A Cooperative Multi-Agent Reinforcement Learning Framework for Resource Balancing in Complex Logistics Network

Xihan Li
Key Laboratory of Machine Perception, Peking University
Beijing, China
xihanli@pku.edu.cn

Jia Zhang
Microsoft Research Asia
Beijing, China
Jia.Zhang@microsoft.com

Jiang Bian
Microsoft Research Asia
Beijing, China
Jiang.Bian@microsoft.com

Yunhai Tong
Key Laboratory of Machine Perception, Peking University
Beijing, China
yhtong@pku.edu.cn

Tie-Yan Liu
Microsoft Research Asia
Beijing, China
Tie-Yan.Liu@microsoft.com

ABSTRACT

Resource balancing within complex transportation networks is one of the most important problems in real logistics domain. Traditional solutions on these problems leverage combinatorial optimization with demand and supply forecasting. However, the high complexity of transportation routes, severe uncertainty of future demand and supply, together with non-convex business constraints make it extremely challenging in the traditional resource management field. In this paper, we propose a novel sophisticated multi-agent reinforcement learning approach to address these challenges. In particular, inspired by the externalities especially the interactions among resource agents, we introduce an innovative cooperative mechanism for state and reward design resulting in more effective and efficient transportation. Extensive experiments on a simulated ocean transportation service demonstrate that our new approach can stimulate cooperation among agents and lead to much better performance. Compared with traditional solutions based on combinatorial optimization, our approach can give rise to a significant improvement in terms of both performance and stability.

KEYWORDS

multi-agent, reinforcement learning, resource balancing, logistics network

ACM Reference Format:
Xihan Li, Jia Zhang, Jiang Bian, Yunhai Tong, and Tie-Yan Liu. 2019. A Cooperative Multi-Agent Reinforcement Learning Framework for Resource Balancing in Complex Logistics Network. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

With the rapid growth of logistics industry, the imbalance between the resource's supply and demand (SnD) has become one of the most important problems in many real logistics scenarios. For example, in the domain of ocean transportation, the SnD of empty containers are very unequal due to the world trade imbalance [21]; in the domain of express delivery, there exists severe emerging unevenness of the SnD of carriers within local areas; in the fast-growing car-sharing and bike-sharing areas, the unbalanced SnD of shared taxis and bikes are also explicit due to various temporal and spatial factors [11, 18]. Henceforth, efficient resource balancing has risen to be the critical approach to solve the resource imbalance in the logistics industry. The failure of that will cause large amounts of unfulfilled resource demand, further resulting in reduction of customer satisfaction, increasing resource shortage cost and declining revenue. Persistent unsolved SnD imbalance can give rise to accumulated resource shortage and, even worse, a stalemate of SnD [18] with unexpected amplified price.

Traditional solutions for resource balancing leverage operational research (OR) based methods [21], which are typically multistage: they first use forecasting techniques to estimate the future SnD of each resource agent; then, the combinatorial optimization approach is employed to find each resource agent's optimal action to minimize a pre-defined objective, which is usually formed as the total cost caused by resource shortage; finally, the feasible execution plan is generated by tailoring the raw solution obtained by OR-based models. Nevertheless, the drastic uncertainty of future SnD, complex business constraints in the non-convex form, as well as the high complexity of transportation networks make it extremely challenging to generate satisfying action plans by using traditional OR solutions.

More concretely, the first crucial challenge, i.e., the uncertainty of future SnD, is mainly caused by multiple external highly dynamic factors, either temporal or spatial, such as special days/events, emerging market changes, unstable policies [21], etc. Moreover, such uncertainty can be even aggravated due to the inherent mutual dependency between the OR-based model and future SnD. Particularly, the future SnD can be dramatically deviated by action plans generated by the OR model, which in turn heavily relies on the future SnD. Henceforth, the uncertainty of future SnD, as drastically increasing the difficulty of accurate SnD forecasting, tends to fail the effectiveness of the traditional multistage OR-based method.

The second major challenge is reflected by many important but complex business rules in real logistics services. On the one hand,

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

they are hard to be formulated in constraints of linear or convex forms, which, therefore, makes it quite hard to model and solve the problem precisely using traditional OR-based method such as linear programming and convex optimization. On the other hand, ignoring these necessary constraints is unacceptable since it will cause a big gap between the model and the real world, leading to significant performance drop and even unfeasible solutions.

Furthermore, since the transportation networks in real logistics services are usually very complex, consisting of various types of terminals and complex connecting routes, the consequential complicated dependencies among terminals rise another vital challenge when building effective OR-based model. Specifically, those complicated dependencies make it quite difficult to create acceptable number of constraints and variables to balance between the individual and the collective objectives in the OR-based model.

