研究速報

多制約分離形離散最適化問題のための近似解法

仲川　勇二†（正員）

A Heuristic Method for Multi-Constraint Separable Discrete Optimization

Yuji NAKAGAWA†, *Member*

† 関西大学総合情報学部，高槻市
Faculty of Informatics, Kansai University, 2–1–1 Ryouzenji, Takatsuki-shi, 569–1095 Japan

あらまし　代理双対ギャップをもつ大規模な多制約分離形離散最適化問題（多次元非線形ナップザック問題）を厳密かつ効率良く解く改良代理制約法（ISC 法）が提案された．本論文では ISC 法を近似解法として利用する．その有効性を示すために Chu と Beasley のテスト問題を用いて計算機実験を行う．難しい問題として知られている 5 制約条件で 500 変数の 0–1 ナップザック問題 30 問を解いた結果は，平均 13.6 秒の計算時間の場合正答率は 80%，平均計算時間 55.8 秒の場合正答率は更に向上し 100%となったことを報告する．

キーワード　多制約分離形離散最適化問題，多次元非線形ナップザック問題，組合せ最適化，代理制約法，近似解法

1. まえがき

代理制約法は，複数の制約条件式をもつ原問題を代理乗数を用いて単一制約の代理問題からなる代理双対問題へ変換し，この双対問題を解くことで原問題の最適解を求めようとする方法である [3]．しかし多くの離散最適化問題では，代理双対問題に双対ギャップ (duality gap) が存在することが多く，代理制約法は解決が困難な問題点をもつ解法と考えられていた [5]．この問題点を解決するために，仲川 [9], [10] は標的（標的に当たった解を列挙する）問題を解くことで代理双対ギャップを閉じ，原問題の厳密解を求めることができる改良代理制約法（ISC 法）を提案した．この解法により 3 個の制約条件式をもち 1000 変数で各変数 20 個の代替項目をもつ問題や 8 制約で 500 変数 50 代替項目の問題のように，極めて大規模な多制約非線形ナップザック問題も厳密解を求めることが可能になった．

本論文で提案する近似解法は，ISC 法を変形し標的値を大きくし解空間を狭めることで計算時間を減らす試みを行っている．本近似解法は，計算実験より解の品質をあまり下げることなしに，計算時間を大幅に削減できることを報告する．

2. 多次元非線形ナップザック問題と近似解法

多次元非線形ナップザック (MNK) 問題は以下のように記述できる．

$$\mathbf{P} : \max\{f(\mathbf{x})|\mathbf{g}(\mathbf{x}) \leq \mathbf{b}, \mathbf{x} \in \mathbf{K}\}$$

ただし，$f(\mathbf{x})$，$\mathbf{g}(\mathbf{x})$ は変数分離関数で，

$$\mathbf{x} = (x_1, x_2, \ldots, x_n),$$
$$\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_m(\mathbf{x})),$$
$$\mathbf{b} = (b_1, b_2, \ldots, b_m),$$
$$\mathbf{K} = \{\mathbf{x}|x_i \in K_i \text{ for } i = 1, 2, \ldots, n\},$$
$$K_i = \{1, 2, \ldots, k_i\}.$$

ここで，K_i は変数 x_i がとり得る代替項目集合である．この問題に対応した代理双対問題 $\mathbf{P}^{\mathrm{SD}}$ は，次のように書ける．

$$\mathbf{P}^{\mathrm{SD}} : \min\{v^{\mathrm{Opt}}[\mathbf{P}^{\mathrm{S}}(\mathbf{u})]|\mathbf{u} \in \mathbf{U}\},$$

ただし，$v^{\mathrm{Opt}}[\bullet]$ は問題 $\bullet$ の最適値（最適解の目的関数値），

$$\mathbf{P}^{\mathrm{S}}(\mathbf{u}) : \max\{f(\mathbf{x})|\mathbf{ug}(\mathbf{x}) \leq \mathbf{ub}, \mathbf{x} \in \mathbf{K}\},$$
$$\mathbf{u} = (u_1, u_2, \ldots, u_m),$$
$$\mathbf{U} = \{\mathbf{u}|\textstyle\sum_{j=1}^{m} u_j = 1, \mathbf{u} \geq \mathbf{0}\}.$$

問題 $\mathbf{P}^{\mathrm{S}}(\mathbf{u})$ は原問題 $\mathbf{P}$ の代理問題で制約条件式が一つである．この代理問題は，モジュラー法 [7], [8] を用いて高速に解くことが可能である．モジュラー法は分枝限定法の分枝操作で幅優先探索を用いた場合を拡張したもので，次の（1）と（2）の操作を繰り返して原問題をより規模の小さい等価問題へ変換する．

