問題解決における最適解

池上　敦子

本稿では，現在の興味や研究テーマについて，著者らの最近の論文や学会発表の内容を利用して紹介したい．具体的には，問題解決における最適解の提供方法について議論する．意思決定者がもつ暗黙的な評価尺度を後から反映しやすい仕組みとして，暫定的に与えた目的関数に対する最適解列挙を行い，それら最適解同士の関係を明らかにしよう，という試みである．まだまだ初歩的ではあるが，これを探るために行った計算実験の結果も紹介したい．

キーワード：問題解決，多様な解と類似な解，最適解列挙，評価尺度，ナーススケジューリング

1. はじめに

4 つの数字を使って 10 を作る遊びがある．たとえば，1234 なら，$1+2+3+4=10$ とか，$2\times4-1+3=10$ といった具合である．もう少し面白いのは，3467 で，$4\times7-3\times6=28-18=10$, 9999 で，$(9\times9+9)\div9=90\div9=10$，8888 で，$(8+8)\div8+8=2+8=10$ などである．

小さな頃から数字が好きだった私は，母にこの遊びを教えられ，昔の鉄道切符に印字されていた番号や，車のナンバープレートを使って遊んでいた．そのおかげか，中学校あたりで習う因数分解では，瞬時に答えを出すことができた．そして，今なお，4 桁の数字を見ると自動的に計算してしまうのである．

さて，この遊びを通じて，最近，自分の思考の偏り（癖）に気づくことがあった．

大槻兼資さん著「パズルで鍛えるアルゴリズム力」[1] という本の最初に，この遊び（テンパズル）の紹介がある．この遊びについて自信満々の私は，出題例を簡単に解いていったものの，最後に「たいへん難しい問題」として挙げられていた 1 問目で，つまづいてしまった．通常は「答え 1 つ（複数）」もしくは「実行不可能である」を秒でいえるのだが，実行可能である前提で出題されているはずなのに，なぜが答えが出てこないのである．

この問題（4 桁数字）は，1337 だった．答えがすぐ出てこないので，つい，先に本を読んだ人に「これって整数制約ありますか」ときいてしまった．そして，「途中解に整数制約はない」との情報（ヒント）を聞いてから答えがわかったので，いまでも悔しい思いが残っている．この答えの詳細については，大槻さんの本を参照していただきたい．

そこで，気付いたのは，私には勝手に「整数制約を加える」癖があるということだった．小数や分数を学ぶ前に，この遊びが身についていた弊害かも知れない．

考えてみると，子供の頃から整数に興味が集中していた．π の意味には興味があったが，物理でたくさん出てくる 9.8 や 4.9 という数字には，どうもしっくりこなかった憶えがある．一方，小学校 2 年生で書いた「数は不思議だ」という詩では，「1 番大きいと思った数字に 1 を足すとさらに大きな数字ができて，それが繰り返せること」を不思議がっていた．高校生か大学生になってこれを読んだとき，「これって帰納法？」「整数好き？」などと面白く感じた．

さて，だいぶ寄り道したが，本稿では，現在の自分の興味や研究テーマについて書いてみたい．

私は専門分野をたずねられたとき，OR，数理最適化，組合せ最適化とこたえ，「モデリングに興味があります」と言っている．解決すべき問題の中の「組合せ構造」を見つけ出し，モデル化するのが楽しいからだと思う．

自分の研究テーマのせいか「理論には興味がなく応用だけ好き」と思われがちだが，理論と応用を別扱いする気もないうえ，両方が好きである．問題解決が大好きなので，両方に興味があり，好きなのである（優れているという意味ではない）．因みに，問題解決が好きなのは，パズルの解を出すのと同じような感覚である．

寄り道ついでに本音を言うと，距離最小化とか，コスト最小化といったように，評価軸が 1 つで明確であり，最適解がそのまま現実世界の最適解になる問題に心惹かれる．鉄道運賃計算の研究 [2, 3] では，モデリングが勝負だったので，アルゴリズム的にはシンプルなものだったが，与えた解がそのまま駅の自動改札機

いけがみ　あつこ
成蹊大学名誉教授
atsuko@seikei.ac.jp

で利用されるといった設定は，とても心地よかった．

一方，これと矛盾するようではあるが，現在，強い興味で取り組んでいるのは，人の主観を無視できない「評価尺度が規定しにくい問題」に対する解の提供方法についてである．

本稿では，この興味に対し，これまで取り組んできたこと，現時点で考えていることを紹介したい．

2. 最適解が多数存在

最適化の魅力は，なんといっても，最適解を与えることによって「この条件の下では，これより良い解は存在しない」と自分にもほかの人にも説明でき，納得を基にした意思決定を支援できることである．この魅力は，研究や実務を進めるうえで強い原動力になると感じている．