To address these challenges, in this paper, we formally formulate the resource balancing problem in complex logistics networks as a *stochastic game* and then propose a novel cooperative multi-agent reinforcement learning (MARL) framework. With the dedicated design of the agent set, joint action space, state set, reward functions, transition probability functions, and discount factor, respectively, our multi-agent reinforcement learning framework provides an end-to-end and high-capability solution, which can not only compensate the imperfect forecasting results to avoid further error propagation in multistage OR methods, but also enable to optimize the obtained action plans towards complicated constraints based on real business rules. Moreover, in contrast to applying MARL under some easier logistics scenarios, a blind employment of reinforcement learning approach may not produce satisfactory results in complex logistics networks, because of its incapability of enhancing cooperation among highly dependent resource agents. To tackle this challenge, we further introduce three levels of cooperative metrics and, accordingly, improve the state and reward design to better promote the cooperation in the complex logistics networks.

To demonstrate the superiority of the MARL framework, we implement our approach under an empty container repositioning (ECR) task in a complex ocean transportation network. In fact, such maritime transportation is essential to the world's economy as 80% of global trade is carried by sea [22]. By far, maritime transportation is the most cost-effective way to move bulk commodity and raw materials around the world. Extensive experiments show that our method can achieve nearly optimal resource balancing results, which yields a significant improvement over the traditional OR baseline.

Our major contributions can be summarized as follows:

- Formulating the resource balancing problem in a complex transportation network as a stochastic game.
- Introducing a cooperative multi-agent reinforcement learning framework as an end-to-end and high-capability solution to the resource balancing problem, as it is not only more robust to the imperfect SnD forecasting but yields higher capability and flexibility compared with the traditional multistage OR-based methods.
- Proposing three levels of cooperative metrics to provide guidance to improve state and reward design, in order to better promote the cooperation in the complex logistics network.
- Conducting extensive experiments on the empty container repositioning task in the scenario of real-world ocean logistics industry.

2 RELATED WORKS

Resource balancing in transportation network, which can be regarded as a branch of scheduling problem, is comprehensively studied in the field of OR [2, 5, 9, 19]. Among them, Epstein et al. [5] studied the ECR problem, and developed a logistics optimization system to manage the imbalance with a multicommodity network flow model based on demand forecasting and safety stock control. For more works about ECR, Song and Dong [21] provides an in-depth review of the OR-based literature.

With the prosperity of deep learning, deep reinforcement learning (RL) methods like DQN [16] has achieved great success in modeling and solving many intellectual challenging problems, such as video games [16] and go [20]. However, they are not widely applied to complicated real-world applications, especially for those who have high-dimensional action spaces and need cooperation between lots of agents.

In recent years, motivated by the great success of deep RL, some methods have been proposed based on RL to address resource balancing problem, especially rebalancing homogeneous, flexible vehicles. Pan et al. [18] proposed a deep reinforcement learning algorithm to tackle the rebalance problem for shared bikes, which learns a pricing strategy to incentivize users to rebalance the system. Lin et al. [11] proposed a contextual multi-agent reinforcement learning framework to tackle the rebalance problem for online ride-sharing platforms, in which every taxi is treated as an agent that learns its action to move to its neighboring grids. Xu et al. [24] proposed a learning and planning approach in on-demand ride-hailing platforms, which combines RL for learning and combinatorial optimizing algorithm for planning. These works have successfully modeled and handled large-scale and real-world traffic scenarios. However, compared with resource balancing in complicate logistics network, the environments in their scenarios are much looser, and the dependency of agents is simple and straightforward. Thus their methods can hardly be applied to solve the resource balancing problem.

To apply MARL in resource balancing, one of the main obstacles is to deal with collaboration of agents with complicated dependency. This dependency is mainly caused by complicated logistics network structures. In the area of traditional multi-agent system, fruitful works are done by dealing with collaboration of multi-agents. Among them, FF-Q [12], Nash-Q [7] and Correlated-Q [6] are famous methods achieving convergence and optimum. However, all of them adopt the joint action approach, which is hardly applied in real-world multi-agent system with lots of agents, due to the extremely large joint action space. Similar limitation occurs in other joint action or best response based methods [8, 23]. Some other works [3, 4, 14] managed to apply potential based reward shaping in MARL to stimulate cooperation. Methods in these works achieve performance improvement in their own scenarios. However, in resource balancing scenarios, where agents' actions have a long-term and immeasurable effect on the ultimate results, more efforts should be put to understand the problem and design rewards.

3 PROBLEM STATEMENT

In this section, we will formally define the resource balancing problem in a complex logistic network.

A typical logistic network can be defined as $G = (P, R, V)$, in which P, R and V stand for the set of terminals, routes, and vehicles, respectively. More specifically,

- Each terminal $P_i \in P$ represents a place that can store resources and generate corresponding SnD. We denote the initial resources in stock at P_i as C_i^0, and we use C_i^t, D_i^t, and S_i^t $(t = 1 \cdots T)$ to represent the numbers of stocks, resource demands, and resource supplies at different time, respectively.
- Each route $R_i \in R$ is a cycle in the logistic network, consisting of a sequence of consecutive terminals $\{P_{i_1}, P_{i_2}, \cdots, P_{i_{|R_i|}}\}$, where $|R_i|$ is the number of stops on R_i and the next destination of $P_{i_{|R_i|}}$ is P_{i_1}. Each route can intersect with others in the network.
- On each route R_i, there is a fixed set of vehicles $V_{R_i} \subseteq V$, each of which, $V_j \in V_{R_i}$, yields an initial position, a duration function $d_j(P_u, P_v) : P \times P \rightarrow N^+$ (mapping from an origin terminal P_u and a destination one P_v into the transit time), a capacity Cap_j^t (the maximum number of resources it can convey). When a vehicle arrives at a terminal, it can either load resources from or discharge its resources to the terminal.

