「『特集名』特集号」 解　説

数理計画ソルバーを用いたメタ解法

久保 幹雄[†]・村岡 秀紀[†]

1. はじめに

最適化関連の実際問題を短時間で解決して欲しいと依頼されたときに，数理計画ソルバーを用いるか，メタ解法を設計するかは分かれ目だ．前者は，問題のサイズが大きくなると解けなくなることがあるし，後者はアルゴリズムを一から設計する必要があるので面倒だ．さらに言うと，後者は実務家への技術移転が難しい．両者の弱点を補うために，この 2 つを合わせたアプローチを使うという手がある．ここでは，そういった手法を幾つか紹介する．

多くの実際問題は，混合整数計画（Mixed Integer Programming: MIP）問題として定式化される．通常，実際問題は，モデリング言語を用いてモデル化され，MIP ソルバーで求解できる．この意味で，MIP ソルバー＋モデリング言語の組合せは，実務的な意味での汎用性が高く，ほとんどの実際問題が（小規模ならば）求解可能である．しかし，問題の規模が大きくなってくると，市販の MIP ソルバーではすぐに限界がきてしまう．これは，多くの MIP ソルバーが分枝限定法とよばれる列挙法を基礎にしていること，ならびに実務上のほとんどの問題が $\mathcal{NP}$-困難族の中でも特に難しいものであることに起因する．(定式化できることと解けることは全く別物なのだ．)

一方，対象とする問題に特化した解法（特にメタ解法）を設計すれば，大規模な問題でも高速に良好な近似解を得ることができる．メタ解法は，比較的コーディングの負担が少ない最適化手法であるが，問題の構造を利用して正しい設計を行わないと十分な性能が出ないという特徴をもつ．また，実際問題においては，プロジェクトの進捗とともに，対象とする問題が徐々に変化していくのが通例である．この場合，問題に特化したメタ解法は，解法の設計を一からやり直さなければならなくなる可能性が出てくる．そうならないように幅広い問題を扱え，また容易に拡張できるようにしておくことが，実務的なメタ解法の設計にとって重要なことであるが，一般にはこれを行うには高度な技術（職人技）を要する．仮に，職人が高性能のメタ解法を設計したとしても，プログラムを作成した本人でないと修正が難しいため，長期のメンテナンスを必要とするプロジェクトにおいては，メタ解法の適用を躊躇する実務家が多い．

実際問題に最適化を適用する際の，典型的な流れは以下のようになる．

(1) 実際問題からモデルを抽出し，モデリング言語を用いて記述する．
(2) MIP ソルバーにかける．ここで解ければ終了．解けないときには，諦めるか，より求解能力の高いソルバーに買い換えるか，強い（良い下界もしくは上界を算出する）定式化に変えるか，パラメータチューニングを行うか，分枝切除価格法などを実装するか，Lagrange緩和などの構造を利用した解法を（MIP ソルバーを部品として）実装する．いずれにせよ，これらの作業は（諦めることとソルバーの買い換え以外は）実務家の手には余ることが多い．
(3) それでも駄目なときには，問題に特化した近似解法を設計する．しかし，上手にメタ解法を設計することも，しばしば実務家の手に余る．また，この場合には今まで作成してきたモデリング言語を経由した工夫はすべて水の泡になる．

喩えて言うなら，MIP ソルバーは万能ナイフであり，メタ解法は切る対象ごとに特化された日本刀や中華包丁や彫刻刀のようなものである．木像は万能ナイフでも彫ることができるが，彫刻刀を使った方が綺麗にかつ早く仕上がる．この 2 つの最適化技術はほとんど独立のグループによって研究されており，対象とする問題が同一であるにもかかわらず，数理計画とメタ解法の接点は少ない．数理計画の技術（たとえば問題固有の多面体構造の解析による切除平面の導出）は，メタ解法の設計にはほとんど影響を与えないし，逆に，高性能のメタ解法の開発は分枝限定法の初期上界（もしくは下界）や固定テストに使えるくらいである．

ここでは，上で述べたような実務と理論の乖離を埋め，かつ数理計画アプローチとメタ解法を自然に融合するために，数理計画に基づくモデリングの工夫の蓄積をメタ解法にフルに利用できるような枠組みを考える．このアプローチは，単純性，頑強性，拡張の容易さ，実装の容

* 東京海洋大学　海洋工学部 流通情報工学科 Department of Logistics and Information Engineering: Tokyo University of Marine Science and Technology 2-1-6 Etsuujima, Koutou, Tokyo JAPAN 135-8533, JAPAN
Key Words: mixed integer programming, metaheuristics.

