G-05

非線形ナップサック問題に対するグローバルグリーディ法
Global greedy method for nonlinear knapsack problem

伊藤　直也†　　米澤 朋子††　　岡田佑一‡　　仲川勇二††
Naoya ITo　　Tomoko Yonezawa　　Yuichi Okada　　Yuji Nakagawa

1. はじめに

本稿では組み合わせ最適化問題に属する非線形ナップサック問題に対する近似解法について検討する．質の良い近似解を得ることによって，厳密解を求めるための消費メモリや計算時間を大幅に短縮することができる．本稿では，単一制約非線形ナップサック問題と二制約非線形ナップサック問題に対して[1]で提案されているアルゴリズムを改良したものを適用しその結果を示す.

2. まえがき

組み合わせ最適化問題は NP 困難であり，大規模な問題を厳密に解くことは難しいため，短時間で解くことができるアルゴリズムの開発が必要とされてきた．非線形ナップサック問題はナップサック問題の中でも複雑な問題として分類されている．非線形ナップサック問題の厳密解法として動的計画法や分枝限定法が主に使われているが，計算時間は問題のサイズに大きく依存する.多次元非線形ナップサック問題に至っては，一般的な解法を用いると小規模の問題でなければ厳密解を得ることは難しい.

多次元非線形ナップサック問題の解法の一つとして Glover[2]によって代理制約法が提案された．代理制約法とは，それぞれの制約条件式に重みづけを行い,単一制約問題に置き換えた問題（代理問題）を解くというものである．代理問題を解くことによって得られる解には，制約を満たさず実行不可能であるものが得られる場合があり，これを代理ギャップと呼ぶ．この問題に対し，仲川らの研究[3]により提案された改良代理制約法（ISC 法[10]）により代理ギャップを閉じ，厳密解を得ることに成功している[4] [12].

近年の非線形ナップサック問題に対する近似解法の研究の多くは遺伝的アルゴリズム[5]の派生形が主であった．遺伝的アルゴリズムやその他の近似解法では，問題を解く際に多くのパラメータ設定が必要になり，使用者の経験が問われる場合が多い．近年では局所探索法と遺伝的アルゴリズムを組み合わせた方法や問題に合わせたコーディングをする遺伝的アルゴリズムが主流となってきているが，局所最適解からの脱出を図るためアルゴリズムに確率的要素を用いることは免れない．乱数シードの値によって，通常それぞれの試行において実行結果が変わるのも特徴である．初期解や交叉率，突然変異確率など解が左右される要因が多い．遺伝的アルゴリズムでは多次元非線形ナップサック問題において，変数の数が多くなると解の品質が低下することが報告されている[9].

本稿で紹介するグローバルグリーディ法は[1]で提案されている解法を改良したもので，解を出すために確率的要素を用いず，使用者によるパラメータ設定も行わない．またアルゴリズムの性質上，グローバルグリーディ法によって出された解は大域的最適解に近い解が得られる.グローバルグリーディ法では再帰した問題を解くことで解の更新を行うが,一度目の試行において最大許容資源量が 0 になった時,得られた解は厳密解となる.比較として用いた DGR 型グリーディ法では上界値計算も行うことができるため,問題の上界値とグローバルグリーディ法によって出された解の比較を行った.

また[1]では信頼性の問題においての検討しか行われていないが，本稿では一般的な非線形ナップサック問題に対して検討し，二制約の非線形ナップサック問題の結果も示す.

本稿ではグローバルグリーディ法の有用性を示すために単一制約問題に対しては greedy ベースの 3 種類のアルゴリズムを用い，二制約問題に対しては greedy ベースの 2 種類のアルゴリズムを用いてそれぞれ比較を行った.

3. 非線形ナップサック問題とその従来解法

3.1 非線形ナップサック問題

多次元制約非線形ナップサック問題は次のように定式化される.

$$\text{maxmize} \quad f(x) = \sum_{i \in N} f_i(x_i) \tag{1}$$

subject to

$$g_j(x) = \sum_{i \in N} g_{ji}(x_i) \leqq b_j (j = 1,2,\cdots,m) \tag{2}$$

$$x_i \in A_i (j = 1,2,\cdots,m) \tag{3}$$

ここで，N={1,2,…,n}は変数の番号の集合であり，
$A_i = \{1,2,\cdots,a_i\}(i \in N)$は各変数の項目集合である.
また m=1 のとき単一非線形ナップサック問題となり，本稿では単一制約非線形ナップサック問題と二制約非線形ナップサック問題について示す.

