情報処理学会論文誌 Vol.60 No.10 1595–1602 (Oct. 2019)

車群経路計画のための空間構造特徴を用いた発見的解法

大滝 啓介[1,a)] 大社 綾乃[1,b)] 西 智樹[1,c)]

受付日 2019年1月17日, 採録日 2019年7月3日

概要：既存の道路資源をサービスの観点から活用して新しい価値を提供するコンセプトとしてMaaSが注目されている．単一の車両に注目するのではなく，複数車両の移動経路を同時に最適化する車群経路計画（vehicle platooning problem; VPP）は，燃料消費量などの移動コスト削減の観点から注目されている．しかしVPPでは，車両が最短経路以外の経路を移動する可能性があるため計算量が大きく，これまでにいくつかの定式化や近似解法が研究されてきた．本稿では空間的・構造的な特徴を利用して既存手法を高速化したうえで，非効率な解を取り除くための前処理手法を提案し，より効率的な近似解法を提案する．提案手法が効率的かつ高速であることを，人工グラフ構造と実グラフ構造を利用した数値実験によって確認した．

キーワード：車群経路計画問題，複数ユーザのための経路計画，近似解法

Heuristics with Structural and Spatial Features for Vehicle Platooning

Keisuke Otaki[1,a)] Ayano Okoso[1,b)] Tomoki Nishi[1,c)]

Received: January 17, 2019, Accepted: July 3, 2019

Abstract: Cooperation among multiple vehicles is a promising approach to realize efficient Mobility-as-a-Service. Vehicle Platooning Problem (VPP) is an optimization problem to find routes when the travel cost is discounted by platooning. To solve the intractable VPP, several solvers have been studied, but they are still inefficient. We develop new heuristic solvers for VPPs using structual and spatial features of given requests and an underling graph. We demonstrated that our methods found routes in around 30-40% shorter computational time with comparable travel costs for synthetic graphs. Further, we confirmed that our solvers could find routes in a few seconds for a larger real-world network than synthetic graphs.

Keywords: Vehicle platooning problem, multi-user routing, heuristic solvers

1. はじめに

1.1 背景

交通や物流における社会問題は，都市化にともなって悪化することが予測されている[*1]．これらの社会問題に対処する有効な手段として，既存の交通システムをより効率的に運用するMaaS[*2]が注目されている [1]．MaaSでは様々な意思決定の最適化が必要となる．たとえばライドシェアでは，車両の待機場所や顧客への訪問経路/順，相乗り方法などを最適化することで，移動コストや待機時間を削減できると期待されている [2], [3], [4], [5]．また自動化配送では，トラックとドローンの連携方法や，配送順序/割当を最適化することで，ドライバ不足などの社会問題に対応できる可能性がある [6], [7]．これらのように，限られた道路資源を効率的に利用し，道路混雑がもたらす悪影響を最小化することは，利用者側・サービス提供者側の双方に価値がある．

本稿では移動サービス利用者の移動経路に着目し，経路の共有による移動コスト削減を目的とした**車群経路計画問題**（vehicle platooning problem; VPP）[9] を扱う．**車群**

[1] 株式会社豊田中央研究所
Toyota Central R&D Labs., Inc., Bunkyo, Tokyo 112–0004, Japan
a) otaki@mosk.tytlabs.co.jp
b) okoso@mosk.tytlabs.co.jp
c) nishi@mosk.tytlabs.co.jp

[*1] 都市化について，たとえばWorld Urbanization Prospects: The 2014 Revision Population Databaseが参考になる．
[*2] Mobility as a Service．車両の利用形態を所有から共有へと移行させ，サービスに基づく効率的運用を試みるコンセプトを指す．

図 1 Bonnet らによる実験概要（文献 [8] の Fig. 2 を引用）
Fig. 1 Experiments by Bonnet et al. (from Fig. 2 in Ref. [8]).

（platoon）とは，何らかの機構によって車間距離を短く保ったまま移動する複数車両の集合を指す．車群の例を**図 1** に引用する．車群の形成とは，個別の車両が通る経路を調整し，同じ経路を同時刻に通るように調整する行為である．車群を形成する主な目的に長距離トラックの**燃料消費削減**がある．Bonnet らは，2 台のトラックが車群を形成して移動した場合の運動/空気抵抗を検証し，燃料消費量を実測したうえで，車群形成により燃料消費量を最大で 2 割程度削減できることを確認した [8]．その他様々な条件において，燃料消費の削減効果が確認されている [10], [11], [12]．

1.2 既存研究の課題と本稿の目的

Larsson らは車群形成による燃料削減効果を用いて，VPP を最適化問題として定式化した [9]．つまりできるだけ共通の経路を同時刻に通過することで燃料消費量を削減する．Larsson らは車群形成を目的とした整数計画問題（Integer Linear Programming; ILP）を提案し，経路のみを最適化する問題と，より実用的に時刻も同時に最適化する問題を研究した．提案された ILP は汎用的なものであるが，VPP が NP 困難であるため，最新の ILP ソルバーを利用しても解くことができる問題のサイズは限られる．そのため Larsson らは，特に経路のみを最適化することに注目し，大規模問題を解くための近似解法を研究し，数値実験によって評価した [9]．しかしながら，Larsson らの手法 [9] や，後に研究された近似解法 [13] には計算方法に不明な点があるため自由度が存在し，改良の余地がある．

VPP は複数車両の経路最適化問題であるため，グラフ上の組合せ最適化として解釈できる [14], [15], [16], [17], [18]．特に瀧瀬らの問題設定は経路のみを最適化する VPP と類似している [19], [20]．瀧瀬らは，出発地（もしくは目的地）が同一である場合の合流経路問題（Multi-user routing; MUR）を定式化して研究した．本稿で述べる通り，経路のみを最適化する同一の出発地（もしくは目的地）の VPP（SDVPP と呼ぶ）は MUR と同一視できる．そのため，瀧瀬らの手法を用いて大規模な SDVPP を解くことができるが，一般の VPP を扱うことができない．

