Proceedings of Machine Learning Research vol 242:1–13, 2024 6th Annual Conference on Learning for Dynamics and Control

Global Rewards in Multi-Agent Deep Reinforcement Learning for Autonomous Mobility on Demand Systems

Heiko Hoppe HEIKO.HOPPE@TUM.DE
Technical University of Munich, Germany

Tobias Enders TOBIAS.ENDERS@TUM.DE
Technical University of Munich, Germany

Quentin Cappart QUENTIN.CAPPART@POLYMTL.CA
Polytechnique Montréal, Canada

Maximilian Schiffer SCHIFFER@TUM.DE
Technical University of Munich, Germany

Editors: A. Abate, K. Margellos, A. Papachristodoulou

Abstract

We study vehicle dispatching in autonomous mobility on demand (AMoD) systems, where a central operator assigns vehicles to customer requests or rejects these with the aim of maximizing its total profit. Recent approaches use multi-agent deep reinforcement learning (MADRL) to realize scalable yet performant algorithms, but train agents based on local rewards, which distorts the reward signal with respect to the system-wide profit, leading to lower performance. We therefore propose a novel global-rewards-based MADRL algorithm for vehicle dispatching in AMoD systems, which resolves so far existing goal conflicts between the trained agents and the operator by assigning rewards to agents leveraging a counterfactual baseline. Our algorithm shows statistically significant improvements across various settings on real-world data compared to state-of-the-art MADRL algorithms with local rewards. We further provide a structural analysis which shows that the utilization of global rewards can improve implicit vehicle balancing and demand forecasting abilities. An extended version of our paper, including an appendix, can be found at `https://arxiv.org/abs/2312.08884`. Our code is available at `https://github.com/tumBAIS/GR-MADRL-AMoD`.

Keywords: multi-agent learning, credit assignment, deep reinforcement learning, autonomous mobility on demand

1. Introduction

Within the coming years, autonomous mobility on demand (AMoD) promises to significantly improve urban passenger transportation. An AMoD system enables its operator to exercise full control over the vehicle fleet, thus improving vehicle dispatching performance, provided that the operator uses an effective control algorithm. This ultimately yields increased vehicle utilization while preserving the comfort of individual mobility. By deciding which requests to accept and assigning vehicles to these requests, the operator faces a contextual multi-stage stochastic control problem. Various approaches to solve this problem exist, ranging from model predictive control (e.g., Alonso-Mora et al., 2017) to deep reinforcement learning (DRL) (e.g., Xu et al., 2018; Wang et al., 2018; Enders et al., 2023). The latter shows better performance due to its model-free nature. In this context, multi-agent approaches allow to achieve scalability to large instances. To train the agents' policy, existing work uses local, egoistic, per-agent rewards, which can lead to sub-optimal behavior from the perspective of the central operator interested in maximizing the system-wide profit.

In this work, we improve upon the state-of-the-art local-rewards algorithm (LRA) of Enders et al. (2023) by proposing a novel way to train agents with global rewards. The core of our methodology is a new advantage function, which estimates the individual agents' contribution to the global reward, as such aligning their goal with the central operator, ultimately following a system-optimal policy. Our algorithm outperforms LRA by up to 2% on average across test dates and up to 6% on individual dates, which is a substantial performance improvement in AMoD. Additionally, our algorithm is as scalable as LRA, accordingly setting a new state-of-the-art for vehicle dispatching in AMoD systems.

1.1. Related work

Our work relates to two research streams: from an application perspective to AMoD fleet control, and methodologically to multi-agent deep reinforcement learning (MADRL). We review related literature from both streams in the following.

Algorithms for (A)MoD fleet control range from greedy heuristics (e.g., Liao, 2003; Lee et al., 2004), to (stochastic) model predictive control (e.g., Alonso-Mora et al., 2017; Tsao et al., 2018), combinations of optimization and supervised learning (e.g., Zhang et al., 2017; Jungel et al., 2023), and DRL. Existing DRL-based algorithms differ from our approach, as they often use value-based learning (e.g., Xu et al., 2018; Wang et al., 2018; Sadeghi Eshkevari et al., 2022; Meneses-Cime et al., 2022), which handles large discrete action spaces less efficiently (cf. Akkerman et al., 2024). Others focus on explicit rebalancing instead of request assignment (e.g., Jiao et al., 2021; Liang et al., 2022; He et al., 2022). Furthermore, most previous work considers non-autonomous MoD, aiming to maximize revenues (e.g., Wang et al., 2018; Xu et al., 2018; Tang et al., 2019), while operators of AMoD systems commonly focus on maximizing operational profit (cf. Enders et al., 2023). Finally, the work of Enders et al. (2023) uses policy-based hybrid MADRL for a profit-maximizing AMoD operator, but trains agents on local rewards, possibly leading to sub-optimal agent behavior. To the best of our knowledge, no work exists that addresses the problem of profit-maximizing vehicle dispatching in AMoD without the aforementioned shortcomings. We address this research gap by extending the work of Enders et al. (2023) to incorporate global rewards, thus creating goal congruence between the agents and the system operator.

