F-050

回収と配達の両方に時間枠をもつ MAPD
MAPD with Time Windows for both Pickup and Delivery

渡邉 伸二† 平山 勝敏† 沖本 天太†
Shinji Watanabe Katsutoshi Hirayama Tenda Okimoto

1. はじめに

Multi-Agent Pickup and Delivery 問題（MAPD）とは，グラフ上のある頂点におけるアイテムの回収（pickup）と別の頂点におけるアイテムの配達（delivery）の両方を伴う複数のタスクに対し，グラフ上の任意のエージェントを各タスクに割り当て，かつ，互いに衝突のない最適な回収・配達経路を求める問題である．最近，MAPD 問題の回収に時間枠を導入した問題の定式化とアルゴリズムが提案された[1]．一方，現実には，フードデリバリー等の例に見られるように回収と配達の両方に時間枠がある場合も少なくない．本研究では，従来の定式化に修正を加えることで，MAPD 問題において回収と配達の両方に時間枠を導入した新たな問題の定式化とこの問題を解く列生成法に基づくアルゴリズムを提案する．また，評価実験を通して，従来の定式化と新たな定式化のそれぞれのモデルの規模およびアルゴリズムの性能を比較する．

2. MAPD 問題

文献[1]では，グリッド状の空間を複数のロボットが移動するという想定の下で問題が設定されている．その空間には，ロボットが通行できない箇所を示す障害物と回収されるべきアイテムが複数配置されている．各アイテムには，大きさと回収することで得られる報酬，そして回収されるべき時間帯を定める時間枠が設定されている．各ロボットは発進地点（launcher）と呼ばれる地点から移動を始める．ただし残存ロボット（extant robot）と呼ばれるロボットは例外的に発進地点以外の場所から移動を始めることが許されている．残存ロボットはアイテムの回収の有無にかかわらず，必ず発進地点に帰還しなければならない．各ロボットには容量が設定されており，回収したアイテムの大きさの合計がその容量を上回ることはない．なお残存ロボットの容量は通常のロボットの容量以下である．通常，各ロボットは発進地点から移動を始め，空間内に存在するアイテムを回収し，それらを発進地点まで持ち帰ってくることが求められる．ロボット 1 台が稼働するには稼働コストがかかり，それに加えて移動するごとに移動コストが必要になる．この問題の目的は，ロボット同士の衝突を避けたうえで，アイテムの回収によって得られる報酬の和と全ロボットの移動コストと稼働コストの和の差で計算される利益が最大になるような複数のロボットの移動経路を求めることにある．

本研究では上記の問題設定に加えて，各アイテムに固有の配達位置とその時間枠を設定した．つまり各ロボットは発進地点からアイテムの回収に向かい，アイテムを指定の位置に配達した後，発進地点に帰還することになる．そしてその回収と配達はそれぞれの時間枠内で行われる．

† 神戸大学大学院海事科学研究科 Graduate School of Maritime Sciences, Kobe University

3. 時間枠付き MAPD

3.1 回収時間枠付き MAPD

3.1.1 回収時間枠付き時間拡張グラフ

グラフ$G=(V,E)$において，頂点集合Vに含まれる任意の頂点vに対して，vと隣接する頂点とv自身を合わせて次の時刻に到達可能である頂点とする．時間拡張グラフとは，時刻 0 から上限時刻μまでの各時刻tについて，グラフ$G=(V,E)$の頂点集合Vをコピーした系列$\{V^{(0)},\dots,V^{(t)},V^{(t+1)},\dots,V^{(\mu)}\}$を作り，$V^{(t)}$に含まれているすべての頂点$v$から次の時刻に到達可能である$V^{(t+1)}$のすべての頂点に有向枝を張ることで得られるグラフである．図 1 に簡単なグラフとその時間拡張グラフ（上限時刻は 2）を示す．

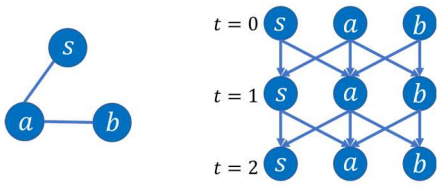

図 1：グラフ（左側）とその時間拡張グラフ（右側）

3.1.2 マスター問題

ロボットの移動経路のうち発進地点が終着点となるものをルートと呼ぶ．文献[1]では，考えうるルートが列挙されているものとして，MAPD 問題を 0-1 整数計画問題として定式化している．以下，この 0-1 整数計画問題（マスター問題）で使用される変数および定数の意味を示す．

