日本オペレーションズ・リサーチ学会
2005年春季研究発表会

固定費付き複数ナップサック問題
── 分枝費用法によるアプローチ ──

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

n個のアイテムと，m個のナップサックを考える．各アイテムjには価値p_jと重量w_jが，また各ナップサックiには重量制限b^iと固定費f^iが与えられている．このとき**固定費付き複数ナップサック問題**（Fixed charge Multiple Knapsack Problem: FMKP）は(P)のように定式化できる．

$$(P)\quad \max \quad \sum_{i=1}^{m}\sum_{j=1}^{n} p_j x_j^i - \sum_{i=1}^{m} f^i y^i \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} w_j x_j^i \le b^i y^i \quad i=1,\ldots,m \tag{2}$$

$$\sum_{i=1}^{m} x_j^i \le 1 \qquad j=1,\ldots,n \tag{3}$$

$$x_j^i, y^i \in \{0,1\} \tag{4}$$

FMKPは複数ナップサック問題（MKP）の発展型であるが，固定費が設定されていることによりMKPで有効とされてきた上界値算法[2,3]の活用が困難になる．

一方，1995年頃以降，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)[1]が，一般化割当問題[4]や固定費付きの問題[5]などで成功を収めている．本研究ではFMKPに対し**列生成による上界値**を用い，分枝費用法による厳密解法を提案する．

2　列生成による上界値 $G(P)$

MKPの上界値を得る手法には線形緩和，ラグランジュ緩和，代理制約緩和などがあるが，固定費が付くと上界値の精度や計算量の問題から，これらの緩和法をFMKPへ適用することは難しい．

一方，列生成による上界値算法は，固定費を目的関数の係数に反映させることで，**使用するナップサックの組合せ要因を消去できる**利点がある．$\mathcal{X}_k^i$を(2)式を満足する0-1列ベクトル$(\boldsymbol{w}\mathcal{X}_k^i \le b^i,\ k=0,\ldots,K^i,\ i=1,\ldots,m : K^i$はベクトルの総数)とする．$(P)$の実行可能解$\boldsymbol{x}^i$の凸包は，このベクトル$\mathcal{X}_k^i$の凸結合で，$\boldsymbol{x}^i = \sum_{k=0}^{K^i} \mathcal{X}_k^i \mu_k^i$, $\sum_{k=0}^{K^i} \mu_k^i = 1, \mu_k^i \ge 0$のように表現でき，これを$(P)$に代入すると$G(P)$のように再定式化（**マスター問題**）できる（図1）．

$$G(P)\quad \max \quad \sum_{i=1}^{m}\sum_{k=1}^{K^i} (\boldsymbol{p}\mathcal{X}_k^i - f^i)\mu_k^i \tag{5}$$

$$\text{s.t.} \quad \sum_{i=1}^{m}\sum_{k=0}^{K^i} \mathcal{X}_k^i \mu_k^i \le \mathbb{1} \tag{6}$$

$$\sum_{k=0}^{K^i} \mu_k^i = 1 \qquad i=1,\ldots,m \tag{7}$$

$$\mu_k^i \ge 0 \tag{8}$$

$G(P)$は非常に多くの変数を持つことになるが，全ての$\mathcal{X}^i$が陽にわからなくても，適切な非基底変数の列を選ぶことができる．改訂単体法を行う逆行列の目的行が

$$(1, \alpha_1, \ldots, \alpha_j, \ldots, \alpha_n, \beta^1, \ldots, \beta^i, \ldots, \beta^m)$$

のような行ベクトルであるとすれば，次のナップサック問題（**列生成子問題**）を解き，$z^i > 0$（単体判定基準）になるような$\mathcal{X}^i$を求めればよい．

$$z^i = \max \quad (\boldsymbol{p}-\boldsymbol{\alpha})\mathcal{X}^i - (\beta^i + f^i)$$
$$\text{s.t.} \quad \boldsymbol{w}\mathcal{X}^i \le b^i \tag{9}$$
$$\mathcal{X}^i \in \{0,1\}^n$$

$G(P)$は上界値としては有効だが，退化のために単体法の無駄な繰返しが多い．そこで，初期解法の導入や$G(P)$の双対問題を考慮した上界値による見切り，ナップサックの巡回などの工夫を施し有効なものにしている．

3　分枝費用法による厳密解法

3.1　分枝変数と分枝ルール

分枝費用法とは，列生成法による上界値を用いた分枝限定法のことをいう．分枝するための変数には，(P)におけるx_j^iと，$G(P)$におけるμ_k^iとが考えられるが，(P)が0-1計画問題の場合には次の定理が成り立つ．

定理 0-1計画問題では，マスター問題のμ_k^iが分数解であれば，凸結合した元問題のx_j^iも分数解である．□

μ_k^iは膨大な数がある上に，ほとんどが0になる．またμ_k^iの固定は，列生成子問題で何番目かの最適解を出す難しい問題になってしまう．したがって，FMKPでは元問題におけるx_j^iを分枝変数とする方が賢明である．