本稿の目的は，Larsson らの汎用的な定式化や近似解法のアイデアと，瀧瀬らが提案した効率的な近似解法の双方に基づき，一般的な VPP に対して適用可能な効率的な近似解法を構築することである．同時に，瀧瀬らの手法において計算量が高い「どの位置で隊列を形成する/離れるか」という頂点探索を効率化することで，より高速な手法を効率することを目指す．我々の手法は，需要位置関係とグラフ構造に着目した空間的・構造的な特徴を利用し，探索空間を小さく抑えることで高速に経路を探索する．

本稿は以下のように構成される．2 章では議論に必要な概念を整理する．3 章では，空間的・構造的な特徴に基づく近似解法を提案し，4 章では数値実験によって評価を行う．最後に 5 章で結論や将来の課題を述べる．

2. 準備

自然数 $n \in \mathbb{N}$ に対して $[n] = \{1, \cdots, n\}$ とする．記号 $\langle \cdot \rangle$ はリストを，演算子 $+$ でリストの連結演算を表す．

2.1 グラフ

本稿では $G = (V, E, w)$ を単純な重み付き有向グラフとする．辺 $(u, v) \in E$ の重みを $w_{u,v}$ と略記する．2 頂点 $s, t \in V$ 間の経路 $p_{s,t}$ を頂点の列として $p_{s,t} = \langle v_1, v_2, \ldots, v_{k+1} \rangle, v_1 = s, v_{k+1} = t, (v_i, v_{i+1}) \in E$ $(1 \leq i \leq k)$，i 番目の頂点 v_i を $p_{s,t}(i)$ で表す．また $u, v \in V$ の間のすべての経路を集合 $\Pi(u, v)$ で，ある 1 つの最短経路を $\pi(u, v) \in \Pi(u, v)$ で表し，u, v 間の**距離**は最短経路 $\pi(u, v)$ 上の辺の重みの総和として $d(u, v) = \sum_{1 \leq i \leq k} w(v_i, v_{i+1})$ と定義する．値 k は $p_{s,t}$ の辺数であり，$|p_{s,t}| = k$ と記す．ある経路 p に，辺 $(u, v) \in E$ が含まれる場合，$(u, v) \in p$ で表す（つまり $(u, v) \in p \iff \exists j$ s.t. $p(j) = u, p(j+1) = v$）．

2.2 移動リクエストと VPP の関連問題

個別の車両は移動の**リクエスト** $r = (o, d) \in V \times V$ を持ち，o は出発地，d は目的地を表す．ある経路 p が $p \in \Pi(o, d)$ であるとき，p は r を達成する**充足経路**と呼ぶ．本稿では，与えられたすべての車両に対する充足経路の集合を，問題の**解**と呼ぶ．以下に VPP の定義を [9] より引用する．

問題 1（VPP） グラフ G，N 台の車両とリクエスト $r_n = (o_n, d_n)$，およびパラメータ η $(0 < \eta < 1)$ に対し，式 (1) のコスト関数を最小化する解 $\mathcal{P} = \{P_n \mid n \in [N], P_n \in \Pi(o_n, d_n)\}$ を計算する問題を VPP(η) と呼ぶ．

$$c(\mathcal{P}) = \sum_{\substack{(u,v) \in E \\ \text{if } N_{\mathcal{P}}(u,v) > 0}} w_{u,v}(1 + \eta(N_{\mathcal{P}}(u, v) - 1)), \qquad (1)$$

ただし $N_{\mathcal{P}}(u, v) = |\{n \in [N] \mid (u, v) \in p_n\}|$ は解 $\mathcal{P}$ 中で辺 $(u, v) \in E$ を移動する際に，車群を形成する台数を意味する．項 $\eta(N_{\mathcal{P}}(u, v) - 1)$ は燃料削減効果を表し，先頭車両以外の後続車両が必要な移動コストを示す．パラメータが $\eta \to 1$ であるとき削減効果はない最短経路問題である．逆

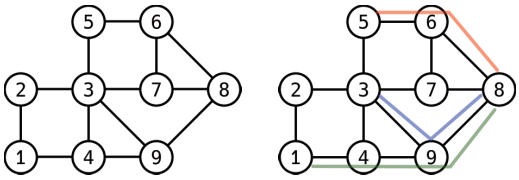


図 2 例題グラフと解（文献 [9] 中の例を筆者が描画）
Fig. 2 A graph example and a VPP solution from Ref. [9].

に $\eta \to 0$ であるほど削減効果が大きいことを意味する．

次に**単一終点合流問題**（MUR）を [19] に従って説明する．

問題 2（単一終点合流問題） グラフ G，N 人の顧客からの移動リクエスト $r_n = (o_n, d_\circ)$（$d_\circ$ は**共通**の目的地），および評価関数 $\alpha : 2^{[N]} \to \mathbb{R}$ に対し，以下のコスト関数を最小化する解 $\mathcal{P} = \{P_n \mid n \in [N], P_n \in \Pi(o_n, d_\circ)\}$ を計算する問題を単一終点合流問題 MUR(α) と呼ぶ．

$$c(\mathcal{P}) = \sum_{n \in [N]} \sum_{i=1}^{|P_n|} \alpha(Gr(\mathcal{P}, M, n, i)) w(e(P_n, i)) \qquad (2)$$

ただし P_n は顧客 n の充足経路，$e(P_n, i)$ は i 番目の辺，$Gr(\mathcal{P}, M, n, i)$ は i 番目の辺において車群を構成する顧客の集合，$M \subseteq [N] \times [N] \times \mathbb{N} \times \mathbb{N}$ は合流の有無を表す．