（1）　等価問題に対して深測操作を適用し決定空間（変数の項目空間）を縮小する
（2）　等価問題の変数の中から 2 個の変数を選び一つの変数に統合することで，変数の数を一つ減らした等価問題を作る．

ここで，決定空間とは探索する解空間のことであり，深測操作とは等価問題の決定空間を狭めるために，変数の項目を固定してできた部分問題に対して次の三つのテストを行う操作である．

（1）　実行可能性テスト
　　部分問題が実行可能解を含むかどうかテストする．

（2） 優越テスト
　他の部分問題と比較して，その部分問題が明らかに劣っていないかどうか，すなわち原問題の最適解を含むかどうかテストする．

（3） 限界値テスト
　部分問題の限界値を求め，現在の暫定解の値と比較し暫定解よりも良い解を含むかどうかを判定する．

また，選ばれた二つの変数を一つの変数に統合する統合政策は，各変数に対して上界値の最大と最小の差（上界値差）の大きい二つの変数を優先選択する．すなわち，現在の問題を $\mathbf{P}^{\mathrm{C}}$ と書き，変数集合を N^{C}，各変数の代替項目集合を K_i^{C} $(i \in N^{\mathrm{C}})$ とおくと，

$$\min\{p_{i_1}, p_{i_2}\} \geq \max\{p_i | i \neq i_1, i \neq i_2, i \in N^{\mathrm{C}}\}$$

となる i_1，i_2 を選択する，ここで

$$p_i = \max\{v^{\mathrm{UB}}[\mathbf{P}^{\mathrm{C}} : x_i = a] | a \in K_i^{\mathrm{C}}\} - \min\{v^{\mathrm{UB}}[\mathbf{P}^{\mathrm{C}} : x_i = a] | a \in K_i^{\mathrm{C}}\},$$

$v^{\mathrm{UB}}[\mathbf{P}' : \bullet]$ は制約 $\bullet$ を追加した問題 $\mathbf{P}'$ の上界値である．また最適な代理乗数 $\mathbf{u}^*$ は COP 法 [4]〜[6] を用いて求めることができる．得られた最適代理乗数 $\mathbf{u}^*$ に対する標的問題は

$$\begin{aligned}[\mathbf{TP}^{(0)}(f^{\mathrm{T}})] : \text{Enumerate all solutions } \mathbf{x} \text{ hitting} \\ \text{a target} \quad & f(\mathbf{x}) \geq f^{\mathrm{T}} \\ \text{subject to} \quad & \mathbf{u}^* \mathbf{g}(\mathbf{x}) \leq \mathbf{u}^* \mathbf{b}, \\ & \mathbf{x} \in \mathbf{K},\end{aligned}$$

と書ける．標的問題がモジュラー法で比較的簡単に解けるように，提案する近似解法では標的値 f^{T} を変化させながら，与えられた変数の数のときに与えられた数の代替項目合計数になるような標的値 f^{T} を探す．その後得られた標的値を用いて標的問題を解き近似解を求める．本近似解法においては問題の単調性は仮定していないことに注意されたい．ここで，この標的問題をモジュラー法で解く過程で，深測操作と統合操作が ℓ 回 $(0 \leq \ell \leq n)$ 繰り返された後変数の数が $n - \ell$ 個になった等価問題を $\mathbf{TP}^{(\ell)}(f^{\mathrm{T}})$ と書く．このとき提案する近似解法の計算手順は擬似コードを用いて下記のように表現できる．

