Vol.2022-MPS-137 No.7
2022/3/3

工事立会者手配問題に対する制約生成法および集合被覆アプローチ

高須賀 将秀[1,2,a),b)]　呉 偉[3,c)]　柳浦 睦憲[2,d)]

概要：近年，道路等のインフラ設備の老朽化は深刻な社会問題となっており，補修工事が日々行われている．そのような工事には立会者が必要であるが，複数の工事への立会者の割当を定める際には様々な業務要件を考慮しなければならず，高度な技能を有する手配者によって割当が行われていた．この工事立会者手配業務に対し，以前我々は実用的な手配結果を算出可能な数理モデルを提示し，実業務への導入を行った．その数理モデルを用いた運用を続けた結果，各立会者に割り当てられる工事件数を今より大きくしたいという新しい要望があがった．本研究では，そのような要望に応える実践的な手法を得るため，制約生成法に基づく数理モデルや，集合被覆アプローチに基づく数理モデルに対して汎用ソルバーを用いて求解する手法の比較検証を行う．計算実験により，立会者に割り当てる工事件数が大きい場合でも，それらの手法によって現実的な時間で良質の解を得ることを確認した．

1. はじめに

近年，道路等のインフラ設備の老朽化は深刻な社会問題となっており，施設保全度の確保およびライフコストの縮減と予算の平準化を図るため，これらのインフラ設備の補修工事が計画的に行われている．本研究で対象とする工事では，工事を行う際に，設備に関する知識を有した立会者を派遣しており，各工事にどの立会者を割り当てるかを定める必要がある．割当を行う際は，各立会者が担当する工事を全て巡回したときの総移動時間を小さくしたいという要望や，スキルの低い人に難しい工事を割り当てると事故が起こりやすくなるため各工事を適切なスキルを持つ人に割り当てたいという要望など，様々な業務要件を考慮する必要があり，割当の決定には高度な技能が必要になる．そのような高度な技能を有している手配者と呼ばれる専門家により意思決定が行われている工事立会者手配業務に対し，実用的な手配結果を算出可能な数理モデルを以前 [1] で構築した．その数理モデルから得られる手配結果を実業務で試用したところ，各立会者に割り当てる工事件数をより大きくしたいという新しい要望があがった．立会者に割り当てる工事件数を大きくすると，1 人の立会者が巡回できるルート候補の個数が指数的に増加する．そのため，先行研究 [1] の，実行可能なルート候補の全列挙に基づく集合被覆アプローチは実用的でない．

本研究では，各立会者に割り当てる工事件数が大きい場合でも実践的な手法を得るため，制約生成法に基づく数理モデルや，集合被覆アプローチに基づく数理モデルに対して汎用ソルバーを用いて求解する手法の比較検証を行う．計算実験により，各立会者に割り当てる工事が 4 件以上と [1] より大きな場合でも，現実的な時間で良質の解を得ることを確認する．

2. 工事立会者手配問題について

本節では，本研究で取り扱う工事立会者手配問題について説明した後，実業務で一定期間運用した際に出てきた新しい要望について述べる．

2.1 手配者による工事立会者の手配

与えられた全ての工事に対して立会者の割当を決定する問題を工事立会者手配問題と呼ぶ．先行研究 [1] で数理モデルを構築した際に手配者へのヒアリングで聞いた，人手で立会者を工事に割り当てる手配業務について述べる．まず，地図上に点で記された工事を，3 つ以下の工事より成るグループに分けたのち，各グループに立会者を割り当てる．立会者は，事務所を出発して，3 つ以下の工事を数回巡回した後，事務所に戻ってくる．このとき立会者は，工事が正常に進捗していることを確認し，必要に応じて指示

[1] 西日本電信電話株式会社
[2] 名古屋大学大学院情報学研究科数理情報学専攻
[3] 静岡大学大学院総合科学技術研究科工学専攻
a) masahide.takasuka.tc@west.ntt.co.jp
b) takasuka@nagoya-u.jp
c) goi@shizuoka.ac.jp
d) yagiura@nagoya-u.jp

を出すことが役割であるため，工事が開始してから終了するまで張り付いて立ち会うわけではない．つまり，各工事を 1 回ずつ訪れたのち事務所に戻るのではなく，1 時間に 1 回程度各工事の様子を見回りに行く．その結果担当する工事の間を何度も巡回する必要があるため，事務所を除いて担当する工事のみを回る巡回路の移動時間の全ての立会者に対する和を小さくしたいという要望がある．また，各工事には 3 段階の難易度が設定されており，各グループ内の工事の難易度の和が 9 以下となるようにしている．各グループへの立会者の割当の決定は，各立会者が各工事に対して持っているスキルレベルの情報に基づいて行われる．ある工事に対して，その工事に対するスキルの低い立会者を割り当てると事故が起こりやすくなるため，そのような割当をできるだけ避けたいという要望がある．