しかし，ナーススケジューリングのように，最適解を 1 つ与えたからといって，このような納得を生み出せるわけではない問題も，現実には多い．

ナーススケジューリングは，病棟ナースの勤務表を作成する問題である．各ナースの健康状態，社会生活を守るための多くの制約を満たしながら，各日各シフトに適切なスキルレベルのナースを必要数配置する必要がある．さらに，目的関数の設定も難しい．

われわれの研究では，満たせない制約に対する違反度にペナルティ重みをつけて，便宜上の目的関数を設定して解く．目的関数を線形式で記述できれば，変数がさほど多くないこの問題は，今日の最適化ソルバーで比較的速く最適解が得られるからである．しかし，性質が全く違う複数の制約に対する違反ペナルティの線形加重和で，真の目的関数を表せるわけもなく，得られるのは便宜上の最適解にほかならない．

ここで，真の目的関数と書いたが，この問題にそれが存在するとは考えていない．「適切な式や重みの設定さえすれば真の目的関数になる」わけではないという意味である．「適切な式や重み」は評価者の主観に依存するだけではなく，同じ評価者でも時間やちょっとした環境の変化で目指すものが微妙に異なってしまうのだ．つまり，評価者本人すら意識していない，もしくは，具体的な解を見てからでないと意識できない，暗黙的な制約や評価尺度が存在するので，真の目的関数を一意には決められないのである．

ちょっと脱線するが，「真の目的関数を明確にすればよい，するべきだ」という意見は，この研究を行ってきて何度も投げかけられた．「それは違う」と言い続けてきたが，説明が悪いのか伝わらないことも多い．したがって，本稿で紹介するテーマにこだわる意味は少なからずあると思う．

さて，便宜上であっても，得られた最適解に意味がないのか，というと，そうではない，…というより，枝葉を切り落としたとしても，本質的な構造を適切にモデリングすることで，最終的に望まれる解に近い解（望む修正のための有効情報としての解）を得るべきと考える．

ここで，この研究を進めてみてわかった大きな事実として，1 つのモデル（定式化）においても，**最適解が多数存在**していたことが挙げられる．最適化ソルバーの性能が今ほどではない頃，1 つの最適解を得るために多くの年月と労力をかけていた人間としては，最適解が 1 つではないということすら最初は驚きだったが，今は，その数が膨大で，それをどう捉えるかが重要な問題になってきた．

3. 広い意味での「最適化」

昔のことではあるが，ナーススケジューリング研究で，勤務表を作って現場にもっていくと，「数学でこんな勤務表ができるなんてすごい！」「だいたいいい感じ！」といった褒め言葉からスタートするものの，しばらくすると，「でも，ちょっと，ここがダメ（体調を崩しているこの人にとって，この夜勤並びはキツい）」「あ，ここもダメ（この人は，先月もたくさん休み希望を出しているので，少し抑えるべき）」といった意見が出て，「だいたいいいけど，もうちょっと違うのはないの？」「もっとたくさん見たいんだけど」とくる．

それでは，といって，勤務表を 30 種類くらい作ってもっていくと，「わー，こんなたくさん見ることはできないので，いいものだけ見せてほしい」といわれる．これは，何度も繰り返したことである．

現場でいろいろ議論していくうちに，勤務表を与えられたときに「感じること」としては，大きく分けて

1. 少し修正したいが，簡単にできるのか
2. 全く違ったものがあるか，あるなら比べたい

の 2 つがあることがわかった．

そして，複数作った勤務表（B4 判に印刷したもの）を重ねて，あかりに向けてかざして，勤務表間の相違や共通部分を見ながら，共通部分について「やっぱり，ここは変えられないのか」とか，そうでない部分について「ここって，いろいろな選択肢があるようだ」とか話し，単純ながら，こういった方法は，それなりにいい情報を生むかもしれない，と感じていた．