3.2 従来の解法

これまで非線形ナップサック問題に対して，グリーディ法，遺伝的アルゴリズム，タブーサーチ，局所探索法などさまざまな近似解法が研究されてきた．またこれらの近似解法の良い部分を組み合わせた解法の研究も行われている.近年では，群知能[7]アルゴリズムの研究が活発になっている．[1]，[6]，[8]では多次元非線形ナップサック問題についてグリーディ法を改良したアルゴリズムを用いている．[6]も多次元非線形ナップサック問題に対して ISC 法を用いている．[6]で提案されているスマートグリーディ法では,バランス係数というパラメータを用い，バランス係数を変化させることでいくつもの解を生成し，その中で一番良い解をスマートグリーディ解としている.

†関西大学大学院　総合情報学研究科 知識情報学専攻, Kansai University, Graduate School of Informatics
†† 関西大学 総合情報学部，Kansai University, Faculty of Informatics
‡株式会社トータス，大阪府高槻市 Tortoise Inc.

4.グローバルグリーディ法

以下に単一制約の非線形ナップサック問題におけるグローバルグリーディ法のアルゴリズムを記す.

f(x): 目的関数
g(x): 制約関数
b: 最大許容資源量

(step1) 前処理

- 目的関数値 z を 0 で初期化する.
- $f_i(x_i)$ の値が昇順になるようにソートする.
- 解を $x_i = 1$ (i = 1, 2, …, N) で初期化する.

(step2) 優越テスト

$f_i(x_i)$ に対して優越テストを行う．ある項目に対して $f_i(p) \leq f_i(q)$ かつ $g_i(p) \geq g_i(q)$ となる $f_i(p)$, $g_i(p)$ が存在する場合は解の候補から外す.

(step3) 実行可能テスト

$g_i(k)$ が b を超えている場合，その代替案は選ばれる可能性がないため解候補から削除する．またこの際暫定解(最初の実行では初期解) x を用いて次の操作を行う.

$$f_i(k) = f_i(k) - f_i(x_i) \ (k = 1, \cdots, a_i)$$
$$g_i(k) = g_i(k) - g_i(x_i) \ (k = 1, \cdots, a_i)$$
$$\mathrm{z} = \mathrm{z} + \sum_{i=1}^{n} f_i(x_i)$$
$$\mathrm{b} = \mathrm{b} - \sum_{i=1}^{n} g_i(x_i)$$

各変数に対して少なくとも 1 つの項目を選択する必要があるため，この操作は各変数に対して最も値の小さい項目を最初に選択する意味を持つ.

(step4) DGR 型項目の生成

$f_i(k), g_i(k)$ において $\mathrm{k} \to \mathrm{k}+1$ に変化したときの制約関数値の変化量に対する目的関数値の変化量の比 $w_i(k)$ を

$$w_i(k) = \frac{f_i(k+1) - f_i(k)}{g_i(k+1) - g_i(k)}$$

とする．このとき $w_i(x_i)(x_i \in A_i)$ が単調減少する（DGR 型: Decreasing Gain Ratio）関数になるようにする．単調減少しない場合は $k+1$ をA_iから削除し $k+s$ $(s = 2, 3, \ldots, a_i)$ として再計算を行う.

(step5) グリーディ解の生成

以下の疑似コードのようにw の値が大きいものから順に選び，グリーディ解 x を更新する.

```
While ( w is not empty )
    w* = max(w), w.pop(w*)
    i, k = index_of(w*)
    IF ( b − g_i(k) < 0 ) THEN break
    z = z + f_i(k)
    b = b − g_i(k)
    x_i = j
EndWhile
```

(step6) 再帰処理

$b < 0$ または選択できる w がなければ終了する．そうでなければ(step3)～(step6)を再帰的に呼び出す.

本稿において 1 回目の試行によって得られた解を DGR 型グリーディ解とし，最終的に得られた解をグローバルグリーディ解とする．つまり 1 回目の試行において処理が終了した場合は DGR 型グリーディ解とグローバルグリーディ解は一致することになる．非線形ナップサック問題は基本的に非凸であるが，DGR 型への変換を行うことで凸性が保証される．DGR 型グリーディ解を得る際に $b = 0$ であればその解は厳密解となるため，グローバルグリーディ解は大域性を保持した近似解を得ることが可能となる．[11]

5. 例題

例題を用いてグローバルグリーディ法のアルゴリズムを具体的に示す．なお，例題で用いる集合は目的関数値によって昇順にソートされているものとする.

