

## 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  $\Delta$ , 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.

## Vejledende løsninger

**Svar 1** Genstandene og deres profi t-vægt forhold er:

$j$	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
$p_j/w_j$	3.0000	1.6667	1.3333	1.2857	1.2000	0.7778	0.7500

dvs. genstandene er allerede sorteret efter  $p_j/w_j$ . Den grådige algoritme finder  $s = 3$  og dermed den primale løsning

$x_1$	1.000000
$x_2$	1.000000
$x_3$	0.666667

den tilhørende løsningsværdi er

$$z^{LP} = 6 + 5 + (9 - 5) \frac{8}{6} = 11 + 5.3333 = 16.3333$$

■

**Svar 2** Den grådige løsning medtager genstande 1, 2, 7. Den tilhørende løsningsværdi er  $z^G = p_1 + p_2 + p_7 = 6 + 5 + 3 = 14$  ■

**Svar 3** Primale problem for eksempel 1:

```

maximize
  6 x1 + 5 x2 + 8 x3 + 9 x4 + 6 x5 + 7 x6 + 3 x7
subject to
  2 x1 + 3 x2 + 6 x3 + 7 x4 + 5 x5 + 9 x6 + 4 x7 <= 9      y'
    x1 <= 1          y1
      x2 <= 1          y2
        x3 <= 1          y3
          x4 <= 1          y4
            x5 <= 1          y5
              x6 <= 1          y6
                x7 <= 1          y7
  
```

Duale problem for eksempel 1:

```

minimize
  9 y' + 1 y1 + 1 y2 + 1 y3 + 1 y4 + 1 y5 + 1 y6 + 1 y7
subject to
  2 y' + y1 >= 6
  3 y' + y2 >= 5
  6 y' + y3 >= 8
  7 y' + y4 >= 9
  5 y' + y5 >= 6
  9 y' + y6 >= 7
  4 y' + y7 >= 3
  
```

Generelt gælder primale problem:

$$\text{maximize } \sum_{j=1}^n p_j x_j \quad (12)$$

$$\text{subject to } \sum_{j=1}^n w_j x_j \leq c, \quad (13)$$

$$0 \leq x_j \leq 1, \quad j = 1, \dots, n. \quad (14)$$

duale problem:

$$\text{minimize } cy' + \sum_{j=1}^n y_j \quad (15)$$

$$\text{subject to } w_j y' + y_j \geq p_j, \quad j = 1, \dots, n \quad (16)$$

$$y_j \geq 0, \quad j = 1, \dots, n. \quad (17)$$

$$y' \geq 0 \quad (18)$$

■

**Svar 4** Duale løsning er

$y'$	1.333333
$y_1$	3.333333
$y_2$	1.000000

■

**Svar 5** Den duale variabel  $y'$  angiver hvor meget  $z^{LP}$  kan øges hvis vi forøger  $c$  med 1. Dette vil netop svare til  $p_s/w_s$  ■

**Svar 6** Den duale variabel  $y_j$  angiver hvor meget  $z^{LP}$  kan øges hvis vi forøger højresiden i (6).

Hvis  $x_j < 1$  i LP-løsningen, så vil  $z^{LP}$  ikke blive øget uanset hvor meget vi øger højresiden i (6), og dermed  $y_j = 0$ .

Hvis  $x_j = 1$  i LP-løsningen, vil  $z^{LP}$  blive øget med  $p_j - w_j p_s/w_s$  hver gang vi øger højresiden i (6). Intuitivt set, får vi profitten af genstand  $j$  men vi skal fjerne  $w_j$  vægtenheder af genstand  $s$  fra rygsækken. Altså er  $y_j = p_j - w_j p_s/w_s$ . ■

**Svar 7** Følsomheds intervallerne er:

$j$	$p_j^{min}$	$p_j$	$p_j^{max}$
1	2.6667	6.0000	$+\infty$
2	4.0000	5.0000	$+\infty$
3	7.7143	8.0000	10.0000
4	$-\infty$	9.0000	9.3333
5	$-\infty$	6.0000	6.6667
6	$-\infty$	7.0000	12.0000
7	$-\infty$	3.0000	5.3333

■

**Svar 8** Da den duale værdi  $y'$  angiver hvor meget  $z^{LP}$  øges/mindskes ved forøgelse/formindskelse af kapaciteten  $c$  får vi direkte at

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

■

**Svar 9** Antag at  $j < s$  dvs.  $p_j/w_j \geq p_s/w_s$ . Da kan numerisktegnet hæves og vi får kriteriet

$$p_j - w_j \frac{p_s}{w_s} \geq z^{LP} - z^G$$

hvilket kan omskrives til

$$z^G \geq U(c + w_j) - p_j$$

her angiver højresiden en øvre grænseværdi for det fraktionelle knapsack problem hvis vi sætter  $x_j = 0$ . Med andre ord har vi en sædvanlig upper bound test for forgreningen  $x_j = 0$ . Hvis knuden kan forkastes, er det tilladt at sætte  $x_j = 1$ .