MUR(α) は車両と顧客を同一視し，VPP の目的地を共通にすることで，経路のみを最適化する VPP(η) と同一視できる．このとき顧客集合 U に対して評価関数 $\alpha(U) = 1 + \eta(|U| - 1)$ を用いる．これはある辺を通過する顧客の集合 U の移動コストを，VPP と同様の考え方で係数 η を用いて割引して評価することを表している．以下に，同一目的地に対する SDVPP[*3] と MUR を比較する．結果として瀧瀬らの手法を VPP に利用できる．

例 3（VPP と MUR の対応関係） **図 2**（左）のグラフ $G = (V, E, w)$ を用いる．なお $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ および $E' = \{12, 14, 23, 34, 35, 37, 49, 56, 67, 68, 89\}$（ただし，たとえば 12 は辺 $(1, 2)$ を示す）とし，$E = E' \cup \{(v, u) \mid (u, v) \in E'\}$ である．すべての辺 $e \in E$ について，移動コストは $w(e) = 1$ とする．3 つのリクエスト $r_1 = (5, 8)$，$r_2 = (3, 8)$，$r_3 = (1, 8)$ が与えられるときの解 $\mathcal{R}$ を図 2（右）に示す．充足経路はそれぞれ $R(u_1) = \langle 5, 6, 8 \rangle$，$R(u_2) = \langle 3, 9, 8 \rangle$，$R(u_3) = \langle 1, 4, 9, 8 \rangle$ である．たとえば合流点の集合 M には，$(u_2, u_3, 2, 3)$ が含まれる．式 (2) 中の $Gr(\cdot)$ の例として，$Gr(R, M, u_2, 1) = \{u_2, u_3\}$ がある．

これまでの例では経路のみを最適化しているが，時刻も同時に最適化する手法としては，移動/滞在時間に関する情報をグラフに埋め込む時拡張ネットワークが応用できる．そのため本稿では経路の最適化に着目して議論を行う．

[*3] 文献 [9] では同一出発地の VPP を SSVPP として議論した．本稿では MUR と合わせるため，対称的な問題として SDVPP を扱う．

Algorithm 1 ハブ探索法 [9]
Input: 顧客集合 U，リクエスト集合 $R = \{(o_n, d_n) \mid n \in U\}$
Output: ハブ $h \in H$ と解 $\mathcal{R}$
1: ハブ頂点 $h \in V$ を求める
2: 出発地から h までの解 $\mathcal{R}_{org}$ を求める
3: 目的地から h までの解 $\mathcal{R}_{dst}$ を求める
4: **for** 各ユーザ $n \in U$ について **do**
5: $\mathcal{R}[n] \leftarrow \mathcal{R}_{org}[n] + \mathcal{R}_{dst}[n][:-1]$
▷ ただし経路 $p = \langle v_1, \cdots, v_k \rangle$ に対する $p[:-1]$ は部分経路 $\langle v_1, \cdots, v_{k-1} \rangle$ を示す．ハブ h が双方に含まれるため除外する．
6: **return** ハブ頂点 h と解 $\mathcal{R}$

2.3 既存手法

本節では，提案手法の元となる既存手法を説明する．

2.3.1 Larsson らの手法 [9]

Larsson らは ILP を提案しただけではなく，一般の VPP を解くための枠組みである**ハブ探索法**を提案した．ハブ探索法では，ハブと呼ばれる頂点を定め，VPP 問題を「出発地からハブへ移動する問題」と，「ハブから目的地に移動する問題」の 2 つの SDVPP 問題に分割し，全体の解を構成する．擬似コードを Algorithm 1 に示す．しかし著者らは，ハブ計算の詳細（Algorithm 1 の 1 行目）を述べておらず，適切なハブ計算方法を考察する必要がある．

2.3.2 瀧瀬らの近似解法 [20]

瀧瀬らの手法は，目的関数が評価関数 α で表現される一般的なモデルであるが，SDVPP のみ扱うことができる．彼らは SDVPP を解くため，以下の考え方による厳密解法を提案した．同一目的地 $d_\circ$ に向かう顧客の集合を $[N]$，同一目的地を持つリクエスト集合を $R = \{(o_n, d_\circ) \mid o_n \in V, n \in [N]\}$ で表す．ある顧客の集合 U' と頂点 $u \in V$ に注目するとき，U' が u に最小コストで到達する経路計画とは

(1) 2 つの集団 $U_1, U_2 \subseteq U'$，$U_1 \sqcup U_2 = U'$ について，U_1 と U_2 が頂点 u に移動した結果 U' を形成する（合流）
(2) 集団 U' が別の頂点 u' から u に移動する（移動）

のどちらかのパターンによって達成される．そのため (1) と (2) の可能な経路を動的計画法によって走査し，コスト最小を達成する場合のみを記録することで厳密解を求める．

しかしこの動的計画法は計算量が依然として高く，N が大きい問題に対して適用できない [20], [21]．そこで瀧瀬らは以下のような近似手法 `AP` を提案した．手法 `AP` では，ある U_1 と U_2 のペアが合流するかしないか，を逐次的に計算する．まず全体の顧客集合 U から，2 人の顧客 u_1, u_2 の組合せに対して，最も合流が効果的な顧客 2 人を選択し，新しい顧客（つまり顧客群）と置き換える．この際，選ばれた両者については 1 つの合流点が選択され，固定される．置き換えにより，次の計算ステップでは U から 1 人分少ない問題を再帰的に解くことになる．計算の最中で以下の式 (3) と (4) を計算し，その差分 $c_2 - c_1$ をペアの優先度とし，優先度の高い顧客のペアから順番に車群を形成する．

