

Følsomhed af Knapsack Problemet

David Pisinger, *Projekt opgave 1*

Dette er den første obligatoriske projektopgave på kurset "DATV: Introduktion til optimering og operationsanalyse". Opgaven stilles fredag 16. februar 2007 og skal afleveres senest tirsdag 27. februar 2007 kl. 12.00 i DIKU's førstedelsadministration. For at blive godkendt skal der være gjort et reelt forsøg på at løse samtlige spørgsmål. Besvarelsen skal udarbejdes i grupper på to til tre deltagere. Grupper med én deltager kræver skriftlig accept fra instruktoren mindst en uge før opgaven skal afleveres. Læs venligst hele opgaveformuleringen igennem inden du går i gang. Hints til opgaverne kan fås ved øvelserne, hvor der er afsat tid til at arbejde med projektopgaven.

Indledning

Knapsack problemet har utallige anvendelser indenfor økonomi (budgetlægning) transport (pakning, ladning), og som underproblem ved løsning af mere komplekse problemer [4, 3]. Specielt bruges knapsack problemet til at separere lovlige uligheder i heltalsprogrammerings-løsere og det opstår som pricing problem når man løser bin-packing problemet ved Dantzig-Wolfe dekomponering. Knapsack problemet er et heltalsprogrammeringsproblem (IP-problem) som er NP-hårdt at løse.

Formelt kan *knapsack problemet* defineres på følgende vis: Lad der være givet n genstande som hver har en tilknyttet profit p_j og vægt w_j . Udvælg en delmængde af genstandene således at den samlede profit-sum bliver maksimeret uden at den tilhørende vægt-sum overstiger en given grænse c , kaldet kapaciteten. Hvis vi bruger binære variable x_j til at angive om en genstand vælges eller ej, får vi følgende matematiske definition af problemet:

$$\text{maximize } \sum_{j=1}^n p_j x_j \quad (1)$$

$$\text{subject to } \sum_{j=1}^n w_j x_j \leq c, \quad (2)$$

$$x_j \in \{0, 1\}, \quad j = 1, \dots, n. \quad (3)$$

Den optimale løsningsværdi vil blive betegnet z^* .

Det antages normalt at alle koefficienter p_j , w_j og c er positive heltal. Se evt. Cormen [2] afsnit 16.2 for yderligere beskrivelse af problemet.

Eksempel 1 I det følgende eksempel er $c = 9$ og der er givet $n = 7$ genstande med følgende profiler og vægte:

j	1	2	3	4	5	6	7
p_j	6	5	8	9	6	7	3
w_j	2	3	6	7	5	9	4

Den optimale løsning er at vælge genstandene 1 og 4 hvilket giver en optimal løsning på $z^* = 15$. ■

Løsning af det fraktionelle knapsack problem

LP-relaxeringen af (1) – (3) svarer til det *fraktionelle knapsack problem*, hvor det er tilladt at medtage brøkdeler af genstande:

$$\text{maximize } \sum_{j=1}^n p_j x_j \quad (4)$$

$$\text{subject to } \sum_{j=1}^n w_j x_j \leq c, \quad (5)$$

$$0 \leq x_j \leq 1, \quad j = 1, \dots, n. \quad (6)$$

Selv om problemet (4) – (6) er et normalt LP-problem, og derfor kan løses med Simplex algoritmen, kan det bedre betale sig at løse problemet med en grådig algoritme. Først sorteres genstandene efter aftagende effektivitet p_j/w_j således at

$$\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \frac{p_3}{w_3} \geq \dots \geq \frac{p_n}{w_n} \quad (7)$$

hvorpå rygsækken fyldes som følger: Elementerne $1, 2, 3, \dots$ lægges i rygsækken indtil man støder på den første genstand s , som der ikke er plads til. Den optimale løsning er da at medtage de første $s - 1$ genstande (dvs. $x_j = 1$ for $j = 1, \dots, s - 1$) mens en brøkdel af genstand s medtages således at hele den tilbageværende kapacitet udnyttes:

$$x_s = \frac{c - \sum_{j=1}^{s-1} w_j}{w_s}$$

Ingen af genstandene efter s medtages (dvs. $x_j = 0$ for $j = s + 1, \dots, n$). Det fraktionelle knapsack problem kan dermed løses i $O(n \log n)$ tid, hvor den tungeste beregning er sorteringen (7).

