

マルチエージェント深層強化学習を用いたライドシェアのサービスエリア制御とスケーラビリティの確保

吉田 直樹^{1,a)} 野田 五十樹² 菅原 俊治¹

概要: ライドシェアは、乗客はモバイル端末から配車を要求でき、不特定多数の乗客が乗り合うことで安価なサービスが提供できるため、世界的に広がった。ライドシェアに多くのドライバが参加し発展した一方で、駅や空港に過剰なドライバが集まり、渋滞やエネルギーの浪費を引き起こす。また、住宅地や校外ではドライバ不足が生じ、利用者の利便性が低下する。これらを解決し質の高いサービスを実現するために、エージェントが乗客を待つ待機地域を適切に決定する必要がある。そのために、我々は深層強化学習を用いてエージェントに待機地域を割り当てる service area adaptation method for ride share with transfer learning (SAAMS) を提案した。エージェントごとに学習器を持つことで、エージェントは自身の待機地域を学習し、前述の過剰供給、供給不足を解消する。他方、エージェント数が増えるほど学習時間が増加し、ライドシェアのようにエージェント数が膨大な環境ではこれを無視できない。そこで本稿では、あらかじめ少ないエージェント数で学習をし、それらの学習器を複製することで学習コストの削減を行う service area adaptation method for ride share with transfer learning (SAAMSwT) を提案する。

キーワード: マルチエージェント強化学習, ライドシェアリング, Deep Q-network, 転位学習

Service area adaptation for a ride-share by using multi-agent deep reinforcement learning and ensuring scalability

1. はじめに

広域無線通信の発達、GPS を搭載したスマートフォンが世間に広まったことで、オンデマンド型のライドシェア (Ride-share, 以降 RS) の普及が進んでいる。RS では不特定多数の乗客が 1 台の車に乗り合うことで、乗客へのサービス提供の機会を増やし、同時に道路上の車の数や乗車費用の削減が期待できる。これらのサービス提供者と利用者の両者の利点により、今後も更に普及すると考えられる [1]。他方、Uber-Pool や Lyft-Line 等のサービスのよう、様々な企業の参加により急成長した RS は、海外での主要都市において渋滞を引き起こし社会問題となっている [2]。たとえば、ドライバは利益を増やすため、乗客を見つけやすい駅や

空港に集まる傾向がある。このとき、地域の需要に見合ったドライバの数であれば問題にならない。しかし、適切な制御が無い環境下での個々のドライバ間の調整は困難であり、結果として都市部におけるドライバの過剰供給は渋滞の原因となり、郊外ではドライバ不足が機会損失を発生させる。

ライドシェアが普及したことで、大量の乗客の乗降車位置のデータを解析することが可能となり、このデータから需要予測を行う研究が活発化している ([3], [4])。この需要予測を用いて車を制御することで、前述の供給の偏りを解決できる可能性がある。たとえば [5], [6] は、各地域における乗客の予測発生数と、その地域の車の数の差を減少させる手法を提案した。しかし、完全に需要予測に従う制御は予測の誤りに脆弱となる。

我々はこの課題に対し、強化学習を用いた空車エージェントの待機地域 (Service Area, 以降 SA) の制御手法として service area adaptation method for ride-share (SAAMS) を提案した [7]。ここでエージェントは、将来の自動運転を想定した RS のドライバ制御エージェントとする。SAAMS では

¹ 早稲田大学基幹理工学研究科, 東京都
Fundamental Science and Engineering, Waseda University, 3-4-1
Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan

² 産業総合技術研究所, 茨城県
National Institute of Advanced Industrial Science and Technology, Ibaraki 3058560, Japan

^{a)} n.yoshida@isl.cs.waseda.ac.jp

エージェントごとにニューラルネットワークを持ち、環境の情報として全エージェントの SA, 需要予測をニューラルネットの入力とし、各エージェントは自身の SA の制御を学習する。エージェントごとにネットワークを持つことで分業が進む一方、エージェント数が増えるほど学習コストは増加していき、特に RS のようにエージェント数が大規模な環境ではこの増加が無視できない。