A crucial challenge in our setting is deriving per-agent contributions to global success from a shared reward signal, i.e., a credit assignment problem (Weiß, 1995; Wolpert and Tumer, 1999; Chang et al., 2003). Solution approaches like inverse reinforcement learning (e.g., Ng and Russell, 2000; Hadfield-Menell et al., 2017; Lin et al., 2018) or value decomposition (e.g., Kok and Vlassis, 2006; Sunehag et al., 2018; Son et al., 2019; Rashid et al., 2020) are not applicable to our setting, because we cannot observe the behavior of an optimal agent and do not use Q-learning. An alternative approach is reward marginalization, based on difference rewards (Wolpert and Tumer, 2001), which uses functions (e.g., advantage functions) to estimate the contribution of individual agents to global rewards (e.g., Nguyen et al., 2018; Wu et al., 2018; Foerster et al., 2018). Reward marginalization approaches often use actor-critic algorithms. However, reward marginalization approaches often suffer from high learning variance and poor sample efficiency, as they are typically built on top of basic DRL algorithms. To the best of our knowledge, no work exists that combines reward marginalization approaches with low-variance actor-critic algorithms so far. We address this

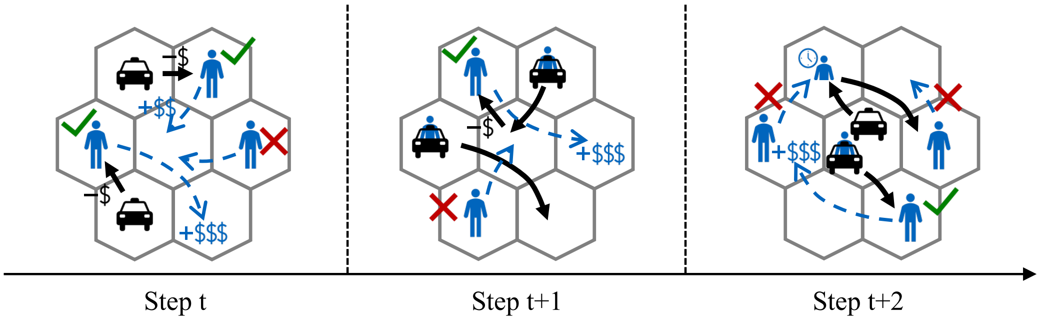

Figure 1: Exemplary vehicle dispatching process.

research gap by embedding reward marginalization into a Soft Actor-Critic (SAC) framework for discrete actions.

1.2. Contribution

To close the research gap outlined above, we develop the first scalable MADRL-based algorithm for vehicle dispatching in profit-maximizing AMoD that trains agents via global rewards. In this context, we combine SAC for discrete actions with credit assignment based on a counterfactual baseline to resolve goal conflicts between the trained agents and the operator's global objective. Our algorithm combines the benefits of low learning variance and sample efficiency of SAC with the benefits of credit assignment via a counterfactual baseline. Since this algorithm based on purely global rewards scales only to medium-sized problem instances, we additionally develop a scheduled algorithm that combines local rewards and global rewards with our counterfactual baseline. Thus, we obtain a powerful new MADRL algorithm with possible applications beyond AMoD. We evaluate our algorithm on real-world taxi data and show that it outperforms LRA of Enders et al. (2023) by up to 2% on average across test dates and up to 6% on individual dates. This constitutes a substantial performance improvement in an AMoD context, where a single percent improvement yields significant daily monetary gains. We further provide a structural analysis which shows that the utilization of global rewards can improve implicit vehicle balancing and demand forecasting abilities.

2. Problem setting

We focus on the contextual multi-stage stochastic control problem from Enders et al. (2023), illustrated in Figure 1, to ensure comparability. In this problem setting, a central operator controls an AMoD fleet and dispatches vehicles to serve requests. We consider a discrete time horizon. During each time step, customers submit a variable number of new requests for point-to-point transportation, of which the operator has no prior knowledge. At the beginning of each time step, the operator makes a decision for the batch of requests that were placed during the previous time step. For each request, this decision is either to reject the request or to assign it to a vehicle. The operator can base its decision on fully observable state information, including the requests' origins and destinations as well as the vehicles' positions and already assigned but not yet completed requests. Each request has to be decided upon immediately and customers have a fixed maximum waiting time until they must be picked up. Based on the operator's decision, the system transitions to the next state: vehicles move towards their next destination and potentially pick up or drop off customers. Customers place new requests, while rejected requests leave the system. We simulate the requests by replaying historical data. Finally, the operator receives the system-wide profit as a reward. The profit is calcu-

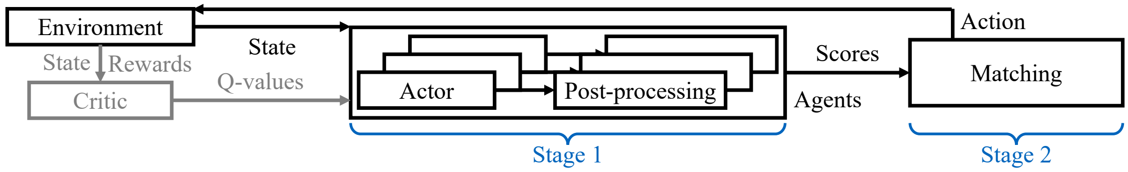

Figure 2: Outline of base algorithm. Black parts are used during training and testing, gray parts only during training.

lated as the sum of all revenues from accepted requests minus the costs for all vehicle movements. Revenues and costs are linear functions of the distance travelled. Since fixed costs do not depend on the dispatching problem, we only include operational costs (e.g., for fuel and maintenance). For further details, including a formal Markov decision process, we refer to Appendix A.