Algorithm 2 AP（文献 [19] を元に論文上の表記を利用）

Input: 顧客集合 $[N]$，リクエスト集合 $R = \{(o_n, d_\circ) \mid n \in [N]\}$
Output: コスト $c_{\mathcal{R}}$ と近似解 $\mathcal{R}$

```
1: 全ての顧客群 P = {{u_n} | n ∈ [N]} の集合を初期化
2: 優先度付きキュー q = ⟨⟩ を初期化
3: コスト C = 0 を初期化
4: for all U_1 ≠ U_2 である全ての U_1, U_2 ∈ P について do
5:     U_1 と U_2 の優先度 p_{U_1,U_2} を式 (3) と式 (4) から計算
6:     q ← (p_{U_1,U_2}, U_1, U_2) を格納
7: while |q| > 1 do
8:     (v, U_1, U_2) ← q を優先度順で取得
9:     P から U_1 と U_2 を削除する
10:    v_i を U_i 中の顧客が取る経路の末尾として取得
11:    v' として U_1 と U_2 の合流点を全探索して固定
12:    q から (U_1, U_2) が含まれる 3 つ組を削除
13:    C ← C + d(v_1, v')|U_1|α(U_1) + d(v_2, v')|U_2|α(U_2)
14:    for all U' ∈ P do
15:        U_1 ∪ U_2 と U' の優先度 p_{U_1∪U_2,U'} を計算
16:        q ← (p_{U_1∪U_2,U'}, U_1 ∪ U_2, U') を新たに格納
17:    U_1 と U_2 について，v' を用いて解 R を更新
18: 最後の合流点から d_∘ までの経路を用いて C と解 R を更新
19: return R のコスト C と近似解 R
```

$$c_1 = \min_{w \in V} d(v_1, w)|U_1|\alpha(U_1) + d(v_2, w)|U_2|\alpha(U_2) + d(w, v_T)|U_1 \cup U_2|\alpha(U_1 \cup U_2) \quad (3)$$

$$c_2 = d(v_1, v_T)|U_1|\alpha(U_1) + d(v_2, v_T)|U_2|\alpha(U_2) \quad (4)$$

式 (3) は，U_1 の所在地 v_1 と U_2 の所在地 v_2 から，$w \in V$ に移動し，そこから $U_1 \cup U_2$ の集団として v_T まで移動するコスト，式 (4) は U_1 と U_2 が，それぞれの現在位置から v_T まで単独移動するコストである．よって優先度が高い順とは，合流することで最も移動コストが削減できるペアの順である．擬似コードを Algorithm 2 に示す．

2.3.3 既存手法の課題

既存手法の課題を以下にまとめる．

(a) リクエストの位置にかかわらず，優先度を計算する際に V を走査する必要がある（Algorithm 2 の 11 行目）
(b) AP は，合流問題（同一目的地）にのみ適用できる
(c) AP は MUR のための手法として開発されているため，VPP に転用する際に効率的ではない

3. 提案手法

本章では一般の VPP を効率的に解くために，空間的および構造的な特徴を利用した近似解法を構築する．まず 3.1 節では課題 (a) に対し，既存の近似解法を高速化するために，探索空間を削減する手法を提案する．次に 3.2 節では課題 (b) に対し，既存手法を併用して一般の VPP を解く手法を提案する．最後に 3.3 節では課題 (c) に対し，探索空間を縮小し，かつ経路の枝刈りを新規に導入することで，効率的な経路構築を可能にする手法を構築する．

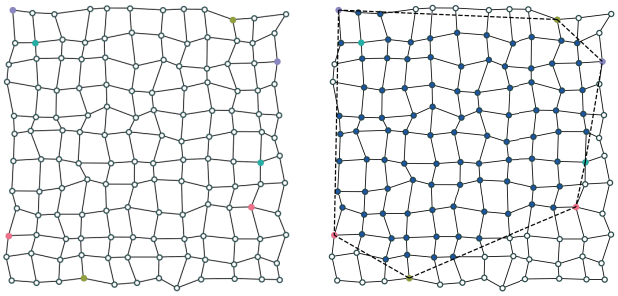


図 3 （左）グラフ上の出発地/目的地のプロット（右）凸包
Fig. 3 (Left) Problem instance (Right) Convex hull of $\bar{R}$.

3.1 探索範囲を限定する Region AP

車群を形成するとき，車両の最短経路と比較して，遠回りする可能性がある．しかしコスト削減を目的とする場合には，最短経路から遠く離れた頂点を経由することができない．そのためグラフ G 全体で合流や車群が発生することは考えにくい．直感的には，互いの位置を最短経路や直線でつないだ線上の近傍において合流する可能性が高い．我々はこの考えに基づいて，優先度計算の探索範囲を限定することで課題 (a) に対処する効率的な手法を構築する．

本稿では顧客集合 U に対するリクエスト $R = \{(o_n, d_n) \in V \times V \mid n \in [N]\}, \bar{R} := \{o_1, \cdots, o_n, d_1, \cdots, d_n\}$ が与えられたとき，各地点の緯度経路といった**空間的情報**が得られると仮定する．それらの情報から平面上の（幾何）**凸包** $conv(\bar{R})$ を計算し，凸包に含まれる頂点からなる集合 V' を構成する．頂点 $v \in V$ の探索において，集合 $V' \subseteq V$ のみを利用する近似解法を Region AP と呼ぶ．**図 3** に例題と，これらの位置情報から作成した凸包を描画した例を示す．

空間的特徴の凸包は，構造上の特徴に置き換えることができる．このとき，グラフ G とリクエスト R にかかわる頂点の集合 $R \subseteq V$ が所与のとき，$G = (V, E)$ 上の **geodesic convex set**（**最短経路凸集合**）[22] を利用することで，同様に探索空間を制限することができる．利用する集合の種類によって，空間的/構造的特徴と区別する．

3.2 ハブ探索により複数目的地へ対応する Hub AP

手法 AP は一般の VPP を解くことができないが，ハブ探索法を併用することでこの問題点を解消できる．Algorithm 1 ではハブとなる頂点 $h \in V$ を探索する方法が重要となる．我々は本稿で，2 つの手法を提案する．

1 つ目の手法は，頂点 h を探索しながら 2 つの SDVPP 問題を解き，最良コストとなるハブ h を全探索する．これを Algorithm 3 に示す．Region AP と同様に，すべての頂点をハブの候補とする必要がないと考えられるため，凸包を利用する．これを Hub Traversal AP（HT-AP）と呼ぶ．