そこで本稿では、あらかじめ少人数で学習したエージェント群のモデルを複製する service area adaptation method for ride-share with transfer learning(SAAMSwt) を提案する。実際に運用する台数よりも少ない台数で学習し、複製することで問題となる学習時間を削減することができる。シミュレーションによって、SAAMSwt は SAAMS より学習に要する時間を抑えながら、同等の性能を発揮したことを示す。

2. 関連研究

RS における配車問題の当初の関心は、乗り合いバスによる巡回経路決定問題であった ([8], [9])。ここでは Dial-a-ride-problem と呼ばれる問題を定式化し、予約等であらかじめ判明しているデマンドを辿る順序を決定する。巡回経路の多さから最適解の求解は難しく、準最適解を求めるメタヒューリスティックな手法が多く提案されてきた ([10], [11])。GPS およびモバイル端末の普及により、多くの車と多くの乗客の間のマッチングに課題が移った ([12], [13])。たとえば [12] は RS における配車車両の決定手法として逐次最適挿入法を提案し、それを用いた RS の実運用を行った。[13] はニューヨーク市内におけるタクシーサービスに RS を導入すると、必要なタクシーの台数を約 70 %削減でき、車台数の観点から RS がタクシーより効率的であることを示した。これまでの配車戦略は、到着したデマンドの処理に着目した短期的戦略であり、将来の需要の不確実さからその戦略をそのまま需要予測に統合することが適切とは限らない。

不確実性の影響を抑えるため、RS における需要予測が重要となり、これに着目した研究が近年多く存在する ([3], [4])。これらにより、予測結果の活用法が提案され始めた。RS では、車の供給と乗客の需要のバランスが崩れ効率が低下する。[5] は需要と供給の差の最小化を目的とした位置制御手法 Receding Horizon Control Approach (RHC) を提案し、将来の需要に対応した空車移動を導入させることで効率的な配車が可能なことを示したが、完全に需要予測に従う配車は予測の誤りに弱いという課題もあった。

[14] で提案された Deep Q-network (DQN) は強化学習手法の一種であり、環境情報のみからエージェントが戦略を学習できるため、ゲームやロボットの制御で広く活用されている ([15], [16])。RS でも DQN を用いた手法が提案されており、既存手法と同等かそれ以上の配車効率を実現した ([17], [18], [19])。

RS に強化学習を導入した既存研究では、たとえば [17] の

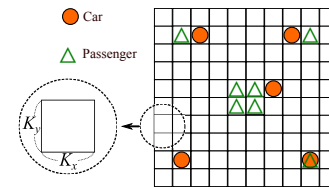


図 1: ライドシェア問題の環境

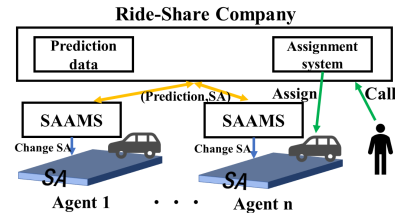


図 2: 配車システムの概略図

ように、効率よく配車するためにはエージェントは乗客の出発地から目的地までの最短経路を求める。学習した最短経路は道路状態により必ずしも一定ではなく、待機地域の制御と最短経路の動的決定はエージェントの学習を複雑にする。本稿で用いる SAAMS では、エージェントの待機地域である SA のみを学習するため、エージェントの行動に関与しない。

3. 実験設定

3.1 問題設定

RS の配車問題を定式化する。図 1 のように、環境上には車であるエージェント (円) と、ある目的地へ移動したい乗客 (三角形) が存在する。環境は 1 辺が $K_x \times K_y$ の大きさのセルが集まった格子として構成され、セルはある一定の領域 (区域) に対応し、各セルに複数のエージェントや乗客が存在できると仮定する。エージェントが乗客を出発地から目的地へと運び、得られる報酬を最大化することが目的である。

図 2 にシステムの概略図を示す。RS 企業は乗客からの乗車要求を受け取り、その要求をエージェントへと割り当てる。また、需要予測等のサービス提供に必要な情報をエージェントへと伝える。RS 企業は乗客からの要求を受け付けたとき、乗客を即座にエージェントに割り当て、エージェントは割り当てられた乗客のみを運ぶ。エージェントは上下左右のセルに移動でき、同一セル上の乗客の乗降ができる。乗車地点から降車地点のセルに向かうことで、1 人の乗客のサービスが完了する。エージェントは乗客の最大容量 Z を持ち、その人数以内で乗り合いを提供できる。サービスの品質を保証するため、追加条件として乗客の最大トリップ時間 t_l を定義し、乗客の輸送に要する時間が t_l を超えると予想されるエージェントには配車しない。ただしこの判断は RS システム側で行うため、各エージェントはこの制約を考える必要はない。