3. Methodology

In the following, we first motivate our baseline algorithm (cf. Enders et al., 2023) and extend it with a naive global reward allocation scheme. We then focus on reward marginalization and show how to modify the existing Counterfactual Multi-Agent Policy Gradient (COMA) paradigm to SAC architectures. We finally discuss enhancements to scale our algorithm to large-scale instances.

3.1. Base algorithm

Our problem setting has two peculiarities influencing the construction of an effective vehicle dispatching algorithm: the global action space is too large to use single-agent DRL and the number of requests varies between time steps. When addressing the first peculiarity with multi-agent learning, the need for agent coordination arises. The algorithm of Enders et al. (2023) therefore uses a two-stage architecture, see Figure 2. The first stage consists of DRL-trained agents, the second one of an optimization-based central matching. In the first stage, each agent is the combination of a vehicle and a request. Therefore, there exists an agent for every possible vehicle-request combination. An agent can either accept the assignment of the respective request to the respective vehicle or reject this assignment, resulting in a per-agent action space of size two. As the number of requests varies, so does the number of agents. Each agent observes the global state, using an encoding and an attention mechanism to cope with the varying number of requests. Using an actor neural network with a softmax function, the agent obtains probabilities for acceptance and rejection. The per-agent post-processing selects the action by sampling during training or taking the action with the highest probability during testing. It then sets the agent's score to the acceptance probability in case of an acceptance and to zero in case of a rejection. The scores of all agents are submitted to the global matching, which coordinates the agents. This bipartite matching selects the global action by maximizing the submitted scores under the constraints that every request can be assigned at most once and that every vehicle can take at most one new request.

We train the agents using SAC for discrete actions (Haarnoja et al., 2018; Christodoulou, 2019). As the use of multiple agents only serves to handle the large action space, we train a single actor and a single critic network and use them for all agents. The rewards we use for training are the profits obtained by the algorithm: when a request is assigned to a vehicle and served within the maximum waiting time, the respective agent immediately receives the profit as a reward.

To transition from this algorithm to our contribution, we first follow a naive approach to include global rewards in training by replacing the per-agent rewards with global rewards when training the critic, resulting in a basic global-rewards algorithm (GRA). We obtain global rewards by summing the profits of all agents at one time step. As this number can be substantially larger than per-agent rewards, we divide it by the average number of non-zero rewards per observation in the replay buffer to stabilize learning. Straightforwardly using this approach leads to a credit assignment problem, as the reward given to agents now depends on the actions of all agents, which generally complicates an agent's learning task. To mitigate this, we explore a credit assignment paradigm in the following.

3.2. Naive COMA

A suitable credit assignment paradigm for our setting is COMA, proposed by Foerster et al. (2018). COMA fits our setting best, because it has a similar structure as SAC and is especially suitable for small per-agent action spaces. Following the main rationale of COMA, agents should maximize their contribution to the global reward instead of maximizing the global reward directly. As obtaining global rewards for several actions is computationally infeasible, COMA trains the critic on global rewards to approximate global state-action-values. The contribution of an agent to this global value is defined as the value of taking an action in contrast to the value of taking a default action. This default action is calculated as the policy-weighted average value of all possible actions the agent can take (counterfactual baseline). The advantage function of COMA therefore is

$$A_i(a_i|s,i) = Q_\theta(a_i|s,\bar{a}_{-i}) - \sum_{a_i'} \pi_\phi(a_i'|s,i)\, Q_\theta(a_i'|s,\bar{a}_{-i}). \tag{1}$$

In this and all following equations, s denotes the global state, a_i' an action agent i can take as a reject/accept decision before the global matching, a_i the action the agent actually takes, and $\bar{a}_{-i}$ the actions of all agents except agent i after the global matching. Let $\pi_\phi(a_i|s,i) \in [0,1]$ be the probability that agent i takes action a_i in state s, following policy π_ϕ parameterized by network ϕ. $Q_\theta(a_i|s,\bar{a}_{-i}) \in \mathbb{R}^2$ is the global state-action-value (Q-value) of action a_i taken by agent i in state s estimated using network θ, given other agents' actions $\bar{a}_{-i}$.