易さが特徴である．まず，モデリング言語で定式化された問題を入力とするので，モデルが多少変更されてもプログラムを修正する必要はほとんどない．また，最も実装が困難な最適化の部分にはMIP ソルバーを利用するので，実装は極めて容易で単純である．

ここで紹介する手法の弱点としては，性能（解の良さと計算速度）が特定のベンチマーク問題を解くために一から設計されたメタ解法と比べると若干劣ることがあげられる．また，使われているアイディアも，従来の様々な近似解法やメタ解法の研究の中で示唆されてきたものであるので新規性も乏しい．しかし，数多く提案されてきたアイディアの中から，MIP ソルバーをサブルーチンとして容易に設計できるものを選択して，汎用のアイディアとした点が特徴である．また，通常のメタ解法は，組合せ的な構造だけを有する整数計画問題に対しては極めて有効であるが，実数変数を一部含むMIP 問題に対しては適用が難しいことも，ここでMIP ソルバーを基礎としたメタ解法を推奨する理由の 1 つである．

一般に，メタ解法は大きく構築法と改善法に分けられる．以下で紹介する変数固定法（2. 節），緩和固定法（3. 節），容量スケーリング法（4. 節）は構築法の範疇に含まれ，MIP 近傍局所探索法（5. 節），局所分枝法（6. 節）は改善法に含まれる．もちろん，実務的には，構築法によって生成された初期解に対して改善法を適用するので，これらの解法はセットとして適用される．その際には，MIP 併合法（7. 節）が役に立つ．

2. 変数固定法

MIP ソルバーを基礎とした近似解法としては，線形計画緩和問題の解をもとにして，変数を固定する方法が，古くから用いられている．以下の 0-1 変数[1] y と実数変数 x から成る MIP 問題を例として解説する．

$$\begin{aligned} &\text{minimize} \quad fx+gy \\ &\text{subject to} \ Ax+By \leq b \\ &\qquad\qquad x \in \mathrm{R}^m, y \in \{0,1\}^n \end{aligned}$$

0-1 変数 y の添え字集合を N とし，アルゴリズムの途中で 1 に固定された変数の添え字集合を N_1，0 に固定された変数の添え字集合を N_0 と記す．まず，一般的な枠組みで変数固定法を記述する．

（変数固定法：一般形）

(1) N_1, N_0 を空集合とし，固定されていない変数の添え字集合 J を N に初期設定する．

(2) $J=\emptyset$ になるまで，以下の操作を繰り返す．

(a) J から適当な部分集合 J' を選択する．

(b) 各 $j \in J'$ に対して，変数 y_j を 1 に固定するのか，0 に固定するのかを，適当な方法（たとえば線形計画緩和解を用いて）決める．0 に固定する場合には $y_j=0$ の制約を付加し，1 に固定する場合には $y_j=1$ の制約を付加する．

(c) 制約を付加した線形計画問題（もしくはMIP 問題）を解く．

(d) 線形計画問題（もしくはMIP 問題）が実行可能であれば，$J := J \setminus J'$ とする．実行不能であれば，追加した制約を外し，バックトラックする．

実務家が用いる最も単純な MIP ソルバーを利用した近似解法は，MIP ソルバーの探索を途中で打ち切り，それまでに得た最良の実行可能解を近似解として用いる方法である．この方法は，打ち切り分枝限定法（truncated branch-and-bound method）とよばれ，上の枠組みでは，選択する変数の集合 J' の位数が 1 の場合に相当する．

