数理最適化とメタヒューリスティクス

久保　幹雄

ここでは，最適化問題を解くための 2 つの主なアプローチである数理最適化（数理計画）と（メタ）ヒューリスティクスの使い分けと，これらを用いて実際問題を解決する際の手順について論じる．また，代表的な最適化問題に対する実験的解析に基づいて，アプローチの選択法と限界について考える．

キーワード：数理最適化，メタヒューリスティクス，制約最適化，スケジューリング，実験的解析

1. はじめに

最近，メタヒューリスティクスと数理最適化（数理計画）についての本を 1 冊ずつ書いた．1 つは「メタヒューリスティクスの数理」[3]，もう 1 つは「あたらしい数理最適化―Python 言語と Gurobi で解く―」[4] だ．両者とも共通の最適化問題を扱っているため，一部の読者から「では，どちらを使ったらよいの？」と聞かれることがままある．両者とも Python 言語[1] によるソースコードがサポートページ[2] で開示されている．自分で実験をして比較することもできるので，「自分の解きたい問題と規模に応じて，両方とも試したうえで，自分で選んでください」というのが一番簡単な回答である．本稿は，それでも「試すのが面倒だからおおよその指針がほしい」とリクエストしてきた人たちのためのガイドである．

2. 解法の分類

本題に入る前に，本稿で用いる用語の整理をしておこう．対象とするのは離散（整数）変数を含んだ最適化問題である．（離散変数を含まない）連続最適化は比較的簡単なので，（二次錐最適化については 4.8 項で簡単に触れるが）本稿の対象外である．

離散最適化問題に対する解法は大きく 2 種類に分類される．1 つは最適性の保証を持った厳密解法であり，もう 1 つは最適性の保証を持たない近似解法である．おのおのに対して，ほかの人が作成した汎用ソルバーを用いるのか，それとも解法を自分で構築するかの 2 つの選択肢がある．

厳密解法を搭載した汎用ソルバーの代表例が混合整数計画 (MIP: Mixed Integer Programming) ソルバーである．「あたらしい数理最適化」では，最速の MIP ソルバーである Gurobi[3] を用いて解説しているが，SCIP[4]，Cbc[5]，GLPK[6] などのオープンソースの MIP ソルバーも選択肢である．MIP ソルバーで採用されている解法は分枝限定法（より正確に言うと分枝カット法）である．実際問題を解く際に，実務家がこれを一から構築することは推奨できない．厳密解法を自分で構築する場合はごく限られてくるが，動的計画に基づく解法が適用できる場合には，しばしば有効である．動的計画は Bellman 方程式の記述ができれば，プログラムは非常に短く書けるので，短時間で開発できるからだ．筆者は，鉄鋼業における実務的なスケジューリング問題（実は非対称な巡回セールスマン問題に帰着できる）を依頼されたときに，動的計画に基づいた 30 行程度の

くぼ　みきお
東京海洋大学大学院海洋工学系
〒 135–8533 東京都江東区越中島 2–1–6

[1] 初学者に優しい超高水準プログラミング言語．ほかのプログラミング言語と較べて簡潔で可読性が高いので，拙著 [3, 4] で採用した．Python についての詳細は http://www.python.jp/ 参照．

[2]「あたらしい数理最適化」のサポートページは http://www.logopt.com/book/gurobi.htm，「メタヒューリスティクスの数理」のサポートページは http://www.logopt.com/mikiokubo/programming.htm．

[3] Zonghao Gu, Edward Rothberg, Robert Bixby によって開発された MIP ソルバー．http://www.gurobi.com/ 参照．日本における商用ライセンスについては，オクトーバースカイ社 http://www.octobersky.jp/ 参照．

[4] Solving Constraint Integer Programs の略．ZIB Academic License であり非商用は無償．http://scip.zib.de/ 参照．

[5] Coin-or branch and cut の略．Eclipse Public License であり商用・非商用にかかわらず無償．https://projects.coin-or.org/Cbc 参照．