「私が神様で，魅力的な勤務表を山ほど知っているの

なら，現場の好みをキーにして，検索できるのに」とも思った（今の気持ちとしては，もうちょっと研究を頑張れば，これも，まんざら夢ではない）．

さて，これらの経験からも，現場で実用できる解を得るためには，最適解を 1 つ得るといった単純な最適化方法では不十分だと考えられる．与えた解に対して，わずかに修正された解が望まれる場合と，全く異なる解が望まれる場合があることから，できるだけ多くの最適解を得る必要があるうえ，解同士の多様性と類似性を考慮する必要がある．もう少しいうと，ナーススケジューリングに限らず，重要な意思決定において，

1. 最適解を複数得る（最適解集合を得る）こと
2. 解の多様性と類似性を議論すること

は，**広い意味での「最適化」**として重要だと考える．

このことを主張し始めてからずいぶん経つが，それを広める論文を出せてこなかった力不足が悔しい思いである．とはいえ，最近，関わる論文を 2 つ出版 [4, 5] できたので，残りの節では，それらと，それに続く研究内容（まだ論文になっていない内容）を簡単に紹介したい．

なお，最近あった，うれしかったことを報告すると，若く優秀な研究者が，すでにこの視点を持ち，多数の最適解を得ることや，最適解集合が満たす条件を研究しているのである [6]．その展開を考えると，非常に楽しい気持ちになる．

4. まずは類似な解

ナーススケジューリングのモデルは複数あり，同じモデルであっても複数の定式化が考えられる．ここでは，1 週間分のシフト（日勤，準夜勤，深夜勤，休みなど）並びをパターンとよび，列挙した実行可能なパターンを各ナースの各週に割り当てる定式化を利用する．そして，類似な解を多数作成する．

I をナースの集合，S をシフトの集合，D を日の集合，G をナースグループ（たとえば，ベテラン，新人，東病室担当，西病室担当）の集合，I_g をグループ g に所属するナースの集合とし，日 d のシフト s にグループ g のメンバーが勤務すべき人数の下限と上限を，それぞれ，a^{lb}_{gds}，a^{ub}_{gds}，ナース i が対象期間内にシフト s で勤務すべき回数の下限と上限を，それぞれ，b^{lb}_{is}，b^{ub}_{is} とする．そして，W を週の集合，D_w を週 w に含まれる日の集合，P_{iw} をナース i の週 w に割り当て可能なパターンの集合とする．パターン p については，シフト s が含まれる数を ρ_{ps}，その構成を，δ_{pds}（パターン p の日 d がシフト s なら 1，そうでないなら 0）で表す．

この問題では，シフト並びに多くの制約が課されている[1]ため，隣り合う週で連続実施が不可能なパターンの組合せが存在する[2]．そこで，パターン p の前週に採用できないパターンを集合 R_p に用意する．

意思決定変数 λ_{iwp} は，ナース i の週 w にパターン p を採用するとき 1，そうでないとき 0 となる．変数 z^{lb}_{gds} と z^{ub}_{gds} は，日 $d \in D$ のシフト s にグループ g のメンバーが勤務すべき人数の下限と上限を守れなかった度合い（それぞれ，不足数，過剰数）を表し，これらの変数が値をもったときのペナルティを，それぞれ，α^{-}_{gds}，α^{+}_{gds} とする．少し端折った説明にはなったが，定式化 [4] を以下に示す[3]．

$$\text{Minimize} \quad \sum_{g\in G}\sum_{d\in D}\sum_{s\in S}(\alpha^{-}_{gds}z^{\mathrm{lb}}_{gds} + \alpha^{+}_{gds}z^{\mathrm{ub}}_{gds}) \tag{1}$$

$$\text{subject to}$$

$$a^{\mathrm{lb}}_{gds} - z^{\mathrm{lb}}_{gds} \le \sum_{i\in I_g}\sum_{p\in P_{iw}} \delta_{pds}\lambda_{iwp} \le a^{\mathrm{ub}}_{gds} + z^{\mathrm{ub}}_{gds}, \quad g \in G, w \in W, d \in D_w, s \in S \tag{2}$$

$$b^{\mathrm{lb}}_{is} \le \sum_{w\in W}\sum_{p\in P_{iw}} \rho_{ps}\lambda_{iwp} \le b^{\mathrm{ub}}_{is}, \quad i \in I, s \in S \tag{3}$$

$$\lambda_{i,w-1,p'} + \lambda_{iwp} \le 1, \quad i \in I, w \in W \setminus \{1\}, p \in P_{iw}, p' \in P_{i,w-1} \cap R_p \tag{4}$$

$$\sum_{p\in P_{iw}} \lambda_{iwp} = 1, \quad i \in I, w \in W \tag{5}$$

$$\lambda_{iwp} \in \{0, 1\}, \quad i \in I, w \in W, p \in P_{iw} \tag{6}$$

$$z^{\mathrm{lb}}_{gds}, z^{\mathrm{ub}}_{gds} \ge 0, \quad g \in G, d \in D, s \in S. \tag{7}$$