2.2 先行研究の数理モデル

工事立会者手配問題では，立会者の集合 S および工事の集合 J，工事 $k \in J$ から $l \in J$ への移動時間 d_{kl}，立会者 $s \in S$ に工事 $k \in J$ を割り当てたときの割当ペナルティ c_{sk}，工事 $k \in J$ の難易度 w_k，各立会者に割り当てられる工事数の上限 ν，各立会者に割り当てられた工事の難易度の和に対する上限 W，および目的関数（総移動時間と総割当ペナルティの重み付き和）内の重み係数 α (≥ 0) が与えられる．工事間の移動時間は，一方通行などの道路状況により工事 k から l のルートと l から k へのルートが異なる場合があり，このとき，$d_{kl} \neq d_{lk}$ となる．また，同一工事間の移動時間は $d_{kk} = 0$ とする．割当ペナルティの値 c_{sk} は大きいほど立会者 s の工事 k に対するスキルが不足していることを表し，難易度 w_k は大きいほど難しい工事であることを表す．工事立会者手配問題は，各工事にちょうど 1 人の立会者を割り当てる問題であり，その際各立会者に割り当てられる工事数は ν 以下，工事の総難易度は W 以下でなければならない．このとき全ての立会者の総移動時間と総割当てペナルティの重み付き和を最小化することが目的である．

対象とする実データが，立会者はたかだか 3 箇所しか巡回しないこと（すなわち $\nu = 3$）に着目し，実行可能なルートを全て列挙することによって得られる数理モデルを [1] では提案した．以下ではそのモデルを説明する．列挙の際，総難易度が W 以下となるルートのみを実行可能なルートとして候補に含める．また，列挙した各ルートの合計移動時間と合計割当ペナルティを計算しておく．そして，各工事が選択したルートの少なくとも一つに含まれる（すなわち被覆される）ように，列挙したルート候補の中から立会者の数 $|S|$ 以下のルートを選択する問題として定式化するのである．

実行可能なルートの集合を R とする．ルート $r \in R$ の合計移動時間を $\tilde{d}_r$，立会者 $s \in S$ がルート $r \in R$ を担当したときの合計割当ペナルティを $\tilde{c}_{rs}$，ルート r が工事 $k \in J$ を含むとき $a_{kr} = 1$，含まないとき $a_{kr} = 0$ とする．立会者 $s \in S$ がルート r を担当するとき $y_{rs} = 1$，担当しないとき $y_{rs} = 0$ となる決定変数 y_{rs} を用いて，問題は

$$\min \quad \sum_{r \in R} \sum_{s \in S} (\tilde{d}_r + \alpha \tilde{c}_{rs}) y_{rs} \tag{1}$$

$$\text{s.t.} \quad \sum_{r \in R} \sum_{s \in S} a_{kr} y_{rs} \geq 1, \qquad \forall k \in J \tag{2}$$

$$\sum_{r \in R} y_{rs} \leq 1, \qquad \forall s \in S \tag{3}$$

$$y_{rs} \in \{0, 1\}, \qquad \forall r \in R, \ \forall s \in S \tag{4}$$

と定式化できる．この問題は一般化上界制約付き集合被覆問題と呼ばれる問題であり [2]，集合被覆問題 (1), (2), (4) に，各立会者 s に対応するルートの中からたかだか一つしかルートを選択できないことを表す制約 (3) が加わっている．

2.3 新しい要望

対象とする実データでは，立会者はたかだか 3 箇所しか巡回しないことに着目し，集合被覆アプローチによる数理モデルで工事立会者手配問題の最適解の算出を図った [1]．しかしながら，その手配結果を実業務で運用したところ，不要な制約が存在することがわかった．具体的には，各立会者に割り当てられる工事件数は 3 件までという制約である．この問題に限らず，人手でスケジュールや割当を考える際には，現実的な時間で結果を出せるよう，人間が考えやすいように候補を絞るための制約を入れることがしばしばあり，この制約もそのようなものであったことが類推される．本制約がない場合，立会者の担当する工事件数が 4 件以上になることも可能となり，事前に実行可能なルートを全て列挙すると，列挙数が指数的に増加する．そのため，先行研究での集合被覆アプローチによる実行可能なルートを全列挙する手法は，実用的でない．

3. 制約生成法の適用

本節では，巡回セールスマン問題に対する制約生成法を紹介したのち，工事立会者手配問題に対する制約生成法に基づく解法ついて述べる．

3.1 巡回セールスマン問題に対する制約生成法

巡回セールスマン問題（traveling salesman problem, TSP）に対して Dantzig らが提案した数理モデル [3] を紹介する．巡回セールスマン問題は，訪問すべき点集合 J と点 $k, l \in J$ 間の移動コスト d_{kl} が与えられたとき，全ての点を 1 度ずつ訪問して出発地点に戻る巡回路の中で，総移動コストが最小のものを求める問題である．巡回路において点 k の直後に点 l を通るとき $x_{kl} = 1$，そうでないとき $x_{kl} = 0$ となる決定変数 x_{kl} を用いて，TSP は

$$\min \quad \sum_{k,l\in J} d_{kl}x_{kl} \tag{5}$$

$$\text{s.t.} \quad \sum_{l\in J} x_{kl} = 1, \qquad \forall k \in J \tag{6}$$

$$\sum_{l\in J} x_{kl} = \sum_{l\in J} x_{lk}, \qquad \forall k \in J \tag{7}$$

$$\sum_{k,l\in J'} x_{kl} \le |J'| - 1, \qquad \forall J' \subsetneq J,\ J' \neq \emptyset \tag{8}$$

$$x_{kl} \in \{0,1\}, \qquad \forall k,l \in J \tag{9}$$

と定式化できる．目的関数 (5) は総移動コストを最小化することを，制約 (6) と (7) は各点 k において，セールスマンが k 以外の点から枝上を移動して k に至り，k から枝上を移動して k 以外の点へ移ることを表す．制約 (8) は部分巡回路，すなわち J の点の一部のみを巡回する閉路が存在してはならないことを表し，部分巡回路除去制約と呼ばれる．この制約は TSP や配送計画問題を解く際にしばしば用いられる [4], [5], [6]．制約 (8) は訪問すべき点数に対し，制約数が指数オーダとなる [7]．そのため，制約 (8) を明示的に全て書き下すのは訪問すべき点数が大きい場合現実的でない．