[6] GNU Linear Programming Kit の略．GNU General Public License であり商用・非商用にかかわらず無償であるが，コピーレフト（二次的著作物の頒布条件を同一のライセンスに限る）の条件が付いている．http://www.gnu.org/software/glpk/ 参照．

C 言語のソースコードを提供したことがある．実際にその解法は，最近聞いたところによると 20 年近く現場で動いているそうだ．

近似解法を用いた汎用ソルバーの多くはメタヒューリスティクスを用いている．「あたらしい数理最適化」では，メタヒューリスティクスに基づく SCOP[7] と OptSeq[8] について解説している．SCOP は制約最適化を対象としたものであり，OptSeq はスケジューリング最適化を対象としたものである．両者とも Nonobe–Ibaraki [1, 2] によって開発されたものに Python によるインターフェイスを付加したものである．メタヒューリスティクス，もしくはより単純なヒューリスティクスを自分で構築することももちろん可能である．ただし，メタヒューリスティクスは問題の構造をうまく利用して設計しないと，十分な性能が出ない場合もある．特にメタヒューリスティクスのアナロジー面だけに囚われ，データ構造や計算時間短縮の工夫を疎かにしたメタヒューリスティクスの実装は，単純な局所探索にも及ばないこともあるので注意を要する．

3. 実際問題解決の手順

実際問題を依頼されたときに，どのようなアプローチを選択するかはアートである．ここでは，筆者の経験に基づくおおまかな指針を与える．

どのアプローチを選ぶかは，問題の難しさと規模に依存する．複雑で大規模な実際問題を解く際に，いきなり MIP ソルバーでモデル化を始める実務家をみかけるが，これは危険である．MIP ソルバーは厳密解法に基づくものなので，問題例（問題に数値を入れたもの）の規模がある程度大きくなると急激に計算量が増大するという「性質」を持つ．実務の問題はほとんどが $\mathcal{NP}$-困難であり，厳密解法を追求する限り，これは避けられない．いかに計算機が速くなろうが，ソルバーの性能が上がろうが，これは宿命なのである．

どのくらいの規模の問題例で計算量の増大が発生するかの予測は難しい．同じ問題でも，問題例の構造によって解ける規模が全く異なるからだ．以下の節で，典型的な問題に対して解きうる規模の指針を与えるので，実際問題がどの典型問題に似ているかによってある程度の判断はできるが，やはりやってみなければわからないというのが現実である．

したがって，大規模な問題に対して MIP ソルバーを用いる際には，問題を小さく分割して適用する必要がある．どのように問題を分割するかもアートであるが，（MIP ソルバーを用いる場合には）分解された問題が（想定する規模のどのような問題例を与えたとしても）容易に解けるというのが基準になる．

実際問題を実務で運用している場合には，人間が意思決定できる範囲に分割して扱っている場合が多い．したがって，実務家への十分なヒヤリングによって問題の構造を十分に理解するとともに，分割された子問題が最適化で解決することができるトレードオフを含むように問題を分割することが望ましい．また多くの実際問題は，意思決定レベルに分けて考えると整理しやすいので，扱う時間軸によって長期（ストラテジック；年次以上），中期（タクティカル；月次），短期（日次以下）で分けてモデル化する方法も有効である．ただし，異なる意思決定レベルの間でどのような情報をやりとりさせて全体最適化を目指すかを十分に考える必要がある．

メタヒューリスティクスを用いる場合も同様である．解きうる規模は MIP ソルバーよりは大きいが，あまり大規模な問題例をそのまま解いても，出てくる解の質は保証できない．一からメタヒューリスティクスを構築するか，汎用ソルバーを使うかも分かれ目であるが，開発にかけることができる時間が短い場合には，汎用ソルバーを選択したほうがよい．アルゴリズムをチューニングする手間が省けるからである．

