

DAT-VA opgaver i FPTAS

David Pisinger, 2004/05

Knapsack Problem

I knapsack problemet er der givet

- en mængde $N = \{1, \dots, n\}$ af genstande
- en profit $p_i \in \mathbb{Z}^+$ for hver genstand $i \in N$
- en vægt $w_i \in \mathbb{Z}^+$ for hver genstand $i \in N$
- en knapsack (rygsæk) med kapacitet $c \in \mathbb{Z}^+$

Problemet er at vælge en delmængde $S \subseteq N$ således at $\sum_{j \in S} p_j$ bliver maximeret idet kapacitetsbegrænsningen $\sum_{j \in S} w_j \leq c$ skal overholdes.

Eksempel:

i	1	2	3	4
p_i	2	4	5	3
w_i	1	2	3	2

$$N = \{1, 2, 3, 4\}, c = 4$$

- 1 Find den optimale løsning $z^*(I)$ til ovenstående instans I
- 2 *Approximationsalgoritme.* Den grådige algoritme sorterer først elementerne efter aftagende p_i/w_i forhold, og tilføjer herefter genstande til rygsækken $i = 1, 2, \dots$ så længe der er plads til dem. Lad z^G betegne den fundne løsning. Vis at den grådige algoritme ikke er en approximations algoritme med fast faktor ρ .
- 3 En forbedret grådig algoritme finder først

$$p_{\max} = \max_{i \in N} p_i$$

og returnerer herefter den approximative løsning

$$z^A = \max\{p_{\max}, z^G\}$$

Vis at den forbedrede algoritme er en approximations algoritme med approximations faktor $\rho = 2$.

- 4 *Absolut performance garanti.* Vi siger at en polynomieltids approximationsalgoritme \mathcal{A} har absolut performance garanti $k \geq 0$, hvis

$$z^*(I) - z^A(I) \leq k$$

for all instances I . Her betegner $z^*(I)$ den optimale løsning, mens $z^A(I)$ betegner den approksimerede løsning.

Vis at der ikke kan findes en polynomieltids approximationsalgoritme for knapsack problemet med absolut performance garanti. (Hint: antag at der fandtes en algoritme med $k \geq 0$, og vis at man så kunne løse ethvert knapsack problem til optimalitet. Benyt eventuelt skalering af problemet).

5 *Dynamisk programmering i profitter.* For hvert $i \in \{1, \dots, n\}$ og $p \in \{1, \dots, n \cdot p_{\max}\}$ definer:

$$A(i, p) = \min \left\{ \sum_{j=1}^i w_j \mid \sum_{j=1}^i p_j = p \right\}$$

dvs. $A(i, p)$ er den mindste vægtsum der skal bruges for at finde en løsning med profitsum p , hvis vi kun må bruge genstandene $\{1, \dots, i\}$.

Initielt sætter vi

$$\begin{aligned} A(0, p) &= c + 1 \text{ for } p = 1, \dots, n \cdot p_{\max} \\ A(0, 0) &= 0 \end{aligned}$$

og benytter derefter rekursionsformlen

$$A(i+1, p) = \min \begin{cases} A(i, p) \\ A(i, p - p_{i+1}) + w_{i+1} \text{ if } p_{i+1} \leq p \end{cases}$$

Kør rekursionsligningen for ovenstående eksempel.

6 Vis at den optimale løsning til knapsack problemet kan findes som

$$z^* = \max\{p \mid A(n, p) \leq c\}$$

7 Vis at køretiden for den dynamiske programmeringsalgoritme er $O(n \cdot z^*) \leq O(n^2 p_{\max})$.

8 *FPTAS for knapsack problemet.* Betragt følgende algoritme

1. Given instance I and $\epsilon > 0$. Let $K = \max\{1, \frac{\epsilon p_{\max}}{n}\}$.
2. Let $p'_i = \lfloor \frac{p_i}{K} \rfloor$ for each $i \in N$.
3. Find the optimal solution set S corresponding to the knapsack problem defined on (p', w, c) , using dynamic programming in profits.
4. Output $z^A = \sum_{j \in S} p_j$.

Kør algoritmen for ovenstående eksempel med $\epsilon = \frac{9}{10}$

9 Vis at $\frac{|z^A - z^*|}{z^*} \leq \epsilon$.

10 Vis at algoritmen kører i fuldt polynomieltid $O(n^3 \frac{1}{\epsilon})$.