そこで，制約 (8) について，全ての制約を満たす解空間を探索するのではなく，(6), (7), (9) 式および制約 (8) の一部を満たす最適解 x' に対しまだ満たしていない (8) の制約が存在するかどうかを確認する．存在しない場合は解 x' を最適解として出力し，そうでない場合は制約 (8) の中で x' が満たしていない制約を生成して数理モデルに追加する．解 x' に部分巡回路が存在しなくなるまで，この操作を繰り返す．このような制約の追加を逐次行う手法は制約生成法と呼ばれている [8]．

3.2 工事立会者手配問題への適用

本節では，工事立会者手配問題に対して，制約生成法に基づく解法について述べる．

問題入力の記号 $(J, S, d, c, w, \nu, W, \alpha)$ は 2.2 節と同じである．立会者 $s \in S$ が工事 k の次に l を訪れるとき $x_{skl} = 1$，そうでないとき $x_{skl} = 0$ となる決定変数 x_{skl} を用いて，工事立会者手配問題は

$$\min \quad \sum_{s\in S}\sum_{k,l\in J} (d_{kl} + \alpha c_{sk})x_{skl} \tag{10}$$

$$\text{s.t.} \quad \sum_{s\in S}\sum_{l\in J} x_{skl} = 1, \qquad \forall k \in J \tag{11}$$

$$\sum_{l\in J} x_{skl} = \sum_{l\in J} x_{slk}, \qquad \forall s \in S,\ k \in J \tag{12}$$

$$\sum_{k,l\in J} w_k x_{skl} \le W, \qquad \forall s \in S \tag{13}$$

$$\sum_{k,l\in J} x_{skl} \le \nu, \qquad \forall s \in S \tag{14}$$

$$\sum_{k,l\in J'} x_{skl} \le |J'| - \sum_{k'\in J\setminus J'} x_{sk'l'}, \qquad \forall s \in S,\ \forall J' \subsetneq J,\ J' \neq \emptyset,\ \forall l' \in J \setminus J' \tag{15}$$

$$x_{skl} \in \{0,1\}, \qquad \forall s \in S,\ \forall k,l \in J \tag{16}$$

Algorithm 1 工事立会者手配問題に対する制約生成法

1: $T \leftarrow \emptyset$.
2: **repeat**
3: 問題 (10)–(14)，(16)，(17) を解き，最適解 x' を求める．
4: **if** 解 x' の中である立会者のルートが 2 つ以上の部分巡回路に分かれている **then**
5: 1 人のルートが 2 つ以上に分かれている立会者全員のそのような部分巡回路全てに対し，そのような部分巡回路の各々に含まれる工事の集合を J' として，$T \leftarrow T \cup \{(s, J', l') \mid s \in S, l' \in J \setminus J'\}$ とする．
6: **end if**
7: **until** x' は問題 (10)–(16) の実行可能解である．

と定式化できる．目的関数 (10) は総移動時間と総割当ペナルティの重み付き和の最小化を表す．制約 (11)–(12) は各工事 k がちょうど 1 人の立会者に割り当てられることを，制約 (13) は各立会者 s に割り当てられる工事の総難易度が W 以下となることを，制約 (14) は各立会者が訪問する工事数が ν 以下となることを，制約 (15) は各立会者のルートが 2 つ以上の部分巡回路に分かれてはならないことを表す．

工事立会者手配問題に対する制約生成法について述べる．集合 $T \subseteq S \times (2^J \setminus \{\emptyset, J\}) \times J$ を用いて，制約 (15) のうち，T に含まれる (s, J', l') に対するもののみを考慮する制約を

$$\sum_{k,l\in J'} x_{skl} \le |J'| - \sum_{k'\in J\setminus J'} x_{sk'l'}, \qquad \forall (s, J', l') \in T \tag{17}$$

と定義する．まず，$T \leftarrow \emptyset$ とし，問題 (10)–(14)，(16)，(17) を解き，最適解 x' を求める．得られた解 x' が制約 (15) を満たす否かを確認し，満たさない場合，すなわち，立会者のルートの中で 2 つ以上の部分巡回路に分かれているものが存在する場合は，そのような部分巡回路の各々（例えば 2 人の立会者のルートがそれぞれ 2 つおよび 3 つの部分巡回路に分かれていたら合計 5 つの部分巡回路のそれぞれ）に対して，それに含まれる工事の集合を J' として，$J \setminus J'$ に含まれる l' と S に含まれる立会者 s の全ての組合せに対して (s, J', l') を T に追加する，すなわち，各部分巡回路に含まれる工事の集合 J' に対して，$T \leftarrow T \cup \{(s, J', l') \mid s \in S, l' \in J \setminus J'\}$ としたのち，問題 (10)–(14)，(16)，(17) を解き直す．解 x' に部分巡回路が存在しなくなるまで，この操作を続ける．工事立会者手配問題に対する制約生成法を Algorithm 1 に示す．