The objective of resource balancing is to minimize the resource shortage among all terminals. At a specific time t, the terminal can only use the stock in the last day, i.e, C_i^{t-1}, to fulfill the current demand D_i^t. [1] Once the stock is not enough, the shortage happens. Thus, we denote the number of shortage as $L_i^t = \max\left(D_i^t - C_i^{t-1}, 0\right)$. Accordingly, the objective of resource balancing is to minimize the total resource shortage: $L = \sum_{P_i \in P, t \in T} L_i^t$.

After the current demand is processed, new resource supplies and those discharged from the vehicle will be added to the stock, thus we can compute the new stock amount as $C_i^t = \max\left(C_i^{t-1} - D_i^t, 0\right) + S_i^t - \sum_{j=1}^{|V|} I(i,j,t)x_j^t$, where $x_j^t \in N$ denotes the number of resources loaded onto vehicle V_j at time t. x_j^t can be negative to denote the discharged amount of resources from the vehicle, and $I(i,j,t)$ is a indicator variable defined as

$$I(i,j,t) = \begin{cases} 1, & V_j \text{ arrives at } P_i \text{ at time slot } t \\ 0, & \text{otherwise.} \end{cases}$$

We further define $C_{V,j}^t$ as the amount of resources on vehicle V_j at time slot t, and clearly, $C_{V,j}^t = C_{V,j}^{t-1} + x_j^t$.

4 COOPERATIVE MULTI-AGENT REINFORCEMENT LEARNING FRAMEWORK

As aforementioned, traditional solutions for resource balancing employ combinatorial optimization with SnD forecasting. However, it suffers from failures in front of uncertainty of SnD, complex business constraints, and high complexity of transportation networks. To address these challenges, in this section, we first model the resource balancing in complex logistic network as a stochastic game and then propose a novel cooperative multi-agent reinforcement learning (MARL) framework to solve it.

4.1 Resource Balancing as a Stochastic Game

The resource balancing problem can be formally modeled as a stochastic game $\mathcal{G} = (N, \mathcal{A}, \mathcal{S}, \mathcal{R}, \mathcal{P}, \gamma)$, where N is the agent set, $\mathcal{A}$ is the joint action space, $\mathcal{S}$ is the state set, $\mathcal{R}$ is the reward function, $\mathcal{P}$ is the transition probability function, and γ is the discount factor. More formally definitions are shown below:

Agent set N. We define each vehicle as an agent, which yields two major advantages: (1) As each vehicle agent continuously sails circularly along the certain route, it can be aware of the larger scope of information within the whole route such that optimizing towards maximizing its own reward, i.e., minimizing the shortage, can benefit the total reward of the entire route. (2) Since multiple vehicle agents navigating along the same route usually share the similar environment, it is natural for them to share the same policy so as to significantly reduce the model complexity in MARL and boost the learning process.

Joint action space $\mathcal{A}$. We define the action of a vehicle agent V_j as loading or discharging resources when it arrives at a terminal P_i. Similar to Menda et al. [15], we apply the idea of event-driven reinforcement learning. To be more concrete, we treat agents' each arrival at a terminal as a trigger event, and an agent only needs to take action once a trigger event happens. Under this event-driven setting, we use a_j^t to denote the action taken by agent $N_j \in N$ at t-th arrival event. For agent N_j, we define its action space as $A_j = [-1, 1]$, where $a_j^t \in [-1, 0)$ means discharging a portion of a_j^t resources from the vehicle, $a_j^t \in (0, 1]$ means loading a portion of a_j^t resources onto the vehicle, and $a_j^t = 0$ means no loading or discharging. Then, the joint action space is $\mathcal{A} = A_1 \times A_2 \times \cdots \times A_{|N|}$, where $|N|$ is the number of agents. The total amount of resources that can be discharged or loaded at t is usually restrictively determined by the dynamic values of C_i^t, Cap_i^t, $C_{V,j}^t$ as well as some other external factors, which are controlled by domain-specific business logics.

State set $\mathcal{S}$. The state $\mathcal{S}$ is a finite set that stands for all possible situations of the *whole* logistics network. Note that, from a practical point of view, it is not necessary for the agents to take action based on the whole state information, due to the extremely large state space and the potential noise introduced by unrelated information. We will elaborate more on the practical state design later in this section.

Rewards function $\mathcal{R}$. The objective of the resource balancing problem is to minimize the accumulated shortage for all terminals. With respect to each individual action, i.e., loading or discharging some resources at a terminal, the impact can be spread to its follow-up periods. To model such delayed reward, it usually leverages rewards shaping to guide the learning process [17], a typical specification of which is to measure the difference of the ultimate accumulated shortage between with and without this action. However, this reward is very hard to compute in practice. Thus, we find

[1] This is because new supplies and discharged resources at time t are usually unavailable temporarily for realistic reasons, such as inner terminal transportation and maintenance. This logic can change with specific application scenarios, and will not affect our framework.

