日本オペレーションズ・リサーチ学会
2021年 春季研究発表会

整数線形計画問題に対する重みつき局所探索法

05009910 NTT データ数理システム 神谷俊介 KAMIYA Shunsuke
01014954 大阪大学 梅谷俊治 UMETANI Shunji
NTT データ数理システム 藤井浩一 FUJII Koichi
NTT データ数理システム 石橋保身 ISHIBASHI Yasumi

1. はじめに

整数線形計画問題を以下の形式で定義する．

$$\begin{aligned} \text{min.} \quad & \sum_{j\in N} c_j x_j \\ \text{s.t.} \quad & \sum_{j\in N} a_{ij}x_j \le b_i \qquad (i \in M_L) \\ & \sum_{j\in N} a_{ij}x_j \ge b_i \qquad (i \in M_G) \\ & l_j \le x_j \le u_j,\ x_j \in \mathbb{Z} \quad (j \in N) \end{aligned} \tag{1}$$

ここで M_L, M_G は制約式の添字集合とし，制約式全体は $M = M_L \cup M_G$ で表す．この最適化問題に対して，大規模な問題でも短時間で比較的良い実行可能解を得るために，重みつき局所探索法 (weighting local search, WLS) と呼ばれるヒューリスティクス [1] が提案されている．そこでは問題クラスに対し以下の仮定が施され，それに合わせアルゴリズムが設計されている．

仮定 1. $a_{ij} \in \{0, 1\} \qquad (i \in M,\ j \in N)$

仮定 2. $l_j = 0,\ u_j = 1 \quad (j \in N)$

WLS は集合被覆問題と集合分割問題に対して高い性能が確認されており，その後中村ら [2] が集合分割問題に対しさらなる改良を加えた．

一方で数理最適化ソルバーとしては，これらの強みを保ちながらより広範な問題クラスが扱える方が好ましい．そこで本発表では，上記 2 つの仮定を取り払った形式が扱えるようアルゴリズムを拡張し，当社製品である Numerical Optimizer V23 へ組み込み性能を検証したことを報告する．

2. 従来の重みつき局所探索法

上述した 2 つの仮定の下で，梅谷 [1] の手法について概説する．定式化 (1) に関して，**ペナルティ重み** $w_i^+, w_i^- \ge 0$ を各制約式に対し導入し以下の定式化 (2) を考える．定式化中の $|\cdot|_+$ は中身と 0 のうち大きい方を返す関数であり，y_i^+, y_i^- は**制約の違反量**を表す．

$$\begin{aligned} \text{min. } z(\boldsymbol{x}) = & \sum_{j\in N} c_j x_j + \sum_{i\in M_L} w_i^+ y_i^+ + \sum_{i\in M_G} w_i^- y_i^- \\ \text{s.t. } & y_i^+ = |\sum_{j\in N} a_{ij}x_j - b_i|_+ \qquad (i \in M_L) \\ & y_i^- = |b_i - \sum_{j\in N} a_{ij}x_j|_+ \qquad (i \in M_G) \\ & l_j \le x_j \le u_j,\ x_j \in \mathbb{Z} \qquad (j \in N) \end{aligned} \tag{2}$$

重みが十分大きければ，(2) の最適解は (1) の最適解と一致する．

この手法の大まかな流れは次の通りである．まず初期化として，$\boldsymbol{x} \leftarrow \boldsymbol{0}$ とし，各重み $\boldsymbol{w}^+, \boldsymbol{w}^-$ には十分大きな値を入れておく．以降は (i) 局所最適に達するまで定式化 (2) の下で $\boldsymbol{x}$ について局所探索を行い，(ii) 各重み $\boldsymbol{w}^+, \boldsymbol{w}^-$ を更新する，という操作を繰り返す．なお，これとは別に最良の (1) の実行可能解 $\boldsymbol{x}^*$ も保持する．