次に，制約 (15) とは異なる 2 つの部分巡回路除去制約と Algorithm 1 中の 5 行目の動作が異なる制約追加方針について述べる．

1 つめの部分巡回路除去制約は

$$\sum_{k,l\in J'} x_{skl} \le |J'| - \sum_{k'\in J} x_{sk'l'}, \qquad \forall s \in S,\ \forall J' \subsetneq J,\ J' \neq \emptyset,\ \forall l' \in J \setminus J' \tag{18}$$

である．制約 (15) では，右辺の第 2 項で，$J \setminus J'$ に含まれる k' に対する $x_{sk'l'}$ の総和をとっているのに対して，制約 (18) では，J に含まれる k' に対する $x_{sk'l'}$ の総和をとっており，制約 (15) より制約 (18) のほうが強い制約となる．

2 つめの部分巡回路除去制約は

$$\sum_{k,l\in J'} x_{skl} \leq |J'| - \frac{1}{|J \setminus J'|} \sum_{k'\in J} \sum_{l'\in J\setminus J'} x_{sk'l'}, \quad \forall s \in S,\ \forall J' \subsetneq J,\ J' \neq \emptyset \ (19)$$

である．制約 (19) は制約 (18) の代理制約であり，s と J' の組合せの各々に対して (18) では $|J \setminus J'|$ 個の制約があるのに対し，(19) ではこれらが 1 本に集約されている．

Algorithm 1 の 5 行目に示した集合 T への要素の追加方法を制約追加方針 1 と呼び，以下に示すものを制約追加方針 2 と呼ぶ．制約追加方針 2 では，Algorithm 1 中の 5 行目において，見つかった部分巡回路の各々に対して，それに含まれる工事の集合を J'，それらの工事を担当する立会者を s' として，$J \setminus J'$ に含まれる l' の各々に対して (s', J', l') を T に追加する，すなわち $T \leftarrow T \cup \{(s', J', l') \mid l' \in J \setminus J'\}$ とする．

4. 制約生成法による集合被覆モデルの改善

2.2 節では，重複しないルートを全列挙する集合被覆アプローチを用いた定式化を述べた．このとき，各立会者に対して可能なルートの数は，$|J|$ 個の工事から k $(1 \leq k \leq \nu)$ 個の工事を選ぶ $\binom{|J|}{k}$ 個の組合せの各々に対して $(k-1)!$ 通りのルートがあることから，$\sum_{k=1}^{\nu} \binom{|J|}{k}(k-1)!$ となる．しかし，各立会者のルートを生成する際，$|J|$ 個の工事から k $(1 \leq k \leq \nu)$ 個の工事を選ぶ組合せの各々に対して，合計割当ペナルティはルートによらず一定であるため，選んだ k 個の工事に対する最短巡回路のみを候補として考慮すれば十分である．この最短巡回路を得るためには，巡回セールスマン問題を解けばよい．そこで，各組合せに対する最短巡回路を求めるときに 3.1 節で述べた制約生成法を用いることで，高速に最短巡回路を得ることを狙う．

5. 計算実験

本節では計算実験の方法とその結果を述べる．

5.1 計算環境と問題例

全ての計算実験は，Intel Core i9-9900k CPU (3.60 GHz)，48 GB メモリを搭載した計算機上で行った．各手法の実装には，Python (ver. 3.6.5) と汎用ソルバーである Gurobi Optimizer (ver. 8.1) を用いた．制約生成法を実装する際には，Gurobi Optimizer の Lazy Constraints 機能を利用した．使用した問題例では，パラメータ W と α を $W = 9, 18$, $\alpha = 1$ と設定した．計算の制限時間は，3,600 秒とした．

本研究では，過去に現場の手配者が実際に割当を行った 36 件の工事と 14 人の立会者からなる実データをもとに計算実験で用いる問題例を生成した．これより規模が小さい問題例は実データの工事と立会者からランダムにいくつかを抽出したデータである．より規模の大きい問題例は実データに別の工事データの工事と立会者を加えたデータである．

5.2 制約生成法の計算結果

まず，3.2 節で紹介した制約生成法の比較結果を示す．実験結果を示す表 1 において，“$|J|$” は工事数を，“$|S|$” は立会者数を，“ν” は各立会者数に割り当てられる工事数の上限を，“Opt” は最適値を，“CGx-(y)” は制約追加方針 x と部分巡回路除去制約 (y) を用いた制約生成法を，“ArrTime” は最適解または最良解に初めて到達するまでに要した計算時間 (秒) を，“SolTime” は問題例が厳密に解けて探索が終了するまでに要した時間 (秒) を，“UB” は制限時間までに得られた最良の上界値を，“*” は制限時間内に得られた最良解が最適であったことを，“TL” は最適性の保証が得られなかったことを表す．“ArrTime” の表示が整数であるのは，Gurobi Optimizer の分枝カット探索のログに小数部の表示がなく情報が得られないためである．