時刻 t ごとに、SAAMS はエージェント $\forall i \in D$ に、SA で

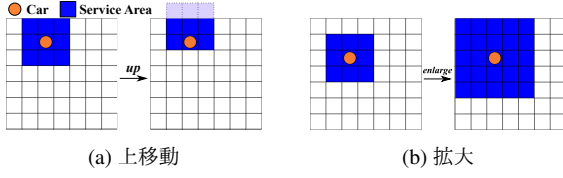


図 3: サービスエリアの制御

ある $T \times T$ のセルで構成される領域 $C_{i,t}$ を割り当てる。 $C_{i,t}$ は正方形の領域であり、最大の大きさは $T \times T$ ($T > 0$)、最小の大きさは 1×1 である。 i は上下左右のセルに移動でき、 i は $C_{i,t}$ が割り当てられると、乗客が割り当てられていない時は即座に $C_{i,t}$ に含まれるセルへ移動し乗客を待つ。 他方、乗客が割り当てられているときには $C_{i,t}$ を考慮せずに輸送を継続し、全ての乗客を降ろした後 $C_{i,t}$ に移動する。 実験では、 i は $C_{i,t}$ の中心のセルに移動する。 このように、SAAMS は $C_{i,t}$ を動的に変化させることで間接的にエージェントを制御する。

RS 企業は乗客からのデマンドを受け取ったとき、即座に割り当て可能な i を探す。 割り当て可能な i は、 $C_{i,t}$ が乗客の出発地を含む、定員制限 Z に違反しない、かつ予想乗車時間が規定の制限時間 t_l を超えない、これら 3 つの条件を満たす。 条件を満たす i が 2 体以上いるときは、乗客の出発地に一番近い i の中から任意の 1 体を選ぶ。 割り当て可能な i がいないとき、要求は即座に拒否され、サービスは提供されない。 エージェントは乗客を割り当てられたとき、即座に乗客の出発地へ向かう。 このときの輸送経路は SAAMS が制御せず、 i 自身の戦略で決定される。

3.2 SA の制御

SAAMS は、サービス提供によって得られる収益を最大化するように SA を制御する。 A を全エージェントの SA の制御の集合とし、時刻 t における $a_t \in A$ は $a_t = (a_t^1, \dots, a_t^n) \in A = A^1 \times \dots \times A^n$ と表現される。 ここで A^i は i の SA 制御の集合 $A^i = \{ \text{拡大, 縮小, 上, 下, 左, 右, 静止} \}$ である (例を図 3 に示す)。 拡大と縮小の際に、 $C_{i,t}$ の大きさの制約に違反する制御は静止とみなす。 状態 s_t において、SAAMS は DQN から得る Q 値に基づき $C_{i,t}$ を変化させる。

4. SAAMS による SA の調整

4.1 報酬

SAAMS は、SA の学習に DQN を用い、乗客から得られる収入を最大化する制御を実現する。 DQN は見積もり報酬を最大化するため、報酬の設計が重要である。 SAAMS では、報酬として以下の重み付き報酬を用いる。