μ_0^s	μ_0^1	μ_1^1	$\cdots$	μ_k^1	$\cdots$	$\mu_{K^1}^1$	μ_0^2	μ_1^2	$\cdots$	μ_k^2	$\cdots$	$\mu_{K^2}^2$	$\cdots \mu_j^s \cdots$		rhs
1	0	$-p\mathcal{X}_1^1$ $+f^1$	$\cdots$	$-p\mathcal{X}_k^1$ $+f^1$	$\cdots$	$-p\mathcal{X}_{K^1}^1$ $+f^1$	0	$-p\mathcal{X}_1^2$ $+f^2$	$\cdots$	$-p\mathcal{X}_k^2$ $+f^2$	$\cdots$	$-p\mathcal{X}_{K^2}^2$ $+f^2$	$\cdots 0 \cdots$	$=$	0
0	**0** $(\mathcal{X}_0^1)$	$\mathcal{X}_1^1$	$\cdots$	$\mathcal{X}_k^1$	$\cdots$	$\mathcal{X}_{K^1}^1$	**0** $(\mathcal{X}_0^2)$	$\mathcal{X}_1^2$	$\cdots$	$\mathcal{X}_k^2$	$\cdots$	$\mathcal{X}_{K^2}^2$	$\cdots e_j \cdots$	$=$	**1**
0	1	1	$\cdots$	1	$\cdots$	1	0	0	$\cdots$	0	$\cdots$	0	$\cdots 0 \cdots$	$=$	1
0	0	0	$\cdots$	0	$\cdots$	0	1	1	$\cdots$	1	$\cdots$	1	$\cdots 0 \cdots$	$=$	1

図 1: $G(P)$ の単体表（$m=2$ のときのイメージ）

分枝の方法は，アイテム j がナップサック 1〜m のいずれかに入るか，どこにも入らないかの $m+1$ 通りとする．分枝を効率よく見切るため，アイテムは効率の逆順，ナップサックも効率の逆順に固定していくことで，早い段階で選ばないアイテム，使わないナップサックを特定できる．また，分枝頂点の探索は深さ優先探索とし，分枝を見切るための条件は標準的な分枝限定法の枠組みにしたがう．

3.2 数値実験と結果

表 1: m,n の変化とアイテムの相関（秒）

		m			
例題	n	$0.2n$	$0.4n$	$0.6n$	$0.8n$
	50	23.52	18.75	15.39	13.77
無相関	100	281.98	158.15	132.20	102.82
	200	—	—	—	2389.56
	50	21.25	17.12	13.42	12.17
弱相関	100	223.38	128.12	98.15	92.11
	200	—	—	—	1755.86
	50	18.33	12.58	10.21	8.28
強相関	100	132.68	93.81	75.23	55.14
	200	—	1532.05	1238.15	832.58

（—：計算時間 2500 秒以上で計測打ち切り）

表 1 は FMKP の厳密解を得るまでの CPU 時間を示す．n と m を変化させ，p_j, w_j 共に [1,100] の乱数（無相関），$w_j = p_j + [1,20]$ の乱数（弱相関）および $w_j = p_j + 10$（強相関）とする．b^i は $\left[w_j\text{の最小値}, \frac{n}{m+i}\sum w_j\right]$ の乱数，また，$\sum f^i$ はナップサックに入る平均的な総価値の 0.5 倍であり，表の各数値は 100 回実験を行った平均値である．

表 1 より，$n=100$ 程度まで実用的な時間で解けた．また，アイテムの相関関係が強まるほどよい結果が得られており，一般にナップサック問題では強相関が解き難いという性質とは逆の結果が得られた．

m と n の関係では，m の割合が多いほどよくなっている．これは従来の MKP の解法では m が増加すると急激に解き難くなる性質とは逆である．一般化割当問題の列生成法では，生成列の非ゼロ要素が少ない方が効果的だと報告されているが [4]，この現象は使用するナップサック数が多いほど効果的だという結果と合致する．

表 2: 固定費の変化　$m=60, n=100$　（秒（個））

	固定費の割合 $(\sum_i f^i) / (\sum_i \sum_j p_j x_j^i)$		
例題	0.25	0.50	0.75
無相関	100.14(41)	132.20(39)	149.35(34)
弱相関	89.87(43)	98.15(40)	130.73(35)
強相関	60.48(45)	75.23(42)	94.86(37)

（括弧内は採用された袋の個数）

表 2 に，固定費の割合を変化させた結果を示す．表より $\sum f^i$ が大きくなると解き難くなっていることがわかる．これは，固定費が増えると採用したほうが有利なナップサック数が少なくなるため，m の割合が少ない場合と同様の問題を解くことになるからだと考えられる．

4 結論と考察

固定費付き複数ナップサック問題に対し，列生成法を用いた上界値の有効性を示した．その上界値を利用した分枝費用法のアルゴリズムを開発し，実用的な時間で厳密解を求めることに成功した．実行効率は通常の（複数）ナップサック問題とは逆に，強相関の例題やナップサック数の多い方が解きやすい性質が分った．

問題点としては，列生成の際の改訂単体法の無駄な繰返しが依然として多く，この問題の解決によりさらなる効率的なアルゴリズムの開発が期待できる．

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] H.Kellerer *et al.*: *Knapsack Problems*, (Springer, 2004).

[3] S.Martello and P.Toth: *Knapsack Problems — Algorithms and Computer Inplementations*, (John Wilay & Sons, 1989).

[4] M.Savelsbergh: A branch-and-price algorithm for the generalized assignment problem. *Ope. Res.*, **45**(1997)831-841.

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