制約追加方針については，制約追加方針 1 と制約追加方針 2 の間に大きな差は見られなかった．部分巡回路除去制約については，(15) と (18) の間に大きな差は見られず，(19) に比べて (15) と (18) のほうが厳密な最適解を得られた問題例が多く（紙面の都合上省略した $|J| \leq 16$ の問題例に対してそのような傾向を観測した），厳密な解が得られない問題に対しては，ほとんどの問題例で UB は同程度であったものの，(19) による UB が (15) と (18) に比べてかなり悪い問題例があった ($|J| = 36$, $\nu = 4$)．

5.3 集合被覆アプローチの計算結果

2.2 節の集合被覆アプローチと，それに 4 節の手法を導入したものの比較結果を示す．以下では，“Enum” は 2.2 節で紹介した，実行可能なルートを全列挙した上で集合被覆モデル (1)–(4) を解く手法を，“Enum+” は 4 節で紹介した，最短巡回路のみをルートとして列挙し，集合被覆モデルを解く手法を表す．表 2 に実験結果を示す．表中，“EnmNum” は生成されたルート数を，“EnmTime” はルート生成に要した時間を，“MdlTime” は集合被覆モデルを解くのに要した時間を，SolTime は EnmTime と MdlTime の合計を表す．また，“MO” は計算中にメモリ不足となり実行可能解が得られなかったことを表す．Enum，Enum+ ともに，MO と記したものを除いた全ての問題例に対して，制限時間内に厳密な最適解を得ており，その結果それらの問題例に対する UB は Opt と一致するため，UB 欄を省略した．

表 2 より，Enum はルート生成の時間は小さいが，ルート生成数が大きく，集合被覆モデルを解く時間が大きいこ

表 1 制約生成法における制約追加方針と部分巡回路除去制約の比較 ($W = 9$)

				CG1-(15)			CG1-(18)			CG1-(19)			CG2-(15)			CG2-(18)			CG2-(19)		
$\|J\|$	$\|S\|$	ν	Opt	SolTime	ArrTime	UB	SolTime	ArrTime	UB	SolTime	ArrTime	UB	SolTime	ArrTime	UB	SolTime	ArrTime	UB	SolTime	ArrTime	UB
18	7	3	7,363	103.16	13	*7,363	147.08	3	*7,363	630.98	41	*7,363	162.07	31	*7,363	132.21	10	*7,363	443.43	29	*7,363
18	7	4	6,889	TL	188	*6,889	TL	265	*6,889	TL	494	*6,889	TL	203	*6,889	TL	240	*6,889	TL	591	*6,889
18	7	5	6,741	TL	282	*6,741	TL	226	*6,741	TL	99	*6,741	TL	45	*6,741	TL	1,941	*6,741	TL	132	6,742
18	7	6	6,741	TL	402	*6,741	TL	58	*6,741	TL	1,871	*6,741	TL	150	*6,741	TL	410	*6,741	TL	231	*6,741
18	7	7	6,741	TL	101	*6,741	TL	1,006	*6,741	TL	1,438	6,742	TL	670	*6,741	TL	333	*6,741	TL	860	*6,741
18	7	8	6,741	TL	683	*6,741	TL	72	*6,741	TL	1,229	*6,741	TL	277	*6,741	TL	320	*6,741	TL	454	6,742
20	8	3	7,230	93.95	15	*7,230	TL	6	*7,230	379.22	48	*7,230	302.62	149	*7,230	219.55	49	*7,230	TL	111	*7,230
20	8	4	6,856	TL	3,488	*6,856	TL	1,568	*6,856	TL	3,090	*6,856	TL	1,244	*6,856	TL	771	*6,856	TL	1,225	*6,856
20	8	5	6,626	TL	85	6,627	TL	2,174	*6,626	TL	2,703	*6,626	TL	1,509	6,627	TL	809	*6,626	TL	2,237	*6,626
20	8	6	6,626	TL	823	*6,626	TL	2,543	*6,626	TL	473	6,627	TL	1,038	*6,626	TL	1,104	6,627	TL	288	6,629
20	8	7	6,626	TL	785	*6,626	TL	1,182	6,627	TL	769	6,627	TL	679	*6,626	TL	2,333	*6,626	TL	3,165	6,627
20	8	8	6,626	TL	892	6,629	TL	1,599	6,627	TL	1,408	6,723	TL	2,051	*6,626	TL	1,139	6,627	TL	2,099	6,627
36	14	3	14,251	TL	1,078	*14,251	TL	490	*14,251	TL	947	*14,251	TL	910	*14,251	TL	909	*14,251	TL	2,616	14,507
36	14	4	13,727	TL	3,341	13,844	TL	2,032	13,803	TL	293	17,706	TL	3,068	13,771	TL	3,411	13,779	TL	3,579	31,729
36	14	5	13,387	TL	2,463	13,389	TL	3,463	13,478	TL	1,239	13,395	TL	1,931	13,389	TL	1,977	13,779	TL	2,800	13,854
36	14	6	13,387	TL	2,978	13,782	TL	2,554	13,813	TL	3,175	13,392	TL	2,284	13,403	TL	2,641	13,390	TL	3,574	13,467
36	14	7	13,387	TL	3,036	13,522	TL	1,547	13,804	TL	3,576	14,331	TL	3,568	13,389	TL	3,140	13,840	TL	3,471	13,401
36	14	8	13,387	TL	2,229	13,389	TL	1,786	13,532	TL	3,010	13,393	TL	2,797	13,388	TL	2,333	13,397	TL	3,367	13,781