なお，汎用ソルバーの選択には，ある程度の定石があるので付記しておく．ほとんどの変数が実数（連続）変数で，双対ギャップ（上界と下界の差）が比較的小さいと推測されるタイプの混合整数計画問題は，MIP ソルバー（たとえば Gurobi）を使う．連続変数がなく，組合せ的な構造のみの離散最適化問題で，かつ近似でもよいならメタヒューリスティクスに基づく制約最適化ソルバー（たとえば SCOP）を使う．特に，制約最適化ソルバーは割り当てタイプの問題や非凸の二次項を含む問題に強い．一方，順序付けを主目的としたスケジューリングタイプの問題に対しては，メタヒューリスティクスに基づくスケジューリング最適化ソルバー（たとえば OptSeq）が有効となる．

なお，問題のヒヤリングの前にソルバーを選定するようなことは，決してしてはならない．これは，自分の専門分野を実務で使いたい研究者，もしくは自社でソルバーを販売している実務家にありがちな落とし穴であり，「我田引水シンドローム」と呼ばれ，最適化を用いたプロジェクトが失敗する要因の 1 つである．

[7] `http://www.logopt.com/scop.htm` 参照．
[8] `http://www.logopt.com/OptSeq/OptSeq.htm` 参照．

4. 個別問題に対する指針

ここでは，代表的な問題に対する実験的解析に基づくアプローチの選択法と限界について述べる．MIP ソルバーは定式化（数式に記述する仕方）によって性能が異なる場合がある．本稿ではスペースの都合上，定式化ならびに実験的解析の詳細については省略する．定式化の詳細については「あたらしい数理最適化」[4] を，実験的解析の結果については，前述した「あたらしい数理最適化」のサポートページ `http://www.logopt.com/book/gurobi.htm` を参照されたい．また，以下で述べるベンチマーク問題例についての情報も，サポートページに記述してある．

4.1 施設配置問題

「あたらしい数理最適化」で最初の例として取り上げたのは施設配置問題である．施設配置問題は，実務におけるさまざまな応用を持つ．施設の立地の際にかかる固定費と輸送費用のトレードオフを最適化するのが標準の固定費用付き施設配置問題である．類似する問題として，施設の数を与えてその中で輸送費用の合計を最小化する k-median 問題や，顧客までの最大距離を最小化する k-center 問題などが代表的なバリエーションである．われわれは，固定費用付き施設配置問題，k-median 問題，k-center 問題の 3 つのタイプの問題に対して実験を行った．結果から言うと，中規模の問題例までは MIP ソルバーで十分である．min–max 型の目的関数を持つ k-center 問題の場合には，通常の定式化をすると計算時間が急激に増加してしまい 100 点あたりで組合せ爆発が起きる[9]．これを避けるための工夫として，min–sum 型の問題（被覆問題）をサブルーチンとした二分探索などの工夫をする必要がある．こういった工夫をすれば，250 点を超える問題例まで組合せ爆発が起きない．

施設配置問題の実際問題では，さまざまな付加条件が付くので，モデル化して MIP ソルバーで求解するのがお薦めである．この際，需要地点（顧客）が数千から数万のオーダーになる場合には，適当に集約して扱う必要がある．施設配置問題は長期の意思決定モデルであるので，集約せずに個々の需要地点のデータを扱うのはセンスが良くない．これは，「木を見て森を見ないシンドローム」と呼ばれる．常に，大きな問題例を解きたいという要求があるが，適当な近似と集約のセンスは重要であり，集約した後の施設配置問題は，モダンな MIP ソルバーで簡単に最適解を求めることができる．

4.2 グラフ最適化問題

「メタヒューリスティクスの数理」でも「あたらしい数理最適化」でも，導入としてグラフ分割，安定集合問題（最大クリーク問題），グラフ彩色問題の 3 つのグラフ問題を例として扱っている．この 3 つのグラフ最適化は，メタヒューリスティクスの比較の定番である．安定集合問題とグラフ彩色問題は，第 2 回の DIMACS チャレンジ[10]で，グラフ分割問題は第 10 回の DIMACS チャレンジでベンチマーク問題の収集や比較が行われた．

本来ならばベンチマーク問題を用いて実験を行ったほうがよいのだが，MIP ソルバーでは解ける範囲が限定されるので，ランダムグラフを対象として実験を行った．ここで用いるランダムグラフは，点数と枝の発生確率を与えたとき，各（無向）枝を一定の確率で発生させたものである．