局所探索では，暫定解 $\boldsymbol{x}$ の近傍の中で $z(\boldsymbol{x})$ の値が下がる解への遷移を繰り返す．ここで $\boldsymbol{x}$ の近傍は，$\boldsymbol{x}$ の要素のうち k 個の変数の値の 0, 1 を反転 (k-*flip*) することで得る．WLS は $k = 1, 2, 4$ の順で試行し改善解を発見次第，特定の近傍における最良解へ移動する（以下では $k = 4$ は考えない）．

重み更新では，基本的には次式に従い違反制約式に対応する重みを増加する．

$$w_i^{\pm} \leftarrow w_i^{\pm} + \frac{z(\boldsymbol{x}^*) - z(\boldsymbol{x})}{\sum_{l\in M_L} {y_l^+}^2 + \sum_{l\in M_G} {y_l^-}^2} y_i^{\pm}$$

一方で，$z(\boldsymbol{x})$ が目的関数値の下界として機能しなくなった場合は，各重みを一斉に β 倍 $(0 < \beta < 1)$ し探索をリセットする．

局所探索の高速化について述べる．k-flip 近傍は $O(|N|^k)$ 個存在するため，$k = 2$ では近傍の範囲を有望な候補に絞ることが重要である．変数

x_{j_1}, x_{j_2} を 2-flip する際，まず以下の定理から $x_{j_1} = 1, x_{j_2} = 0$ のときに限定できる．

定理 1. $\boldsymbol{x}$ が 1-*flip* において局所最適であるとき，2-*flip* における目的関数の差分値が負値となるのは $x_{j_1} \neq x_{j_2}$ のときに限る．

また，**隣接リスト**という概念を用いた近傍の取捨も本質的である．$S_{j,j'}$ を「A の j 列目と j' 列目で共通して 1 が立つ行 i の集合」とし，変数 x_j の隣接リスト $L(j)$ には $|S_{j,j'}|$ が大きい j' たちを格納する．以上を併せ，2-flip 近傍を $j_1 \in X,\ j_2 \in L(j_1) \cap \overline{X}$ を満たす変数組に限定する．ここで $X = \{j \in N \mid x_j = 1\}$，$\overline{X}$ は X の補集合である．

最後に，差分値計算の高速化について述べる．2-flip の目的関数の差分値 $\Delta z^{\downarrow\uparrow}(\boldsymbol{x})$ は次式で表せる．

$$\Delta z^{\downarrow\uparrow}_{j_1,j_2}(\boldsymbol{x}) = \Delta z^{\downarrow}_{j_1}(\boldsymbol{x}) + \Delta z^{\uparrow}_{j_2}(\boldsymbol{x}) - \sum_{i \in S_{j_1,j_2};\ \sum_j a_{ij}x_j = b_i} (w_i^+ + w_i^-) \tag{3}$$

ここで $\downarrow, \uparrow$ はそれぞれ $1 \to 0,\ 0 \to 1$ の flip を表す．式 (3) により，1-flip の計算結果を用いて 2-flip の差分値計算を高速に行える．

3. アルゴリズムの拡張

定式化 (1) から仮定 1 と仮定 2 を緩和する上で，アルゴリズムに施した修正について説明する．

仮定 1：実数係数への対応．「$a_{ij} \in \mathbb{R}$」として仮定を緩めたとき，2-flip における近傍の取捨と目的関数の差分計算をいかに修正するか考える．なお，ここでは便宜的に変数はすべて 0-1 変数とみなす（仮定 2 を認める）．

まず近傍取捨の土台であった定理 1 は，実数係数において次のとおりに一般化できる．

定理 2. すべての $i \in M$ について $a_{ij_1}a_{ij_2} \geq 0$ (≤ 0) とする．$\boldsymbol{x}$ が 1-*flip* において局所最適であるとき，2-*flip* における目的関数の差分値が負値となるのは $x_{j_1} \neq x_{j_2}$ $(x_{j_1} = x_{j_2})$ のときに限る．