目的関数 (1) は，制約式 (2) と併せることで，各日各シフトに対する各グループからの勤務人数の過不足を最小化する．制約式 (3) は，各ナースの各シフトに対する勤務回数の下限上限を設定し，制約式 (4) は，禁止されるシフト並びを避け，制約式 (5) は，各ナースの各週に実行可能なパターンを 1 つ割り当てる．

誤解を避けるため説明しておくと，最適化ソルバーで問題を解く場合，求解速度的に，この定式化がよいのかというと，そういうわけではない．

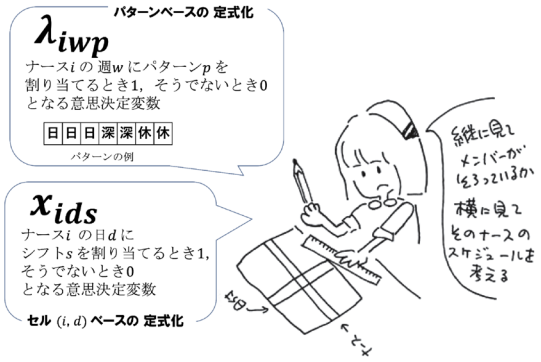
以前，この定式化の制約式 (4) の部分がわずかに異なる 3 種類と，意思決定変数 x_{ids}（ナース i の日 d に

[1] 同一シフトに関して，連続回数の下限と上限，間隔日数の下限と上限，そして，準夜勤や深夜勤の後の日勤といったように，異種シフトを含む禁止並び，がある．

[2] パターン長が短かすぎると，隣り合う週の関係だけでは，禁止されるシフト並びの制約を確認できないが，多くの場合，週（7 日）の長さをもてば問題ない．詳細は，論文 [4] を参照されたい．

[3] 記号の定義が，論文 [4] と異なる部分があるが，本質的には同じである．

シフトを割り当てるとき 1，そうでないとき 0）を利用した定式化を，2 種類の最適化ソルバーを使って比べたことがある．ある問題インスタンスに対し，前者の 3 つは，最適解を暫定解として見つけるのに 1 時間以上かかり，最適解の保証を得るにも 1 週間では足りない場合があったが，x_{ids} に基づく定式化は，最適解を数分で見つけ，数十分でその保証を得ることができた．



そこで，最適化ソルバーで直接解く場合は，x_{ids} の定式化を利用することにしているが，前述したとおり，本節の最適解列挙では，定式化 (1) 〜(7) に基づいて話を進める．

研究の過程で利用した内容を以下にまとめる．

1. 目的関数 (1) と制約式 (2) を除くと，ナースごとに独立な，実行可能スケジュールを求める部分問題になる [7]
2. 各ナースにおいて，各週の各パターンを，異なるシフト累積回数情報をもたせた複数のノードで表し，それらの隣接可能関係をアークで表すことにより，対象期間前後に加えたソースノードとシンクノードの間のすべてのパスが，そのナースの実行可能スケジュールになる，かつ，すべての実行可能スケジュールが含まれるようなネットワークを構築できる [8]
3. 何らかの解（勤務表）を 1 つ与えて，対象ナース 1 人以外のスケジュールをその解に従って固定すると，目的関数 (1) と制約式 (2) に従って対象ナースの各パターンにコストが設定できるので，対象ナースの最適スケジュールを得る問題を，最短路問題，動的計画問題として解くことができる [8]
4. さらに，この部分問題において，制約式 (3) を緩和すると，「異なるシフト累積回数情報をもたせた複数のノード」を 1 つで表現できるので，ネットワークサイズが大幅に削減できる．さらに，最短パスを全列挙した中に，制約式 (3) を満たすものが多く存在する [4]