$(\mathcal{P},\mathcal{E})$：時間拡張グラフ
$g\in\mathcal{G}$：実行可能なルートの集合
$d\in\mathcal{D}$：アイテムの集合
$r\in\mathcal{R}$：残存ロボットの集合
$t\in\mathcal{T}$：時刻の集合
$p\in\mathcal{P}$：時間拡張グラフ上の頂点
$e\in\mathcal{E}$：時間拡張グラフ上の有向枝
$\bar{e}\in\bar{\mathcal{E}}$：交換移動を示す枝の組み合わせ

N：利用可能なロボットの数の上限
Γ_g ：ルートgを採用することによる利益
G_{dg} ：ルートgでアイテムdが回収されていれば 1
そうでなければ 0 となる変数
G_{rg} ：ルートgが残存ロボットrによるものであれば 1
そうでなければ 0 となる変数
G_{tg} ：ルートgにおいて，時刻tで移動中であれば 1
そうでなければ 0 となる変数

G_{pg} ：ルートgが$\mathcal{P}$上の頂点pを通っていれば 1
そうでなければ 0 となる変数
G_{eg} ：ルートgが$\mathcal{E}$上の枝eを通っていれば 1
そうでなければ 0 となる変数
$[t_d^-, t_d^+]$：アイテムdの回収されるべき時間枠
θ_d：アイテムdを回収することで得られる報酬
θ_1：ロボットの 1 単位時間当たりの稼働コスト
θ_2：ロボットの 1 単位時間当たりの移動コスト
c_0：各ロボットの容量
c_d：アイテムdの大きさ

γ_g：解にルートgが選ばれれば 1，そうでなければ 0 となる決定変数

そして以下の式(1)から式(6)が文献[1]で提案されたマスター問題である．

$$\max_{\gamma_g \in \{0,1\}\ \forall g \in \mathcal{G}} \sum_{g \in \mathcal{G}} \Gamma_g \gamma_g \tag{1}$$

$$\sum_{g \in \mathcal{G}} G_{dg} \gamma_g \leq 1 \quad \forall d \in \mathcal{D} \tag{2}$$

$$\sum_{g \in \mathcal{G}} G_{tg} \gamma_g \leq N \quad \forall t \in \mathcal{T} \tag{3}$$

$$\sum_{g \in \mathcal{G}} G_{rg} \gamma_g = 1 \quad \forall r \in \mathcal{R} \tag{4}$$

$$\sum_{g \in \mathcal{G}} G_{pg} \gamma_g \leq 1 \quad \forall p \in \mathcal{P} \tag{5}$$

$$\sum_{g \in \mathcal{G}} (G_{e_1 g} + G_{e_2 g}) \gamma_g \leq 1 \quad \forall (e_1, e_2) \in \bar{\mathcal{E}} \tag{6}$$

式(1)は利益の総和が最大になるようなルートの組み合わせを得るための目的関数である．式(2)は各アイテムが回収される回数を 1 回以下とする制約である．式(3)はどの時刻においても同時に稼働できるロボットを最大N台までとする制約である．式(4)は残存ロボットが必ず発進地点に帰還しなければならないことを示している．式(5)ではある頂点に存在できるロボットは 1 台まで，つまり複数のロボットが 1 つの頂点に同時に存在することを禁止している．式(6)はロボットの交換移動を禁止している．交換移動とは，実際のグラフにおいて 1 つの枝を 2 台のロボットが同時に逆方向へ移動することである．例えば，あるロボットが頂点aから頂点bへ移動すると同時に，他のロボットが頂点bから頂点aへ移動することが交換移動に該当する．

3.1.3 列生成法

3.1.2 での定式化により，実行可能なルートがすべて列挙されていれば MAPD 問題を解くことができる．しかし現実には実行可能なルートは無数に考えられるため，それらを列挙することは困難である．そこで文献[1]では列生成法を用いたアルゴリズムが提案されている．このアルゴリズムでは，制限されたマスター問題（RMP） と価格問題の 2 つを解くことになる．