Antag at  $j > s$  dvs.  $p_j/w_j \leq p_s/w_s$ . Da hæves numerisktegnet og vi får kriteriet

$$-p_j + w_j \frac{p_s}{w_s} \geq z^{LP} - z^G$$

hvilket kan omskrives til

$$z^G \geq U(c - w_j) + p_j$$

her angiver højresiden en øvre grænseværdi for det fraktionelle knapsack problem hvis vi sætter  $x_j = 1$ . Med andre ord har vi en sædvanlig upper bound test for forgreningen  $x_j = 1$ . Hvis knuden kan forkastes, er det tilladt at sætte  $x_j = 0$ . ■

**Svar 10** Vi udregner værdierne

$j$	$ p_j - w_j p_s / w_s $
1	$ 3\frac{1}{3} $
2	$ 1 $
3	$ 0 $
4	$ -\frac{1}{3} $
5	$ -\frac{2}{3} $
6	$  -5 $
7	$  -2\frac{1}{3} $

Vi har endvidere  $z^{LP} - z^G = 16\frac{1}{3} - 14 = 2\frac{1}{3}$ . Dermed ses det at  $x_1 = 1$  og  $x_6 = 0$  i en optimal IP-løsning. Endvidere vil  $x_7 = 0$  i enhver forbedret løsning, men det er ikke det vi blev spurgt om. ■

**Svar 11** Den reducerede instans er

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

med kapacitet  $c = 9 - 2 = 7$  ■

**Svar 12** Lad  $i = 7$  være det sidste element. Da  $x_i = 0$  i den optimale løsning, ønsker vi i recursion (11) at det første led bliver valgt i max-operatoren:

$$f_i(d) = \max \left\{ f_{i-1}(d), f_{i-1}(d - w_i) + p_i \right\} \quad (19)$$

Dette sker hvis

$$f_{i-1}(d) \leq f_{i-1}(d - w_i) + p_i$$

Da  $i = 7$ ,  $w_i = 4$  og  $d = c = 9$  skal der gælde at

$$f_6(9) \leq f_5(9 - 4) + p_i$$

hvilket er ensbetydende med  $p_i \geq f_6(9) - f_5(9 - 4) = 15 - 11 = 4$  ■

**Svar 13** Samme metode bruges for alle elementer. ■

```

maximize
  6 x1 + 5 x2 + 8 x3 + 9 x4 + 6 x5 + 7 x6 + 3 x7
subject to
  2 x1 + 3 x2 + 6 x3 + 7 x4 + 5 x5 + 9 x6 + 4 x7 <= 9
    x1 <= 1
      x2 <= 1
        x3 <= 1
          x4 <= 1
            x5 <= 1
              x6 <= 1
                x7 <= 1
end

```

```

CPLEX> opt
Tried aggregator 1 time.
LP Presolve eliminated 7 rows and 1 columns.
Reduced LP has 1 rows, 6 columns, and 6 nonzeros.
Presolve time = 0.00 sec.

```

```

Iteration log . . .
Iteration: 1 Dual infeasibility = 0.000000
Iteration: 2 Dual objective = 16.333333

```

```

Dual simplex - Optimal: Objective = 1.6333333333e+01
Solution time = 0.00 sec. Iterations = 2 (1)

```

```

CPLEX> dis sol var -
Variable Name      Solution Value
x1                 1.000000
x2                 1.000000
x3                 0.666667
All other variables in the range 1-7 are zero.

```

```

CPLEX> dis sol dua -
Constraint Name    Dual Price
c1                 1.333333
c2                 3.333333
c3                 1.000000
All other dual prices in the range 1-8 are zero.

```

```

CPLEX> dis sen obj 1-7
Variable Name      Reduced Cost      Down      Current      Up
x1                 zero           2.6667     6.0000     +infinity
x2                 zero           4.0000     5.0000     +infinity
x3                 zero           7.7143     8.0000     10.0000

```

x4	-0.3333	-infinity	9.0000	9.3333
x5	-0.6667	-infinity	6.0000	6.6667
x6	-5.0000	-infinity	7.0000	12.0000
x7	-2.3333	-infinity	3.0000	5.3333

CPLEX> dis sen rhs 1-8

RHS Sensitivity Ranges				
Constraint Name	Dual Price	Down	Current	Up
c1	1.3333	5.0000	9.0000	11.0000
c2	3.3333	zero	1.0000	3.0000
c3	1.0000	0.3333	1.0000	2.3333
c4	zero	0.6667	1.0000	+infinity
c5	zero	zero	1.0000	+infinity
c6	zero	zero	1.0000	+infinity
c7	zero	zero	1.0000	+infinity
c8	zero	zero	1.0000	+infinity

CPLEX> dis sen lb 1-7

Lower Bound Sensitivity Ranges				
Variable Name	Reduced Cost	Down	Current	Up
x1	zero	-infinity	zero	1.0000
x2	zero	-infinity	zero	1.0000
x3	zero	-infinity	zero	0.6667
x4	-0.3333	-0.2857	zero	0.5714
x5	-0.6667	-0.4000	zero	0.8000
x6	-5.0000	-0.2222	zero	0.4444
x7	-2.3333	-0.5000	zero	1.0000

CPLEX> dis sen ub 1-7

Upper Bound Sensitivity Ranges				
Variable Name	Reduced Cost	Down	Current	Up
x1	zero	1.0000	+infinity	+infinity
x2	zero	1.0000	+infinity	+infinity
x3	zero	0.6667	+infinity	+infinity
x4	-0.3333	zero	+infinity	+infinity
x5	-0.6667	zero	+infinity	+infinity
x6	-5.0000	zero	+infinity	+infinity
x7	-2.3333	zero	+infinity	+infinity