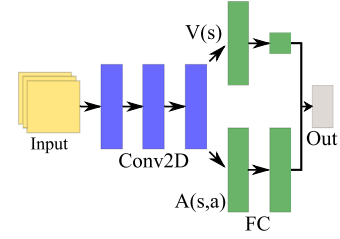


図 4: ネットワーク概略図

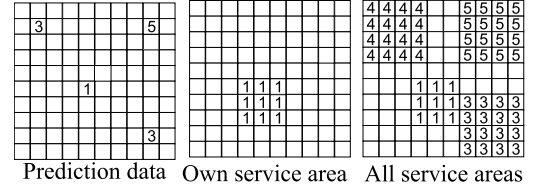


図 5: ネットワークへの入力

$$r_t^i = \sum_{p \in P_t^i} w_b f_b^p + w_d f_d^p + w_t f_t^p \quad (1)$$

$$r_t'^i = -w_g f_g \quad (2)$$

w_b, w_d, w_t, w_g は非負の重みである。 P_t^i は時刻 t に i へ割り当てた乗客の集合、 f_b はそれらの乗客の配車手数料、 f_d はそれらの乗客の想定所用距離、 f_t はそれらの乗客の想定所用時間である。 所用距離と所要時間は乗車地点から降車地点までを最短で移動した場合の経路であり、乗り合いによる遠回りは考慮しない。 r_t^i は乗客から得る乗車費用を想定した正の報酬である。 この報酬は RS 企業が決定するもので、乗客を割り当てたエージェントに対し与える。 他方、 f_g は移動のコストを表し、エージェントが移動するごとに負の報酬として $r_t'^i$ を与える。 RS 企業として得られる乗車費用の最大化を目指す、報酬はエージェントごとに与える。

エージェントは C_i を拡大することで乗客が割り当てられやすくなる。 他方、複数の割り当て候補がいる場合は乗客にもっとも近いエージェントが選ばれるために競合が発生する。 エージェントは他エージェントの SA を考慮しながら、乗客が発生しやすい地域に C_i を移動させる必要がある。

4.2 DQN の構造

本稿では、[20] が拡張した DQN である Dueling-Network を用いる。 これは画像表現の入力を処理する Convolutional Neural Network(CNN) 層、CNN 層からの出力を処理し Q 値を出力する Fully Connected Neural Network(FC) 層からなる。 SAAMS に用いたネットワーク構造を図 4 に示す。 SAAMS ではエージェントごとに図 4 のネットワークを持ち、独立して Q 値を学習する。

入力の例を図 5 に示す。 図における数値は、需要予測ではそのセルから発生する乗客数を表し、SA の情報ではセルに存在する SA の数を表す。 SA は大きさを持つため、1 体のエージェントの SA の情報が複数のセルに含まれる (図 5 の Own Service Area では、1 体のエージェントの SA が 3×3

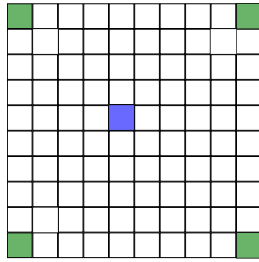


図 6: 実験環境

のセルを含む). エージェントはネットワークへの入力として, 需要予測, 自身の SA の位置 C_i , および全エージェントの SA の位置 $\sum_{i=1}^n C_i$ を得る. これにより, 個々のエージェントは他エージェントの全ての SA の位置を得ることができ, 他エージェントの SA を考慮し学習する.

4.3 SAAMSwT

SAAMSwT は, 学習済みエージェントのモデルを複製することで環境のエージェント数を増加させる. 例えばマンハッタンでは 5000 台程のエージェントを運用するように [13], RS ではエージェント数が非常に多く, 全てのエージェントが 0 から学習するには大幅な時間を要する. そこで, 学習済みモデルを活用することでこの時間を削減したい.

エージェント数 N の環境において, 学習の流れは以下の通りである.

- (1) 想定する環境において, $N > n$ である n 台のエージェントで学習させる
- (2) n 台のエージェントから任意のモデルを選び複製し, エージェントを追加する
- (3) (2) をエージェント台数が N 台に達するまで繰り返す
- (4) N 台で学習を行う

ここで, どのエージェントのモデルを複製すべきかという問題がある. 本実験では想定するエージェント数 N の半分, もしくは十分の一を学習させ, それら全てを複製する. n 台のエージェントは担当地域となるデマンドが発生しやすい地点, および 1 日の発生の変化を学習する. これはエージェント数が変化しても変わらないため, 知識を他のエージェントに活用することができる. これにより, 追加するエージェントは 0 から学習するよりも短い時間で待機地域の学習が可能となる.