グラフ分割問題とは，無向グラフの点集合を 2 つに等分割したとき，分割をまたぐ枝数を最小にする問題である．グラフ分割問題に対しては，線形化を用いた定式化が最も良いが，枝の発生確率を 0.1 とした 70 点のランダムグラフの問題例を解くことができない．グラフが密になると，40 点あたりでさえ限界だ．したがってこのタイプの問題を解く際には，必然的にメタヒューリスティクスに頼ることになる．「メタヒューリスティクスの数理」に記述したお薦めのメタヒューリスティクスは，戦略的振動を用いた禁断探索法であり，これを使えば 1,000 点規模の問題例なら楽々解くことができる（もちろん最適性の保証はない）．より大規模な（数百万点規模の）問題例を解く必要がある場合には，多レベル法と呼ばれる解法が推奨される．多レベル法をさまざまなグラフ分割問題の変形に対して実装したプログラム (METIS)[11] を利用してもよい．

安定集合問題は，与えられた無向グラフの最大の安定集合（点の部分集合で，点間に枝のないもの）を求める問題であり，同じ目的関数値をもつ解がたくさんある可能性が高いという特徴がある．これは平坦部と呼ばれ，問題を（メタヒューリスティクスにせよ数理最適化にせよ）解きにくくする構造である．小規模な

[9] 使用した計算機は Intel Xeon CPU E5-2687W, 3.10 GHz, RAM 64 GB, 言語は Python 2.7 である．以下の実験も同じ環境で行った．

[10] The Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) が主催する，さまざまな問題に対するプログラミングコンテスト．

[11] `http://glaros.dtc.umn.edu/gkhome/`

問題例に対しては，数理最適化ソルバーでも求解が可能である．枝の発生確率が 0.05 のランダムグラフに対しては，点の数が 200 近辺で組合せ爆発が起こる．枝の密度（発生確率）が 0.1 付近で問題例が難しくなり，小さくなると（疎になると）簡単になり，大きくなっても簡単になる．中規模以上の場合には（どうしても厳密解が欲しい場合を除いては）何らかのヒューリスティクスを用いる必要がある．最適解に近い近似解を得たい場合には，「メタヒューリスティクスの数理」で記述した平坦部からの脱出を考慮した解法が推奨される．

グラフ彩色問題は，枝の両端には異なる色を塗らなければならないという制約の下で，与えられたグラフの点に最小数の色を塗る問題である．MIP ソルバーで単純に定式化すると，点の色を入れ替えても同じ解になるので，この問題は対称性を持つ．解の対称性は，MIP ソルバーが基礎とする分枝限定法にとってはやっかいな性質である．対称性を除去するためには，特殊な制約を付加するなどのテクニックが必要となるが，色々な工夫をしたとしても解きにくいという性質には変わりがない．最も良い定式化とアプローチは．4.1 項で述べた k-center 問題に対する二分探索法と同じアプローチであり，彩色数を固定して条件を破っている枝数を最小化する問題を繰り返し解く．それでも，枝の発生確率を 0.5 としたランダムグラフに対しては，60 点あたりで組合せ爆発が起こる．グラフ彩色問題の応用には，時間割作成や周波数割当などがあるが，小規模な問題例以外では，メタヒューリスティクスが推奨される．また，メタヒューリスティクスに基づく汎用の制約最適化ソルバーも，付加条件を加えたときの拡張性やメンテナンスのしやすさの観点から推奨される．

4.3 巡回路問題

巡回セールスマン問題とは，与えられたグラフのすべての点を最短距離で訪問する巡回路を求める問題であり，巡回路型の最適化問題の代表選手である．実験に用いたのは TSPLIB[12] と呼ばれるベンチマーク問題集である．距離が非対称なベンチマーク問題例に対しては，持ち上げ操作によって強化した Miller–Tucker–Zemlin 定式化が最も良く，最大で 1,000 点の問題例まで解くことができた．一般にランダムに生成された非対称巡回セールスマン問題は，比較的容易であり，より大規模な問題例まで解けると思われる． なお，求解の安定性では単品種定式化も推奨される．非対称巡回セールスマン問題に対する局所探索に基づくメタヒューリスティクスは，1,000 点を超える問題例でも比較的良い近似解を算出するので，最適性にこだわらない場合には，選択肢の 1 つとなる．