Foerster et al. (2018) use a sampling-based approach to estimate the actor's loss function and base its computation only on the action taken by the agent. The loss function thus reads $J_\pi(\phi) = \mathbb{E}_{s\sim D}\big[\sum_i A_i(a_i|s,i)\big]$, with D denoting the replay buffer. In contrast to that, Enders et al. (2023) use SAC with discrete actions, which considers all possible actions of an agent in the loss function

$$J_\pi(\phi) = \mathbb{E}_{s\sim D}\left[\sum_i \sum_{a_i} \pi_\phi(a_i|s,i)\left(\alpha \log \pi_\phi(a_i|s,i) - \min_{j\in\{1,2\}} \left\{Q_\theta^j(a_i|s,\bar{a}_{-i})\right\}\right)\right], \tag{2}$$

with $\alpha \in \mathbb{R}$ being the entropy coefficient. To use credit assignment via COMA in combination with the proven reliability and low variance of SAC, we need to integrate the baseline of COMA into the loss function of SAC. In the following, we use $\pi(a_i) := \pi_\phi(a_i|s,i)$ and $Q(a_i) := \min_{j\in\{1,2\}} \left\{Q_\theta^j(a_i|s,\bar{a}_{-i})\right\}$ for conciseness. Then, considering one instance of a batch and one agent, the loss function of SAC extended by the advantage of COMA reads

$$J_\pi(\phi|s,i) = \sum_{a_i} \pi(a_i)\left(\alpha \log \pi(a_i) - Q(a_i) + \sum_{a_i'} \pi(a_i')\, Q(a_i')\right). \tag{3}$$

Proposition 1 *The loss function $J_\pi(\phi|s,i)$ as defined in Equation (3) is equivalent to the entropy $J_\pi(\phi|s,i) = \sum_{a_i} \pi(a_i)\, \alpha \log \pi(a_i)$ of a plain SAC architecture.*

For a proof of Proposition 1, we refer to Appendix B. Accordingly, using the loss function as derived in Equation (3) does not allow to learn a meaningful policy such that we cannot simply apply the COMA paradigm to our SAC framework. This observation motivates us to study a novel approach to combine SAC for discrete actions with COMA to learn a good policy with global rewards.

3.3. Adjusted COMA for SAC architectures

To solve the convergence problem outlined above, we have to adjust the loss function in Equation (3). Using only the action taken by the agent for the loss function as in Foerster et al. (2018) does not lead to convergence even in small experimental instances, as it increases the loss function's variance. We therefore adjust $\pi(a_i')$, changing the weighting of the default action in the baseline. This is possible from a theoretical perspective, as the exact specification of a default action is not derived from the idea of difference rewards (Wolpert and Tumer, 2001), but left to the user's discretion.

Straightforwardly, we can define the default action by using an equally-weighted average instead of a policy-weighted average, resulting in the advantage function $A_i^{\text{equ}}(a_i) = Q(a_i) - \sum_{a_i'} \frac{1}{n_{a_i'}} Q(a_i')$, with n_{a_i} being the number of actions per agent. We call this algorithm COMA$^{\text{equ}}$, which resolves the convergence problem, but has a disadvantage: when the actor network estimates different probabilities for single actions, weighting all actions equally is not a reasonable default action. This problem is especially pronounced during late training, when the actor network is better at estimating action probabilities. We solve this issue by defining a second default action with the use of a target actor network $\bar{\phi}$. This network has the same structure and initialization as the actor network ϕ and is updated using exponential averages of the actor network's parameters, similar to target networks in Q-learning. The advantage function of this algorithm (COMA$^{\text{tgt}}$) is $A_i^{\text{tgt}}(a_i) = Q(a_i) - \sum_{a_i'} \pi_{\bar{\phi}}(a_i')\, Q(a_i')$. Since $\bar{\phi}$ differs from ϕ, this algorithm solves the convergence problem as well. During early training, COMA$^{\text{tgt}}$ is not as suitable as COMA$^{\text{equ}}$, since the sub-optimal action probabilities of an untrained target actor network are a disadvantage compared to equally-weighted actions. Later in training, $\bar{\phi}$ can estimate better action probabilities, making COMA$^{\text{tgt}}$ superior to COMA$^{\text{equ}}$.

Since we now have one algorithm especially suitable for early learning and one especially suitable for later learning, we combine these two and obtain COMA$^{\text{adj}}$, based on a dynamic combination of the two newly introduced advantage functions. The advantage function of COMA$^{\text{adj}}$ thus reads

$$\begin{aligned} A_i^{\text{adj}}(a_i) &= (1-\beta) A_i^{\text{equ}}(a_i) + \beta A_i^{\text{tgt}}(a_i) \\ &= Q(a_i) - (1-\beta) \sum_{a_i'} \frac{1}{n_{a_i}}\, Q(a_i') - \beta \sum_{a_i'} \pi_{\bar{\phi}}(a_i')\, Q(a_i'). \end{aligned} \tag{4}$$

Here, the hyperparameter $\beta \in [0, 1]$ is the weight of the COMA$^{\text{tgt}}$ baseline and follows a schedule depending on the training iteration, starting at zero and ending at one. Using a simple linear schedule usually works best, see Appendix C.4. Then, the loss function of COMA$^{\text{adj}}$ reads

$$J_\pi^{\text{adj}}(\phi|s, i) = \sum_{a_i} \pi(a_i) \left(\alpha \log \pi(a_i) - A_i^{\text{adj}}(a_i) \right). \tag{5}$$

With this loss function, COMA$^{\text{adj}}$ solves the credit assignment problem. In our experiments, COMA$^{\text{adj}}$ performs better than LRA, but has a scalability problem: when the number of agents increases beyond medium-sized problem instances, COMA$^{\text{adj}}$ fails to converge. Reasons for this are the diminishing influence of a single agent on global rewards and the overlap of many agents' actions when the number of agents increases, making learning per-agent Q-values difficult (cf. Rashid et al., 2020). We therefore investigate how to scale COMA$^{\text{adj}}$.