RMP はマスター問題の決定変数γ_gを 0 以上 1 以下の連続変数に緩和し．扱う実行可能なルートの集合を$\mathcal{G}$ではなく$\mathcal{G}$の部分集合である$\hat{\mathcal{G}}$と置き換えることで得られる．文献[1]で提案されている列生成法の手続きにおいて，価格問題を解く目的は，RMP の目的関数の値を改善させる可能性がある新しいルートを生成し，それを$\hat{\mathcal{G}}$に追加することである．ここで追加されるルートは式(8)に示す$\bar{\Gamma}_g$の最大化問題を解くことで得られる．

$$\bar{\Gamma}_g = \Gamma_g - \sum_{i \in \mathcal{I}} \lambda_i G_{ig} \tag{8}$$

なお$i \in \mathcal{I} = \{\mathcal{D} \cup \mathcal{T} \cup \mathcal{P} \cup \mathcal{E} \cup \mathcal{R}\}$として，$\lambda_d, \lambda_t, \lambda_r, \lambda_{e_1 e_2}$はそれぞれ RMP の制約式(2)から(6)に対応する双対変数である（$\lambda_{e_1} = \lambda_{e_2} = \lambda_{e_1 e_2}$とする）．文献[1]で提案されているアルゴリズムの擬似コードを Algorithm 1 に示す．

Algorithm 1: Optimization via Column Generation
1: repeat
2: $\gamma, \lambda \leftarrow$ Solve the RMP over $\hat{\mathcal{G}}$
3: $g^* \leftarrow \arg\max_{g \in \mathcal{G}} \bar{\Gamma}_g$
4: $\hat{\mathcal{G}} \leftarrow \hat{\mathcal{G}} \cup \{g^*\}$
5: until $\bar{\Gamma}_{g^*} \leq 0$
6: $\gamma \leftarrow$ Solve ILP in (1)-(6) over $\hat{\mathcal{G}}$ instead of $\mathcal{G}$
7: Return γ

図 2：文献[1]のアルゴリズムの擬似コード

3.1.4 価格問題

文献[1]では，価格問題は式(8)の最大化を目的とした elementary resource-constrained shortest-path problem[2]として定式化されている．この問題ではマスター問題の時間拡張グラフ$(\mathcal{P}, \mathcal{E})$に頂点と枝を追加した時間拡張グラフ$(\mathcal{P}^+, \mathcal{E}^+)$を扱う．なお追加される頂点は次の 4 種類である．

P_{dt}：アイテムdをその時間枠$[t_d^-, t_d^+]$内の時刻tに回収したことを示す頂点
P_r：残存ロボットの使用を示す頂点
P_+：ソースノード
P_-：シンクノード

以下に文献[1]における価格問題での時間拡張グラフの枝の重みと決定変数の設定について説明する．時間拡張グラフ中の時刻t_iにおける頂点p_iと時刻$t_j = t_i + 1$における頂点p_jについて，p_iからp_jの移動が現実には単一の頂点での待機を示している場合，枝(p_i, p_j)の重みは$k_{p_i p_j} = \theta_1 - \lambda_{t_j} - \lambda_{p_j}$となる．$p_i$から$p_j$の移動が現実には異なる 2 頂点間の移動を示している場合，枝(p_i, p_j)の重みは$k_{p_i p_j} = \theta_1 + \theta_2 - \lambda_e - \lambda_{t_j} - \lambda_{p_j}$となる．またどちらの場合においても，決定変数$x_{p_i p_j g}$はルート$g$に枝$(p_i, p_j)$が含まれており，かつ，$p_i$でアイテムが回収されていない場合に 1，そうでなければ 0 となる．時間拡張グラフ上の頂点pはアイテムdが配置されている頂点を示しているものとする．枝(p, p_{d_t})の重みは$k_{p p_{d_t}} = \theta_d - \lambda_d$となる．決定変数$x_{p p_{d_t} g}$はルート$g$において，アイテム$d$が時刻$t$に回収されていれば 1，そうでなければ