表 5.1 目的関数と制約関数

Index	1	2	3	4	5	6
f_1/g_1	24/21	26/20	37/21	41/23	48/24	54/25
f_2/g_2	7/10	13/11	16/12	18/14	18/15	19/16
f_3/g_3	30/31	34/32	36/34	37/33	38/34	40/35

$b = 66, z = 0$

各変数に対して優越テストを行った後，各変数ついて目的関数・制約関数の最小値を差し引き，差し引いた目的関数値を z に加え，制約関数値は b から差し引く.

表 5.2 優越テストと実行可能テスト後の目的関数と制約関数

Index	1	2	3	4	5
f_1/g_1	0/0	11/1	15/3	22/4	28/5
f_2/g_2	0/0	6/1	9/2	11/4	12/6
f_3/g_3	0/0	4/1	7/2	8/3	10/4

$b = 66 - 61 = 5, z = 0 + 63 = 63$

上記の f_n, g_n を用いて w_n を計算する.

$$w_1(2) = \frac{f_1(2) - f_1(1)}{g_1(2) - g_1(1)} = 11.0$$

となり，同様に $w_1(3) = 2.0, w_1(4) = 7.0$ となるので,

$$w_1(4) = \frac{f_1(4) - f_1(2)}{g_1(4) - g_1(2)} = 3.66$$

とする．以下同様にすべての w を求める.

表 5.3$w_i(k)$の計算結果

Index	1	2	3	4	5
w_1	0.00	11.00	-	-	4.25
w_2	0.00	6.00	3.00	1.00	0.50
w_3	0.00	4.00	3.00	-	1.50

w を降順に並べると,

$$w_1(2) = 11.0, w_2(2) = 6.00, w_1(5) = 4.25, w_3(2) = 4.00, \ldots$$

となる． $w_1(2)$ に対応する $f_1(2), g_1(2)$ を解として選ぶと

$$b = 5 - 1 = 4, z = 63 + 11 = 74$$

となり， $w_2(2)$ に対応する $f_2(2), g_2(2)$ を解として選ぶと

$$b = 4 - 1 = 3, z = 74 + 6 = 80$$

となる． $w_1(5)$ に対応する $f_1(5), g_1(5)$ を解として選ぶと

$$b - 4 = 3 - 4 < 0$$

となり制約条件を超えるため停止する．よって 1 回目の試行後の解は $\mathbf{x} = \{2,2,1\}$ となり．目的関数値は 80 となる．これを DGR 型グリーディ解とする．次に残りの $\mathbf{b}(=3)$ を制約条件として実行可能テストを行い，各変数の候補 f_i, g_i のリストを生成する．このとき暫定解として選択された $f_i(x_i), g_i(x_i)$ を offset 値として f_i, g_i, z, b を更新する．

表 5.4 再帰問題に優越テストと実行可能テストを行った値

Index	1	2	3	4	5
f_1/g_1	-	0/0	4/2	11/3	-
f_2/g_2	-	0/0	3/1	5/3	-
f_3/g_3	0/0	4/1	7/2	8/3	-

$b = 3, z = 80$

表 5.5$W_i(k)$の計算結果

Index	1	2	3	4	5
w_1	-	0.00	-	3.66	-
w_2	-	0.00	3.00	1.00	-
w_3	0.00	4.00	3.00	1.00	-

最も大きい $w_3(2) = 4.00$ に対応する $f_3(2), g_3(2)$ を選び，$b = 3 - 1 = 2, z = 80 + 4 = 84$ とする．次に $w_1(4) = 3.66$ に対応する $f_1(4), g_1(4)$ を選ぶと， $b = 2 - 3 < 0$ となり制約条件を超えるため，解を $\mathbf{x} = \{2,2,2\}$ と更新してさらに再帰する．

表 5.6 再帰問題に優越テストと実行可能テストを行った値

Index	1	2	3	4	5
f_1/g_1	-	0/0	4/2	-	-
f_2/g_2	-	0/0	3/1	-	-
f_3/g_3	-	0/0	3/1	4/2	-