5. 実験

5.1 実験設定

本稿では, 実験として図 6 の環境での実験, およびニューヨークのマンハッタンにおける実データ ([21]) を用いた 2 つの実験を行う. マンハッタの実データを用いることで提案手法の性能を評価できるが, 乗客の発生が複雑であり解析がしづらい. そこでより単純な図 6 の環境で実験を行う

表 1: グリッド環境における実験パラメータ

Parameter	Value
エージェント数, N	20
環境サイズ (km)	5.0
定員, Z	4
SA の最大サイズ, T_x, T_y	5
配車手数料, f_b	3
移動コスト, f_g	1
1 試行ステップ長	150

表 2: マンハッタン環境における実験パラメータ

Parameter	Value
エージェント数, N	100
環境サイズ (km)	22×5
定員, Z	4
SA の最大サイズ, T_x, T_y	5
配車手数料, f_b	3
移動コスト, f_g	1
1 試行ステップ長	1440

表 3: シナリオ 1 における発生率 λ (30 ステップあたり)

セル	$t = 0$	$t = 30$	$t = 60$	$t = 90$	$t = 120$
緑	4	4	4	4	4
青	0.025	0.025	0.025	0.025	0.025
その他	0.025	0.025	0.025	0.025	0.025

表 4: シナリオ 2 における発生率 λ (30 ステップあたり)

セル	$t = 0$	$t = 30$	$t = 60$	$t = 90$	$t = 120$
緑	0.25	0.25	0.25	0.25	0.25
青	0.25	0.25	0.25	0.25	0.25
その他	0.25	0.25	0.25	0.25	0.25

表 5: シナリオ 4 における発生率 λ (30 ステップあたり)

セル	$t = 0$	$t = 30$	$t = 60$	$t = 90$	$t = 120$
緑	4	2.5	0.025	2.5	4
青	0.025	8	18	8	0.025
その他	0.025	0.025	0.025	0.025	0.025

表 6: 需要予測に用いるニューラルネットワーク

Layer	Input	Filter size	Stride	Activation	Output
Convolutional	$22 \times 11 \times 2$	3×3	1	rectifier nonlinearity	$22 \times 11 \times 32$
Convolutional	$22 \times 11 \times 32$	2×2	1	rectifier nonlinearity	$22 \times 11 \times 64$
Convolutional	$22 \times 11 \times 1$	1×1	1	rectifier nonlinearity	$22 \times 11 \times 1$

ことで, 性能を解析する.

グリッド環境での実験における各種パラメータを表 1 に示す. 図 6 の環境では, 乗客はポアソン分布 $P_o(\lambda)$ に従い発生する. 発生率 λ の値は各セルごとに異なり, 発生の分布が違う 4 つの環境を用いる. 静的な環境として, 実験中で各セルの発生率 λ が変化しない環境を 2 種類用意した. 1 つは図 6 の緑のセルからデマンドが発生しやすい環境であり, 他

方は環境全体の発生率が一定の環境である。これらの環境の分布を表 3, 4 に示す。動的な環境として、2 種類の環境を作成した。図の青色のセルを駅、緑色のセルをランドマークと想定し、駅とランドマーク間のデマンドが多く発生する環境とする。1 つは、30 ステップごとに駅、ランドマークの 5 点から任意の一点を選び、選んだセルの発生率 $\lambda = 12$ 、選ばれなかった他の 4 点が $\lambda = 1$ 、色が無いセルを $\lambda = 0.025$ とする。この環境はイベント等により、1 点にたくさんの乗客が発生することを想定している。もう 1 つは、時間に応じて発生する場所がランドマークから駅へと規則的に変化していく環境であり、このときの各セルの発生率 λ の変化を表 5 に示す。動的環境では発生しやすい場所が変化するため、エージェントは発生率の変化に従い SA を変化させる必要がある。

エージェントには需要予測を与えるが、本実験では表 3, 4 のような発生率 λ を各セルの需要予測としてエージェントに与える。発生率に合わせてデマンドを生成するため、予測された数と実際に発生する数は必ずしも一致しない。需要予測は 30 ステップごとに更新する。配車時間の制限である t_l は、乗り合いを考慮せずに最短距離で配車したときの倍の時間とした。

グリッド環境のシミュレーションはエージェント数 20 台で行う。実験では、提案手法となる SAAMSwT は、事前にエージェント数 10 体で 1500 エピソード学習し、そのモデルをコピーしエージェント数 20 体で 500 エピソード学習した。比較手法として、最初からエージェント数 20 台で 2000 エピソード学習した SAAMS, RHC[5] を用いる。RHC は、セルごとの需要予測と車の台数が小さくなるように線形計画問題を解くことで車の位置を制御する手法である。需要予測は SAAMS と同様の情報を与える。配車を SAAMS と同様に扱うために、エージェントを中心とした 5×5 の範囲の乗客を受け入れる。