Løsningsværdien til det fraktionelle knapsack problem betegnes z^{LP}

$$z^{LP} = \sum_{j=1}^{s-1} p_j + p_s \frac{c - \sum_{j=1}^{s-1} w_j}{w_s}$$

Opgave 1 Find z^{LP} for instansen fra eksempel 1. ■

Grådig løsning

Sorteringen (7) kan bruges til at finde en lovlig (men ikke nødvendigvis optimal) løsning til knapsack problemet (1) – (3). Vi gennemløber genstandene i rækkefølgen $1, 2, 3, \dots, n$ og medtager genstand j (dvs. sætter $x_j = 1$) hvis den kan tilføjes rygsækken uden at overskride kapaciteten c . Den tilsvarende løsningsværdi vil blive betegnet z^G .

Opgave 2 Find z^G for instansen fra eksempel 1. ■

Duale problem

Opgave 3 Opskriv (i generel form) det duale problem af (4)–(6). Lad y' være den duale variabel svarende til begrænsning (5) og y_j for $j = 1, \dots, n$ være de duale variable svarende til begrænsningerne (6). ■

Opgave 4 Angiv værdierne af y' og y_1, \dots, y_n for instansen fra eksempel 1. ■

Opgave 5 Vis at der generelt gælder at den duale værdi svarende til begrænsning (5) er givet ved $y' = p_s/w_s$. ■

Opgave 6 Bestem i generel form de duale værdier y_i svarende til begrænsningerne (6). ■

LP-følsomhedsanalyse

Ofte kender man ikke profitter p_j og vægte w_j af genstandene med fuldstændig nøjagtighed. Det kan derfor være relevant at bestemme hvor meget profitten eller vægten af en genstand j kan ændres uden at den optimale LP-løsning ændrer sig. Dette kaldes *følsomhedsanalyse* (sensitivity analysis), jf. Taha [5] afsnit 4.5.

Opgave 7 Angiv for hver genstand $j = 1, \dots, 7$ i eksempel 1 den mindste og største værdi af profit p_j således at LP-løsningen er uændret. Det er tilstrækkeligt kun at redegøre for detaljerne i udregningen for den første genstand, og herefter at opstille alle resultater i tabel-form. ■

Reduced-cost fixing

Opgave 8 Vis at hvis vi øger/mindsker kapaciteten c af rygsækken med Δ , da er en øvre grænseværdi U for det fraktionelle knapsack problem givet ved

$$U(c + \Delta) = z^{LP} + y'\Delta$$

■

Antag at vi har fundet en LP-løsning til det fraktionelle knapsack problem (4)–(6). Hvis der for en genstand j gælder at

$$\left| p_j - w_j \frac{p_s}{w_s} \right| \geq z^{LP} - z^G \quad (8)$$

så vil x_j i en IP-løsning (knapsack problemet) have samme værdi som x_j i den tilhørende LP-løsning (fraktionelle knapsack problem).

Opgave 9 Vis at ovenstående påstand er korrekt. ■

Metoden kaldes *reduced cost fixing* og den kan anvende til at formindske en instans af knapsack problemet idet de variable hvis IP-løsningsværdi er kendt kan fjernes fra problemet ([6] section 7.8 exercise 7).

Opgave 10 Udregn værdien af $|p_j - w_j p_s/w_s|$ for $j = 1, \dots, n$ i instansen fra eksempel 1, og anvend reduced cost fixing til at bestemme de optimale IP-løsningsværdier for så mange variable som muligt. ■

Opgave 11 Fjern de reducerede variable fra knapsack problem instansen i eksempel 1 og opskriv det tilbageværende problem. Husk at reducere kapaciteten c af rygsækken med en passende værdi. ■

Dynamisk programmering

Bellman viste i 1957 at knapsack problemet kan løses med dynamisk programmering [1] (se også Taha [5] afsnit 10.3.1).