0 となる．p_{d_t}から時間拡張グラフ上の頂点pへの枝(p_{d_t}, p)の重み$k_{p_{d_t}p}$は同一の頂点での待機・移動を示す枝と同じ様に設定される．ただしアイテムの回収を示す頂点間に枝が張られることはない．決定変数$x_{p_{d_t}p_j g}$はルートgにおいて，アイテムdを時刻$t_j - 1$に回収した後p_jに待機・移動していれば 1，そうでなければ 0 となる．時刻tにおける発進地点を示す頂点をp_{0t}とする．ソースノードp_+からの枝(p_+, p_{0t})の重みは$k_{p_+p_{0t}} = \theta_1 - \lambda_t - \lambda_{p_{0t}}$となる．決定変数$x_{p_+p_{0t}g}$はルート$g$において発進地点からの移動が時刻$t$に始まっていれば 1，そうでなければ 0 となる．またソースノードから残存ロボットの使用を示す頂点p_rへの枝(p_+, p_r)の重みは$k_{p_+p_r} = \theta_1 - \lambda_r - \lambda_{t=1} - \lambda_{p_r}$となる．決定変数$x_{p_+p_r g}$はルート$g$が残存ロボット r を使用していれば 1，そうでなければ 0 となる．時刻 0 に残存ロボットが存在する頂点をp_{0r}とする．p_rから出ていく枝は(p_r, p_{0r})のみであり，その重みは$k_{p_r p_{0r}} = 0$となる．p_{0t}からシンクノードp_-への枝(p_{0t}, p_-)の重みは$k_{p_{0t}p_-} = 0$となる．決定変数$x_{p_{0t}p_- g}$はルートgの終着点がp_{0t}なら 1，そうでなければ 0 となる．

上記を式(8)の最大化問題の定式化に用いたものが以下となる．なおどのルートにおいても$\bar{\Gamma}_g = \sum_{(p_i, p_j)\in\varepsilon^+} \kappa_{p_i p_j} x_{p_i p_j g}$となる[1].

$$\max_{x_{p_ip_j}\in\{0,1\}} \sum_{\forall(p_i,p_j)\in\mathcal{E}^+} \kappa_{p_ip_j} x_{p_ip_j} \tag{9}$$

$$\sum_{(p_i,p_j)\in\mathcal{E}^+} x_{p_ip_j} - \sum_{(p_j,p_i)\in\mathcal{E}^+} x_{p_jp_i} = [p_i = p_+] - [p_i = p_-] \quad \forall p_i \in \mathcal{P}^+ \tag{10}$$

$$\sum_{d\in D} c_d \sum_{t_d^-\le t\le t_d^+} \sum_{(p,p_{d_t})\in\mathcal{E}^+} x_{pp_{d_t}} \le c_0 + \sum_{r\in\mathcal{R}} (c_r - c_0) x_{p_+p_r} \tag{11}$$

$$\sum_{t_d^-\le t\le t_d^+} \sum_{(p,p_{d_t})\in\mathcal{E}^+} x_{pp_{d_t}} \le 1 \quad \forall d \in D \tag{12}$$

式(9)はルートの利益を示す目的関数である．式(10)は各頂点の流量制約を表す．式(11)はルートgにおけるロボットの容量制約である．c_rは残存ロボットrの容量である．式(12)は各アイテムが回収されるのは高々1 度であることを約束している．式(9)から式(12)を用いた最適化は NP 困難であり，アイテムの総数$|\mathcal{D}|$に対して指数関数的に計算複雑度が増すことが示されている[3].

3.2 回収配達時間枠付き MAPD

本研究では，文献[1]で提案された価格問題の時間拡張グラフと定式化に修正を加えることで，各アイテムの配達の時間枠を扱えるようにした．以下，その修正点について説明する．

3.2.1 回収配達時間枠付き時間拡張グラフ

本研究では，$(\mathcal{P}^+, \mathcal{E}^+)$にアイテムの配達を示す頂点とそれに接続する枝を追加する．アイテムdの回収されるべき時間枠を$[t_{d_pic}^-, t_{d_pic}^+]$，配達されるべき時間枠を$[t_{d_del}^-, t_{d_del}^+]$とする．また$[t_{d_pic}^-, t_{d_pic}^+]$内の時刻$t$におけるアイテム$d$の回収を示す頂点を$p_{d_{t_pic}}$，$[t_{d_del}^-, t_{d_del}^+]$内の時刻$t$におけるアイテム$d$の配達を示す頂点を$p_{d_{t_del}}$とする．このとき，アイテム$d$を回収したことを示す枝$(p, p_{d_{t_pic}})$の重み$k_{pp_{d_{t_pic}}}$と，$p_{d_{t_pic}}$から出る枝の重みは 3.1.4 で述べた設定と同じである．また枝$(p, p_{d_{t_del}})$の重みは 0 としており，決定変数$x_{pp_{d_{t_del}}g}$はルートgにおいて，アイテムdが時刻tに配達されていれば 1，そうでなければ 0 となる．枝$(p_{d_{t_del}}, p)$の重みは$p_{d_{t_pic}}$から出る枝と同様である．なお本研究ではアイテムの回収，配達にはそれぞれ 1 単位時間かかるものと考えて，$p_{d_{t_pic}}$と$p_{d_{t_del}}$からは待機状態を示す枝のみを張ることとした．後の評価実験は既存研究と本研究のどちらもこの設定の下で行っている．