表 2 集合被覆アプローチにおけるルート生成法の比較 ($W = 9$)

				Enum				Enum+			
$\|J\|$	$\|S\|$	ν	Opt	EnmNum	EmnTime	MdlTime	SolTime	EnmNum	EmnTime	MdlTime	SolTime
18	7	3	7,363	5,052	0.01	0.36	0.37	959	0.73	0.13	0.87
18	7	4	6,889	44,604	0.08	1.19	1.27	2,607	2.17	0.17	2.34
18	7	5	6,741	153,324	0.29	4.97	5.26	3,513	3.20	0.25	3.45
18	7	6	6,741	218,124	0.37	6.53	6.90	3,603	3.24	0.24	3.49
18	7	7	6,741	218,124	0.38	6.63	7.02	3,603	3.25	0.24	3.49
18	7	8	6,741	218,124	0.41	6.60	7.01	3,603	3.32	0.24	3.56
20	8	3	7,230	7,024	0.02	0.23	0.25	1,314	1.02	0.09	1.11
20	8	4	6,856	73,024	0.15	2.31	2.47	4,064	3.36	0.29	3.65
20	8	5	6,626	320,344	0.66	7.55	8.21	6,125	5.59	0.45	6.04
20	8	6	6,626	601,144	1.26	24.59	25.85	6,515	6.22	0.47	6.69
20	8	7	6,626	651,544	1.61	35.32	36.93	6,525	6.39	0.50	6.89
20	8	8	6,626	651,544	1.23	35.93	37.15	6,525	6.38	0.50	6.88
36	14	3	14,251	43,110	0.13	2.73	2.86	7,635	5.50	0.90	6.40
36	14	4	13,727	667,350	1.52	36.95	38.47	33,645	29.70	11.94	41.64
36	14	5	13,387	3,626,670	7.09	MO	MO	58,306	55.77	14.53	70.31
36	14	6	13,387	8,022,990	16.97	MO	MO	64,412	66.37	14.92	81.30
36	14	7	13,387	9,529,950	25.42	MO	MO	64,711	73.27	9.25	82.51
36	14	8	13,387	9,529,950	44.92	MO	MO	64,711	94.79	10.26	105.05

とが観測できる．一方，Enum+はルート生成の時間は大きいが，ルート生成数が小さく，集合被覆モデルを解く時間が小さいことが観測できる．求解に要する総時間 SolTime を比べると，ν が 4 以下の問題例では，Enum でもルート生成数は比較的小さく，Enum+より Enum のほうが高速に最適解を得ている一方，ν が大きくなるにつれ Enum のルート生成数が大きくなり，ν が 5 以上の問題例では，Enum より Enum+のほうが高速に最適解を得ている．この傾向は $|J|$ が大きいほど強く，$|J|$ や ν が大きい問題例に対しては Enum+のほうが有効であることが確認できた．

本研究で用いた問題例は実データに近いデータであり，工事の数が立会者の数の 3 倍以内に収まっている．ν を大きくすると計算により得られた最適値は改善しているが，表 2 に示す全ての問題例において ν を 5 より大きな値にしても $\nu = 5$ の場合と最適値は変わらないことがわかった．これは，$W = 9$ という制約によって，実行可能なルートが制限され，ν を大きくしても生成されるルートがそれほど増えないことが原因と思われる．実際，ルート生成数を示す EnmNum の値は，6 以上の ν ではそれほど変化していない．

5.4 CG と Enum+の計算結果

CG と Enum+の比較結果を表 3–5 に示す．表 3 は $W = 9$ の場合，表 4 は $W = 18$ であること以外は表 3 と同じ問題例で，表 5 はこれらに比べて $|S|$ が小さい (その結果平均ルート長が表 3 と表 4 の問題例より長い) 問題例である．また，[1] で構築した数理モデルによる計算結果も比較のためこれらの表に示す (“TOM59Model” と記す)．このモデルを解くのに [1] では汎用ソルバーとして NUOPT を用いたが，本研究では計算環境を揃えるために Gurobi Optimizer を用いた．表中の “UNK” は本研究で検証したいずれの方法を用いても，制限時間内に最適解が判明しなかったことを表す．Enum+は，$W = 9$ の全ての問題例 (表 3) を厳密に解いて探索を終了した．$W = 18$ の問題例 (表 4 と 5) についても，表 4 の $|J| = 36$, $\nu = 6, 7, 8$ の 3 つの問題例を除く全ての問題例が Enum+によって厳密に解けた．それらの問題例を厳密に解くのに要した時間 SolTime は，表 3 と表 4 の問題例では Enum+が最も小さいが，表 5 の問題例では Enum+は CG よりもやや大きい時間を要した．上述の 3 つの問題例については，Enum+は計算中にメモリ不足となり実行可能解が得られなかった．CG と TOM59Model は，問題例を厳密に解くのに要する時間は Enum+よりも大きい場合が多く，制限時間内に計算が終了せず (つまり最適性の保証が得られず) TL と記されているものが多いものの，表 3–5 の全ての問題例に対して制限時間内に実行可能解を得た．TOM59Model は，UB 欄に記した暫定値が最適値に一致している (i.e., *印のついた) 問題例が，表 3 と表 4 それぞれで 18 問中 2 問，表 5 で 10 問中 4 問と少ないものの，$|J| \leq 20$ の問題例に対しては，いずれも最適値からの相対誤差 10% 以下の解が得られている．一方，$|J| = 36$ の問題例に対しては，最適値 (最適値がわからない問題例については最良値) からの相対誤差が 10% を超えた問題例が表 3 と表 4 それぞれで 6 問中 5 問であった．それに対して，CG は，厳密に解けた問題例は表 3 で 18 問中 1 問，表 4 で 18 問中 4 問と少ないものの，UB 欄に記した暫定値が最適値に一致している問題例は，表 3 と 4 それぞれで 18 問中 11 問であった．また，最適値が判明している問題例に対して最適値からの相対誤差は表 3 で最大 3.2% ，表 4 で最大 1.1% であった．表 5 の問題例は CG によって全て厳密に解けており，求解に要した時間 SolTime は Enum+より小さい．