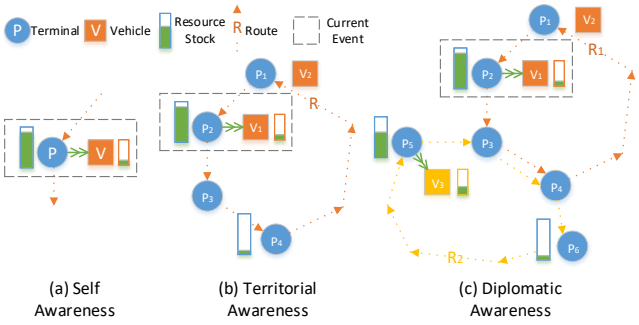


Figure 1: Illustration of three levels of cooperative metrics. (a) Self awareness agent V only consider information of (P, V) to make decision. (b) Territorial agent V_1 will make decision based on information within its territory. It could load more resources at arrival port P_2 with the awareness that port P_4 on its route R has low stock. (c) Agent V_1 with diplomatic awareness can look far beyond its route. It could load more resources at current port P_2 and discharge them at transshipment port P_3 or P_4 later with the awareness that port P_6 on its neighboring route R_2 needs support.

other more realistic rewards shaping methods, which will also be discussed later in this section.

Transition probability function $\mathcal{P}$. It is defined as a mapping $\mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, which can be specified by the definition of S, R, V and the distribution behind SnD within particular logistics networks.

4.2 Cooperative Metrics for State and Reward Design

After formulating the resource balancing problem as a stochastic game, applying MARL approach to the real world, however, requires a dedicated design on the game state and the action's reward to promote cooperation and improve performance. Based on the scope of agents' awareness of cooperation, we identify three levels of cooperative metrics: *self awareness*, *territorial awareness*, and *diplomatic awareness*. In general, agents with self awareness are fully selfish and shortsighted and only consider immediate information and interests; agents with territorial awareness have a broader vision and make decision based on information belonging to their territories, i.e., routes in this problem. At last, agents with diplomatic awareness even overlook beyond their own routes and conduct resource balancing, in a diplomatic way, by cooperating with intersecting routes so that resources can flow from *fertile* routes to *barren* routes.

4.2.1 Self Awareness. When agent V_j arrives at terminal P_i, it is natural that V_j makes decisions just based on the information of itself and P_i. Regarding the reward of this action, a straightforward metric is to consider whether shortages will happen before next vehicle's arrival at P_i. Obviously, this is a very shortsighted agent.

Suppose the time of k-th arrival event of a vehicle agent V_j is t_k and the arrival terminal is P_i. The state $s_{P,i}^{t_k}$ for terminal P_i can be formed up by:

- Current available resources $C_i^{t_k}$.
- Historical information of available resources $\phi\left(C_i^1, \cdots, C_i^{t_k-1}\right)$ and shortages $\psi\left(L_i^1, \cdots, L_i^{t_k-1}\right)$.
- Other domain-specific information, such as terminal ID, berth length, etc.

where $\phi(\cdot)$ and $\psi(\cdot)$ denote some statistical function (Mean, Median, etc.) or more advanced sequential data processing models (CNN, RNN, etc.). Specific implementation should depend on the application scenario.

State $s_{V,j}^{t_k}$ for vehicle V_j can be comprised of:

- Current available resources onboard $C_j^{t_k}$.
- Available space $Cap_j^{t_k} - C_j^{t_k}$.
- Other domain-specific information, such as vehicle ID, vehicle type, etc.

Concatenating the above information, we get the state $s_I = \left[s_{P,i}^{t_k}, s_{V,j}^{t_k}\right]$ for self awareness agents. The self awareness agents only concerns if shortage happens between t_k and t_k' where $t_k' \geq t_k$ stands for the time of next vehicle's arrival at P_i. Besides, inspired by the idea of safety stock in traditional methods, we add a small positive reward if no shortage happens. This reward is calculated according to a function $f : N \to R$ that has diminishing marginal gain[2]. The purpose is to encourage the agents to put some safety stock with upper limit on terminals. In summary, the reward can be written as follows:

$$r_I = f\left(C_i^{t_k'}\right) - g\left(\sum_{t=t_k}^{t_k'} L_i^t\right), \tag{1}$$

where $g : N \to R$ is the loss defined on the total shortage.

4.2.2 Territorial Awareness. According to the problem definition, a vehicle needs to navigate along with the certain route and is obliged to balance the SnD within its own territory, i.e., the terminals in its route. Apparently, each agent with self awareness, with no consideration on other terminals and vehicles in its route, cannot balance the resources SnD within its route. Thus, we introduce territorial awareness agent to minimize the total shortage of all terminals in the route. Specifically, for an agent V_j on route R_q, we hope the agent to get the accurate information of neighboring environment on the route, which is more likely to influence the current decision. We add extra successive information as follows:

- Information about n successive terminals $\{s_{T,i'}^{t_k} | P_{i'} \in \text{Sc}_{i,j}(n)\}$ where $\text{Sc}_{i,j}(n)$ is the set of n terminals to which vehicle V_j will travel after terminal P_i.
- Information about m future vehicles $\{s_{V,j'}^{t_k} | V_{j'} \in \text{Fu}_{i,j}(m)\}$ where $\text{Fu}_{i,j}(m)$ stands for the set of m vehicles that will arrive at P_i just after V_j's arrival.