2 つ目の手法は，AP の内部でハブ探索を考慮した計算を行う手法であり，Dual-AP と呼ぶ．元々の AP は，同一目

Algorithm 3 Hub Traversal AP（HT-AP）

Input: 顧客集合 $[N]$，リクエスト集合 $R = \{(o_n, d_n) \mid n \in [N]\}$
Output: ハブ $h \in V$ と近似解 $\mathcal{R}$

```
凸包 hull = conv(S ∪ T) を計算する
minCost = ∞                                  ▷ 最小コストの初期化
V_h := {v ∈ V | v は hull に含まれる位置の頂点 }
S := {o_n | n ∈ [N]}, T := {d_n | n ∈ [N]}
for v ∈ V_h do
    c_S, R_S ← AP([N], {(o_n, v) | n ∈ [N]})
    c_T, R_T ← AP([N], {(d_n, v) | n ∈ [N]})
    if c_S + c_T < minCost then
        minCost ← c_S + c_T とし，経路 R を構築する
return 最小コスト minCost を達成したハブ v と解 R
```

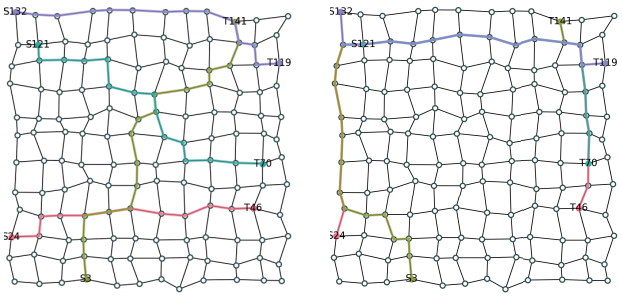


図 4 （左）最短経路移動（右）VPP の近似解による移動
Fig. 4 (Left) Shortest paths (Right) A solution of VPP.

的地の問題のみを対象として設計されているため，優先度を次のように拡張し，対応する経路の構築も修正することで，Algorithm 2 と同種の擬似コードが得られる．

$$c_1^{(\mathrm{dual})} = \min_{w,u \in V} d(v_1, w)|U_1|\alpha(U_1) + d(v_2, w)|U_2|\alpha(U_2) + d(u, v_3)|U_1|\alpha(U_1) + d(u, v_4)|U_2|\alpha(U_2) + d(w, u)|U_1 \cup U_2|\alpha(U_1 \cup U_2) \quad (5)$$

$$c_2^{(\mathrm{dual})} = d(v_1, v_3)|U_1|\alpha(U_1) + d(v_2, v_4)|U_2|\alpha(U_2) \quad (6)$$

式中の v_1, v_2 は AP と同一であり，v_3, v_4 は分岐側について v_1, v_2 と同等の頂点である．合流と分岐の両方を扱うために，u という新しい頂点を利用し，合流と分岐の両方のコストを計算して探索する．ハブ探索法について，**図 4**（左）に描画した最短経路移動と比較した経路を図 4（右）に示す．

3.3 顧客の位置関係を用いる分割

複数目的地に対応した経路計画においてハブ探索を単純に用いると，不必要な迂回経路が計算される場合がある．これは優先度計算後の経路計画の際，AP が MUR のための手法であることから位置情報を利用せず，探索が冗長になるためである．この問題を解消するため，リクエストの位置情報を用いた顧客のグループ化を提案する．

基本的なアイデアは，別方向へ移動している場合など，合流する利得が明らかに小さいユーザを前処理で取り除くことである．Algorithm 4 に擬似コードを示す．例を**図 5** に示す．図 5（左）は，直接 HT-AP を適用して得られた経路を，図 5（右）は Onestep-Cosine-Grouping（OCG と記す）を適用した後に，問題を分割して HT-AP を適用した（これを HT-AP-OCG と記す）結果を図示する．結果より，コスト削減に寄与しない迂回を抑制する効果がある．

Algorithm 4 Onestep-Cosine-Grouping（OCG）

Input: 顧客集合 $[N]$，リクエスト集合 $R = \{(o_n, d_n) \mid n \in [N]\}$．各頂点 $p \in \bar{R}$ は位置情報ベクトル (x_p, y_p) を持つ
Output: 顧客 $n \in [N]$ のクラスラベル $l_n \in \{0, 1\}$

```
顧客 n ∈ [N] について，方向ベクトル v⃗_n を計算
ある顧客 n' ∈ [N] の方向ベクトルを選択
U_1 = {n ∈ [N] | v⃗_{n'} · v⃗_n ≥ 0} および U_2 = [N] \ U_1 を計算
n ∈ U_1 であれば l_n = 0，u ∈ U_2 であれば l_n = 1 とする
return ラベル l_n
```

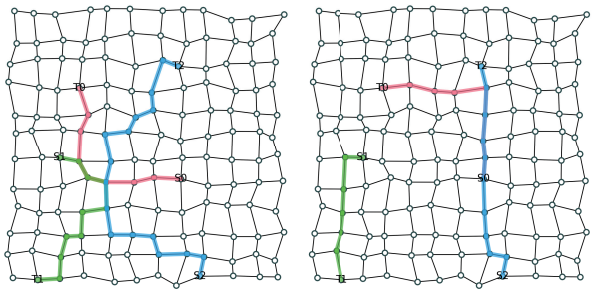


図 5 （左）前処理なしの経路（右）前処理ありの経路
Fig. 5 A solution (Left) without pre-processing (Right) with pre-processing.

表 1 既存手法と提案手法の特徴比較（その 1）
Table 1 Comparison of methods (perspective 1).

	AP (Alg. 2)	Region AP
探索範囲	V 全体	(空間) 凸包 V_h (構造) 最短経路凸集合

表 2 既存手法と提案手法の特徴比較（その 2）
Table 2 Comparison of methods (perspective 2).