$b = 2, z = 84$

表 5.7$W_i(k)$の計算結果

Index	1	2	3	4	5
w_1	-	0.00	-	-	-
w_2	-	0.00	3.00	-	-
w_3	-	0.00	3.00	1.00	-

次に $w_2(3) = 3.00, w_3(3) = 3.00$ を選ぶと $b = 2 - 1 - 1 = 0$ となり $b = 0$ となるためプログラムは停止し，解は $\mathbf{x} = \{2,3,3\}$ つまり (f_i, g_i) の選択された値は $\{(37,21),(16,12),(37,33)\}$ であり，グローバルグリーディ値は 90 となる．

尚，本問の厳密解は 91 であり，非常に近い値が近似解として得られた．また DGR 型グリーディ法で上界値を計算した場合 92.75 となる．

6.計算精度比較実験

6.1 単一制約非線形ナップサック問題

本解法の有効性を示すために OR-library に基づいて作られている Beasley のテスト問題[5]を用いてグリーディ法，DGR 型グリーディ法，グローバルグリーディ法の解の精度比較を行った．グローバルグリーディ法はグリーディ法をベースとしたアルゴリズムのため上記の 2 つのアルゴリズムを比較の対象とした．Beasleyのテスト問題では複数制約問題を扱っているため制約関数それぞれを一つの目的関数とすることや，目的関数であったものを制約関数とすることで多様な問題を 15 問作成した．一つの問題は 100 変数 11 項目となっている．計算結果は表 6.1 のようになっている．問題に対する精度比較としてグローバルグリーディ法を用いて出された上界値と比較した．

6.2 二制約非線形ナップサック問題

単一制約非線形ナップサック問題と同様に Beasley のテスト問題を用いて 10 問のテスト問題を作成した．それぞれの問題は 100 変数 11 項目となっている．単一制約問題においてグリーディ法はグローバルグリーディ法と DGR 型グリーディ法との違いが大きく見られたため，二制約問題では比較対象から外した．代理制約法を用いて単一制約問題に変えた代理問題にグローバルグリーディ法を用いた．二制約問題では代理ギャップが見られた問題もあるため,制約を超えて出された解は色を付けて示している.また二制約問題では厳密解法との比較を行い,グローバルグリーディ法の解の精度を測定した．

表 6.2 では代理問題を解いた実行結果を示し,表 6.3 では表 6.2 で扱った問題と同じ問題を扱い,代理ギャップが出ないように制約を調整した問題の実行結果を載せている．

表 6.1 単一制約問題実行結果

	test1	test2	test3	test4	test5
greedy	71952	75874	62986	59081	62365
DGRgreedy	85222	94578	85920	84102	81264
globalgreedy	85484	94895	86224	84346	81646
Upper Bound	85563.4545	94915.8682	86263.1384	84383.2113	81685.0058
	test6	test7	test8	test 9	test10
greedy	71748	61752	66415	62728	59081
DGRgreedy	89452	68197	78751	88389	75163
globalgreedy	89712	68669	79236	88720	75199
Upper Bound	89742.5082	68813.6793	79287.4667	88758.6483	75227.3506
	test11	test12	test13	test14	test15
greedy	80920	66552	67473	68848	69815
DGRgreedy	94390	84874	86847	80661	88608
globalgreedy	94795	85114	87008	80900	89012
Upper Bound	94813.9128	85164.8626	87019.4223	80988.9974	89029.5714

表 6.2 二制約問題実行結果

	test1	test2	test3	test4	test5
DGRgreedy	54094	53545	54937	80193	81643
Globalgreedy	54576	53802	55022	80230	81643
Exact solution	54668	53800	55037	80233	81653
	test6	test7	test8	test9	test10
DGRgreedy	80658	80307	94125	94166	94612
Globalgreedy	80719	80465	94125	94322	94672
Exact solution	80704	80479	94115	94318	94683

表 6.3 制約調整後の二制約問題実行結果

	test1	test2	test3	test4	test5
DGRgreedy	54094	52686	54903	79686	80872
Globalgreedy	54094	53435	54903	79790	81303
	test6	test7	test8	test9	test10
DGRgreedy	80079	79944	93792	94166	94499
Globalgreedy	80456	80080	94034	94232	94536