Lad $f_i(d)$ være en optimal løsning til (1) – (3), hvor kapaciteten er begrænset til $c = d$ og hvor kun de første i genstande tages i betragtning. Med andre ord har vi

$$f_i(d) = \max \left\{ \sum_{j=1}^i p_j x_j : \sum_{j=1}^i w_j x_j \leq d, x_j \in \{0, 1\} \right\} \quad (9)$$

for $i = 0, \dots, n$ og $d = 0, \dots, c$. For $i = 0$ kan man naturligvis kun opnå profit-summen 0 uanset værdien af d så der gælder

$$f_0(d) = 0 \text{ for } d = 0, \dots, c \quad (10)$$

Hvis vi kender de optimale løsninger for f_{i-1} kan vi finde de optimale løsninger for f_i ved brug af følgende rekursion

$$f_i(d) = \begin{cases} f_{i-1}(d) & \text{hvis } d < w_i \\ \max \{ f_{i-1}(d), f_{i-1}(d - w_i) + p_i \} & \text{hvis } d \geq w_i \end{cases} \quad (11)$$

Den optimale løsningsværdi til (1) – (3) findes dermed som $z^* = f_n(c)$.

Eksempel 2 Tabellen $f_i(d)$ for eksempel 1 løst med dynamisk programmering bliver:

$d \setminus i$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	0	6	6	6	6	6	6	6
3	0	6	6	6	6	6	6	6
4	0	6	6	6	6	6	6	6
5	0	6	11	11	11	11	11	11
6	0	6	11	11	11	11	11	11
7	0	6	11	11	11	12	12	12
8	0	6	11	14	14	14	14	14
9	0	6	11	14	15	15	15	15

■

IP følsomhedsanalyse

Modsat følsomhedsanalyse af LP-problemer er det ikke nemt at lave følsomhedsanalyse af IP-problemer. For knapsack problemet kan det dog lade sig gøre relativt effektivt ved brug af dynamisk programmering.

Betragt knapsack problemet fra eksempel 1 og den tilhørende dynamisk programmerings tabel fra eksempel 2.

Opgave 12 Antag at vi vil finde den mindste og største værdi af p_i således at IP-løsningen er uændret. Vi betragter den sidste genstand $n = 7$ i problemet. Betragt de sidste to kolonner i dynamisk programmerings tabellen, og angiv på basis af disse den mindste og største værdi af p_7 så IP-løsningen er uændret. ■

Dynamisk programmerings rekursionen (11) fungerer uanset rækkefølgen af genstandene. Vi kan derfor på skift lade hver genstand i være den sidste genstand i instansen.

Opgave 13 Benyt denne ide til for samtlige $j = 1, \dots, 7$ at finde mindste og største værdi af p_j således at IP-løsningen er uændret. ■

Noter

Det kan være hensigtsmæssigt at udvikle nogle små programmer til f.eks. at løse knapsack problemet med dynamisk programmering. Dermed spares mange beregninger, og man undgår at lave regnefejl. Udskrifter af sådanne programmer må godt vedlægges besvarelsen, men besvarelsen skal kunne læses uden at kigge i programmet.

Litteratur

- [1] R.E. Bellman (1957), *Dynamic programming*, Princeton University Press, Princeton, NJ.
- [2] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein (2003), "Introduction to Algorithms, second edition", MIT-Press, Cambridge, England.
- [3] H. Kellerer, U. Pferschy, and D. Pisinger (2004), "Knapsack Problems", Springer, Berlin, Germany.
- [4] S. Martello and P. Toth (1990), "Knapsack Problems: Algorithms and Computer Implementations", Wiley, Chichester, England.
- [5] H.A. Taha (2003) "Operations Research, an introduction", Prentice Hall, London, England.
- [6] L.A. Wolsey (1998) "Integer Programming" Wiley, Chichester, England.