	ハブ探索 (Alg. 1)	HT-AP	Dual-AP
優先度計算	$c_2 - c_1$	$c_2 - c_1$	$c_2^{(\mathrm{dual})} - c_1^{(\mathrm{dual})}$
ハブ探索	V 全体	凸包 V_h	–

表 3 既存手法と提案手法の特徴比較（その 3）
Table 3 Comparison of methods (perspective 3).

	HT-AP	HT-AP-OCG
顧客組合せ	すべて	同方向のみ抽出して分割

3.4 提案手法の位置づけと性質

本章の最後に，提案した手法と既存手法の対応関係を**表 1**，**表 2** および**表 3** に示す．表 1 に示すとおり，空間的特徴である凸包や構造的特徴である最短経路凸集合のみを用いて頂点の探索を行うことで，計算の効率化を図る手法が Region AP である．表 2 に示すとおり，ハブ探索の考え方を用いて一般の VPP を解くための拡張が HT-AP お

よび Dual-AP である．表 3 に示すとおり，すべての顧客の組合せを探索する既存手法（例として HT-AP）に対して，方向性によってグループ分けを行ったうえで個別の問題を解く手法を提案した（例として HT-AP-OCG を記す）．

最後に手法の性質について補足する．我々の提案手法は内部の経路最適化に AP を利用しており，AP が停止するため我々の提案手法はいずれも停止し解を返す．しかし問題によっては，同じコストを持つ複数の解が最適解になることがある．

4. 数値実験

本章では既存手法 AP を提案手法と比較する．実験に利用するため，人工グラフと OpenStreetMap [*4] から取得したグラフを作成した．移動リクエストとして，空間的に左右に移動するリクエストをランダム作成した．すべての実験は Intel Core i7-3770K（3.50 GHz）の CPU，32 GB のメモリ，Python 3.5.2 を備えた PC 上で実施した．

実験に先立って，グラフ G に対する頂点間の距離や最短経路，リクエスト R から構成される空間的な特徴である凸包 $conv(\bar{R})$ に含まれる頂点の集合 $V' \subseteq V$ などは事前に前処理として計算し[*5]，$\mathcal{O}(1)$ でアクセス可能なデータ構造に保存する．既存研究の実験 [20] と同じく，前処理の時間は実験時間には含んでいない．また実験結果におけるエラーバーは，複数試行した結果の標準偏差を示す．

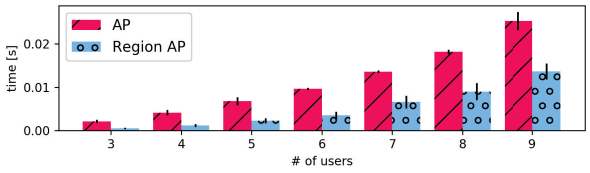
4.1 人工データによる検証

人工データとして，図 3（左）のように歪んだ格子グラフを作成する．それぞれ 1 辺を 12，20，30 と変更した格子世界を作成した．まずは格子グラフ上で，顧客数 $N = 3$ から 9 に対して，それぞれ 20 個ずつのリクエストをランダムに作成し，数値実験に利用した．アルゴリズムの比較として，具体的には以下の 3 点を目的として数値実験を行う．

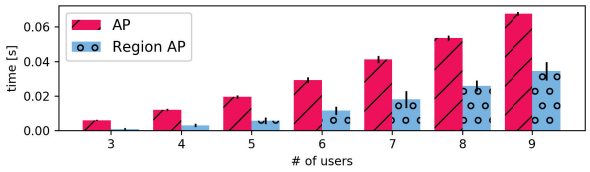
(I) AP と Region AP を比較し，探索範囲を制限することによる利点と欠点を把握する
(II) ハブ探索の新たな実装方法として，HT-AP と Dual-AP を比較し，利点と欠点を把握する
(III) 提案した前処理 OCG の利点と欠点を把握する

4.1.1 課題 (I) 探索範囲の削減に関する実験

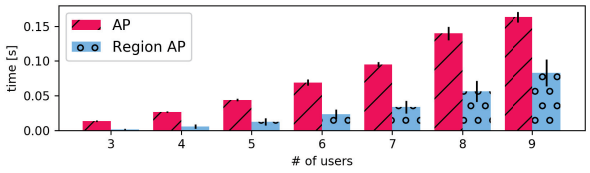
課題 (I) について確認するため，AP と Region AP を比較した．この実験ではハブ探索を利用せず，問題インスタンスはすべて SDVPP として設定した．**図 6** に計算時間の比較結果を示す．今回の実験においては，全体を通して効率的な計算が可能になったことが分かる．具体的には，1 つの SDVPP インスタンスを解くための必要な計算時間が最大で 50%程度高速化された．計算時間を計測すると同時に，探索範囲を制限することによる移動コストへの影響を確認したところ，総コストが変化するインスタンスは存在しなかった．以上の結果より，AP を Region AP に置き換えることで，高速に近似解を求めることができることが明らかになった．また SDVPP の場合は，総移動コストが変化しないことを実験的に確認した．



(a) 12 × 12 の計算時間 (Results on 12 × 12)



(b) 20 × 20 の計算時間 (Results on 20 × 20)



(c) 30 × 30 の計算時間 (Results on 30 × 30)

図 6 計算時間の比較（AP と Region-AP）
Fig. 6 Computational time (AP and Region-AP).