変数固定法の中では，以下の方法が最も単純であるが，問題によっては有効である．

（単純丸め法）

線形計画緩和問題の解を $\bar{y}$ とする．$\bar{y}_j$ が 1 に近い変数は 1 に固定し，0 に近い変数は 0 に固定する．変数を固定することによって縮小された問題に対する最適解を MIP ソルバーを用いて求め，それを近似解として出力する．

また，線形計画緩和問題の解 $\bar{y}$ を用いて，確率 $\bar{y}_j$ で 1 に固定し，それ以外のとき 0 にする確率的丸め法や，一部の変数を固定した後，再び線形計画問題を解くことによって新たな緩和解を得て，それをもとに変数の固定を行う連続丸め法も有効である．

いずれの丸め法を用いるにせよ，上の方法では，困難な MIP 問題に対する良好な近似解を得ることは難しいが，改善法の初期解生成や，実際問題を解く際の第一刀としての価値はあると思われる．実際に，連続丸め法に確率的な丸めを加味し，さらに適当な改善法を組み込んだメタ解法は，貪欲ランダム適応型探索法（Greedy Randomized Adaptive Search Procedure: GRASP）とよばれ，一般の MIP 問題に対してある程度の成果をあげている．

3. 緩和固定法

緩和固定法（relax and fix method）とは，上で紹介した変数固定法において，固定する変数を決める部分に，MIP を用いた構築法である．

まず，一般の MIP 問題で緩和固定法を解説する．

MIP 問題に含まれる整数変数を，自由変数，固定変数，連続緩和変数の 3 つに分けて考える．自由変数はオリジナルの問題と同じ意味をもつ整数変数であり，すべての変数が自由変数である問題は，元の問題に他ならない．固定変数は，問題の規模を縮小するために一時的に固定された変数である．連続緩和変数は，実数変数として緩和された変数であり，すべての整数変数を連続緩和

[1] 正確にはベクトルであるが，以下では次元は適当に読み替えるものとする．

変数にした問題は，線形計画緩和問題に他ならない．

まず，固定変数がない状態からはじめ，一部の変数を自由変数とし，残りの変数を連続緩和変数として MIP ソルバーで最適化する．求まった解の自由変数のすべて，もしくは一部分を得られた最適解に固定し固定変数とする．この操作をすべての変数が固定変数になるまで繰り返す．

ここで重要になるのは，自由変数の選択方法とその順序である．順序は，重要な（言い換えれば目的関数やボトルネック制約に与える影響が大きい）変数から順番に自由変数とする方法が有効である．また，自由変数の選択法には，以下の 2 点を考慮する必要がある．

(1) 解を構成するにあたって，互いに密接な関係があり，同時に決定することが必要であると考えられる変数を，重要な（目的関数や制約条件に与える影響が大きい）順に自由変数を選択する．
(2) 選択された自由変数の個数がそれほど多くなく，MIP ソルバー（に内蔵されている分枝限定法）である程度高速に求解できる．

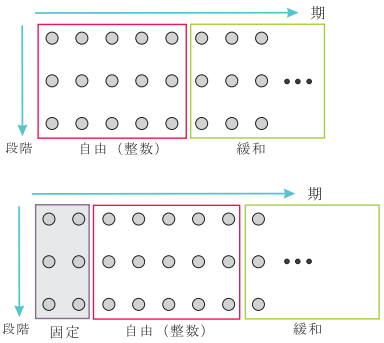
変数の重要性，変数間の関係や分枝限定法で求解可能な規模は，問題依存であるので，対象とする MIP 問題のクラスに応じて指針をもっておく必要がある．

ここでは，緩和固定法の適用例として，ロットサイズ決定問題を考える．ロットサイズ決定問題とは，多期間に跨る生産計画問題であり，各期における段取り替えの有無が 0-1 変数として表された MIP 問題として定式化される [5]．