対称な巡回セールスマン問題に対しては，分枝ノードで整数解が得られたときに部分巡回路制約を追加する分枝カット法が最も良く，1,000 点を超えるようなベンチマーク問題例に対しても最適解を求めることができた．もちろん，小規模でも難しい問題例（たとえば平坦部が多いと予想される ts225）もあり，安定して短時間で求解したい場合には，（メタ）ヒューリスティクスが推奨される．対称巡回セールスマン問題に対するヒューリスティクスは，数多く提案されているが，作りやすさの観点から局所探索に基づく解法が推奨される．このタイプの問題に対しては，（厳密解法でもヒューリスティクスでも）CONCORDE[13] と呼ばれるソルバーが準備されているので，商用利用でなければこれを使うのがよい．

100 万点を超えるような超大規模の問題例に対しては，解法の選択よりもデータ構造が重要になる．地点間の距離を保管するだけでも大量のメモリが必要になるので，最適解に含まれる可能性が低い枝（点のペア）は，除外してデータ構造を組み立て，必要に応じて距離を計算するなどの工夫が必要になる．

時間枠付き巡回セールスマン問題は，各顧客（点）にサービスを受けることができる時間枠（最早時刻と最遅時刻のペア）が付加された問題である．最も良い定式化は，持ち上げ操作によって強化した Miller–Tucker–Zemlin タイプの定式化 [4] であり，時間枠が狭く比較的簡単な Dumas らによるベンチマーク問題なら，150 点の問題例まで解くことができた．一方，時間枠が広い Gendreau らによるベンチマーク問題では，制限時間 1 時間で 40 点の問題例までしか最適解を得ることができなかった．

容量制約付き配送計画問題とは，デポと呼ばれる特別な地点を出発した複数人のセールスマン（運搬車）が各顧客を巡回して荷物を届け，すべての顧客に荷物が届くようにする問題である．目的はすべての運搬車の総移動距離を最小にすることであり，さらに（運搬車の運べる荷物の合計量に限界があることを表す）容量制約が付加されている．この問題に対しても，巡回セールスマン問題と同様に分枝カット法を適用できるが，100 点の問題例を解くのは難しい．実際の配送計

[12] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[13] http://www.tsp.gatech.edu/

画問題では，時間枠などのさまざまな付加条件を考慮する必要がある．制約がきつい場合には，ルートを必要に応じて生成する列生成法が適用可能であるが，実務的には局所探索に基づくメタヒューリスティクスが推奨される．

4.4 スケジューリング問題

スケジューリング問題とは，複数の活動（ジョブ，作業）を時間軸を考慮して資源に割り付ける問題の総称である．一般に MIP ソルバーは，スケジューリング問題に対しては無力である．さまざまなスケジューリング問題に対する実装を行い，実験を行ったが，問題ごとに工夫をした定式化でも解ける問題の規模はそれほど大きくはない．

ランダムに生成した 1 機械リリース時刻付きのメイクスパン最小化問題に対しては，離接定式化は 15 ジョブ程度で，時刻添え字定式化は 40 ジョブ程度で組合せ爆発を起こす．一方，1 機械納期遅れ最小化問題に特化した線形順序付け定式化は，200 ジョブまでなら最適解を得ることができた．ほかのスケジューリング問題に対しても，現状で考え得る最良の定式化を試したが，結果はあまり良くない．したがって，実務におけるスケジューリング問題を安定して求解するためには，現状では MIP ソルバーは推奨できない．スケジューリング問題に対しては，問題の構造を利用した専用メタヒューリスティクスか，汎用のスケジューリング最適化ソルバーがよい．

