日本オペレーションズ・リサーチ学会
2004年春季研究発表会

0-1複数ナップサック問題に対する列生成法を用いた上界値

02702090　防衛大学校　*保田　亮　　YASUDA Ryo
01107880　防衛大学校　　片岡　靖詞　KATAOKA Seiji

1　はじめに

Danzig-Wolf の分解原理を整数計画問題に適用した分枝-費用法 (Branch-and-Price method) の研究が 1995 年以降に目だっている [1,4,5]. この手法は，一般化割当問題や乗員スケジューリング問題，固定費のついた問題などにおいて成功している.

本研究では，0-1 複数ナップサック問題 (Multiple Knapsack Problem: MKP) に分解原理を適用し，上界値として優れた値を出すことを検証する. MKP は一般化割当問題の特殊例と見なすこともでき，その特殊性を活用すれば，効果的な解法も期待できる.

n をアイテムの数（N をその集合），m をナップサックの数（M をその集合），p_j を j 番目のアイテムの価値（$\boldsymbol{p}$ を行ベクトル），w_j を j 番目のアイテムの重量（$\boldsymbol{w}$ を行ベクトル），b^i を i 番目のナップサックの重量制限（$\boldsymbol{b}$ を列ベクトル），x_j^i をナップサック i にアイテム j を入れるとき 1，それ以外で 0 をとる変数とする. このとき MKP は次のように定式化できる.

$$MKP \quad \max. \quad \sum_{i=1}^{m}\sum_{j=1}^{n} p_j x_j^i \tag{1.1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} w_j x_j^i \le b^i, \quad i \in M \tag{1.2}$$

$$\sum_{i=1}^{m} x_j^i \le 1, \quad j \in N \tag{1.3}$$

$$x_j^i \in \{0,1\} \tag{1.4}$$

2　上界値

組合せ最適化問題では，最適解法の提案や最適性の保証をするためには（最大化問題であれば）上界値が重要な役割を果たす. 通常，上界値は元の問題の緩和問題を解くことにより得られる. よく用いられる緩和問題としては，(1) 線形緩和:C，(2) 代理制約緩和:S，(3) ラグランジュ緩和:L が挙げられる (アルファベットは各緩和法を示すときの略号. 最適値には z を用いる).

2.1　線形緩和 $C(MKP)$

線形緩和は最も原始的な緩和法であり，0-1 制約 (1.4) を $0 \le x_j^i \le 1$ に置き換えたものである.

$C(MKP)$ は，ナップサックの容量を $\sum_{i=1}^{m} b^i$ とした通常のナップサック問題 (KP) の線形緩和と目的関数値が一致することが簡単に確かめられる.

$$z(C(MKP)) = z(C(KP)) \tag{2.1}$$

計算はとても簡単だが，上界値としては弱く，線形緩和だけで効果的な最適解法の開発は期待できない.

2.2　代理制約緩和 $S(MKP,\pi)$

代理制約緩和は，ナップサック制約 (1.2) の各式に正の乗数 π^i を掛けて，線形結合したものである.

$$S(MKP,\pi) \quad \max. \quad \sum_{i=1}^{m}\sum_{j=1}^{n} p_j x_j^i$$

$$\text{s.t.} \quad \sum_{i=1}^{m}\pi^i \sum_{j=1}^{n} w_j x_j^i \le \sum_{i=1}^{m} \pi^i b^i$$

$$(1.3), (1.4)$$

最も小さな値をとる上界値は，π^i をすべて同じ値 $\pi^i = k(>0)$ にすればよく ([3], Thm6.1, p.159)，したがって $S(MKP,k)$ も容量 $\sum_{i=1}^{m} b^i$ としたナップサック問題と等価になる.

$$z(S(MKP,k)) \le z(C(MKP)) \tag{2.2}$$

代理制約緩和は，線形緩和よりもよい上界値をとるが，その値を得るためには 0-1 型ナップサック問題（1 度でよい）を解かなければならない. ただ，最適に解けたとしても，どのアイテムがどのナップサックに入るか認識不可能で，この緩和を用いた分枝限定法の列挙木は大きくなりがちである.

