **else**
6: **return** $((0, ..., 0, x_k = 1, 0, ..., 0), p_k)$

Satz 7

Algorithmus 7 berechnet für jede Instanz von MAXIMUM KNAPSACK eine zulässige Lösung mit Gewinn G_0'. Im Vergleich zur optimalen Lösung mit Gewinn OPT gilt: $G_0' \geq \frac{1}{2}\text{OPT}$.

Approximations-Algorithmen

Sei Prob ein Maximierungsproblem und Alg ein Algorithmus

- Alg ist ein c-Approximationsalgorithmus für Prob, wenn für jede Instanz I von Prob mit Optimalwert $\text{OPT}(I)$ gilt:
  - Alg liefert in polynomieller Zeit in der Codierungsgröße von I eine zulässige Lösung mit Wert $\text{A}(I)$
  - $\text{A}(I) \geq c \cdot \text{OPT}(I)$ für eine Konstante $c \leq 1$.

Bemerkung: Falls Prob ein Minimierungsproblem fordern wir analog:

- $\text{A}(I) \leq c \cdot \text{OPT}(I)$ für eine Konstante $c \geq 1$.

Approximation von Maximum Knapsack

Algorithmus 8 (GREEDY$_k$)

Eingabe: $z_1, ..., z_n, Z, p_1, ..., p_n$ [Parameter k fixiert]

Ausgabe: $[(x_1, ..., x_n); G_k]$ mit $x_i \in \{0,1\}$, $\sum_{i=1}^{n} x_i z_i \leq Z$, $G_k := \sum_{i=1}^{n} x_i p_i$

1: $G_k := 0$
2: **for all** $(v_1, \ldots, v_n) \in \{0,1\}^n$ mit $\sum_i v_i \leq k$ **do** ▷ $\leq k$-Teilmenge
3: **if** $\left(\sum_{i=1}^{n} v_i z_i \leq Z\right)$ **then** ▷ falls zulässig
4: $I := \big((1 - v_1)z_1, \ldots, (1 - v_n)z_n, (Z - \sum_i v_i z_i), p_1, \ldots, p_n\big)$
▷ modifizierte Instanz (Wahl der Objekte mit $v_i = 1$ 'kostenlos')
5: $(X, G) := \text{GREEDY}_0(I)$ ▷ ergänze v durch Greedylösung
6: **if** $(G > G_k)$ **then** ▷ bei Verbesserung
7: $G_k := G$ ▷ aktualisiere besten Gewinn
8: $x := X$ ▷ aktualisiere beste Lösung
9: **return** $[(x_1, ..., x_n); G_k]$

Approximation von Maximum Knapsack

Satz 8

$\textsc{Greedy}_k$(Algorithmus 8) ist ein $\left(1 - \frac{1}{k+1}\right)$-Approximationsalgorithmus für jedes beliebig aber feste $k \in \mathbb{N}$.

Beweis.

Tafel... □

Für Algorithmus 8 gilt:

- Die Gütegarantie von $\left(1 - \frac{1}{k+1}\right)$ bestmöglich.
- Ersetzen von $\textsc{Greedy}_0$ in Zeile 5 durch modified $\textsc{Greedy}_0$, liefert eine Gütegarantie von $\left(1 - \frac{1}{k+2}\right)$.
- Für jedes $\epsilon > 0$ gibt es eine $(1 - \epsilon)$-Approximation für Maximum Knapsack $\rightarrow$ Approximationsschema.

Polynomial-Time Approximation Scheme (PTAS)

Ein polynomielles Approximationsschema (PTAS) für ein Optimierungsproblem ist eine Familie von Algorithmen, die für jedes beliebig aber feste $\epsilon > 0$ einen $(1-\epsilon)$-Approximationsalgorithmus (bzw. $(1+\epsilon)$) liefert.

Satz 9

$\{\textsc{Greedy}_k \mid k \in \mathbb{N}\}$ ist ein PTAS für Maximum Knapsack.

Beweis.

Wähle k groß genug, sodass $\frac{1}{1+k} < \varepsilon$. Verwende $\textsc{Greedy}_k$. □