4.5 ロットサイズ決定問題

ロットサイズ決定問題は，多期間の生産計画における段取りと生産量の意思決定を行うためのモデルである．生産量を実数変数として扱う場合には，メタヒューリスティクスは作りにくい．したがって，MIP ソルバーが選択肢となるが，段取りに伴う費用が大きい場合には，双対ギャップが大きく解きにくいタイプの問題となる．ロットサイズ決定問題は大きく分けて大バケットのものと小バケットのものに分けられる．これらは，段取りに関する情報を次の期に持ち越さない（大バケット）か，持ち越すか（小バケット）による違いである．

（複数品目や多段階の）大バケットのロットサイズ決定問題に対しては，施設配置定式化と呼ばれる強化版の定式化が推奨される．従来の研究では通常の定式化に対して分枝カット法によって切除平面を追加する方法も推奨されていたが，Gurobi による実装では（制約追加のオーバーヘッドがあるため）切除平面を追加せずにそのまま求解したほうが速い．

小バケットのロットサイズ決定問題は，（単位時間が日ではなく時間の）スタッフスケジューリング問題や発電機の起動停止問題と類似の構造を持っている．数理最適化によるアプローチでは，定式化の仕方に工夫をする必要があり，大バケットよりは難しい問題と言える．大規模な問題例に対しては，MIP ソルバーを用いたメタヒューリスティクス（たとえば「メタヒューリスティクスの数理」で記述した緩和固定法や容量スケーリング法），もしくは MIP ソルバーを途中で打ち切る方法が選択肢となる．

4.6 多制約ナップサック問題

多制約ナップサック問題は，複数の資源上限制約（重量や容量など）を持ったナップサックに，利得が定義されたアイテムを，ナップサックの資源上限制約を満たしつつ，利得の合計が最大になるように詰める問題である．この問題はプロジェクト選択に応用を持つが，アイテム数が 1,000 程度までなら Gurobi のようなモダンな MIP ソルバーを使えば容易に最適解を得ることができる．

Chu–Beasley ならびに Glover–Kochenberger のベンチマーク問題でテストをしたところ，変数の数が 2,500 を超えるようになると厳密解を出すのに多大な時間を要することがわかった．しかし，どの問題例も短時間で上界と下界のギャップは比較的小さくなっており，途中で計算を打ち切って得られる近似解は，十分に実用に耐えうると考えられる．たとえば，最大の問題例である 25 万個の変数を含む問題例に対しては，分枝前のヒューリスティクスで相対誤差 1.5%の解を見つけ，分枝開始から 12 秒後に誤差 0.06%の解を得ている．メタヒューリスティクスでこの性能に対抗するのは難しい．もちろん，商用の MIP ソルバーを購入する予算がなく，無償の MIP ソルバーで不満足な場合には，メタヒューリスティクスも選択肢になる．その場合に推奨されるのは，「メタヒューリスティクスの数理」で記述した戦略的振動を用いた禁断探索法であり，最適解の数パーセントの解を算出することができる．

4.7 非線形関数の区分的線形近似

非線形関数を含んだ問題を MIP ソルバーで求解する際には，区分的線形関数で近似する方法を用いる．1 変数の非線形関数は，区分的線形関数で近似することができ，多変数の場合でも，直線の集まりで近似するかわりに超平面の集まりで近似することによって，同様の定式化が可能である．また，任意の多変数関数は，適当な補助変数を導入することによって，1 変数の（分離可能な）関数の和として記述できることが知られている．

非線形関数の区分的線形近似については，過去にさまざまな定式化が提案されており，それらの中でよいと思われるものに対して比較を行った．比較に用いた問題は，倉庫における費用が出荷量の合計の平方根の容量制約付き施設配置問題である．平方根は凹関数であり，その最小化のためには整数変数を用いた区分的線形近似を行う必要がある．

候補となる定式化は，多重選択定式化，（対数個の変数を用いた）非集約型凸結合定式化，（対数個の変数を用いた）集約型凸結合定式化，タイプ 2 の特殊順序集合 (SOS: Special Ordered Set) [4] を用いた定式化である．