マンハッタンの実データ ([21]) には、24 時間通じた乗客の乗車時間、乗車位置、降車位置などの情報が含まれる。実験ではマンハッタンを 22×11 のセルに分割し、乗客は対応した乗車位置、降車位置のセルから発生する。実験は 1 ステップを 1 分として 24 時間分行い、実データに基づいた時間に乗客を発生させる。用いる実データとして、学習時には 2018 年 2 月 1 日から 2 月 14 日のデータを用い、実験時には 2018 年 2 月 15 日から 2 月 21 日のデータを用いる。実験のパラメータを表 2 に示す。エージェントには需要予測を与えるが、需要予測の生成をニューラルネットワークで行う。このニューラルネットワークは入力として現在から 30 ステップ前までに各セルで発生したデマンド数、および 30 ステップ前から 60 ステップ前までに各セルで発生したデマンド数の 2 枚の画像情報を与える (それぞれ図 5 の需要予測と同様の情報である)。ニューラルネットワークの構造を表 6 に示す。需要予測は 30 ステップごとに更新する。

表 7: 学習に要した時間 (s)

Parameter	Scenario 1	Scenario 2	Scenario 3	Scenario 4
SAAMSwT	6907.1	6688.7	6926.8	6563.5
SAAMS	9529.1	9539.3	9551.0	9766.4

表 8: 獲得総利益 (\$)

Parameter	Scenario 1	Scenario 2	Scenario 3	Scenario 4
SAAMSwT	676.2	768.4	510.3	643.3
SAAMS	682.7	801.3	502.2	728.7
RHC	631.8	766.1	464.9	619.5

表 9: 乗客の旅行時間に対する乗車待ち時間の割合 (%)

Parameter	Scenario 1	Scenario 2	Scenario 3	Scenario 4
SAAMSwT	5.3	19.0	13.4	9.3
SAAMS	5.4	19.0	12.9	7.7
RHC	11.0	17.1	11.3	10.5

マンハッタンのシミュレーションは、エージェント数 100 台で行う。実験では、提案手法となる SAAMSwT は、事前にエージェント数 10 体で 80 エピソード学習し、そのモデルをコピーしエージェント数 100 体で 20 エピソード学習した。比較手法として、最初からエージェント数 100 台で 100 エピソード学習した SAAMS, および RHC を用いる。

SAAMS のトレーニング時における報酬の重みを、 $w_b = 3$, $w_d = 0.2$, $w_t = 0.2$, $w_g = 0.05$ とした。これらは RS 実運用における乗車費用を参考に決定した値であり、単位は米ドルである。エージェントは ϵ -greedy で学習し、 ϵ は学習中に 1 から 0.1 へと変化させる。

実験では評価指標として、獲得総利益、乗客の旅行時間に対する乗車待ち時間の割合、車の平均総移動距離の 3 つを用いる。獲得総利益はエージェントから得た乗車費用の総和であり、RS において最も重要な指標である。利益の計算は 4.1 項の正の報酬の計算式を使用し、各重みは $w_b = 3$, $w_d = 0.2$, $w_t = 0.2$ とした。乗客の旅行時間に対する乗車待ち時間の割合は、乗客が乗車要求をしてから降車地点に移動するまでの旅行時間全体に対して、乗車要求をしてから乗車するまでの時間が占める部分である。エージェントは需要予測があるセルへ移動することで、この割合を小さくすることができる。車の平均総移動距離は、小さいほどガソリン消費が少なく効率の良いサービス提供が可能となる。この距離は乗客の乗車に関わらず、全ての時間の移動距離を含む。

5.2 実験結果

SAAMSwT と SAAMS における 2000 エピソードの学習に要した時間を表 7 に示す。SAAMSwT では、SAAMS と比べ約 30 % 学習時間を削減した。

獲得総利益の結果を図に示す。全てのシナリオにおい

表 10: 車の平均総移動距離 (cell)

Parameter	Scenario 1	Scenario 2	Scenario 3	Scenario 4
SAAMSwT	75.7	88.9	55.5	73.1
SAAMS	76.0	90.9	53.4	80.4
RHC	116.4	143.2	100.4	115.7