したがって変数組 (j_1, j_2) の 2-flip に関して定理の前提条件を満たせば，「$x_{j_1} = 1 \wedge x_{j_2} = 0$」または「$(x_{j_1} = 0 \wedge x_{j_2} = 0) \vee (x_{j_1} = 1 \wedge x_{j_2} = 1)$」として近傍を限定できる．一方でこのままでは前提条件を満たさない変数組も存在する．そこでこの条件を A の列ベクトルの内積 $\boldsymbol{a}_{j_1} \cdot \boldsymbol{a}_{j_2}$ の正負として緩和し，すべての変数組について有望な近傍操作を限定した．

変数 x_j の隣接リストについては，$L^+(j), L^-(j)$ を用意し「内積 $\boldsymbol{a}_{j_1} \cdot \boldsymbol{a}_{j_2}$ の絶対値」が大きい j' たちを内積の正負で分類して格納するような一般化が考えられる．前述の近傍の取捨と隣接リストを組み合わせて用いることで，以下のとおりに 2-flip 近傍を 3 通りのパターンに絞り込んだ．

- $j_1 \in X,\ j_2 \in L^+(j_1) \cap \overline{X}$ について $\downarrow\uparrow$ の 2-flip
- $j_1 \in X,\ j_2 \in L^-(j_1) \cap X$ について $\downarrow\downarrow$ の 2-flip
- $j_1 \in \overline{X},\ j_2 \in L^-(j_1) \cap \overline{X}$ について $\uparrow\uparrow$ の 2-flip

最後に 2-flip 差分式は，式 (3) と同様に 1-flip 差分値を用いた以下のような形式で記述できる．

$$\Delta z^{\downarrow\uparrow}_{j_1,j_2}(\boldsymbol{x}) = \Delta z^{\downarrow}_{j_1}(\boldsymbol{x}) + \Delta z^{\uparrow}_{j_2}(\boldsymbol{x}) + \sum_{i \in S_{j_1,j_2}} p_{i,j_1,j_2}(\boldsymbol{x})(w_i^+ + w_i^-) \tag{4}$$

ここで $p_{i,j_1,j_2}(\boldsymbol{x})$ は a_{ij_1} と a_{ij_2} の符号の組合せにより定義が異なり，例えば $a_{ij_1}, a_{ij_2} \geq 0$ の場合は次式となる．

$$p_{i,j_1,j_2}(\boldsymbol{x}) = -|\min\{a_{ij_1} - y_i^+, a_{ij_2} - y_i^-\}|_+$$

また，$S_{j,j'}$ は「A の j 列目と j' 列目がいずれも 0 でない行 i の集合」と定義し直せばよい．実装上式 (4) の計算効率は式 (3) と比べ劣るが，係数が 0-1 となる行 i については従来の方法で計算を行うため性能が維持される．

仮定 2:整数変数への対応．「$l_j, u_j \in \mathbb{Z},\ l_j < u_j$」として仮定を緩めることを考える．アルゴリズムへの影響が生じるのは局所探索における近傍操作である．そこで k-flip 近傍を「k 個の変数を $+1$ または -1 する近傍」と解釈すれば，概念として自然に拡張できる．また目的関数と制約式は線形であるため，差分計算も同様に定義できる．

以上の修正と改良手法 [2] を併せ，Numerical Optimizer V23 にて実装した．数値実験の結果については口頭発表にて報告する．

参考文献

[1] S. Umetani. Exploiting variable associations to configure efficient local search algorithms in large-scale binary integer programs. *Eur. J. Oper. Res.*, Vol. 263, pp. 72–81, 2017.

[2] 中村健吾, 藤井浩一, 石橋保身, 神谷俊介, 梅谷俊治. 集合分割問題に対する重みつき局所探索法の改良. RIMS 研究集会「数理最適化の理論・アルゴリズム・応用」, 2020.