結果としては，単純なタイプ 2 の特殊順序集合を用いたものが最も良いので，これを推奨する．以前の研究では，対数個の変数を用いた定式化が良いという結論を出しているものもあるが，Gurobi のようなモダンなソルバーでは，単純に特殊順序集合を用いるほうが，良い結果を出す．

次いで，対数個の変数を用いた集約型凸結合定式化，対数個の変数を用いた非集約型凸結合定式化である．3 変数以上の（分離不能な）非線形関数の近似の際には，（特殊順序集合を用いた定式化が使えないので）これらの定式化を使うべきである．ほぼ同じ性能を持つ集約型凸結合定式化も，定式化が簡単なので推奨される．この定式化は下界が弱いので使うべきではないと，一部の従来の研究では結論づけられていたが，変数の少なさや簡潔さがそれを補うらしい．理論的には良い性質を持っていると従来の研究で言われてきた非集約型凸結合定式化は，推奨できない．

もちろん，ここで示したのは，1 つの問題（凹費用関数を持つ容量制約付き施設配置問題）に対する実験の結果である．最終的な結論を得るには，ほかの非線形関数を含んだ問題に対する実験結果によって補完する必要があり，それは今後の課題と言える．

4.8 二次錐最適化

（Gurobi をはじめとする）モダンな MIP ソルバーは，二次錐制約を含んだ（整数最適化）問題も求解することができる．例として，古典的な Weber 問題（平面上に施設の最適な立地地点を求める問題）を二次錐最適化問題として定式化して，実験を行った．点の数を 10 万まで増やしたが，計算時間はほとんど線形に増加した．同じような計算時間の増加傾向は，（確率制約を二次錐制約として定式化した）ポートフォリオ最適化問題に対しても確認された．結論を言うと，Gurobi の二次錐最適化は高速であり，単純な二次錐最適化問題として定式化できる問題に対しては，（メタ）ヒューリスティクスなどに頼るべきではない．

一方，整数変数を含んだ二次錐最適化問題に対しては，単純な二次関数や線形関数の場合と比べて計算時間を要する．多くの複雑な非線形制約が，工夫次第で二次錐制約として表現できることが知られている（これを二次錐化と呼ぶ）が，整数変数を含んだ問題に対しての適用の可否を判断するには，まだ多くの実験を重ねる必要がある．

5. おわりに

近年の数理最適化技術の進歩は目覚ましい．しかし，多制約ナップサック問題や二次錐最適化問題のように，MIP ソルバーで容易に求解できるものもあれば，グラフ彩色問題，スケジューリング問題，配送計画問題のように，MIP ソルバーに入れただけでは，実用規模の問題を解けないものもある．またその中間で，ロットサイズ決定問題や非線形関数を含んだ整数最適化問題のように，定式化の工夫次第で（実用的な時間で）解けるか解けないかが決まる問題もある．そのため，実務家が実際問題を解く際には，MIP ソルバーは万能薬とは言えず，まだ敷居が高いのが現状である．本稿は，その敷居を下げることを目的として書いたものだが，スペースの都合上，多少舌足らずな記述となってしまった．筆者は，現実問題の解決を至上の喜びと感じている研究者の端くれである．手に余るような最適化問題をお持ちの実務家は遠慮なくご連絡いただければ，喜んでお手伝いをしたいと考えている．

参考文献

[1] K. Nonobe and T. Ibaraki, “A tabu search approach to the constraint satisfaction problem as a general problem solver,” *European J. Operational Research*, **106**, 599–623, 1998.

[2] K. Nonobe and T. Ibaraki, “A tabu search algorithm for a generalized resource constrained project scheduling problem,” In *MIC2003: The Fifth Metaheuristics International Conference*, 2003.

[3] 久保幹雄，ジョア・ペドロ・ペドロソ，メタヒューリスティクスの数理，共立出版，2009.

[4] 久保幹雄，ジョア・ペドロ・ペドロソ，村松正和，アブドル・レイス，あたらしい数理最適化—Python 言語と Gurobi で解く—，近代科学社，2012.