表 11: マンハッタン環境において学習に要した時間 (s)

Parameter	マンハッタン
SAAMSwT	12290
SAAMS	58117

表 12: マンハッタン環境における実験結果

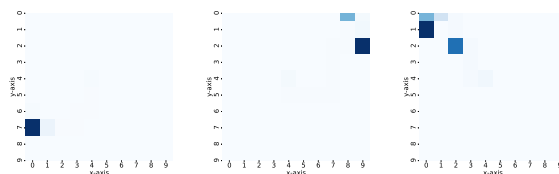
Parameter	獲得総利益 (\$)	待ち時間の割合 (%)	車の平均総移動距離 (cell)
dSAAMS	83714.7	39.2	1152.5
SAAMS	83755.0	39.1	1142.6
RHC	66991.3	44.2	1217.0

て, SAAMSwT, SAAMS は既存手法である RHC を超える性能を示した. シナリオ 1, シナリオ 3 では SAAMSwT は SAAMS と同等の性能を示したが, シナリオ 2, シナリオ 4 では SAAMSwT は SAAMS の性能に及ばなかった.

図 7, 8 に SAAMSwT のエージェントがシナリオ 1 において, 1 エピソード中に訪問したセルをプロットしたヒートマップを示す. ここで, それぞれの画像のエージェント 1 から 3 は同一のエージェントである. シナリオ 1 では角のセルからデマンドが発生しやすいため, エージェントは角にいて乗客を乗せやすい. 図 7 では 10 体で学習しているときのエージェント 1 から 3 であるが, 角から離れた位置に SA がある. 一方, 図 8 は 20 体で学習しているときのエージェント 1 から 3 であるが, 10 体で学習していた図 7 と比べて SA が角に移動している. これは環境中のエージェント数に応じて, 報酬を獲得しやすい位置が変化するためである. この結果から, エージェントを複製し追加したとき, エージェントはより報酬が得られる SA に変化させる. 他方, シナリオ 2 は環境全体から発生する環境であり, 10 体のときの 20 体となったときに, 必要な SA の変化が大きく学習が足りていないと考えられる.

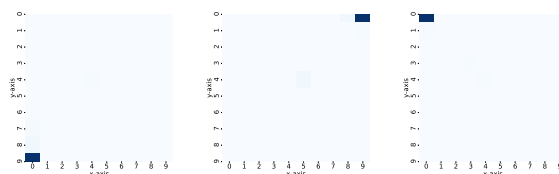
乗客の旅行時間に対する乗車待ち時間の割合の結果を図 9 に, 車の平均総移動距離の結果を図 10 に示す. これらの項目では, SAAMSwT は SAAMS と同等の性能を示した. RHC と比較すると, 移動距離では全ての項目において優位であるが, 乗客の待ち時間では, シナリオ 1 とシナリオ 4 のみ優位となった. これは, シナリオ 1 とシナリオ 4 では発生しやすい地点, およびその変化に傾向があり, 予測に従うことであらかじめ乗客が発生しやすい地点に SA を移動できる. 一方で, シナリオ 2 とシナリオ 3 は発生に規則性が少なく, 待機位置によって待ち時間を削減しづらい.

最後に, マンハッタンにおける学習時間の比較を表 11 に, 実験の結果を表 12 に示す. 台数が増えたことで SAAMS は



(a) エージェント 1 (b) エージェント 2 (c) エージェント 3

図 7: シナリオ 1 におけるエージェントの SA の訪問回数 SAAMSwT・10 体学習時



(a) エージェント 1 (b) エージェント 2 (c) エージェント 3

図 8: シナリオ 1 におけるエージェントの SA の訪問回数 SAAMSwT・20 体学習時

学習時間が伸び, SAAMSwT は学習時間を 79 %削減することができた. また実験結果では, 全ての指標で SAAMSwT は既存手法である RHC よりも良く, SAAMS と同等の性能を示している. SAAMSwT はマンハッタンの実データにおいても, 学習の性能は下げずに, 学習時間を削減することができた.