近似解法

入力：原問題 $\mathbf{P}$，刻み幅 d，変数の個数 s，代替項目数 α；

1. $\mathbf{P}$ の代理双対問題 $\mathbf{P}^{\mathrm{SD}}$ を解き，最適乗数 $\mathbf{u}^*$ と解 $\mathbf{x}^{\mathrm{SD}}$ を求める；
2. **If** $\mathbf{x}^{\mathrm{SD}}$ が $\mathbf{P}$ の実行可能解である **then**
3. 　$\mathbf{x}^{\mathrm{Exact}} \leftarrow \mathbf{x}^{\mathrm{SD}}$ とする；
4. 　このアルゴリズムを終了する；
5. **EndIf**
6. $f^{\mathrm{U}} \leftarrow f(\mathbf{x}^{\mathrm{SD}})$，$f^{\mathrm{L}} \leftarrow f(\mathbf{x}^{\mathrm{SD}}) - d$ とおく；
7. 問題 $\mathbf{TP}^{(n-r)}(f^{\mathrm{L}})$ $(r \leq s)$ の代替項目の合計数が α 以上になるように f^{L} の値を d ずつ下げながら f^{L} の値を設定する；
8. 変数の数が s 個の問題 $\mathbf{TP}^{(n-s)}(f^{\mathrm{L}})$ の代替項目の合計数が α 以上で 1.2α 以下となる f^{T} の値を f^{L} と f^{U} の間で二分探索法を用いて求める；
9. 問題 $\mathbf{TP}^{(n-s)}(f^{\mathrm{T}})$ を厳密に解く．得られた解を $\mathbf{x}^{\mathrm{Near}}$ とする；

ただし，刻み幅 d は経験的に決定する．

この近似解法では，変数が s 個の問題 $\mathbf{TP}^{(n-s)}(f^{\mathrm{T}})$ で比較的簡単に解ける規模の標的値 f^{T} を定め，得られた問題 $\mathbf{TP}^{(n-s)}(f^{\mathrm{T}})$ の標的解（標的に当たった解）の中で原問題 $\mathbf{P}$ の制約条件を満たす最善の解を準最適解とする．

3. 計 算 実 験

提案した近似解法の有効性を評価するために，Chu ら [1] で取り扱われた 5 制約 500 変数の 0–1 ナップザック問題 30 問（http://www.ms.ic.ac.uk/jeb/pub/mknapcb3.txt から得られる）をテスト問題として用いた．このテスト問題は目的関数と制約条件の係数が相関をもって乱数で生成されているため，乱数を独立に用いて作成した問題よりはるかに解くことが難しいことが知られている．この問題では 10 問ごとに制約条件の制約の強さが変えられている．すなわち制約が緩くなると最適解に 1 を多く含むようになる．

今回の実験では，パーソナルコンピュータ (Pentium 4 1.7 GHz，1 GByte) を用いた．本近似解法においてパラメータは $s = 20$ で $\alpha = 5000$，$d = 40$ とした．すなわち標的値 f^{T} を変化させて問題 $\mathbf{TP}^{(n-20)}(f^{\mathrm{T}})$ の代替項目の合計数が 5000 を超えたときの問題を解き，標的に当たった解の中から原問題の制約条件を満たす最善の解を求めた．このときの平均の計算時間は 13.6 秒で正答率は 0.8 であった．また，パラメータを変え $s = 25$ で $\alpha = 5000$，$d = 40$ とした場合の計算結果は，平均 55.8 秒の計算時間で全問正解が得られた．Chu らの GA [3] での平均の計算時間は 1343.5 秒

(SGI workstation R4000，100 MHz，48 MByte) で正解率 0.2 と比較すると，使われた計算機は異なるものの本近似解法の有効性を示した計算結果だといえる．更に商用のソフトウェアである ILOG 社 CPLEX（初期設定精度：相対誤差 0.0001，絶対誤差 1.0e-6）の計算結果は，正答率 0.833 で平均計算時間は 2955.6 秒であった．CPLEX は，商用としては世界最速と評価の高い解法である．今回の実験では計算速度を速めるために，CPLEX を近似解法として用い，有効けた数 4 けたの精度で最適解が求まれば計算を停止する CPLEX の初期設定の状態で実験を行った．また，0–1 ナップザック問題の近似解法としては世界で最高の性能をもつといわれる Vasquez らのタブサーチ (TS) [11] の計算結果は正答率は 0.467 であった．表 1 は計算結果をまとめたものであり，左から順に，厳密解，Chu らの GA [3] を用いたときの解，Vasquez らの TS [11] を用いたときの解，CPLEX で計算したときの解と計算時間，提案した近似解法 ($s = 20$，$\alpha = 5000$，$d = 40$) で計算したときの解と計算時間，提案した近似解法 ($s = 25$，$\alpha = 5000$，$d = 40$) で計算したときの解と計算時間を表している．なお，$*$ 印は厳密解と一致しなかったことを意味する．

4. むすび

本論文で提案した近似解法は高い確率で厳密解を見つけている，また正解が見つからなかった場合でも極めて良い準最適解を見つけている．本解法は計算時間と正答率の両面で優れており，実用的観点から見て極めて良い特質をもつことが分かった．今後の課題としては，更に他のテスト問題を解くことで応用範囲を広めることと，アルゴリズムの更なる改良を予定している．

文　献

[1] P.C. Chu and J.E. Beasley, “A genetic algorithm for the multidimensional Knapsack problem,” J. Heuris-

表 1 計算結果
Table 1 Computational result.