2.3　ラグランジュ緩和 $L(MKP,\lambda)$

制約式 (1.3) に非負の乗数 $(\lambda_1,\ldots,\lambda_n)$ を掛けて目的関数に組み込んだものをラグランジュ緩和という.

$$L(MKP,\lambda) \quad \max. \quad \sum_{j=1}^{n} p_j x_j^i - \sum_{j=1}^{n}\lambda_j\left(\sum_{i=1}^{m} x_j^i - 1\right)$$

$$\text{s.t.} \quad (1.2), (1.4)$$

これは，ナップサック問題を独立に m 個考えることと等価である. もっとも効率の良い（$z(L(MKP,\lambda))$ の値を最も小さくする）ラグランジュ乗数 λ を決定するためには，劣勾配法などを用いる必要があるが，ある程度精度の良いものであれば，線形計画の相補性より，(2.3) ような設定方法も有効である. ただし，s は $C(S(KP))$ で分数値をとるアイテムの添字である.

$$\bar{\lambda}_j = \begin{cases} p_j - w_j(p_s/w_s) & j < s \\ 0 & j \ge s \end{cases} \tag{2.3}$$

この $\bar{\lambda}$ を用いたときの線形緩和においては，次の等式が成り立つ.

$$z(C(L(MKP,\bar{\lambda}))) = z(C(S(MKP))) = z(C(MKP)) \tag{2.4}$$

$L(MKP,\lambda)$ は，$C(MKP)$ よりもよい上界値を出しうるが，0-1 ナップサック問題を何度も解かねばならな

μ_0	μ_1^1	$\cdots$	μ_k^1	$\cdots$	$\mu_{K^1}^1$	μ_{n+1}^1	μ_1^2	$\cdots$	μ_k^2	$\cdots$	$\mu_{K^2}^2$	μ_{n+2}^2	$\cdots\mu_j^s\cdots$		rhs
1	$-\boldsymbol{p}\boldsymbol{y}_1^1$	$\cdots$	$-\boldsymbol{p}\boldsymbol{y}_k^1$	$\cdots$	$-\boldsymbol{p}\boldsymbol{y}_{K^1}^1$	0	$-\boldsymbol{p}\boldsymbol{y}_1^2$	$\cdots$	$-\boldsymbol{p}\boldsymbol{y}_k^2$	$\cdots$	$-\boldsymbol{p}\boldsymbol{y}_{K^2}^2$	0	$\cdots 0\cdots$	=	0
0	$\boldsymbol{y}_1^1$	$\cdots$	$\boldsymbol{y}_k^1$	$\cdots$	$\boldsymbol{y}_{K^1}^1$	**0**	$\boldsymbol{y}_1^2$	$\cdots$	$\boldsymbol{y}_k^2$	$\cdots$	$\boldsymbol{y}_{K^2}^2$	**0**	$\cdots\boldsymbol{e}_j\cdots$	=	**1**
0	1	$\cdots$	1	$\cdots$	1	1	0	$\cdots$	0	$\cdots$	0	0	$\cdots 0\cdots$	=	1
0	0	$\cdots$	0	$\cdots$	0	0	1	$\cdots$	1	$\cdots$	1	1	$\cdots 0\cdots$	=	1

図 1: $m=2$ のときの単体表（イメージ）

い．もし整数解が得られても各アイテムの使用制限が緩和されているので，実行可能解にはつながらない．仮に実行可能解が得られたとしても，不等式制約を緩和しているので，最適性は保証されない．

2.4 線形緩和と切除平面法 $P(MKP)$

整数計画法・組合せ最適化問題に対し，よい上界値を得る手法の代表的なアプローチとして切除平面法がある．MKP に切除平面法を適用した分枝-カット法の研究としては，Ferreira *et al.*[2] がある．

3 列生成による上界値 $G(MKP)$

$\boldsymbol{y}_k^i$ を i 番目のナップサック制約 (1.2) を満足する（つまり $\boldsymbol{p}\boldsymbol{y}_k^i \le b^i$) 0-1 列ベクトル ($k=1,\ldots,K^i$) とする．$K^i$ は一般的に非常に大きな数になる．このベクトル $\boldsymbol{y}_k^i$ の凸結合により，$\boldsymbol{x}^i = \sum_{k=1}^{K^i} \mu_k^i \boldsymbol{y}_k^i$ ($\sum_{k=1}^{K^i} \mu_k^i = 1, \mu_k^i \ge 0$) を MKP に代入すると次のように再定式化（マスター問題）でき，$G(MKP)$ と示すことにする．