As we can see, the larger n and m are, the more information can be used for decision. However, in practice, we usually set small values for n and m to control the model complexity and noise introduced by unimportant information. To compensate the potential information loss, we introduce the overall statistical territory information $s_{R,q}^{t_k}$ for route R_q:

- Information of available resources in all the terminals in the route $\Phi\left(\left\{C_i^{t_k} | P_i \in R_q\right\}\right)$

[2]For example, $f(x) = \sum_{i=0}^{x} \beta^i$ for $0 < \beta < 1$.

- Information of shortage in all the terminals in the route $\Psi\left(\left\{\psi\left(L_i^1, \cdots, L_i^{t_k-1}\right) | P_i \in R_q\right\}\right)$

Similar as $\phi(\cdot)$ and $\psi(\cdot)$, $\Phi(\cdot)$ and $\Psi(\cdot)$ are statistical functions or models based on series data.

We concatenate all information above with s_I to get the territorial state s_T. Territorial awareness agents will make decision based on the state s_T.

4.2.3 Diplomatic Awareness. In real logistics networks, imbalance can also happen among different routes: there may be a large amount of supplies but very few demands on some routes, while some other routes may be opposite, with a large amount of demands that cannot be satisfied with limited supplies. In this case, it is infructuous to attempt balancing SnD within the territory of single route. To solve this problem substantially, agents should learn the diplomacy: solving imbalance collaboratively with agents in intersecting routes.

To this end, more information about neighboring routes should be considered. Assume an event (P_i, V_j, R_q), and denote CR_q as the crossing routes having common terminal(s) with route R_q. First, statistic information for all neighboring routes $\Phi_r\left(\left\{s_{R,p}^{t_k} | R_p \in \mathrm{CR}_q\right\}\right)$ should be involved to represent the general status of crossing routes. Moreover, we add additional information when agents arrive at transfer terminals, that is $\Phi_n\left(\left\{s_{R,p}^{t_k} | R_p \in \mathrm{RT}_i\right\}\right)$ where RT_i is the set of routes that pass through terminal P_i. We concatenate all information above with s_T as the diplomatic state s_D.

To encourage cooperation, we extend the reward by considering cross routes shortage. For an agent V_j on a route R_q, its action not only influences the reward on its own route, but also influences the reward of agents in the neighboring routes in CR_q, especially on the transfer terminals where routes are intersecting. To take neighboring routes into consideration, we use $r_D = \alpha r_I + (1-\alpha) r_C$, where α is a soft hyper-parameter and

$$r_C = f\left(\xi_1\left(\left\{C_i^{t'_k} | P_i \in R_p, R_p \in \mathrm{CR}_q\right\}\right)\right) - g\left(\xi_2\left(\left\{L_i^t | t_k \le t \le t'_k, P_i \in R_p, R_p \in \mathrm{CR}_q\right\}\right)\right),$$

for statistical functions or advanced models $\xi_1(\cdot)$ and $\xi_2(\cdot)$.

The three levels of cooperative metrics are illustrated in Figure 1. The whole cooperative MARL framework for resource balancing is shown in Algorithm 1. From Line 4 to 13, the agents interact with environment by function calls, and collect transition experiences. It should be emphasized that GetState $\left(S_{j,k}, P_i, V_j\right)$ refers to the process of constructing state based on current event (P_i, V_j) and global environment snapshot $S_{j,k}$. This snapshot contains complete information of the environment when the event is triggered. GetDelayedReward $\left(S_{j,k-1}, S_{j,k}\right)$ refers to the process to calculate the delayed reward based on shortage happens between these two snapshots. The detail implementation of GetState$(\cdot)$ and GetDelayedReward$(\cdot)$ will be determined based on the adopted level of cooperative metric.

Algorithm 1 Cooperative MARL Framework

1: Initialize replay memory D_j to capacity M for each agent V_j
2: Initialize action-value function Q_j with random weights θ_j for each agent V_j
3: **for** episode ← 1 to MAX **do**
4: ResetEnvironment()
5: **while** environment is not terminated **do**
6: *// k means the k-th event of agent V_j*
7: $(P_i, V_j, k) \leftarrow$ WaitingEvent()
8: $S_{j,k} \leftarrow$ GetEnvironmentSnapshot()
9: $s_k \leftarrow$ GetState $\left(S_{j,k}, P_i, V_j\right)$
10: $r_{k-1} \leftarrow$ GetDelayedReward $\left(S_{j,k-1}, S_{j,k}\right)$
11: StoreExperience$\left(D_j, (s_{k-1}, a_{k-1}, r_{k-1}, s_k)\right)$
12: $a_k \leftarrow \epsilon$-Greedy$\left(\arg\max_a Q_j(s_k, a)\right)$
13: Execute(P_i, V_j, a_k)
14: **end while**
15: **for** $l \leftarrow 1$ to MAX-TRAIN **do**
16: **for** each V_j in V **do**
17: Sample a batch of data (s, a, r, s') from D_j
18: Compute target $y \leftarrow r + \gamma \max_{a'} Q_j(s', a'; \theta_j)$
19: Update Q-network for agent V_j as $\theta_j \leftarrow \theta_j - \nabla_{\theta_j}(y - Q_j(s, a; \theta_j))^2$
20: **end for**
21: **end for**
22: **end for**