Prob.		GA	TS	CPLEX		The present method			
						$s = 20$		$s = 25$	
No.	f^{Exact}	f^{GA}	f^{TS}	f^{CPLEX}	Time(s)	f^{Near}	Time(s)	f^{Near}	Time(s)
0	120148	120130*	120134*	120148	3744	120148	4	120148	31
1	117879	117837*	117864*	117879	1476	117879	15	117879	39
2	121131	121109*	121112*	121131	5796	121131	16	121131	57
3	120804	120798*	120804	120804	4536	120804	17	120804	78
4	122319	122319	122319	122319	5364	122319	13	122319	21
5	122024	122007*	122024	122024	10332	122024	15	122024	58
6	119127	119113*	119127	119127	4068	119127	15	119127	38
7	120568	120568	120568	120568	7776	120568	14	120568	87
8	121586	121575*	121575*	121575*	3384	121575*	9	121586	30
9	120717	120699*	120707*	120711*	3996	120717	17	120717	66
10	218428	218422*	218428	218428	2016	218428	12	218428	62
11	221202	221191*	221191*	221202	1800	221191*	13	221202	53
12	217542	217534*	217534*	217534*	3528	217534*	11	217542	37
13	223560	223558*	223558*	223560	4644	223560	14	223560	57
14	218966	218962*	218966	218966	1116	218966	12	218966	32
15	220530	220514*	220530	220530	4824	220530	11	220530	69
16	219989	219987*	219989	219989	2016	219989	16	219989	78
17	218215	218194*	218194*	218215	1188	218215	18	218215	97
18	216976	216976	216976	216976	5688	216976	14	216976	70
19	219719	219693*	219704*	219719	3636	219717*	14	219719	15
20	295828	295828	295828	295828	72	295828	14	295828	36
21	308086	308077*	308083*	308086	648	308086	17	308086	55
22	299796	299796	299796	299796	288	299796	15	299796	32
23	306480	306476*	306478*	306480	1836	306480	12	306480	80
24	300342	300342	300342	300342	360	300342	10	300342	51
25	302571	302560*	302561*	302565*	756	302560*	14	302571	48
26	301339	301322*	301329*	301339	468	301339	9	301339	51
27	306454	306430*	306454	306454	576	306454	18	306454	88
28	302828	302814*	302822*	302828	1116	302828	15	302828	67
29	299910	299904*	299904*	299904*	1620	299906*	12	299910	91
Aver.					2955.5		13.6		55.8

tics, vol.6, pp.63–86, 1998.
[2] M.E. Dyer, “Calculating surrogate constraints,” Math. Program., vol.19, pp.255–278, 1980.
[3] F. Glover, “Surrogate constraints,” Oper. Res., vol.16, pp.741–749, 1968.
[4] Y. Nakagawa and S. Miyazaki, “Surrogate constraints algorithm for reliability optimization problems with two constraints,” IEEE Trans. Reliab., vol.R-30, no.2, pp.175–180, 1981.
[5] Y. Nakagawa, M. Hikita, and H. Kamada, “Surrogate constraints algorithm for reliability optimization problems with multiple constraints,” IEEE Trans. Reliab., vol.R-33, no.4, pp.301–305, Oct. 1984.
[6] 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, pp.53–59, Jan. 1984.
[7] 仲川勇二, “離散最適化問題のための新解法,” 信学論 (A), vol.J73-A, no.3, pp.550–556, March 1990.
[8] Y. Nakagawa and A. Iwasaki, “Modular approach for solving nonlinear Knapsack problems,” IEICE Trans. Fundamentals, vol.E82-A, no.9, pp.1860–1864, Sept. 1999.
[9] Y. Nakagawa, “A reinforced surrogate constraints method for separable nonlinear integer programming,” RIMS 1068, Kyoto University, pp.194–202, 1998.
[10] Y. Nakagawa, “An improved surrogate constraints method for separable nonlinear integer programming,” J. Operations Research Society of Japan, vol.46, no.2, pp.145–163, June 2003.
[11] M. Vasquez and J. Hao, “A hybrid approach for the 0-1 multidimensional Knapsack problem,” IJCAI, pp.328–333, 2001.

（平成 15 年 7 月 28 日受付,
16 年 1 月 8 日最終原稿受付）