項目 4 で述べたネットワーク上のノード（各週のパターンに対応）のコスト f_{wp} について簡単に説明する．

n'_{gds} を，与えた解に従って対象ナース以外のスケジュールを固定した場合の，日 d のシフト s にグループ g から勤務する人数（対象ナースは除く）として，その人数過不足に基づくペナルティの値 Δ_{gds} を

$$\Delta_{gds} = \begin{cases} -\alpha^{-}_{gds}, & \text{if } n'_{gds} < a^{\mathrm{lb}}_{gds}, \\ \alpha^{+}_{gds}, & \text{if } n'_{gds} \geq a^{\mathrm{ub}}_{gds}, \\ 0, & \text{otherwise} \end{cases}$$

とすると，コスト f_{wp} は，

$$f_{wp} = \sum_{g \in U_i} \sum_{d \in D_w} \sum_{s \in S} \Delta_{gds} \delta_{pds}$$

と表せる．ここで，U_i は対象のナース i が所属するグループの集合である．

便宜上，コスト f_{wp} を，週 w のパターン p のノードではなく，そのノードから発するアークに設定し，最短路問題を解くことにする．

以下に，このネットワークを利用した最適解列挙の方法を示す．

1. 既知の最適解（勤務表）をシードとして 1 つ与える
2. 各ナースについて，以下を行う
   i) シード解を基に，ネットワークを設定する
   ii) 最短路問題を解き，最短パスをすべて列挙する：最短パスを構成するノードとアーク[4]だけのネットワークを作り，ソースノードからシンクノードまでのパスをすべて列挙する
   iii) 列挙されたパスに対応するスケジュールが，制約式 (3) を満たせば，それをシード解における対象ナースのスケジュールと入れ替えることで新たな最適解が得られるので，その解を最適解プールに保存する
3. 最適解プールから 1 つ解を選び，シードとして 2 に戻る

参考までに，利用ノードのイメージが湧くよう，2 の ii) で作った最短パスを列挙するためのネットワーク

[4] 便宜上，表記を変えるが，ソースノードからノード j までの最短距離を d_j，アーク (i, j) のコストを c_{ij} とすると，$d_i + c_{ij} = d_j$ の関係をもつノードとアークのことをいう．

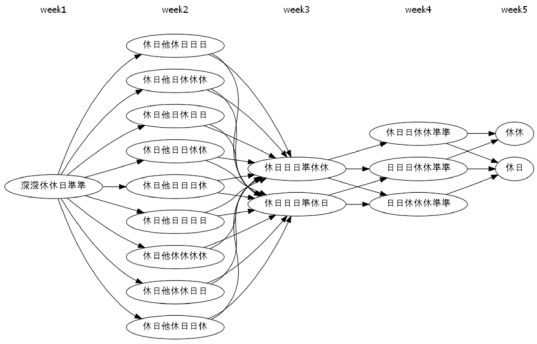


図 1　最短パス列挙用のネットワーク
（日：日勤，準：準夜勤，深：深夜勤，休：休み，他：他勤務）

（ソースノードとシンクノードは省略）を図 1 に示す．

ここでは，終了条件を書かなかったが，「シードとして未使用の最適解が残っている限り」とすると，簡単に終わらない場合がある．与えた問題インスタンスでは，35 日間の計算で 7,000 万以上の最適解を得たにもかかわらず，シード解として使われたのは，最適解プールの 0.3%のみであった．

論文 [4] では，ほかにも計算実験を行ったが，最適解の数は，想像していたより，ずっと膨大であること，1 人のナースのスケジュール変更に絞った探索だったため，異なる部分（ナースと日）は限られており，非常に似ている解，つまり，類似な解ばかり得られていることがわかった．

最初にシードとして与えた解が，ほぼ満足いくものである場合や，実施期間に入っている解（勤務表）だった場合の急な変更に対し，少ない人数，少ない部分の修正を可能にする情報として，それらの解は意味があると考えられる．

5.　多様な解とその間にある解

今度は，逆に，最初に与えられた解がしっくりこない場合や，全く異なる解を比べたいときのことを考える．前節の方法では，探索範囲が限られている可能性が高く，その領域から抜け出すことが難しい．

さて，ナーススケジューリングで対象とする勤務表は，各列が日に対応し，各行がナース 1 人分のスケジュールを表すことが一般的である．たとえば，ナース人数が 25，対象日数が 30 であれば，$25 \times 30 = 750$ のセルに，勤務シフトの種類を書き込むことになる．

