Aalto University
School of Science
and Technology

Trimmitysongelman LP-relaksaation ratkaiseminen sarakkeita generoivalla algoritmilla ja brute-force-menetelmällä

Vesa Husgafvel
19.11.2012

Ohjaaja: DI Mirko Ruokokoski
Valvoja: Prof. Harri Ehtamo

Työn tavoite

- Ratkaistaan trimmitysongelman LP-relaksaatio sarakkeita generoivalla algoritmilla ja brute-force-menetelmällä.
- Brute-force-menetelmä toteutetaan MATLABilla ja CPLEXillä, sarakkeita generoiva algoritmi pelkällä CPLEXillä.
- Vertaillaan numeerisesti menetelmien laskenta-aikoja erilaisissa instansseissa.

Tausta

- Trimmityksellä tarkoitetaan (tässä yhteydessä) kustannustehokasta tuotannonsuunnittelua, jossa tyydytetään kysyntä minimoimalla käytetyn raaka-aineen määrä.
- Esimerkki: miten sahata vaneria, niin että syntyvien hukkapalojen määrä olisi mahdollisimman pieni.
- Ongelma voidaan mallintaa lineaarisena kokonaislukutehtävänä.
- Vanerinsahaus on esimerkki 2D-tehtävästä.
- Tässä työssä tarkasteltava tehtävä on yksiulotteinen.

Trimmitysongelma (1/3)

- Paperitehdas valmistaa emorullaa, jonka pituus on W.
- Asiakasyritys i, $i = 1, \ldots, m$, haluaa ostaa b_i kappaletta rullia, joiden pituus on $w_i \leq W$.
- Pienempiä rullia saadaan viipaloimalla emorullia erilaisilla leikkausmuoteilla.
- Kutakin leikkausmuottia $j = 1, \ldots, n$, vastaa sarakevektori $\mathbf{A}_j$, siten että sarakkeen alkio $a_{ij} \in \mathbb{Z}_+$ ilmaisee, kuinka monta kertaa rulla w_i esiintyy kyseisessä leikkausmuotissa.
- Luonnollisesti leikkausmuotissa olevien pienten rullien yhteispituus ei saa ylittää emorullan pituutta, mistä saadaan rajoitusehdot

$$\sum_{i=1}^{m} a_{ij} w_i \leq W, \qquad\qquad j = 1, 2, \ldots, n.$$

Trimmitysongelma (2/3)

- Kun paperitehdas haluaa minimoida valmistettujen emorullien lukumäärän tyydyttäen samalla asiakasyritysten kysynnät, saadaan tehtävän formulaatioksi

$$\begin{aligned} Z_{IP} \quad = \quad \min \quad & \sum_{j=1}^{n} x_j \\ \text{s.e.} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq b_i, \qquad i = 1, 2, \ldots, m, \\ & x_j \in \mathbb{Z}_+, \qquad\qquad j = 1, 2, \ldots, n, \end{aligned}$$

missä x_j on leikkausmuotin j mukaan leikattujen emorullien lukumäärä.

Trimmitysongelma (3/3)

- Yksinkertaisuuden vuoksi siirrytään tarkastelemaan tehtävän LP-relaksaatiota

$$
\begin{aligned}
Z_{LP} \quad = \quad \min \quad & \sum_{j=1}^{n} x_j \\
\text{s.e.} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq b_i, \qquad i = 1, 2, \ldots, m, \\
& x_j \geq 0, \qquad\qquad\quad j = 1, 2, \ldots, n.
\end{aligned}
$$

- Ongelmana kuitenkin on, että rajoitusmatriisi **A** on tuntematon, eikä tehtävää siksi voida ratkaista suoraan esimerkiksi Simplex-algoritmilla.
- Esitetään seuraavaksi kaksi ratkaisumenetelmää:

Brute-force-menetelmä

- Muodostetaan rajoitusmatriisi **A** kokonaisuudessaan etsimällä kaikki mahdolliset sarakevektorit $\mathbf{A}_j = [a_{1j}, a_{2j}, \ldots, a_{mj}]' \in \mathbb{Z}_+^m$, jotka toteuttavat ehdon

$$\sum_{i=1}^{m} a_{ij} w_i \leq W, \qquad j = 1, 2, \ldots, n.$$

- Geometrisesti tilanne vastaa annetun monitahokkaan sisältämien kokonaislukupisteiden määritystä.
- Menetelmässä joudutaan käymään läpi hyvin suuri määrä pisteitä, mistä johtuen käytetään brute-force-nimeä.
- Kun rajoitusmatriisi on selvitetty, on LP-tehtävä ratkaistavissa Simplexillä.

Sarakkeita generoiva algoritmi

- Tarkoitettu lineaarisen ohjelmoinnin tehtäville.
- Algoritmi alustetaan poimimalla $k \leq n$ kappaletta rajoitusmatriisin $\mathbf{A}$ sarakkeista ja muodostetaan näistä uusi rajoitusmatriisi $\tilde{\mathbf{A}}_k$.
- Ratkaistaan LP-tehtävä rajoitusmatriisin $\tilde{\mathbf{A}}_k$ suhteen.
- Tarkastelemalla ratkaisun $\tilde{\mathbf{x}}_k$ redusoituja kustannuksia, saadaan muodostettua uusi sarake, joka lisätään rajoitusmatriisiin $\tilde{\mathbf{A}}_k$ $(k \rightarrow k+1)$.
- Ratkaistaan LP-tehtävä rajoitusmatriisin $\tilde{\mathbf{A}}_{k+1}$ suhteen.
- Lopulta löydetään alkuperäisen LP-tehtävän optimiratkaisu jollakin arvolla $k = q$, $q \leq n$.
- Menetelmä soveltuu parhaimmin tehtäville, joissa suurin osa päätösmuuttujista saa arvon 0 optimiratkaisussa, jolloin $q << n$.

Instanssien generointi

- Rullien lukumäärän m sekä emorullan koon W vaikutusta LP-tehtävän laskenta-aikaan tutkittiin ratkaisemalla erilaisia instansseja.
- Tarkasteltavissa instansseissa muuttujien m ja W arvojoukoiksi valittiin $\{10, 20, \ldots, 100\}$ ja $\{500, 600, 700, 800, 900\}$.
- Rullakokojen w_i ja kysyntöjen b_i, $i = 1, \ldots, m$, arvoina käytettiin MATLABin satunnaislukugeneraattorin arpomia kokonaislukuja. Luvut tuotettiin väleiltä $[l_w, u_w] = [100, 500]$, ja $[l_b, u_b] = [0, 100]$.

Laskenta-ajat

- Sarakkeita generoivan algoritmin (SG) ja brute-force-menetelmän (BF) laskenta-ajat tarkastelluissa instansseissa:

$m \backslash W$	500		600		700		800		900	
	BF	SG	BF	SG	BF	SG	BF	SG	BF	SG
10	0,14s	2,21s	0,13s	2,31s	0,12s	2,84s	0,13s	2,36s	0,14s	2,75s
20	0,14s	3,04s	0,17s	4,11s	0,21s	6,04s	0,30s	7,20s	0,55s	10,8s
30	0,15s	2,21s	0,20s	4,45s	0,27s	5,08s	0,46s	10,5s	1,19s	18,4s
40	0,35s	5,55s	0,90s	8,25s	5,83s	24,1s	39,8s	1min 5s	4min 25s	4min 16s
50	0,82s	6,68s	6,82s	13,9s	1min 6s	30,0s	9min 44s	2min 34s	1h 23min	6min 1s
60	1,67s	8,42s	9,76s	49,3s	1min 9s	2min 24s	10min 49s	10min 16s	1h 39min	24min 43s
70	2,36s	26,1s	25,1s	1min 57s	4min 47s	9min 26s	1h 22min	22min 29s	#	56min 20s
80	7,00s	28,1s	1min 25s	1min 36s	17min 41s	8min 4s	3h 57min	31min 33s	#	2h 16min
90	20,8s	1min 46s	5min 44s	9min 4s	1h 42min	25min 0s	#	1h 40min	#	2h 12min
100	7,37s	2min 40s	1min 41s	11min 13s	26min 4s	27min 28s	#	1h 42min	#	2h 9min