TOM59Model は整数計画問題を 1 度解けばよく，制約生成のような反復計算を含まないため，数理モデルを記述

表 3　CG と Enum+の比較結果 ($W = 9$)

				CG1-(18)			Enum+		TOM59Model		
$\|J\|$	$\|S\|$	ν	Opt	SolTime	ArrTime	UB	SolTime	ArrTime	SolTime	ArrTime	UB
18	7	3	7,363	147.08	3	*7,363	0.87	0	TL	1,265	*7,363
18	7	4	6,889	TL	265	*6,889	2.34	2	TL	3,205	6,981
18	7	5	6,741	TL	226	*6,741	3.45	3	TL	3,176	6,750
18	7	6	6,741	TL	58	*6,741	3.49	3	TL	960	6,977
18	7	7	6,741	TL	1,006	*6,741	3.49	3	TL	2,682	6,898
18	7	8	6,741	TL	72	*6,741	3.56	3	TL	1,999	6,747
20	8	3	7,230	TL	6	*7,230	1.11	1	TL	162	*7,230
20	8	4	6,856	TL	1,568	*6,856	3.65	3	TL	3,497	7,055
20	8	5	6,626	TL	2,174	*6,626	6.04	6	TL	1,172	6,733
20	8	6	6,626	TL	2,543	*6,626	6.69	6	TL	407	7,337
20	8	7	6,626	TL	1,182	6,627	6.89	6	TL	344	7,281
20	8	8	6,626	TL	1,599	6,627	6.88	6	TL	3,584	6,717
36	14	3	14,251	TL	490	*14,251	6.40	6	TL	3,281	15,321
36	14	4	13,727	TL	2,032	13,803	41.64	39	TL	3,399	15,368
36	14	5	13,387	TL	3,463	13,478	70.31	70	TL	3,189	17,439
36	14	6	13,387	TL	2,554	13,813	81.30	81	TL	3,486	16,739
36	14	7	13,387	TL	1,547	13,804	82.51	81	TL	3,081	15,115
36	14	8	13,387	TL	1,786	13,532	105.05	104	TL	3,597	22,876

表 4　CG と Enum+の比較結果 ($W = 18, |S| \geq 7$)

				CG1-(18)			Enum+		TOM59Model		
$\|J\|$	$\|S\|$	ν	Opt	SolTime	ArrTime	UB	SolTime	ArrTime	SolTime	ArrTime	UB
18	7	3	7,363	TL	3	*7,363	1.39	1	TL	119	*7,363
18	7	4	6,755	TL	398	*6,755	6.34	5	TL	3,476	6,757
18	7	5	6,297	TL	123	*6,297	23.67	23	TL	2,421	6,314
18	7	6	5,652	1,347.61	239	*5,652	71.01	69	TL	1,721	5,736
18	7	7	5,652	936.06	88	*5,652	159.08	158	TL	1,794	5,656
18	7	8	5,652	1,275.87	61	*5,652	241.95	239	TL	2,391	5,845
20	8	3	7,230	90.92	8	*7,230	1.78	0	TL	1,897	*7,230
20	8	4	6,507	TL	29	*6,507	9.34	8	TL	1,036	6,511
20	8	5	5,966	TL	57	*5,966	39.17	38	TL	1,948	6,083
20	8	6	5,823	TL	769	5,827	136.72	132	TL	3,424	5,966
20	8	7	5,630	TL	2,615	*5,630	335.80	335	TL	3,000	6,171
20	8	8	5,630	TL	258	5,632	596.03	595	TL	2,385	5,870
36	14	3	14,251	TL	343	*14,251	11.17	10	TL	1,533	14,540
36	14	4	12,915	TL	3,010	13,046	109.03	104	TL	22	26,666
36	14	5	11,940	TL	2,640	11,944	800.57	799	TL	1,949	14,329
36	14	6	UNK	TL	2,900	11,755	MO	MO	TL	2,980	16,316
36	14	7	UNK	TL	3,314	11,760	MO	MO	TL	3,496	14,711
36	14	8	UNK	TL	3,226	11,299	MO	MO	TL	3,103	17,550