3.4. Reward scheduling

Usually, one could resolve the scalability problem of COMA$^{\text{adj}}$ straightforwardly by adjusting the critic to accommodate value factorization (e.g., Su et al., 2021), but this approach is infeasible in our setting as the number of agents is variable. Similarly, learning the critic on a static mix of local and global rewards in a local-global-rewards algorithm (LGRA) does not solve the scalability problem, since any non-negligible share of global rewards distorts learning when increasing the number of agents. In addition, reward marginalization with a counterfactual baseline is problematic for partially local rewards.

Instead, we can train a single actor network using a weighted average policy loss function, consisting of the loss function for LRA, $J_\pi^{\text{loc}}(\phi|s,i)$, by Enders et al. (2023) and COMA$^{\text{adj}}$. The loss function thus reads

$$J_\pi^{\text{scd}}(\phi|s,i) = (1-\kappa)\, J_\pi^{\text{loc}}(\phi|s,i) + \kappa\, J_\pi^{\text{adj}}(\phi|s,i), \tag{6}$$

with $\kappa \in [0,1]$ being the weight of the loss function of COMA$^{\text{adj}}$. Again, κ follows a schedule depending on the training iteration, increasing linearly, following a power function or jumping from zero to one at a specified point. Power functions with exponents between 0.01 and 0.5 generally work best, with larger exponents being more suitable for large instance sizes, see Appendix C.4. This leads to a new algorithm, we call it COMA$^{\text{scd}}$. COMA$^{\text{scd}}$ solves the scalability problem, as it enables the learning of global Q-values when increasing the number of agents. The reason this works is the utilization of experience collected following a mixed policy: this way, more diverse experience is available than if using solely own experience, thus improving learning without the destabilizing influence of increasing the entropy. We can therefore train four critic networks from the beginning on, two for local and two for global rewards. Two networks each are necessary for SAC, where we always use the minimum of the two state-action-values to avoid value overestimation. Due to the influence of both local and global rewards, COMA$^{\text{scd}}$ can sometimes have a lower performance than algorithms purely based on global rewards, but makes up for this by being as scalable as LRA, thus sacrificing a portion of its performance for scalability.

4. Numerical studies

We benchmark our algorithms using the experimental design of Enders et al. (2023), which bases on New York Taxi data (NYC TLC, 2015) in a hexagonal grid of Manhattan, with 38 large, 11 small, or 5 small zones. In this realm, we study five instances: two edge cases with high (5 zones, 15 vehicles) and low (11 zones, 6 vehicles) acceptance rates, two typical test cases (11 zones, 18 and 24 vehicles), and a comparatively large instance (38 zones, 100 vehicles). For details on the experimental design, we refer to Appendix C.1 and report hyperparameters for all models in Appendix C.2.

Firstly, we test COMA$^{\text{scd}}$ on all five instances and benchmark it against LRA of Enders et al. (2023) and a greedy algorithm, which considers requests in their order of submission, accepting

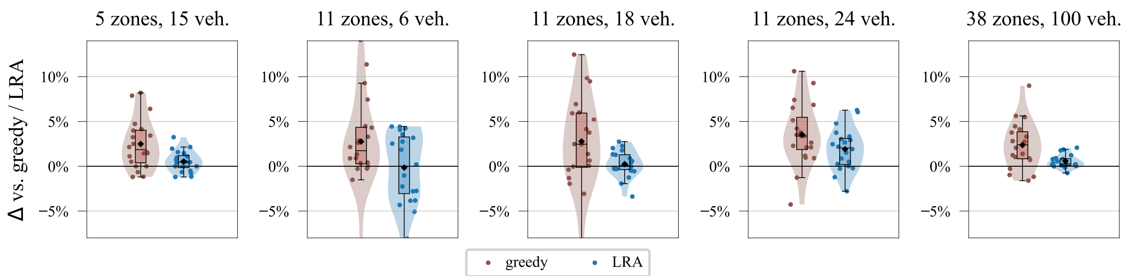

Figure 3: Relative test performance Δ [%] of COMA^{scd} vs. greedy and LRA for multiple test dates.

	5 zones, 15 veh.	11 zones, 6 veh.	11 zones, 18 veh.	11 zones, 24 veh.	38 zones, 100 veh.
vs. LRA	0.5%	-0.2%	0.2%	1.9%	0.6%
p-value	0.05	0.52	0.22	0.00	0.00
vs. greedy	2.5%	2.8%	2.8%	3.5%	2.4%
p-value	0.00	0.00	0.01	0.00	0.00

Table 1: Mean test performance improvement of COMA^{scd} vs. LRA and greedy, including the respective Wilcoxon p-values.

profitable ones and rejecting all others. Secondly, we present an ablation study to show the superiority of COMA^{scd} over our alternative algorithms that use global rewards. Thirdly, we discuss why COMA^{scd} outperforms LRA. For details on the performance metrics, we refer to Appendix C.3.