2 つの勤務表を比べるとき，750 セルのうち，多くのセルに異なるシフトが入っていると「大きく異なる」と思われがちだが，本当にそうだろうか．

以前，与えた勤務表に対して，それと「異なるシフトとなるセルの数の最大化」を目的関数に加えて解いてみた．結果として得られた勤務表を見たとき，比較的制約が緩い部分，たとえば，新人にとってのある日が「休み」か「日勤」かといった違いが多くを占め，現場で出てくる「もっと違うのはないの？」という要望には応えられない，と感じた憶えがある．

現場では，ベテランナースの勤務シフトに注意を払って勤務表が作成される．そして，勤務人数が少なく，メンバーの組合せが重視される準夜勤や深夜勤における違いが，大きな興味である．

論文 [5] では，それぞれが最適解であり，望まれる多様性をもつ複数の解を得ることを目指した．ナースの各日のシフトを扱いやすくするため，意思決定変数 x_{ids} に基づく定式化を利用した．そして，現場で重要視するセル (i,d) の集合を T として設定し，それらのセルのシフトの違いができるだけ多くなるように複数解を得ることを考えた．

実験では，重要視するセルや，望まれる多様性が明らかでない場合にも適用できるよう，定式化に頻繁に登場する意思決定変数に関わるセル (i,d) を集合 T の要素にした．

最適解を 1 つ得る一般的なナーススケジューリングの定式化を以下で表すことにする．

$$\text{Minimize} \qquad f(\boldsymbol{x}) \tag{8}$$
$$\text{subject to} \qquad \boldsymbol{x} \in X. \tag{9}$$

ここで，X は実行可能解の集合，$\boldsymbol{x}$ の要素 x_{ids} はナース i の日 d がシフト s であるとき 1，そうでないとき 0 となる意思決定変数，$f(\boldsymbol{x})$ は勤務表 $\boldsymbol{x}$ を評価する値（最小化）である．

すでに得られている最適解や実行可能解の集合 H に対し，前述の集合 T の要素セル (i,d) において，できるだけ異なるシフトになる解を求める定式化を示す．

$$\text{Minimize} \qquad \theta + \rho f(\boldsymbol{x}) \tag{10}$$
$$\text{subject to} \qquad \boldsymbol{x} \in X \tag{11}$$
$$\sum_{(i,d)\in T} x_{ids^h_{id}} \le \theta, \qquad h \in H. \tag{12}$$

ここで，解 $h \in H$ は，セル (i,d) に割り当てられているシフト s^h_{id} $(i \in I, d \in D)$ で表されているとする．ρ は $f(\boldsymbol{x})$ に対する重み付けであり，変数 θ は H の要素解との T の要素セルにおける一致数の最大値を表す．

1 つの最適解のみを含む集合を H としてこの問題を解き，得られた解を集合 H に追加して解き直すこと

表 1　6 つの解の間の距離

	0	1	2	3	4	5
0	–	124	108	101	102	91
1	450	–	108	101	102	92
2	383	414	–	101	103	99
3	376	413	396	–	105	92
4	405	400	376	387	–	91
5	394	384	385	364	375	–

（右上三角：$|T| = 153$ セル対象，左下三角：全 750 セル対象）

を，必要な数の解が得られるまで繰り返す．

おおまかなイメージを示すため，問題インスタンスの詳細は省略するが，セル数 750（ナース数 25，対象日数 30），$|T| = 153$ の問題に対して，1 つの最適解（解 0）を与え，5 つの最適解（解 1〜5）を得た例を紹介する．最適値は 2 である．

6 つの解の違い = 距離（異なるシフトになったセルの数）を表 1 に示す．$|T| = 153$ セルを対象にした数が右上三角，全 750 セルを対象にした数が左下三角である．対象とする 153 セルだけでなく，全セルを対象にしても，各解が比較的均等に異なっていると感じられた．

ここで紹介したように，解の相違，多様性を何らかの形で定義できれば，最適化ソルバーなどを利用して，特徴が異なる最適解を複数得ることができる．

すると，次なる興味は「非常に多くの数」の最適解の列挙である．以下に 2 つの方向性を示す．

I. 定式化 (10)(11)(12) に基づく方法をずっと続ける
II. 特徴の異なる解の間の解を列挙する

I の方法は，解く方法にもよるが，計算負荷が比較的大きい（実験については省略）．

II の方法では，「解の間の解」の定義によって，その負荷を軽減できると考え，2 つの解 h と k の共通部分（$s^h_{id} = s^k_{id}$ となったセル）は，そのシフトで固定し，それ以外のセルでは，s^h_{id} か s^k_{id} を選ぶことにした．そして，解 h から解 k に向けて距離を 1 ずつ増やしながら解く．ここでの距離は，表 1 の左下の数（全 750 セルを対象とした距離）とする[5]．