$$G(MKP) \quad \max. \quad \sum_{i=1}^{m}\sum_{k=1}^{K^i}(\boldsymbol{p}\boldsymbol{y}_k^i)\mu_k^i \tag{3.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{m}\sum_{k=1}^{K^i}\boldsymbol{y}_k^i\mu_k^i \le 1 \tag{3.2}$$

$$\sum_{k=1}^{K^i}\mu_k^i = 1 \quad i\in M \tag{3.3}$$

$$\mu_k^i \in \{0,1\} \tag{3.4}$$

Danzig-Wolf の分解原理でみるブロック部分には，1 つのナップサック制約しかないので，この $G(MKP)$ はもとの MKP と比べて行の圧縮は行われず，返って変数の数だけ膨大に増やしてしまうことになるので，改訂単体法を行うにしても，計算上のメリットは少ない．

初期実行可能基底解を得るのは簡単である．各ナップサックに何も入れないという解があり，それを初期基底として活用する (m 本)．また，(3.2) が不等式制約なので，それに対するスラック変数に対応する列が n 本ある．初期実行可能基底が理解しやすいように，$m=2$ のときの単体表を図 1 に示す．

改訂単体法を行うための逆行列の目的行が $(1,\beta_1,\ldots,\beta_j,\ldots,\beta_n,\alpha^1,\ldots,\alpha^i,\ldots,\alpha^m)$ のような行ベクトルになっているとすれば，次のようなナップサック問題を解き，$z^i>0$ になるような $\boldsymbol{y}^i$ を求めればよい．

$$CG^i \quad z^i = \max. \quad (\boldsymbol{p}-\boldsymbol{\beta})\boldsymbol{y}^i - \alpha^i \tag{3.5}$$

$$\text{s.t.} \quad \boldsymbol{w}\boldsymbol{y}^i \le b^i \tag{3.6}$$

$$\boldsymbol{y}\in\{0,1\}^n \tag{3.7}$$

これは，制約式の右辺が異なる m 個のナップサック問題が集まっているだけであり，$z^i>0$ となる解であれば，近似解法を適用してもよく，最終的に最適性の保証を得る場合であっても，動的計画法であれば，一番大きな b^i のところだけを解けばよい．

計算機実験では，乱数で生成した問題で適用し（線形緩和 $z(C(MKP))$ の値を 1 として）4 つの上界値を比較したところ，表 1 のような結果を得た．ただし，$G(MKP)$ は (3.4) 式を線形緩和して解いている．問題の規模は $n=20$, $m=2$ とし，ナップサックの容量は，$b^1 = 0.3\sum_{j=1}^{n} w_j$, $b^2 = 0.2\sum_{j=1}^{n} w_j$ とした．表の各数値は 100 回実験を行った平均値である．アイテムの価値と重量について，(a) 強相関，(b) 弱相関とした．

表 1: 上界値の比較 ($z(C(MKP))=1.0$ とする)

(a) **強相関** $90\le p_j\le 110, 90\le w_j\le 110$

$z(C(MKP))$	$z(S(MKP))$	$z(L(MKP))$	$z(C(G(MKP)))$
1.0000	0.9844	0.9992	0.9836

(b) **弱相関** $50\le p_j\le 150, 50\le w_j\le 150$

$z(C(MKP))$	$z(S(MKP))$	$z(L(MKP))$	$z(C(G(MKP)))$
1.0000	0.9858	0.9999	0.9979

この結果から，強相関では列生成による上界値が，他の緩和による上界値とくらべて最も優れた値を出しているが，弱相関では代理制約の方がよい値を得た．

また，$S(MKP)$ や $L(MKP)$ では，得られた解に最適性の保証はないが，列生成による上界値は，整数解が得られた場合にはそれが最適解となる．この割合が，強相関の場合では 66%に到ったが，弱相関の場合には 4%であった．

参考文献

[1] C.Barnhart *et al.*: Branch-and-price — column generation for solving huge integer programs. *Ope. Res.*, **46**(1998)316-329.

[2] C.E.Ferreira *et al.*: Solving multipue knapsack problem by cutting planes. *SIAM J. Opt,*, **6**(1996)858-877.

[3] S.Martello and P.Toth: *Knapsack Problems* algorithms and computer inplementations, (John Wilay & Sons, 1989).

[4] F.Vanderbeck and L.A.Wolsey: An exact algorithm for IP column generation. *O.R. Letters*, **19**(1996)151-159.

[5] L.A.Wolsey, *Integer Programming*, (John Wilay & Sons, 1998).