課題 (I) に対する実験より，グラフサイズは主に計算時間に対して影響していると考えられる．以降の実験ではいくつかの代表的な結果のみを図示する．

4.1.2 課題 (II) ハブ探索に関する実験

課題 (II) について確認するため，ハブ探索法（Hub AP と記す），HT-AP および Dual-AP の計算時間と，解の総移動コストを比較した．特に 30 × 30 の格子世界上で得られた結果を**図 7** に示す．結果から，凸包を利用することでハブとなる頂点の全探索を効率化でき，この場合は総移動コストが劣化しなかった．一方で，AP を一般 VPP に対して適用できるように一般化した Dual-AP は，さらに高速な計算が可能であった．しかし，総移動コストに関しては，ハブを全探索する手法と比較しておおむね 10%程度悪化した．

4.1.3 課題 (III) 前処理 OCG の効用

課題 (III) について，既存手法 AP において考慮されていない探索空間の枝刈りを提案手法 OCG で行う手法の影響を確認する．作成した格子グラフごとのインスタンス 140 個ずつのうち，OCG を適用した結果，顧客が分割されたユーザ数 $N = 3$ から 5 までのインスタンス 49 問を抽出し，計算時間と総移動コストを比較した．実験では一般の VPP

*4 https://www.openstreetmap.org
*5 グラフ上の計算は networkx で，凸包は scipy.spatial によって計算した．

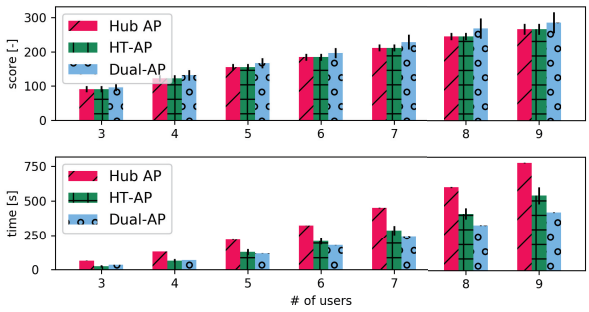


図 7 30 × 30 における総移動コストおよび計算時間の比較（Hub AP, HT-AP および HT-AP）

Fig. 7 Results of total travel costs and computational time (Hub AP, HT-AP, and HT-AP on 30 × 30 graph).

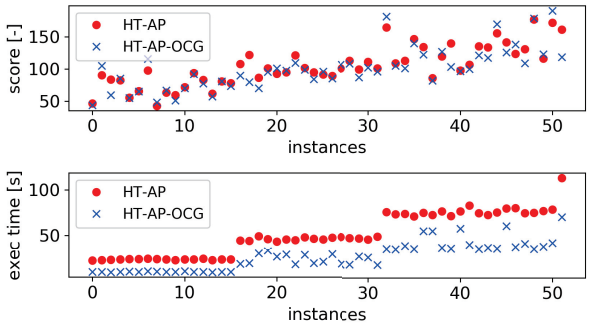


図 8 30 × 30 のグラフにおける前処理の有無による計算時間と総移動コストの比較（HT-AP vs. HT-AP-OCG）

Fig. 8 Comparisons of HT-AP and HT-AP-OCG: Travel costs (above) and computational time (below) on 30 × 30 graph.

を扱うため，HT-AP と HT-AP-OCG と比較を行った．特に 30 × 30 のグラフにおける結果を図 8 に示す．

枝刈りを行わない単純な HT-AP と比較して，出発地と目的地の位置情報に基づく問題の分割を行うために OCG を適用することで，計算時間がすべてのインスタンスについて高速化された．この時の総移動コストについては，良くなる場合と悪くなる場合が見られたが，半数以上のインスタンスについては，総移動コストが改善された．今回作成したインスタンスの中では，スコアが大きく悪化することはなく，最大でも 2%程度の悪化に留まった．その一方で，最大では 50%程度良化する例も見られた．

今回の実験で利用したリクエストはランダムに作成されているが，格子世界上に配置されているため移動方向に偏りが存在し，この場合にはリクエストがいくつかのクラスタに分割される．これは現実の環境において，移動需要や人気のあるスポットの偏りがあることに相当している．今回の実験では，特にリクエストの位置が空間的に大きく離れてクラスタ化されており，クラスタ間を移動する迂回路が除かれた場合に削減効果が大きかったと考えられる．今回は 2 つのクラスタに分割しているため，複数のクラスタへ一般化する手法や，クラスタリングの初期値を与える方法を検討することで，より改善できると考えられる．

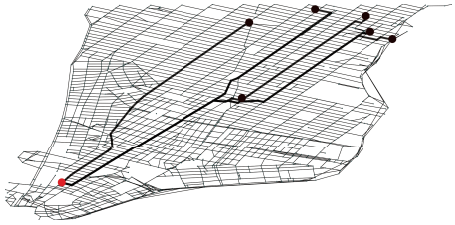


図 9 実グラフ上の SDVPPP の例

Fig. 9 An example of SDVPP in a road network.

4.2 実世界のグラフ構造への適用例

OpenStreatMap よりマンハッタン島の道路ネットワークを一部取得し，ランダムな移動リクエストに対する適用結果を観察する．得られたグラフ構造 $G = (V, E, w)$ は $|V| = 1{,}728$，$|E| = 3{,}044$ であり，重み w は地図上の距離とした．図 9 に SDVPP のランダムに作成した顧客数が 6 の例題を，近似手法 Dual-AP（パラメータ $\eta = 0.8$）によって解いて得られた経路を図示する．この経路を得るための計算時間は 0.39 秒であり，各車両が最短経路で移動した場合の移動コストと比較して，車群を形成した結果として，総移動コストは約 10%削減された．

4.3 実験のまとめ

本章では課題 (I)，(II)，および (III) に対して，数値実験を通じて以下のことを明らかにした．

(1) 頂点の探索範囲を V 全体から，何らかの方法で削減することは，計算時間の削減に大きく寄与する．今回実験で利用した幾何的な凸包は，計算結果を大きく悪化させることなく，高速化を達成できることが分かった．

(2) 一般的な VPP を扱うための手法としてハブ探索が適用できる．一方で，全頂点 V からハブを探索すると計算時間が大きくなるため，同様に範囲の制約が有効であった．また直接的にハブ探索を行う Dual-AP は，より計算時間が効率的であったが，コストが悪化した．

(3) 移動方向ベクトルを用いる前処理は，計算時間を削減するために効率的であった．コストが改善する場合の検討や，一般的な前処理の開発などの改良により，さらに良い解が得られる可能性がある．