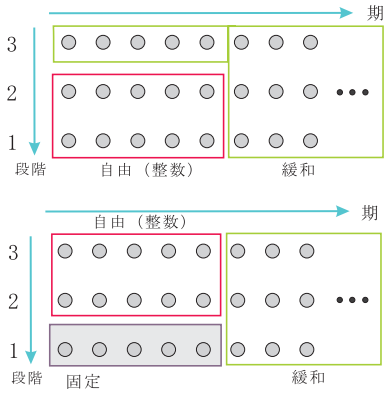
いま，小規模な（期の数が小さい）ロットサイズ決定問題が MIP ソルバーにかけることによって，短時間で求解可能であるとする．ここで，求解可能とは，良好な近似解を短時間で得ることができると読み替えても良い．この小規模問題を逐次最適化することによって，全体の近似解を得ようというのが，緩和固定法の基本的なアイディアである．

ロットサイズ決定問題に対しては，期が近い変数同士は互いに密接な関係があると考えられる．また，近い未来の意思決定が重要であると考えられるので，期の番号の小さい順に自由変数としていく方法が有効であると推測される．上の理由により，自由変数の選択法としては，連続する期内の段取りに関する変数を一度に決めるものとする（図 1）．

たとえば，1 期から 5 期までの段取りを自由変数，残りの期の段取りを連続緩和して MIP ソルバーで最適化した後で，1 期から 2 期を固定し，今度は 3 期から 8 期までを自由変数として再び最適化するといった具合である．これを最後の期までの段取りが固定されるまで繰り返す．ここで，一度に最適化を行う期間（上の例では 5 期）が探索の広さと速度を調節するためのパラメータであり，最適化された自由変数をどこまで固定するか（上の例では 2 期）が探索のきめ細かさ（刻み幅）を制御するためのパラメータになる．



第 1 図　緩和固定法の適用例（探索の広さは 5 期，探索の刻み幅は 2 期の場合）．上が最初の反復における変数の固定・緩和法で，1 期から 5 期までを自由変数，6 期以降を連続緩和変数として最適化する．下が次の反復における変数の固定・緩和法で，上の問題を解いて得られた最適解の 1 期と 2 期の分は固定し，3 期から 7 期までを自由変数，8 期以降を連続緩和変数として最適化する．



第 2 図　3 段階のロットサイズ決定問題に対する緩和固定法の適用例．上が最初の反復における変数の固定・緩和法で，1 期から 5 期までの 1 段目と 2 段目を自由変数，1 期から 5 期までの 3 段目と 6 期以降のすべてを連続緩和変数として最適化する．下が次の反復における変数の固定・緩和法で，上の問題を解いて得られた最適解における 1 期から 5 期までの 1 段目の分は固定し，2 段目と 3 段目を自由変数，6 期以降を連続緩和変数として最適化する．この後は，通常の緩和固定法と同様に，期を後ろにずらしながら，同様の操作を繰り返す．

機械（工程）ごとに全体に与える影響が大きく異なるときには，重要な機械（工程）から順に自由変数として最適化する手法も有効である．これは，ジョブショップスケジューリング問題に対するボトルネックずらし法（shifting bottleneck method）[1] と同じ理由による．多段階の工程をもつロットサイズ決定問題に対しては，需要側（サプライ・チェインの下流側）の変数が重要であると考えられるので，需要側（下流）の段階から順に自由変数にしていく方法も有効である（図 2）．

4. 容量スケーリング法

ここで紹介する容量スケーリング法（capacity scaling method）は，固定費用付きの問題の特殊構造を生かしたものであり，様々な固定費用付きの最適化問題に使うことができる汎用ヒューリスティクスである．ロットサイズ決定問題も固定費用付きの最小費用流問題と考えることができ，ある程度良い近似解を短時間で算出することが知られている [4].