5 EXPERIMENTS

To evaluate the effectiveness of our proposed approach, we conduct experiments on resource balancing in the scenario of ocean container transportation. In this task, the resource balancing mainly corresponds to Empty Container Repositioning (ECR). In the following of this section, we will first introduce the background of ECR, then we will show the experimental results on a part of real ocean logistics network.

5.1 The ECR Problem

As containers are the most important asset in ocean logistics industry, the resource balancing in this scenario corresponds to ECR, which is quite necessary since the SnD of empty containers are very unequal due to the world trade imbalance [21]. In particular, the goal of ECR is to reposition empty containers by container vessels sailing on pre-determined routes within ocean logistics networks to fulfill the dynamic transportation demand of ports. According to Asariotis et al. [1], the estimated cost of seaborne empty container repositioning was about 20 billion dollars in 2009, with 50 million empty containers movement, which has demonstrated the necessity to optimize ECR in ocean logistics industry. More formally, ports, container vessels, and predetermined routes for vessels correspond to terminals P, vehicles V, and routes R, respectively. External demands and supplies of empty containers for port P_i at time slot t correspond to D_i^t and S_i^t, respectively.

Nonetheless, there are several domain-specific feature for the ECR problem. In ECR problem, the external demands and supplies D_i^t and S_i^t are determined by transportation orders O, which are also external and dynamic. An order $o \in O$ is a tuple (P_u, P_v, n, t_o), which denotes departure port, destination port, amount of needed containers and order time. The *container transportation chain* for orders can be described as follows, also illustrated in Figure 2: when

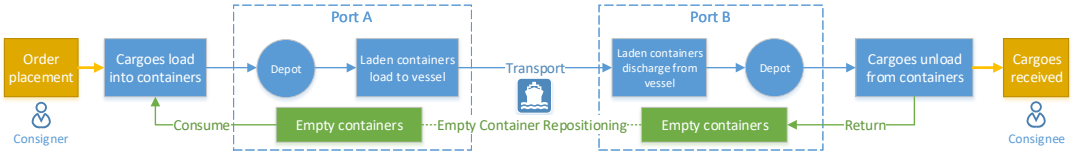


Figure 2: The container transportation chain in ECR problem. Blue lines indicate laden container flows and green lines indicate empty container flows. All flows are under the control of specific business logics in real logistics scenarios.

an order (P_u, P_v, n, t_o) is placed at time slot t_o, the external demand of departure port $D_u^{t_o}$ will be added by n, which means P_u need to provide n empty containers to fulfill the order at time slot t_o. If the order is fulfilled, cargoes will be loaded into these empty containers, and they are transformed to laden containers waiting for vessels to transport them to destination port P_v. Laden containers will be lifted on the arriving vessel V_j on route R_k if $P_v \in R_k$.[3] When the laden containers are discharged to the destination port P_v at time slot t'_o, the cargoes in laden containers will be unloaded and these containers will be returned to P_v as empty containers at time slot $t'_o + t_{\text{ret}}$, in which t_{ret} is a constant. Therefore, the external supply of destination port $S_v^{t'_o+t_{\text{ret}}}$ will be added by n. To summarize, the specification of ECR problem is concluded as follows:

- Empty containers are reusable, which will circulate between ports as receptacles for cargoes;
- Laden containers and empty containers share the same vessel. i.e., the space for empty containers Cap_j^t for vessel V_j will change dynamically depending on the amount of laden containers on the vessel;
- The whole order will fail if not enough empty containers can be served from departure port when the order is placed. The resource shortage L_o for a single order o is defined as $L_o = n$, when $n > C_u^{t_o}$, and $L_o = 0$ for otherwise.

5.2 Experimental Setting

In the following experiments, we extract a main ocean transportation network among Asia, North America and Europe based on the real world service loops of a commercial company. This network consists of 4 route, 17 ports and 31 vessels. The routes are listed as follows and illustrated in Figure 3:

- R1: Pacific Atlantic route, 94 days with 14 vessels.
- R2: Central Asia to Southeast Asia route, 60 days with 9 vessels.
- R3: Japan to America route, 33 days with 5 vessels.
- R4: Japan-China-Singapore route, 19 days with 3 vessels.

The vessels are uniformly distributed with a interval around one week in their routes. Initially, there are 3000 empty containers distributed in the 17 ports based on historical statistic from a commercial ocean logistics company, and all vessels are empty without any laden or empty containers. The distribution of SnD of all 17 ports in the simulated environment is shown in Figure 4 based on information provided by the same company. Every vessel has a capacity of 200 containers. i.e., the total amount of laden and empty containers cannot exceed 200 for every vessel. To assist the

[3]Without loss of generality, we only deal with non-transshipment order, that is we suppose P_u and P_v are always within one route. A transshipment order can be viewed as multiple separated non-transshipment orders.