して汎用ソルバーを適用するためにかかる手間が少なく済む．そのため，$|J|$ が小さい問題例 ($|J| \leq 20$) しか解く必要がなく，解が多少悪くても (e.g., 相対誤差 10% 程度) 許容できる場合に有用である．Enum+は表 4 の 3 問を除き，高速に厳密な最適解を得ることができた．CG では SolTime が TL になっている問題例でも Enum+は短時間で最適解を得ている場合が多く，CG の SolTime が Enum+より小さい問題例に対しても，表 5 の最初の 1 問を除き，Enum+の時間は CG の数倍程度に収まっている．したがって，本論文で用いた問題例に対しては，厳密な最適解を得るためのアルゴリズムの性能は総じて Enum+の方が高いといえる．CG は調べた全ての問題例に対して実行可能解を得ており，最適値が判明した全ての問題例に対してその誤差は 3.2% 以下と小さい．$|J|$, ν および W が共に大きい問題例の場合，Enum+では実行可能解を得られない可能性があることを考慮すると，試した問題例ではそのようなことの起こっていない CG は有用であるといえる．以上より，これらを実践的に利用する方法として，CG と Enum+の 2 つの実装が可能な場合には，まず Enum+を適用し，実行可能解が得られないときには CG を用いる使い方が，2 つの実装が難しい場合には，CG を実装して適用するのが効率的であると考えられる．

6. まとめ

本研究では，工事立会者手配問題に対して，各立会者に割り当てられる工事件数が大きい場合でも現実的な時間で良質な解が得られる手法を得るため，制約生成法に基づく手法と集合被覆アプローチを検証した．工事の部分集合の各々に対して最適なルートのみを生成する集合被覆アプローチ Enum+は，ルート候補が多くメモリ不足になるような問題例を除き，実験に用いたほとんどの問題例に対して高速に厳密な最適解を得ることができた．Enum+では短時間で厳密に解けた問題例でも，制約生成法に基づく手法 CG では制限時間内に最適性の保証が得られなかったものも多く，CG によって Enum+より早く厳密に解けた問題例に対しても両者の差はそれほど大きくなかった．したがって，本論文で用いた問題例に対しては，厳密な最適解を得るためのアルゴリズムの性能は総じて Enum+の方が高いといえる．CG は調べた全ての問題例に対して実行可能解を得ており，最適値が判明した全ての問題例に対してその誤差は 3.2% 以下と小さい．Enum+では実行可能解を得られない可能性があることを考慮すると，試した問題例ではそのようなことの起こっていない CG は有用であるといえる．計算実験に用いた問題例は実データをもとに生成したものであり，検証した手法がそのような問題例に対して，現実的な時間で良質の解を得ることを確認した．

表 5　CG と Enum+の比較結果 ($W = 18, |S| \leq 3$)

				CG1-(18)			Enum+		TOM59Model		
$\|J\|$	$\|S\|$	ν	Opt	SolTime	ArrTime	UB	SolTime	ArrTime	SolTime	ArrTime	UB
15	2	8	7,686	1.56	1	*7,686	33.35	33	TL	154	*7,688
15	3	5	7,690	4.38	2	*7,690	5.78	5	1,064.63	72	*7,690
15	3	6	7,611	10.27	7	*7,611	13.50	13	TL	246	7,613
15	3	7	7,117	17.34	9	*7,117	23.16	23	TL	1,025	*7,117
15	3	8	7,093	15.95	9	*7,093	29.02	29	TL	306	7,289
18	3	6	9,428	27.68	10	*9,428	45.23	45	TL	224	*9,428
18	3	7	9,367	48.32	11	*9,367	101.11	101	TL	2,117	9,428
18	3	8	8,978	44.61	22	*8,978	152.31	152	TL	419	9,018
20	3	7	9,761	156.31	103	*9,761	216.04	216	TL	1,859	10,108
20	3	8	9,761	127.85	36	*9,761	372.46	372	TL	424	10,038

参考文献

[1] 高須賀将秀, 柳浦睦憲, 工事手配業務に対する数理最適化の活用と意思決定の支援, 情報処理学会論文誌 数理モデル化と応用, 14 (2021), 112–120.
[2] S. Umetani, M. Arakawa, M. Yagiura, Relaxation heuristics for the set multicover problem with generalized upper bound constraints, *Computers & Operations Research*, 93 (2018), 90–100.
[3] G. Dantzig, R. Fulkerson, S. Johnson, Solution of a large-scale traveling-salesman problem, *Journal of the Operations Research Society of America*, 2 (1954), 393–410.
[4] U. Pferschy, R. Staněk, Generating subtour elimination constraints for the TSP from pure integer solutions, *Central European Journal of Operations Research*, 25 (2017), 231–260.
[5] R. H. Pearce, Towards a General Formulation of Lazy Constraints, Doctoral Dissertation, School of Mathematics and Physics, The University of Queensland, 2019.
[6] P. Toth, D. Vigo, The Vehicle Routing Problem, *SIAM*, (2002).
[7] D. L. Applegate, R. E. Bixby, V. Chvátal, W. J. Cook, The Traveling Salesman Problem. *Princeton University Press*, (2011).
[8] H. Crowder, M. W. Padberg, Solving large-scale symmetric travelling salesman problems to optimality, *Management Science*, 26 (1980), 495–509.