5. まとめ

本稿では車群経路計画問題に着目し，出発地と目的地のグラフ上での空間特徴と構造特徴を利用する近似解法を提案した．我々は探索空間を削減するために，リクエストに関する場所・方向を用いて，考慮すべき組合せをできるだけ小さく抑える近似解法を提案した．数値実験により，適切な前処理によって，コストの削減が見込まれること結果

を得た．また近似手法は，現実的な街の一部などを想定したサイズのグラフに対しても適応可能であった．

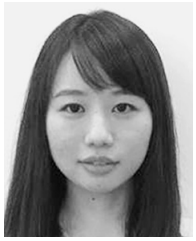
今後の課題として大規模実験や，瀧瀬らの手法とは異なる近似解法である Steinmetz らの手法 [13] との比較，分散最適化のための手法開発がある．また目的関数の定義から生じる解の一意性や，既存の近似手法 AP を含めた解の収束性などの理論的な側面を検証する予定である．

参考文献

[1] Hietanen, S.: Mobility as a service, pp.2–4 (2014).
[2] Agatz, N., Erera, A., Savelsbergh, M. and Xign, W.: Opimization for dynamic ride-sharing: A review, *EJOR*, Vol.223, pp.295–303 (2012).
[3] Ma, S., Zheng, Y. and Wolfson, O.: Real-Time City-Scale Taxi Ridesharing, *IEEE Trans. Knowledge and Data Engineering*, Vol.27, No.7, pp.1782–1795 (2015).
[4] Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X. and Koenig, S.: Ridesharing: The state-of-the-art and future directions, *TRP-B*, Vol.57, pp.28–46 (2013).
[5] Santi, P., Resta, G., Szell, M., Sobolevsky, S., Strogatz, S.H. and Ratti, C.: Quantifying the benefits of vehicle pooling with shareability networks, *Proc. National Academy of Sciences*, Vol.111, No.37, pp.13290–13294 (2014).
[6] Gonzalez-Feliu, J., Samet, F. and Routhier, J.-L.: *Sustainable urban logistics: Concepts, methods and information systems*, Springer (2014).
[7] Langley, J.C.: 2018 Third-Party Logistics Study (2017), available from ⟨http://3plstudy.com⟩ (accessed 2018-10-5).
[8] Bonnet, C. and Fritz, H.: Fuel consumption reduction in a platoon: Experimental results with two electronically coupled trucks at close spacing, Technical Report, SAE Technical Paper (2000).
[9] Larsson, E., Sennton, G. and Larson, J.: The vehicle platooning problem: Computational Complexity and Heuristics, *TRP-C*, Vol.60, pp.258–277 (2015).
[10] Al Alam, A., Gattami, A. and Johansson, K.H.: An experimental study on the fuel reduction potential of heavy duty vehicle platooning (2010).
[11] Liang, K.-Y., Martensson, J. and Johansson, K.H.: Fuel-saving potentials of platooning evaluated through sparse heavy-duty vehicle position data, *Proc. IEEE IV2014*, pp.1061–1068 (2014).
[12] Lammert, M.P., Duran, A., Diez, J., Burton, K. and Nicholson, A.: Effect of platooning on fuel consumption of class 8 vehicles over a range of speeds, following distances, and mass, *SAE International Journal of Commercial Vehicles*, Vol.7, No.2014-01-2438, pp.626–639 (2014).
[13] Steinmetz, D., Burmester, G. and Hartmann, S.: A fast heuristic for finding near-optimal groups for vehicle platooning in road networks, *Proc. DaWaK2017*, pp.395–405 (2017).
[14] Fahnenschreiber, S., Gündling, F., Keyhani, M.H. and Schnee, M.: A multi-modal routing approach combining dynamic ride-sharing and public transport, *Transportation Research Procedia*, Vol.13, pp.176–183 (2016).
[15] Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D. and F. Werneck, R.: Route planning in transportation networks, arXiv:1504.05140 (2015).
[16] Geisberger, R., Luxen, D., Neubauer, S., Sanders, P. and Volker, L.: Fast detour computation for ride sharing, *Proc. ATMOS2010*, pp.88–99 (2010).
[17] Yan, D., Zhao, Z. and Ng, W.: Efficient algorithms for finding optimal meeting point on road networks, Vol.4, No.11, pp.1–11 (2011).
[18] Bit-Monnot, A., Artigues, C., Huguet, M.-J. and Killijian, M.-O.: Carpooling: The 2 synchronization points shortest paths problem, *Proc. ATMOS2013* (2013).
[19] Takise, K., Asano, Y. and Yoshikawa, M.: Multi-user routing to single destination with confluence, *Proc. 24th ACM SIGSPATIAL*, pp.72:1–72:4 (2016).
[20] 瀧瀬和樹，浅野泰人，吉川正俊：合流による利益を考慮した単一目的地への集合経路最適化，日本データベース学会和文論文誌，Vol.15-J, No.6 (2017).
[21] Zhang, X., Asano, Y. and Yoshikawa, M.: Mutually Beneficial Confluent Routing, *IEEE Trans. Knowledge and Data Engineering*, No.10, pp.2681–2696 (2016).
[22] Pelayo, I.M.: *Geodesic convexity in graphs*, Springer (2013).

大滝 啓介 （正会員）

2016 年京都大学大学院情報学研究科知能情報学専攻修了．京都大学博士（情報学）．同年（株）豊田中央研究所入社．意思決定に関するデータ解析と最適化に関する研究に従事．



大社 綾乃

2015 年大阪府立大学工学部知能情報工学科卒業．2017 年同大学大学院修士課程修了．同年（株）豊田中央研究所入社．意思決定の最適化に関する研究に従事．

西 智樹

2005 年大阪大学応用理工学部機械工学科卒業．2007 年同大学大学院修士課程修了．同年（株）豊田中央研究所入社．意思決定の最適化と自動運転および MaaS への応用に関する研究に従事．