4.1. Performance of COMA^{scd}

We present results of our tests of COMA^{scd} in Figure 3 and Table 1. In all instances except the one with 11 zones and 6 vehicles, COMA^{scd} outperforms LRA and the greedy algorithm on average, the former by up to 1.9% and the latter by up to 3.5%. On single test dates, COMA^{scd} can outperform LRA by up to 6%. This improvement is significant, as the Wilcoxon p-values are at most 5% for the respective instances. While these relative improvements appear to be small, they are substantial in AMoD and of a similar magnitude as the improvements of previous state-of-the-art algorithms over their respective benchmarks (cf. Sadeghi Eshkevari et al., 2022; Enders et al., 2023).

In the instance with a high acceptance rate (5 zones, 15 vehicles), the significant performance improvement of COMA^{scd} compared to LRA is an especially positive result, as a vehicle is usually available for each request in this instance, limiting the improvement potential for DRL. In the instance with a low acceptance rate (11 zones, 6 vehicles), the improvement potential is similarly limited, as vehicles are rarely idle. Consequently, the performance of COMA^{scd} is most similar to LRA in this instance. In contrast, the instance of 11 zones and 24 vehicles has a balanced ratio between the number of vehicles and requests. Here, the performance improvement of COMA^{scd} is the largest of all instances, outperforming LRA by on average 1.9% and greedy by on average 3.5%. In the large instance (38 zones, 100 vehicles), COMA^{scd} significantly improves performance by on average 0.6% compared to LRA, proving that the algorithm is applicable to large-scale environments. The lower performance improvement can be explained by the weight of COMA^{adj} being required to increase more slowly in the loss function of COMA^{scd} when the number of agents increases.

4.2. Ablation study

In the following, we discuss the performance of all proposed algorithms with respect to numerical stability and scalability across random seeds (Table 2) as well as computational performance

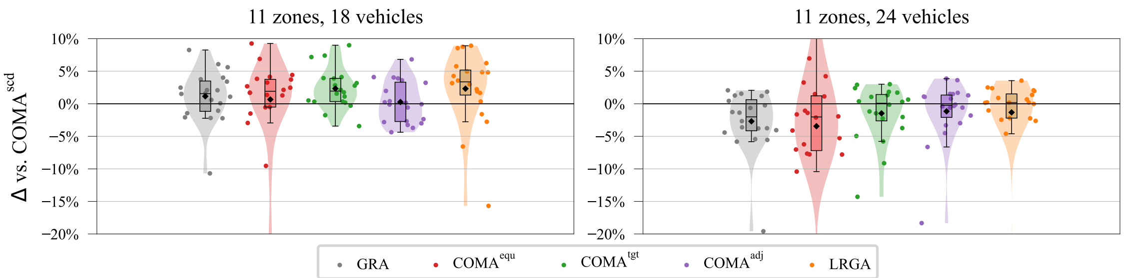

Figure 4: Relative test performance Δ [%] of all algorithms vs. COMA$^{\text{scd}}$.

	5 zones 15 veh.	11 zones 6 veh.	11 zones 18 veh.	11 zones 24 veh.	38 zones 100 veh.
GRA	✓	✓	○	○	–
COMA$^{\text{equ}}$	✓	✓	○	○	–
COMA$^{\text{tgt}}$	✓	✓	○	○	–
COMA$^{\text{adj}}$	✓	✓	○	○	–
LGRA	✓	✓	✓	✓	–
LRA	✓	✓	✓	✓	✓
COMA$^{\text{scd}}$	✓	✓	✓	✓	✓

Table 2: Convergence of algorithms. ✓ denotes stable convergence, ○ unstable convergence (across random seeds), – no convergence.

	11 zones 18 vehicles	11 zones 24 vehicles
GRA	1.2% (0.02)	-2.7% (0.01)
COMA$^{\text{equ}}$	0.7% (0.03)	-3.5% (0.04)
COMA$^{\text{tgt}}$	2.3% (0.00)	-1.5% (0.23)
COMA$^{\text{adj}}$	0.3% (0.42)	-1.2% (0.41)
LGRA	2.3% (0.00)	-1.3% (0.55)

Table 3: Test performance of algorithms vs. COMA$^{\text{scd}}$ (Wilcoxon p-value).

(Table 3 & Figure 4). As can be seen in Table 2, all algorithms show stable convergence for the small instances, while all but LRA, LGRA, and COMA$^{\text{scd}}$ exhibit stability issues already for medium-sized instances, failing to converge for one-third of the seeds and requiring about ten times as many training steps to converge compared to LRA for the remaining seeds. In contrast, LGRA and COMA$^{\text{scd}}$ converge on all seeds and require comparable to at maximum twice as many training steps compared to LRA. For the large instance, only COMA$^{\text{scd}}$ and the LRA baseline converge, with COMA$^{\text{scd}}$ requiring similar to twice as many training steps. Figure 4 and Table 3 show the relative performance of all algorithms compared to COMA$^{\text{scd}}$ for the medium-sized instances over seeds for which all algorithms converged. As can be seen, results are mixed: while pure global-rewards-based algorithms outperform COMA$^{\text{scd}}$ on average on the 18 vehicles instance, COMA$^{\text{scd}}$ outperforms all other algorithms on the 24 vehicles instance. To understand this ambiguous effect, we detail the algorithms' convergence behavior: increasing the instance from 18 to 24 vehicles technically requires to train 570 instead of 430 agents, which significantly challenges all purely global-rewards-based algorithms. In fact, looking at the validation reward variance of each algorithm (see Appendix C.5), we observe converging but less stable learning behavior for all purely global-rewards-based algorithms, which explains the respective performance drop. While this observation manifests the robustness of COMA$^{\text{scd}}$ at a performance that improves upon local-rewards-based algorithms, it also points at a promising direction for future research: if one manages to stabilize and scale the purely global-rewards-based algorithms, one will most likely obtain even better performance.