6. まとめ

本研究では, RS サービスにおいて, 各車のサービスエリアを制御する SAAMS が既存手法以上の利益を得ながら, 乗客の待ち時間, 車の走行距離を削減し, より効率的な運行が可能となることを示した. また, あらかじめ少ないエージェント数で学習をし, それらのモデルを複製することでエージェント数を増やす SAAMSwT は, SAAMS と同等の性能を示しながら, 学習時間を削減することができた.

今回は, 事前に学習させた全てのエージェントを複製することでエージェント数を増加させた. しかし, 実際にはエージェントごとに重要度が異なると考えられる. 例えば, 駅を担当地域とするエージェントは多く複製する必要がある. 郊外を担当したエージェントは追加しなくても十分かもしれない. より適切な複製をすることで, N 台での学習時間をより削減できる可能性があり, 複製する戦略について考えていく必要がある.

謝辞 本研究の一部は, JST, 未来社会創造事業, JP-MJMI19B5, および JSPS 科研費 (20H04245) の支援を受けたものである.

参考文献

- [1] Niam Yaraghi and Shamika Ravi. The Current and Future State of the Sharing Economy. *SSRN Electronic Journal*, 01 2017.
- [2] Gregory D. Erhardt, Sneha Roy, Drew Cooper, Bhargava Sana, Mei Chen, and Joe Castiglione. Do transportation network companies decrease or increase congestion? *Science*

- Advances*, 5(5), 2019.
- [3] Jintao Ke, Hongyu Zheng, Hai Yang, and Xiqun (Michael) Chen. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies*, 85:591–608, 2017.
- [4] Wei Ma, Xidong Pi, and Sean Qian. Estimating multi-class dynamic origin-destination demand through a forward-backward algorithm on computational graphs, 03 2019.
- [5] Fei Miao, Shan Lin, Sirajum Munir, John A Stankovic, Hua Huang, Desheng Zhang, Tian He, and George Pappas. Taxi dispatch with real-time sensing data in metropolitan areas a receding horizon control approach. *IEEE Transactions on Automation Science and Engineering*, 13, 04 2015.
- [6] Ramón Iglesias, Federico Rossi, Kevin Wang, David Hallac, Jure Leskovec, and Marco Pavone. Data-driven model predictive control of autonomous mobility-on-demand systems. *2018 IEEE International Conference on Robotics and Automation*, pages 1–7, 2017.
- [7] 吉田 直樹, 野田 五十樹, and 菅原 俊治. 分散 / 集中制御によるマルチエージェント深層強化学習を用いたライドシェアのサービスエリア制御の比較. 人工知能学会全国大会論文集, JSAI2020:2J4GS204–2J4GS204, 2020.
- [8] Jean-François Cordeau and Gilbert Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594, 2003.
- [9] Gerardo Berbeglia, Jean-François Cordeau, and Gilbert Laporte. A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem. *INFORMS Journal on Computing*, 24:343–355, 07 2012.
- [10] Timo Gschwind and Michael Drexler. Adaptive large neighborhood search with a constant-time feasibility test for the dial-a-ride problem. *Transportation Science*, pages 400–413, 02 2018.
- [11] Diego Muñoz-Carpintero, Doris Saez, Cristián Cortés, and Alfredo Núñez. A methodology based on evolutionary algorithms to solve a dynamic pickup and delivery problem under a hybrid predictive control approach. *Transportation Science*, 49:150310112916006, 03 2015.
- [12] 中島 秀之, 野田 五十樹, 松原 仁, 平田 圭二, 田柳 恵美子, 白石 陽, 佐野 渉二, 小柴 等, and 金森 亮. バスとタクシーを融合した新しい公共交通サービスの概念とシステムの実装. 土木学会論文集 D3 (土木計画学), 71(5):875–888, 2015.
- [13] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467, 2017.
- [14] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dhharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.
- [15] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.
- [16] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik’s cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.
- [17] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD ’18, pages 1774–1783, New York, NY, USA, 2018. ACM.
- [18] Takuma Oda and Carlee Joe-Wong. MOVI: A Model-Free Approach to Dynamic Fleet Management. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 2708–2716, April 2018.
- [19] J. Wen, J. Zhao, and P. Jaillet. Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 220–225, 2017.
- [20] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003, 2016.
- [21] NYC. Nyc taxi limousine commission-trip record data-nyc.gov. <http://www.nyc.gov>, 2020.