3.2.2 価格問題

3.2.1 では既存研究の価格問題における時間拡張グラフ$(\mathcal{P}^+, \mathcal{E}^+)$にアイテムの配達を示す枝と頂点を追加した．またアイテムの配達を考慮するためには価格問題の定式化にも修正が必要になる．以下に修正後の価格問題の定式化を示す．なお赤字で記している部分は修正した箇所である．

$$\max_{x_{p_ip_j}\in\{0,1\}} \sum_{\forall(p_i,p_j)\in\mathcal{E}^+} \kappa_{p_ip_j} x_{p_ip_j} \tag{9}$$

$$\sum_{(p_i,p_j)\in\mathcal{E}^+} x_{p_ip_j} - \sum_{(p_j,p_i)\in\mathcal{E}^+} x_{p_jp_i} = [p_i = p_+] - [p_i = p_-] \quad \forall p_i \in \mathcal{P}^+ \tag{10}$$

$$\sum_{d\in D} c_d \left(\sum_{0\le t\le t^*} \sum_{(p,p_{d_{t\_pic}})\in\mathcal{E}^+} x_{pp_{d_{t\_pic}}} - \sum_{0\le t\le t^*} \sum_{(p,p_{d_{t\_del}})\in\mathcal{E}^+} x_{pp_{d_{t\_del}}} \right) \le c_0 + \sum_{r\in\mathcal{R}} (c_r - c_0) x_{p_+p_r} \quad \forall t^* \in T \tag{11}$$

$$\sum_{t_{d\_pic}^-\le t\le t_{d\_pic}^+} \sum_{(p,p_{d_{t\_pic}})\in\mathcal{E}^+} x_{pp_{d_{t\_pic}}} \le 1 \quad \forall d \in D \tag{12}$$

$$\sum_{t_{d\_del}^-\le t\le t_{d\_del}^+} \sum_{(p,p_{d_{t\_del}})\in\mathcal{E}^+} x_{pp_{d_{t\_del}}} \le 1 \quad \forall d \in D \tag{13}$$

$$\sum_{0\le t\le t^*} \sum_{(p,p_{d_{t\_pic}})\in\mathcal{E}^+} x_{pp_{d_{t\_pic}}} - \sum_{0\le t\le t^*} \sum_{(p,p_{d_{t\_del}})\in\mathcal{E}^+} x_{pp_{d_{t\_del}}} \ge 0 \quad \forall d \in D, \forall t^* \in T \tag{14}$$

$$\sum_{t_{d\_del}^-\le t\le t_{d\_del}^+} \sum_{(p,p_{d_{t\_del}})\in\mathcal{E}^+} x_{pp_{d_{t\_del}}} - \sum_{t_{d\_pic}^-\le t\le t_{d\_pic}^+} \sum_{(p,p_{d_{t\_pic}})\in\mathcal{E}^+} x_{pp_{d_{t\_pic}}} = 0 \quad \forall d \in \mathcal{D} \tag{15}$$

修正した箇所について述べる．式(11)は各時刻におけるロボットの容量制約である．この定式化ではアイテムの配達によってロボットの容量の増加・減少が考えられるため，それに対応した形になる．式(12)，式(13)はそれぞれ各アイテムの回収，配達を 1 度以下に制約している．式(14)はアイテムの配達が回収よりも先に行われることがないことを保証している．式(15)はアイテムの回収と配達は同じ回数だけ行われるという制約である．これは式(12)，式(13)と合わせて，1 度回収されたアイテムは必ず配達されることを保証する．このように価格問題を修正することで配達の時間枠も考慮することが可能になる．