問題設定方法としては，以下の 2 つを考えた（ここで，隣接解とは解 h からの距離が 1 だけ違う解とする）．

1. 隣接解同士の距離は 1：解 h からの距離が t の解は，解 h からの距離が $t-1$ の解を利用し，1 つのセルだけシフトが異なるものとして見つける
2. 隣接解同士の距離に制約なし：解 h からの距離が t の解を，t の値ごとに独立に解く

これらを基に，表 1 の 6 つの解の間を双方向に（$6 \times 5 = 30$ 通りで），解を推移させることを考えた．

方法 1 では，前の距離の解に基づいて解くため，実行可能解が存在しない場合もある（その場合は，次の距離までの解を対象にする）．全列挙であっても高速に解を得るが，解（勤務表）自体の評価値 $f(\boldsymbol{x})$ が悪くなる度合いが大きい．推移開始から距離 50 付近までは，$f(\boldsymbol{x})$ は最適値 2 を保つが，その後，$f(\boldsymbol{x})$ は大きくなり続け，200 を超す場合もある．そして，行き先の解 k に近づいたところで，急に最適値に近づく．大きな山を越えないと，大きく異なる解に到達できないという状況は，30 通りの推移すべてで同様だった．

方法 2 は，解 h からの距離 t の解を，以下の定式化に基づいて求めた．

$$\text{Minimize} \quad f(\boldsymbol{x}) \tag{13}$$

$$\text{subject to} \quad \boldsymbol{x} \in X \tag{14}$$

$$\sum_{i \in I} \sum_{\substack{d \in D \\ s^h_{id} \neq s^k_{id}}} x_{ids^k_{id}} = t, \tag{15}$$

$$x_{ids} = 1, \qquad i \in I, d \in D, s = s^h_{id} = s^k_{id} \tag{16}$$

$$x_{ids^h_{id}} + x_{ijs^k_{id}} = 1, \quad i \in I, d \in D, s^h_{id} \neq s^k_{id}. \tag{17}$$

制約式 (15) は，解 h からの距離が t であること，制約式 (16) は，解 h と解 k で同じシフトになっているセルはそのシフトを割り当てること，制約式 (17) は，同じシフトでないセルでは解 h か解 k どちらかのシフトを割り当てることを示す．

図 2 は，方法 2 の結果である．6 つの解を 6 角形の頂点に表し，各解の間の推移に伴う $f(\boldsymbol{x})$ の値の変化を双方向に示す．各 2 頂点の最適解の間に，400 前後の解の $f(\boldsymbol{x})$ 値がプロットされている．

探索領域が狭いこともあり，1 つの解を得る平均時間は 0.22 秒である．双方向で同じ解が得られる[6]と考えると，単方向 15 通りの約 5,900 の問題を 22 分程度で解き，3,000 以上の最適解を得ることができたことになる（最適化ソルバー：Gurobi 8.1.0，CPU：Xeon E-2144G @3.60 GHz; memory: 64 GB）．また，最適でない場合も，最適値に近い値（高々 8）だった．

多様な解を多数列挙することについては，まだ論文にも書いておらず，2022 年 3 月の OR 学会で発表 [9] しただけである．まだまだ試したいこともあり，まと

[5] 右上の数（$|T|$ の要素を対象とした距離）で考えてもよい．

[6] 双方向で同じ解になる保証はないが，$f(\boldsymbol{x})$ の値は等しいので，図 2 の 2 頂点間を繋ぐ線は 1 本で表現することはできる．

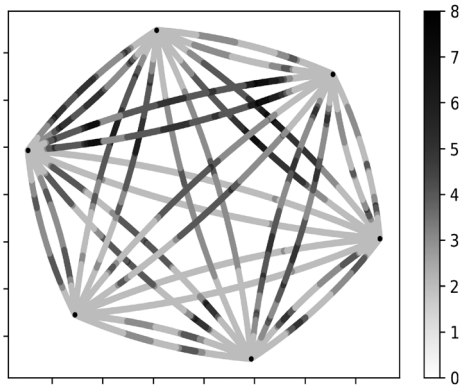


図 2　距離が離れた 6 つの最適解の間の解の $f(\boldsymbol{x})$ の推移

まった成果を出すには時間がかかりそうだが，図 2 に含まれる 3,000 以上の最適解を基に，その特徴を分析し，必要にあわせて，それらの解の間の推移を行うことにより，さらなる最適解で埋め尽くす方法も展開してみたい．そして，それら複数の解を視覚的にどう提示するかが，意思決定を支援できるかどうかに大きく関わると考えている．