7.考察

単一制約問題においては，すべての問題でグローバルグリーディ法を用いた解が DGR 型グリーディ法やグリーディ法で得られた解よりも質の良い解が得られることが確認された．通常のグリーディ法ではすべての品物が選ばれると，同じ問題であれば制約条件値を変えるだけでは解の更新は行われないが，グローバルグリーディ法では制約条件値に応じて解の更新を行うため,問題の上界値に近い近似解を得ることができた．またグリーディ法の性質上，同じ問題であれば解は常に同じであるので誰が使用しても同じ結果が得られる．単一制約問題では本稿で比較を行った 3 種類のグリーディ法では実行時間に大きな差は見られなかった．小さなサイズの問題においては解が一つに定まってしまうという性質のため遺伝的アルゴリズムなどの近似解法のほうが良い解が得られることも考えられるが，グローバルグリーディ法は他の近似解法よりも実行時間と解の品質が問題のサイズに依存しないため，大きなサイズの問題において良い解が得られることが期待される．

二制約問題では,単一制約に置き換えた代理問題をグローバルグリーディ法で解いた場合，解は原問題の二制約のどちらかを満たさず，すべて実行不可能な解であった．DGR 型グリーディ法では実行可能解になるものもあったが，実行不可能解である場合が多かった．どちらの解法も共に二制約のうち制約を満たさない方の最大許容資源量を小さくした問題に置き換えることで代理乗数が変化し代理ギャップを出さない解を得られると考えた.試行したところ，最大許容資源量を変えると, 最大許容資源量を変える前の問題における実行可能で近似解が得られる場合が多く確認された．

8.むすび

本稿では一般的な単一制約非線形ナップサック問題と二制約非線形ナップサック問題に対してグローバルグリーディ法，DGR 型グリーディ法，グリーディ法の 3 種類のアルゴリズムで解の質の比較を行いグローバルグリーディ法の有用性を示した．実験結果からグローバルグリーディ法では単一制約問題に対して上界値に十分近い近似解を得られることが確認された．二制約問題では代理制約法を用いた代理問題に対して代理ギャップが生じるものの, 最大許容資源量を変化させることで原問題に対する近似解が得られることが確認された．二制約問題において制約式を調整すると良い近似解を得られる場合もあることが確認されたので,今後は制約の調整方法を一般化し複数制約問題に応用していきたい．

文　　献

[1]仲川勇二,太田垣博一:”非線形ナップサック問題型信頼性最適化問題に対するグリーディ法の改良”, 電子情報通信学会論文誌 1991/3 Vol. J74–A No.3

[2] Glover, F. “Surrogate constraints”. Oper. Res, 16, pp.741–749(1968)

[3]仲川勇二,疋田光伯,鎌田弘,”代理双対問題を解くためのアルゴリズム”,信学論 (A), vol.J67-A,no.1,pp53-59,Jan,1984

[4]並川哲郎,岩崎彰典,太田垣博一,仲川勇二”変数分離可能な多次元非線形ナップザック問題の解法と 2 次形式ナップザック問題への適用(あいまいさと不確実 性を含む状況の数理的意思決定),” 数理解析研究所講究録,1252,pp64-68,Feb.2002

[5]P.C. Chu and J.E. Beasley, “A genetic algorithm for the multidimensional knapsack problem,” J. Heuristics, vol.6, pp.63–86, 1998

[6]太田垣博一, 岩崎彰典,仲川勇二,亀高哲夫,成久洋之 “多次元非線形ナップザック問題に対するスマートグリーディ法の適用(数理システムにおける最適化理論とその応用), ” 数理解析研究所講究録 , (B), 899, pp.133-141, Mar. 1995.

[7]枇杷木秀,黒岩丈介, 諏訪いずみ,白井治彦,小高知宏, 小倉久和,”粒子群最適化法のマルチナップサック問題への適応”,日本知能情報ファジィ学会,

[8]Yal˛cın Ak˛cay·Haijun Li·Susan H. Xu” Greedy algorithm for the general multidimensional knapsack problem”, Ann Oper Res (2007)

[9]並川哲郎,岩崎彰典,太田垣博一,仲川勇二” 多次元非線形ナップザック問題の GA 解の評価 (不確実性 の下での数理モデルの構築と最適化),” 数理解析研究所講究録,1194,pp105-110,Mar.2001

[10]Y. Nakagawa, An improved surrogate constraints method for separable nonlinear integer programming, Journal of the Operations Research Society of Japan, Vol.46, No.2, pp.145163, June 2003

[11]P. Sinha and A. Zoltners: The Multiple-Choice Knapsack Problem, Operations Research, 27 (1978) 125-131

[12] Yuji Nakagawa, Ross J. W. James, César Rego, Chanaka Edirisinghe: Entropy-Based Optimization of Nonlinear Separable Discrete Decision Models, Management Science, Vol. 60, No. 3, March 2014, pp. 695–707