以下の一般の固定費用付きの MIP 問題を例として解説する．

$$\begin{aligned} \text{minimize} \quad & vx+Fy \\ \text{subject to} \quad & x \le Cy \\ & Ax+By \le b \\ & x \in \mathrm{R}^m, y \in \{0,1\}^n \end{aligned}$$

ここで，v は変動費用，F は固定費用を表すパラメータである．

（容量スケーリング法）

(1) 平滑化パラメータ $\lambda \in (0,1]$ を決め，変化させる容量を表すパラメータ C' をオリジナルの容量 C に初期設定する．

(2) 以下の操作を適当な収束判定基準を満たすまで（たとえば，線形計画緩和問題の解 $\bar{y}$ が 0 もしくは 1 に十分に近づくまで）繰り返す．

(a) 0-1 変数 y_j を $[0, C_j/C'_j]$ に緩和し，制約を $x_j \le C' y_j$ に変更した線形計画緩和問題を解く．これは，固定費用 F_j を容量 C'_j で除した値を変動費用 v_j に加えた目的関数をもち，容量制約を $x_j \le C_j$ とした線形計画問題を解くことと同値である．

(b) 得られた線形計画緩和問題の解 $\bar{x}, \bar{y}$ においては，$\bar{x} = C'\bar{y}$ となっているので，新しい容量 C' を，平滑化パラメータ λ を用いて，現在の C' と $C'\bar{y}$ の間になるように

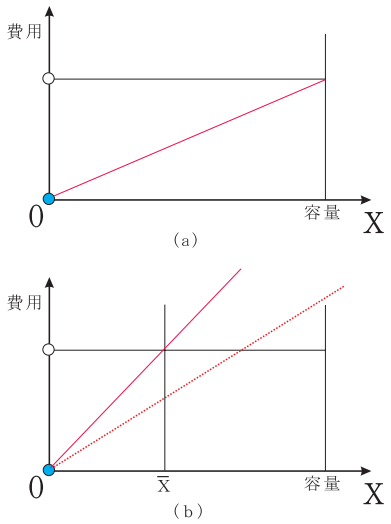
$$C' := \lambda C'\bar{y} + (1-\lambda)C'$$

と変更する（図 3）．

容量スケーリング法は高速であり，かつ得られた解は局所最適解であるので，局所最適解からの脱出を図るための工夫を取り入れることが望ましい．脱出のためのテクニックとしては，目的関数に長期メモリのペナルティを追加する誘導局所探索法や，現在の解と一部が異なることを表す制約を追加する方法が有効であると考えられる．

5. MIP 近傍局所探索法

何らかの構築法によって得られた解を改善する方法として，MIP 問題をサブルーチンとした大近傍局所探索法（MIP 近傍局所探索法）が考えられる．



第 3 図 容量スケーリング法の概念図．(a) 最初の線形計画緩和．固定費用を容量で除した値を傾き（変動費用）に加えた問題を解く．(b) 緩和問題の解 $\bar{x}$ に対する容量の変更．実線は新しい容量を $\bar{x}$ とした場合．点線はもとの容量と $\bar{x}$ の間になるようにした場合．

一般の MIP 問題を考え，MIP 問題に含まれる整数変数を，自由変数，固定変数の 2 つに分けて考える．自由変数はオリジナルの問題と同じ意味をもつ整数変数であり，固定変数は，問題の規模を縮小するために一時的に固定された変数である．MIP 近傍局所探索法では，与えられた実行可能解を改善するために，一部の整数変数を自由変数として MIP 問題を最適化する．ここで重要になるのは，（緩和固定法と同様に）自由変数の選択方法である．

ロットサイズ決定問題に対しては，期が近い変数同士は互いに密接な関係があると考えられるので，近い期に関連する整数変数を自由変数，他の整数変数を固定変数として，MIP ソルバーを用いて最適化する．たとえば，1 期から 5 期までの段取りを自由変数とし，残りの期の段取りを固定して最適化する．ここで，一度に最適化を行う期間（上の例では 5 期）が探索の広さと速度を調節するためのパラメータとなる．これを，開始する期を $2,3,\cdots$ と変化させて，改善解がみつかったら，その解を新たな出発点として，再び探索を行う．

配送計画問題に対する MIT 近傍局所探索法も提案されている [2]．配送計画とは，複数の運搬車による顧客の巡回路（ルート）を決定する問題であり，通常は運搬車の荷量制約や顧客上への到着時刻に対する時間枠制約など，種々の不可条件が追加される．近傍の探索は，現在のルートからランダムに顧客を除き，それらをルートに再挿入することを表す MIP 問題を，MIP ソルバーで最適化することによって成される．