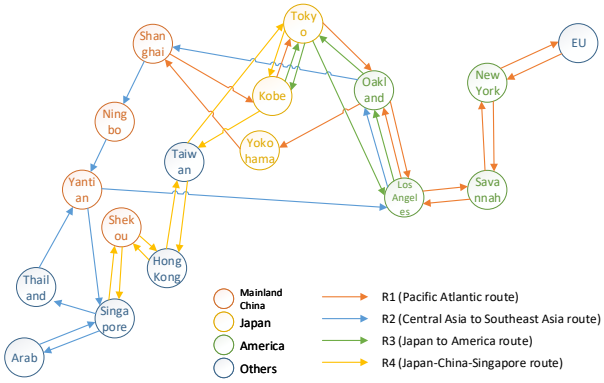


Figure 3: The extracted ocean transportation network among Asia, North America and Europe.

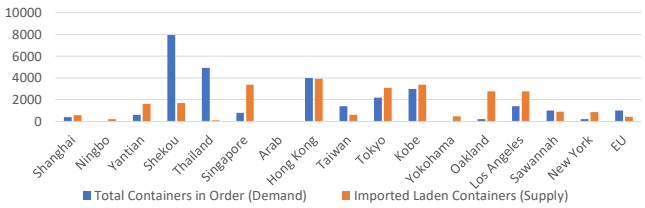


Figure 4: The distribution of demand and supply of all 17 ports in the environment.

training of our cooperative MARL approach, we build a simulated ECR environment based on real historical data from the commercial ocean logistics company.

To measure the performance of our approach, we use the metric of *fulfillment ratio*, which is defined as the ratio of total successfully fulfilled containers compared to all containers requested in one episode (400 time steps, where one time step corresponds to one day). In real-world, there are many other types of cost for container repositioning, including loading/discharging cost, storage cost, etc. Among all of them, however, the cost of shortage, measured by the fulfillment ratio, is the dominant one since it will directly affect the booking acceptance and consequently the transportation company's reputation. Therefore, we focus on minimizing the cost of shortage in this work. Indeed, other types of cost can also be naturally captured by MARL through rewards shaping and specific action space design, which will be one of our future targets.

5.3 Compared Methods

In the following experiments, we compare the following methods on the ECR problem:

- **No Reposition**: Empty containers are never repositioned. The flow of containers will only depends on the laden container transportation.
- **Rule-Based Inventory Control (IC)**. With the idea in inventory management theory, this method sets two inventory thresholds, safety threshold F_i^s and excess threshold F_i^e ($F_i^s \leq F_i^e$), for each port P_i based on the historical information of SnD respectively. When a vessel V_j arrives at P_i at time slot t, it will try to maintain the stock C_i^t located in the range $\left[F_i^s, F_i^e\right]$ by loading or discharging containers. Formally, suppose $x_{i,j}^t$ is the number containers loading from P_i (negative value means discharging to this port), it satisfies

$$x_{i,j}^t = \begin{cases} \min(C_i^t - F_i^e, Cap_j^t - C_{V,j}^t, C_i^t), & C_i^t > F_i^e, \\ -\min(F_i^s - C_i^t, C_{V,j}^t), & C_i^t < F_i^s, \\ 0, & \text{otherwise}. \end{cases}$$

- **Online Linear Programming (LP)**. With some approximation approaches, ECR problem can be modeled in linear programming (LP) by adopting the mathematical definitions in problem statement section. However, it is hard to apply the solution directly due to the gap caused by simplified model. Here, we apply rolling horizon policy described in Long et al. [13] to solve the problem: empty reposition plan are generated for a long period on the planning horizon based on LP model with forecasting information for this period, but only partial planning at the beginning are executed. Repeat this procedure until termination. This is the so called online LP method. Note that, our proposed end-to-end MARL method directly interacts with the simulator with no explicit forecasting stage, therefore, for the purpose of appropriate comparison, we use exact future order information to replace the forecasted future demand in the LP model so as to eliminate the effects of external factors leading to uncertain forecasts, which can be seen as a relatively ideal condition. More details about the online LP can be found in the appendix of the full version on arXiv [10].
- **Online LP with Inventory Control**. In this baseline, we adopt the idea from Epstein et al. [5] which combines LP model with inventory control. This method sets a safety threshold F_i^s for each port P_i based on the historical information of SnD, and then constrains $L_i^t = \max\left(D_i^t - (C_i^{t-1} - F_i^s), 0\right)$.
- **Self Awareness MARL (SA-MARL)**. This is the MARL model described in the previous section with self awareness agents. For terminal (port) state $s_{P,i}^{t_k}$, $\phi(\cdot)$ is an average function while $\psi(\cdot)$ is a sum function. For vehicle (vessel) state $s_{V,i}^{t_k}$, we add amount of laden containers onboard as additional domain-specific information. As for reward, we set $f(x) = 1 - 0.5^x$ and $g(y) = 5y$, where x and y are calculated as in Equation (1).
- **Territorial Awareness MARL (TA-MARL)**. This is the MARL model with territorial awareness agents. For successive terminal information, both m and n are set to 1. $\Phi(\cdot)$ and $\Psi(\cdot)$ in $s_{R,q}^{t_k}$ are set to be average functions.
- **Diplomatic Awareness MARL (DA-MARL)**. This is the MARL model described in previous session with diplomatic awareness agents. $\Phi_r(\cdot)$ and $\Phi_n(\cdot)$ are set to be average functions with $\alpha = 0.5$. Both $\xi_1(\cdot)$ and $\xi_2(\cdot)$ are 2-layer average functions $\textsc{Avg}\{\textsc{Avg}\{\sum_{t=t_k}^{t'_k} L_i^t | P_i \in R_p\} | R_p \in \textsc{Cr}_q\}$.
- **Offline Optimal LP (Upper Bound)**. In this case, the shortage will be directly calculated as objective by LP model mentioned above, which has the knowledge of all orders in advance, without implementation in simulated environment. This can be seen as an upper bound for the problem. i.e., it is not likely for any methods to achieve better performance than this.