4.3. Structural analysis

Finally, we aim to understand the performance difference between the studied algorithms by analyzing the respective policy characteristics in Table 4, which details request rejection rates of profitable requests for our LRA baseline, the pure global-rewards-based algorithm COMA$^{\text{adj}}$, and COMA$^{\text{scd}}$. Both COMA-based algorithms have a lower rejection rate compared to LRA, which explains their improved performance. This finding, as well as the relation between COMA$^{\text{scd}}$ and COMA$^{\text{adj}}$, are

measure	LRA	COMA^{adj}	COMA^{scd}
rejection rate (of profitable requests)	17.6%	16.6%	15.5%
→ rejection rate for destination zones without vehicles	16.4%	12.9%	13.6%
→ rejection rate for destination zones with >2 vehicles	20.2%	19.3%	18.5%
overperformance ratio (rejections / acceptances)	1.75	1.87	1.27

Table 4: Rejection rates of generally profitable requests on the instance with 24 vehicles.

in line with the performance shown for the 24 vehicles instance in Figure 4. To understand operational intricacies, we analyze the difference between average rejection rates of empty destination zones and zones that contain more than two vehicles upon a request's arrival, which is 3.8% for LRA, 6.4% for COMA^{adj} and 4.9% for COMA^{scd}. This indicates a stronger focus of COMA^{scd} and COMA^{adj} on implicit vehicle balancing, as these algorithms are more reluctant to send vehicles to already crowded zones. Such a focus on vehicle balancing stems from global reward structures and partially explains performance improvements: if vehicles are unbalanced, less overall requests can be served; individual vehicles might still obtain high local rewards, but the global reward decreases.

Beyond implicit balancing, we analyze the algorithms' anticipative performance, i.e., their capability to foresee future demand and consider it during decision-making. To do so, we analyze an algorithm's overperformance ratio (see Table 4), which compares the summed theoretical profits of future requests in the same zone following acceptance or rejection of an initially profitable request. We calculate this ratio by dividing the total theoretical profit after rejections by that after acceptances (see Appendix C.6 for details). As can be seen, COMA^{adj} has the highest overperformance ratio, followed by LRA and COMA^{scd}. From the higher ratio, we conclude that COMA^{adj} has better forecasting abilities, as requests after rejections are more profitable than requests after acceptances under its policy. In contrast, COMA^{scd} appears to suffer from the mixture of local and global rewards, which can explain some of the performance gaps compared to algorithms with purely global rewards. A possible reason for the better forecasting abilities of COMA^{adj} is that global Q-values incorporate more implicit information about future demand and thus more prescriptive information, making them more representative: since global rewards are less dependent on the actions of individual agents, it is easier to infer information about demand from them.

5. Conclusion

We study vehicle dispatching for AMoD systems, where a profit-maximizing central operator assigns vehicles to requests or rejects these. We propose a novel MADRL algorithm with global rewards and credit assignment based on a counterfactual baseline. Our algorithm combines the benefits of low learning variance and sample efficiency of SAC with the benefits of credit assignment of COMA. We show that a naive combination of SAC and COMA does not converge, and develop a stable and scalable algorithm that uses reward scheduling with a new advantage function based on an equally-weighted baseline and a target actor network baseline. We show that our algorithm improves upon the current state-of-the-art by up to 2% on average across test dates and up to 6% on individual dates from real-world data sets. This constitutes a substantial performance improvement for the AMoD application case where a single percent improvement yields significant daily monetary gains. We further provide a structural analysis which shows that the use of global rewards can improve implicit vehicle balancing and demand forecasting abilities. Our proposed algorithm is applicable beyond the area of AMoD, as it can be useful in any application where stable multi-agent deep reinforcement learning with credit assignment is required. An interesting avenue for future research would be the comparison of our algorithm to decentralized learning algorithms with local actors and critics. Future work may furthermore apply our algorithm to other areas or focus on stabilizing purely global-rewards-based algorithms.

Acknowledgments

We thank the CORAIL research group at Polytechnique Montréal for valuable comments and discussions. The work of Heiko Hoppe was supported by a scholarship from the Max Weber-Programm Bayern.

References

Fabian Akkerman, Julius Luy, Wouter van Heeswijk, and Maximilian Schiffer. Dynamic Neighborhood Construction for Structured Large Discrete Action Spaces. arXiv preprint at arXiv, 2024. URL arXiv:2305.19891.

Javier Alonso-Mora, Alex Wallar, and Daniela Rus. Predictive routing for autonomous mobility-on-demand systems with ride-sharing. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

Yu-Han Chang, Tracey Ho, and Leslie Kaelbling. All learning is Local: Multi-agent Learning in Global Reward Games. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 16, 2003.