実際には，MIP ソルバーを用いた最適化には計算時間がかかる可能性があるので，近傍の計算時間を打ち切っ

て得られた最良解で評価する方法や，一度探索して改善解がみつからなかった近傍は，次回以降の探索から除外するなどの工夫が必要になる．

6. 局所分枝法

ここでは，2. 節で用いた一般の 0-1 MIP 問題を例として，変数固定法の拡張である局所分枝法 (local branching method) [3] を紹介する．局所分枝法の基礎になるアイディアは，変数を直に固定するのではなく，制約として「ソフト」に固定することである．

いま，何らかの実行可能解が与えられているものとし，それを参照解とよび $\tilde{y}$ と記す．0-1 変数の添え字集合 N のうち $\tilde{y}_j$ が 1 の添え字集合を N_1，$\tilde{y}_j$ が 0 の添え字集合を N_0 と記す．

局所分枝法では，参照解 $\tilde{y}$ と任意の解 y の間の距離

$$\Delta(y,\tilde{y}) = \sum_{j\in N_1}(1-y_j) + \sum_{j\in N_0} y_j$$

を用い，参照解との距離が k 以下であることを表す制約

$$\Delta(y,\tilde{y}) \le k \tag{1}$$

を付け加えることによって,「ソフト」に変数の固定を行う．ここで，k は近傍の広さを制御するためのパラメータである．

集合 N_1 の位数がほぼ一定である場合には， 制約 (1) のかわりに，$j\in N_1$ である変数 y_j の内の高々 k 個だけ 0 に変えることを許す制約

$$\sum_{j\in N_1} y_j \ge |N_1| - k$$

を加えても良い．

また，現在の上界 $\tilde{Z}$ 以下の解を探索するために，以下の制約を付加することによって，さらに探索範囲を限定する．

$$\sum_{j=1}^{m} f_j x_j + \sum_{j\in N} g_j y_j \le \tilde{Z}$$

目的関数値が必ず整数値をとることが分かっている場合（たとえば実数変数 x がない場合）には，上界未満の解を探索するために，以下の制約を付加しても良い．

$$\sum_{j\in N} g_j y_j \le \tilde{Z} - 1$$

これらの制約を付加することによって，分枝限定法の探索範囲が限定されるので，MIP ソルバーによる解の探索は高速に完了することが期待されるが，通常は探索時間（もしくは分枝子問題数）に上限を設けて，MIP ソルバーをよぶ．このとき，MIP ソルバーによる探索の結果によって，以下の 4 通りに分けて考える．

最適解を算出した場合： 制約 (1) を付加した問題に対する最適解 $\tilde{y}$ が得られたので，今後の探索からこの制約を満たす解を除くために，以下の測深制約を追加する．

$$\Delta(y,\tilde{y}) \ge k+1$$

もしくは

$$\sum_{j\in N_1} y_j \le |N_1| - k - 1$$

その後，得られた最適解を新たな参照解として探索を続ける．

（最適の保証はないが）実行可能解を返した場合： 得られた実行可能解を参照解 $\tilde{y}$ として探索を続ける．ただし，同じ解を探索しないようにするため，以下の禁断（tabu）制約を付加する．

$$\Delta(y,\tilde{y}) \ge 1$$

もしくは

$$\sum_{j\in N_1} y_j \le |N_1| - 1$$

実行不能であることが証明された場合： 現在の参照解の近傍には良い解はないことが保証されたので，最適解を算出した場合と同様に，測深制約を追加する．その後，別の参照解を得るために,（後述する）多様化探索に移行する．

制限時間内では実行可能解が得られなかった場合： この場合には，2 通りの方法が考えられる．1 つは，近傍を小さくして再び同じ参照解の周りを探索する方法である．もう 1 つは，実行可能解を返した場合と同様に，禁断制約を付加し，その後，別の参照解を得るために,（後述する）多様化探索に移行する方法である．