- Merkintä # tarkoittaa, että instanssia ei kyetty ratkaisemaan 5 tunnissa.
- Punaisella värillä merkitty tapauksia, joissa SG oli nopeampi.
- Sinisellä värillä merkitty tapauksia, joissa BF:n laskenta-aika oli 45-100 % SG:n laskenta-ajasta.

Generoitujen leikkausmuottien lukumäärä

- Sarakkeita generoivan algoritmin (SG) ja brute-force-menetelmän (BF) käyttämien leikkausmuottien lukumäärä tarkastelluissa instansseissa:

$m \backslash W$	500			600			700			800			900		
	BF	SG	N	BF	SG	N	BF	SG	N	BF	SG	N	BF	SG	N
10	17	14	3072	25	16	3072	45	17	4608	80	22	23328	125	23	34992
20	135	45	$6*10^{7}$	317	47	$5*10^{8}$	728	58	$4*10^{9}$	1576	63	$1*10^{10}$	3279	73	$8*10^{10}$
30	105	44	$3*10^{10}$	275	60	$6*10^{11}$	656	70	$9*10^{12}$	1500	82	$2*10^{14}$	3335	96	$5*10^{15}$
40	736	78	$2*10^{16}$	2231	108	$4*10^{17}$	6877	120	$2*10^{19}$	18842	146	$5*10^{21}$	50862	172	$9*10^{22}$
50	1766	96	$2*10^{20}$	6577	134	$4*10^{22}$	22043	143	$2*10^{24}$	68409	187	$2*10^{26}$	203616	202	$1*10^{29}$
60	1547	130	$2*10^{23}$	5754	158	$9*10^{25}$	19887	202	$5*10^{28}$	64585	246	$5*10^{31}$	198924	280	$6*10^{33}$
70	2743	156	$8*10^{27}$	10761	203	$5*10^{30}$	39690	262	$2*10^{33}$	138821	290	$9*10^{36}$	#	317	$2*10^{40}$
80	4253	195	$2*10^{31}$	18184	204	$5*10^{34}$	72032	280	$1*10^{38}$	267700	330	$6*10^{41}$	#	373	$3*10^{45}$
90	8286	234	$4*10^{36}$	38336	298	$2*10^{40}$	166152	341	$7*10^{43}$	#	393	$1*10^{48}$	#	379	$9*10^{52}$
100	4267	244	$8*10^{37}$	19093	308	$1*10^{42}$	79477	403	$5*10^{46}$	#	351	$5*10^{50}$	#	389	$7*10^{54}$

- Merkintä # tarkoittaa, että instanssia ei kyetty ratkaisemaan 5 tunnissa.
- *N* on teoreettinen yläraja leikkausmuottien lukumäärälle, joka on saatu arvioimalla käypää aluetta *m*-ulotteisella suorakulmiolla.

Johtopäätökset

- Rullien lukumäärällä m sekä emorullan koolla W on voimakas vaikutus laskenta-aikaan, mutta myös rullien koolla w_i ja kysynnällä b_i, $i = 1, \ldots, m$.
- Brute-force-menetelmä on tehokkaampi pienen kokoluokan tehtävissä - sarakkeita generoiva algoritmi taas parempi suuren kokoluokan tehtävissä.
- Generoitujen leikkausmuottien (sarakkeiden) lukumäärä korreloi suoraan laskenta-ajan kanssa.
- Leikkausmuottien teoreettinen yläraja N ei tarjoa mitään käytännön hyötyä.