Petros Christodoulou. Soft Actor-Critic for Discrete Action Settings. arXiv preprint at arXiv, 2019. URL arXiv:1910.07207v2.

Tobias Enders, James Harrison, Marco Pavone, and Maximilian Schiffer. Hybrid Multi-agent Deep Reinforcement Learning for Autonomous Mobility on Demand Systems. In *Proceedings of The 5th Annual Learning for Dynamics and Control Conference (L4DC)*, volume 211 of *Proceedings of Machine Learning Research (PMLR)*, 2023.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual Multi-Agent Policy Gradients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J. Russell, and Anca Dragan. Inverse Reward Design. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Sihong He, Yue Wang, Shuo Han, Shaofeng Zou, and Fei Miao. A Robust and Constrained Multi-Agent Reinforcement Learning Electric Vehicle Rebalancing Method in AMoD Systems. arXiv preprint at arXiv, 2022. URL arXiv:2209.08230.

Yan Jiao, Xiaocheng Tang, Zhiwei (Tony) Qin, Shuaiji Li, Fan Zhang, Hongtu Zhu, and Jieping Ye. Real-world ride-hailing vehicle repositioning using deep reinforcement learning. *Transportation Research Part C: Emerging Technologies*, 130, 2021.

Kai Jungel, Axel Parmentier, Maximilian Schiffer, and Thibaut Vidal. Learning-based Online Optimization for Autonomous Mobility-on-Demand Fleet Control. arXiv preprint at arXiv, 2023. URL arXiv:2302.03963.

Jelle R. Kok and Nikos Vlassis. Collaborative Multiagent Reinforcement Learning by Payoff Propagation. *Journal of Machine Learning Research (JMLR)*, 7, 2006.

Der-Horng Lee, Hao Wang, Ruey Long Cheu, and Siew Hoon Teo. Taxi Dispatch System Based on Current Demands and Real-Time Traffic Conditions. *Transportation Research Record: Journal of the Transportation Research Board*, 1882(1), 2004.

Enming Liang, Kexin Wen, William H. K. Lam, Agachai Sumalee, and Renxin Zhong. An Integrated Reinforcement Learning and Centralized Programming Approach for Online Taxi Dispatching. *IEEE Transactions on Neural Networks and Learning Systems*, 33(9), 2022.

Ziqi Liao. Real-time taxi dispatching using Global Positioning Systems. *Communications of the ACM*, 46(5), 2003.

Xiaomin Lin, Peter A. Beling, and Randy Cogill. Multiagent Inverse Reinforcement Learning for Two-Person Zero-Sum Games. *IEEE Transactions on Games*, 10(1), 2018.

Karina Meneses-Cime, Bilin Aksun Guvenc, and Levent Guvenc. Optimization of On-Demand Shared Autonomous Vehicle Deployments Utilizing Reinforcement Learning. *Sensors*, 22(21), 2022.

Andrew Y. Ng and Stuart J. Russell. Algorithms for Inverse Reinforcement Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, 2000.

Duc Thien Nguyen, Akshat Kumar, and Hoong Chuin Lau. Credit Assignment For Collective Multiagent RL With Global Rewards. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

NYC TLC. Trip Record Data. Online, 2015. URL https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. *Journal of Machine Learning Research (JMLR)*, 21, 2020.

Soheil Sadeghi Eshkevari, Xiaocheng Tang, Zhiwei Qin, Jinhan Mei, Cheng Zhang, Qianying Meng, and Jia Xu. Reinforcement Learning in the Wild: Scalable RL Dispatching Algorithm Deployed in Ridehailing Marketplace. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022.

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

Jianyu Su, Stephen Adams, and Peter Beling. Value-Decomposition Multi-Agent Actor-Critics. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(13), 2021.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018.

Xiaocheng Tang, Zhiwei (Tony) Qin, Fan Zhang, Zhaodong Wang, Zhe Xu, Yintai Ma, Hongtu Zhu, and Jieping Ye. A Deep Value-network Based Approach for Multi-Driver Order Dispatching. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

Matthew Tsao, Ramon Iglesias, and Marco Pavone. Stochastic Model Predictive Control for Autonomous Mobility on Demand. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

Zhaodong Wang, Zhiwei Qin, Xiaocheng Tang, Jieping Ye, and Hongtu Zhu. Deep Reinforcement Learning with Knowledge Transfer for Online Rides Order Dispatching. In *2018 IEEE International Conference on Data Mining (ICDM)*, 2018.

Gerhard Weiß. Distributed Reinforcement Learning. *The Biology and Technology of Intelligent Autonomous Agents*, 1995.

David H. Wolpert and Kagan Tumer. An Introduction to Collective Intelligence. Techreport NASA-ARC-IC-99-63, NASA, 1999.

David H. Wolpert and Kagan Tumer. Optimal Payoff Functions For Members Of Collectives. *Advances in Complex Systems*, 04(02n03), 2001.

Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M. Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance Reduction for Policy Gradient with Action-Dependent Factorized Baselines. In *6th International Conference on Learning Representations (ICLR)*, 2018.

Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

Lingyu Zhang, Tao Hu, Yue Min, Guobin Wu, Junying Zhang, Pengcheng Feng, Pinghua Gong, and Jieping Ye. A Taxi Order Dispatch Model based On Combinatorial Optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.