1.5 Approximation

Wie gut ist Greedy? (Ganzzahlige Variante von Alg 1.4, G_0)

Schon in Übung gesehen: Beliebig schlecht!

Beispiel 1.21

Betrachte: $n = 2$, $z_1 = 1$, $p_1 = 1+\varepsilon$

$z_2 = N$, $p_2 = N$

$Z = N$

Dann liefert Greedy eine Lösung von Nutzen $1+\varepsilon$ (nur Objekt 1), optimal ist aber Nutzen N (nur Objekt 2).

Für $N \gg 1+\varepsilon$ ist das beliebig weit vom Optimum!

Relevante Größe ist dabei nicht die absolute Abweichung, sondern die relative: Wieviele Prozent des Optimums können wir erreichen?

ALGORITHMUS 1.22 ($Greedy_0$)

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n > 0$

Ausgabe: Eine Lösung $S \subseteq \{1, \dots, n\}$ mit $\sum_{i \in S} z_i \leq Z$
und Lösungswert $G_0 := \sum_{i \in S} p_i$

(1) Sortiere $\{1, \dots, n\}$ nach $\frac{z_i}{p_i}$ aufsteigend;
dies ergibt Permutation $\pi(1), \dots, \pi(n)$.
Setze $j := 1$

(2) WHILE $\left(\sum_{i=1}^{j} z_{\pi(i)} \leq Z \right)$ DO {

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

$j := j + 1$;

(3) RETURN

(Vergleiche Algorithmus 1.4 !

(2) WHILE $(j \leq n)$ DO {
IF $\sum_{i=1}^{j-1} z_{\pi(i)} x_{\pi(i)} + z_{\pi(j)} \leq Z$ {
$x_{\pi(j)} := 1$
}
$j := j + 1$
}
(3) RETURN

⚠ $x_j \to x_{\pi(j)}$ ⚠)

ALGORITHMUS 1.23

Eingabe: $z_1, \dots, z_n, Z, p_1, \dots, p_n > 0$ (mit $z_i \leq Z$)

Ausgabe: Eine Lösung $S \subseteq \{1, \dots, n\}$ mit $\sum_{i \in S} z_i \leq Z$

(1) Berechne G_0 mit Algorithmus 1.22

(2) Return $\tilde{G}_0 := \max \{ G_0, \max p_i \}$ und zugehöriges S

SATZ 1.24

Algorithmus 1.23 berechnet eine Lösung mit Wert $\tilde{G}_0$, für den im Vergleich zum bestmöglichen Lösungswert gilt

$$\tilde{G}_0 \geq \frac{1}{2} OPT.$$

Beweis:

Nach Konstruktion ist $\tilde{G}_0 \geq G_0$

und $\tilde{G}_0 \geq \max p_i =: p^*$

Außerdem ist (vgl. Algorithmus 1.4)

$$G_0 + p^* \geq \text{Fract KP} \geq OPT$$

Also ist $2\tilde{G}_0 \geq G_0 + p^* \geq OPT$, d.h. $\tilde{G}_0 \geq \frac{1}{2} OPT$.

Das motiviert die folgende

DEFINITION 1.25

(I) Für ein Maximierungsproblem MAX ist ein Algorithmus ALG ein c-Approximationsalgorithmus, wenn für jede Instanz I von MAX

(i) ALG in polynomieller Zeit in der Größe der Instanz eine Lösung mit Wert ALG(I) liefert.

(ii) Für den Vergleich mit dem zugehörigen Optimalwert OPT(I) gilt

$$ALG(I) \geq c \cdot OPT(I)$$

(2) Entsprechend verlangt man für ein Minimierungsproblem wiederum:

(i) polynomielle Laufzeit

(ii) $ALG(I) \leq c \cdot OPT(I)$

Merke: Maximieren $\rightarrow$ $c \leq 1$, je größer c, um so besser!

Minimieren $\rightarrow$ $c \geq 1$, je kleiner c, um so besser!

Wie gut kann man KNAPSACK approximieren?

ALGORITHMUS 1.26 ($Greedy_k$)

Eingabe: $z_1, \dots, z_n, z, p_1, \dots, p_n$ [Parameter k fixiert]

Ausgabe: Eine Lösung $S \subseteq \{1, \dots, n\}$ mit $\sum_{i \in S} z_i \leq z$

und Lösungswert $G_k := \sum_{i \in S} p_i$

(1) $G_k := 0$, $S := \emptyset$

(2) Für alle $\bar{S} \subseteq \{1, \dots, n\}$ mit $|\bar{S}| \leq k$ DO {

(2.1) IF $\sum_{i \in \bar{S}} z_i \leq z$ DO {

(2.1.1) $G_k := \max \{ G_k, \sum_{i \in \bar{S}} p_i + GREEDY_0 \left(\{z_i \mid i \notin \bar{S}\}, z - \sum_{i \in \bar{S}} z_i, \{p_i \mid i \notin \bar{S}\} \right) \}$,

(2.1.2) Update S

}

}

(3) RETURN G_k, S