6. おわりに

問題解決において，われわれが扱う最適解を意思決定者にどのように提供するかは，OR にとって非常に重要な課題だと感じる．そのためにも，最適化概念を少し広げる工夫，広い意味での最適化について，さらなる議論，研究が必要だと考える．

意思決定者の自由度を広げるためには，解の可能性を感じられる（把握できる）情報が必要であり，具体的には，意思決定者が最も望む解を含むような解空間（解集合）や，望む解に素早く到達できる修正可能性の情報を提供するべきと考える．

4 節の方法や 5 節の方法 1 の実験結果からは，近傍探索のような方法では，特徴の異なる解を見つける際，かなり大きな山を越す必要があり，小さい近傍設定では，たどり着けないことが考えられる．しかし，4 節の最後に述べたように，類似な解にも大きな意味がある．

一方，5 節の方法 2 のように，大きく特徴の異なる解を少数であっても複数与えることにより，推移という発想で，探索範囲を絞り込んだ高速化を図り，最適解列挙に利用することは，今後の「多様な解」の研究や「最適解列挙」に何らかの推進力になると考える．

本特集では，OR 誌読者のみなさんに，なにか素敵なメッセージを発信すべきなのだが，結局，自分の興味のことばかりになってしまったことを，お詫びしたい．

わがままついでに，今後の自分のことを書いてしまうと，やはり，パズルを解くように研究したり，問題解決に取り組み続けていたいのである．

かつて，褒め言葉として「泥臭い」研究といわれたことがあるが，本人的には「いつかきっと！」という気持ちになった．いつか洗練された視点やアイデアを発信できたら，という夢はあきらめず，年齢や立場とは関係なく，大好きな研究に携わっていきたい，と考える次第である．あくまで夢ではあるが♪

最後に，OR 誌は，いつでも頼れる**素敵な機関誌**であり続けていただきたい．

謝辞　本稿における内容は，成蹊大学の院生だった秋田博紀さん，長谷部勝也さん，加藤尚瑛さんの修士論文研究，本稿で紹介した論文の共著者の皆さんとの研究や議論に基づいています．

参考文献

[1] 大槻兼資，『パズルで鍛えるアルゴリズム力』，技術評論社，2022.
[2] 森田隼史，池上敦子，菊地丞，山口拓真，中山利宏，大倉元宏，“鉄道運賃計算アルゴリズム― Suica/PASMO 利用可能範囲の JR 東 510 駅の運賃を対象とした場合―，”日本オペレーションズ・リサーチ学会和文論文誌，**54**, pp. 1–22, 2011.
[3] 池上敦子，森田隼史，山口拓真，菊地丞，中山利宏，大倉元宏，“鉄道運賃計算のための最安運賃経路探索　―複数の鉄道会社を含む場合―，”日本オペレーションズ・リサーチ学会和文論文誌，**51**(2), pp. 1–24, 2008.
[4] M. Hasebe, K. Nonobe, W. Wu, N. Katoh, T. Tanabe and A. Ikegami, “Generating decision support information for nurse scheduling including effective modifications of solutions,” *Journal of the Operations Research Society of Japan*, **64**, pp. 109–127, 2021.
[5] 加藤尚瑛，呉偉，池上敦子，“ナーススケジューリングにおける多様な解の生成，”情報処理学会論文誌「数理モデル化と応用」，**15**, pp. 1–10, 2022.
[6] K. Ando, N. Sukegawa and S. Takagi, “Strong Condorcet criterion for the linear ordering problem,” *Journal of the Operations Research Society of Japan*, **65**, pp. 67–75, 2022.
[7] A. Ikegami and A. Niwa, “A subproblem-centric model and approach to the nurse scheduling problem,” *Mathematical Programming*, **97**, pp. 517–541, 2003.
[8] 秋田博紀，池上敦子，“ナース・スケジューリングにおける部分問題実行可能解空間のネットワーク表現，”統計数理，**61**, pp. 79–95, 2013.
[9] 加藤尚瑛，呉偉，池上敦子，“多様な最適解の間の推移，”日本オペレーションズ・リサーチ学会春季研究発表会 (Online), 1-B-5, 2022.