上の幾つかのケースでは，現在の参照解の近傍には良い解がないので，新たな参照解を求める必要があった．これを行う方法を総称して多様化探索とよぶ．多様化探索には種々の方法が考えられるが，単純なものとしては，近傍を大きくする（パラメータ k を増やす）ことがあげられる．他の方法として，上界以下（未満）の解に限定するための制約を除いて（改悪を許して）MIP ソルバーをよび，最初に見つかった実行可能解を参照解とする方法も有効である．

7. MIP 併合法

最近では，適応型局所探索法や散布探索法に代表されるように，複数の異なる解を保持して，その情報をもとに新たな初期解を生成し，それに対して何らかの改善法を適用するメタ解法が主流である．これらのメタ解法では，複数の相異なる解から新しい初期解を生成する部分が問題になるが，この部分に MIP ソルバーを用いる方

法が MIP 併合法である．

現在までに得られた複数の解に含まれる（$y_j = 1$ になっている）変数だけを自由変数とし，再び MIP ソルバーで解くことによって，それぞれの解の良い部分を部品としたさらに良い解を得ることが期待される．また，現在保持する解とは異なる解を生成するためには，局所分枝法と同様に禁断制約を追加しておけば良い．このように，MIP 併合法を用いることによって，良いメタ解法を設計するための基本原則である集中化と多様化を，MIP ベースのメタ解法に比較的容易に組み込むことができる．これは，遺伝的アルゴリズムにおける交叉（2 つの相異なる解から新しい解を生成する操作）を一般化したものであり，かつ生成される解が実行不能になること（致死遺伝子）を回避したものと解釈できる．

配送計画に対しては，探索の途中で求めた複数の良いルートを保管し，それらのルートから顧客をちょうど 1 回だけ通過するようにルートを再構成する方法がしばしば有効である．これは，MIP 併合法の一種であると考えられる．また，MIP 近傍局所探索法における顧客の再挿入と，MIP 併合法におけるルートの選択を同時に解くタイプのメタ解法も考えられる．

8. 並列化

MIP ソルバーを用いたメタ解法が，他のメタ解法と根本的に異なることは，MIP 自身が比較的重い計算である点である．通常のメタ解法では，近傍を評価するための計算時間は低次の多項式オーダーに抑えられるので，この部分を並列化してもその恩恵は僅かである．一方．MIP を用いた近傍評価は，通常大量の計算時間とメモリを要するため，この部分を並列計算機で処理することによる恩恵は，通常のメタ解法より大きい．

9. おわりに

本稿では，MIP ソルバーを用いたメタ解法を幾つか紹介した．単なるベンチマーク問題の解決だけではなく，実務に耐えうるメタ解法を設計するためには，汎用と特化のバランスを吟味する必要がある．汎用の解法は幅広い問題に適用可能であるが，一般に性能は劣る．逆に，特化した解法は性能は優れている場合が多いが，拡張性に欠ける．

変数固定法や局所分枝法は，一般の MIP 問題を対象としたものであるので，汎用の MIP ベースのメタ解法である．緩和固定法や MIP 近傍局所探索法は，問題に特化した構造を利用することによって，性能を向上されることができる．容量スケーリング法は，固定費用付きの問題の構造を利用するので，適用可能な問題が限定されている．

要は，万能薬はなく，対象とする応用のニーズに（どの程度の汎用性を求めるのか，どの程度の性能を求めるのか）に応じて，適用する手法を決める必要があるのである．今後，ここで解説した手法が，実際問題の解決に少しでも役に立てば嬉しい．

参 考 文 献

[1] J. Adams, E. Balas, and D. Zawack. The shifting bottleneck procedure for job shop scheduling problem. *Management Science*, 34:391–401, 1988.

[2] R. de Franceschi, M. Fischetti, and P. Toth. A new ILP-based refinement heuristic for vehicle routing problems. *Mathematical Programming*, 105:471–499, 2006.

[3] M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–47, 2003.

[4] Y. Pochet and M. Van Vyve. A general heuristic for production planningproblems. *INFORMS Journal on Computing*, 16:316–327, 2004.

[5] 久保幹雄. **ロジスティクス工学**. 朝倉書店, 2001.