4. 評価実験

文献[1]における定式化と本研究における定式化を比較するために，同じ 30 個の問題例に対し評価実験を行った．

ロボットが移動するフィールドは 10×10 のグリッド状空間とし，その中からランダムに 8 つの頂点をロボットが移動できない箇所（障害物）とした．アイテムは 8 個であり，回収地点と配達地点とそれぞれの時間枠はランダムに設定されており，時間枠の幅は 1 から 10 の整数値である．なお回収と配達の時間枠に重複はない．またアイテムの大きさは 1，2，3 のいずれかであり，報酬はすべて 100 とした．全体の上限時刻は 50 である．ロボットの発進地点をランダムに配置し，残存ロボットは 1 台のみランダムに配置した．通常のロボットの容量は 6，残存ロボットの容量は 1 から 6 のいずれかの整数値である．どの時刻においても，同時に存在できるロボットは残存ロボットを含めて 8 台までとする．またロボットの移動コスト，稼働コストはどちらも-1 である．なお，文献[1]による定式化の実験においては，アイテムの配達場所とその時間枠は無視している．すなわちロボットはアイテムを回収した後，それらを発進地点に持ち帰る．またどちらの定式化においても，回収もしくは配達の時間枠にその処理を行えない場合，そのアイテムは存在しないものとして扱われることに注意されたい．今回は CPLEX を用いて実験を行った．計算機環境は Windows 10，Intel® Core™ i7-8565U（1.80GHz，8.00GB メモリ），IBM ILOG CPLEX 22.1.0，python 3.7.9 である．

表 1：30 個の問題例を解いた実験結果の平均値

	文献[1]の定式化	本研究の定式化
RMP の目的関数値	620.577	391.389
マスター問題の目的関数値	613.333	384.067
解の精度	0.988	0.981
回収・配達したアイテム数	7.500	5.733
価格問題の変数の数	19994.867	20102.267
価格問題の制約式の数	4756.100	4876.167
追加された移動経路の本数	25.433	24.900
実行時間（秒）	113.100	493.867

表 1 に 30 個の問題例を解いた実験結果の平均値を示す．表中の解の精度はマスター問題の目的関数値を RMP の目的関数値で除して求められる．本研究では，既存研究よりも複雑なモデルを扱ったが，解の精度と価格問題の変数の数，価格問題の制約式の数に大きな違いが表れなかったことは評価できる点である．一方，回収・配達したアイテム数は文献[1]の定式化よりも少なくなっており，その影響で RMP とマスター問題の目的関数値がどちらも少し悪化している．これは配達の時間枠が設定されたことで，各時間枠内に回収と配達の両方を完了させることが困難になり，存在しないものとされたアイテムが増えたためである．また実行時間が 4 倍以上悪化している．こちらも配達に時間枠が設定されたことに起因するものと考えている．

5. まとめ

本研究では，アイテムに回収と配達の 2 つの時間枠が設定されている MAPD 問題の定式化とこの問題を解く列生成法に基づくアルゴリズムを提案した．評価実験の結果から，本研究の定式化と文献[1]の定式化では解の精度と価格問題の変数の数，価格問題の制約式の数に大きな違いがないことが示された．一方で回収・配達したアイテムの数と実行時間はいずれも文献[1]の定式化に劣る結果となった．

今後の課題は 2 つ考えられる．1 つ目に回収・配達の時間枠に間に合わないアイテムへの対応である．現状，時間枠内に回収・配達できないアイテムはその存在を無視することになっている．そのため本研究のモデルでは 8 個のアイテムのうち，平均して 5.7 個を回収・配達するに留まった．しかし現実には時間枠に違反することになったとしてもアイテムを回収・配達しなければならない状況が考えられる．そうした状況に対応できるように今後モデルを拡張していきたい．2 つ目の課題は実行時間である．実行時間の大半を占めているのが価格問題の求解である．文献[1]でも価格問題の計算量は問題視されていたが，本研究の定式化ではアイテムの個数が 8 個という条件下で，既存研究のおよそ 4 倍の実行時間を要した．よりアイテムが多い問題を扱うためには実行時間を短縮することが重要になる． 文献[1]ではヒューリスティクス解法を取り入れた高速化が提案されており，本研究でもヒューリスティクス解法の導入を検討していきたい．

参考文献

[1] Haghani, N., Li J., Koenig S., Kunapuli G., Contardo C., Regan A., Yarkony J. “Multi-Robot Routing with Time Windows: A Column Generation Approach“, CoRR, abs/2103.08835 (2021)..
[2] Righini G., Salani M. “New dynamic programming algorithms for the resource constrained elementary shortest path problem”, Networks, Vol.51, No.3, pp.155–170 (2008).
[3] Desrochers, M., Desrosiers, J., Solomon, M., “A new optimization algorithm for the vehicle routing problem with time windows”, Operations Research, Vol.40, No.2, pp.342–354 (1992).