Table 1: Performance comparison with different baselines.

Method	Fulfillment Ratio (%)		
	80% Container	100% Container	150% Container
No Reposition	26.58 ± 0.90	29.87 ± 0.85	38.25 ± 1.07
IC	58.30 ± 0.93	61.07 ± 0.98	68.63 ± 0.98
Online LP	76.28 ± 1.54	85.75 ± 1.34	94.48 ± 1.00
Online LP with IC	81.09 ± 1.21	88.99 ± 0.89	96.30 ± 0.80
SA-MARL	65.39 ± 1.20	72.04 ± 1.57	84.21 ± 1.45
TA-MARL	75.25 ± 1.38	83.48 ± 0.94	93.75 ± 0.69
DA-MARL	**82.04 ± 1.69**	**95.97 ± 0.63**	**97.70 ± 0.98**
Offline LP (Upper Bound)	98.32 ± 0.60	98.95 ± 0.31	99.42 ± 0.25

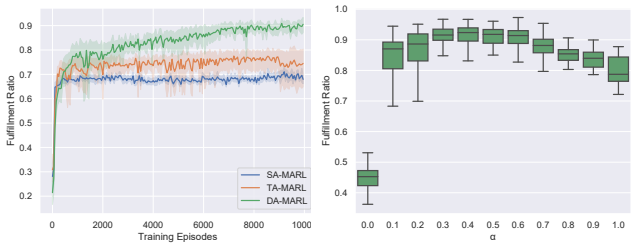
Table 2: Performance comparison with different delay parameter k in DA-MARL

k	Fulfillment Ratio	k	Fulfillment Ratio
1	95.87 ± 0.65	20	94.52 ± 0.89
5	95.76 ± 0.67	30	93.23 ± 1.76
10	95.49 ± 0.65	40	90.39 ± 2.50
15	94.71 ± 0.93	50	85.87 ± 3.23

All MARL methods are trained 10000 episodes with ϵ-greedy exploration. The ϵ is annealed linearly from 0.5 to 0.01 across the first 8000 episodes, and fixed at 0.01 in the rest episodes. We use *Adam Optimizer* with a learning rate of 10^{-4}. Batch size is fixed to 32. All agents in the same route share the same Q-network, and each Q-network is parameterized by a 2-layer MLP with node size of 16 and 16, activated by ReLU. Since DQN works on discrete action space, we discretize the continuous action space $A_i = [-1, 1]$ uniformly by 21 actions, that is $A'_i = \{-1, -0.9, \cdots, 0.9, 1\}$.

5.4 Results Analysis

To compare all the methods aforementioned, we run our trained models and baseline methods in 100 randomly initialized environments. For baseline methods, we run grid search to find suitable parameters. To test the robustness of the learned policy in our framework, we also evaluate the model trained under 100% (3000) empty containers setting by changing the total amount of containers to 80% (2400 containers) and 150% (4500 containers). The results are summarized in Table 1, in which we report the mean and standard deviations of the fulfillment ratios. As we can see, DA-MARL method achieves the best performance in all initial container settings. Even TA-MARL method is comparable with traditional online LP method. The SA-MARL achieves the poorest performance among our MARL methods, while it is still better than rule-based



(a) (b)

Figure 5: (a) Convergence comparison of MARL methods. The X-axis is number of episodes. (b) Performance comparison with different α in Diplomatic Awareness MARL. The X-axis is α. The Y-axis is fulfillment ratio in both figures.

inventory control. The testing of robustness shows that agents have learned efficient policies to deal with dramatic environment changes. The trained DA-MARL model still performs better than the online LP and its IC version, which in fact are built on changed environments.

The convergence comparison of MARL methods are shown in Figure 5a. Each MARL method is trained for 10 times, and we report the mean and standard deviation of performance during training. As we can see, all MARL methods converge very quickly at first 1000 episodes. After that, DA-MARL will get a much larger improvement than the others.

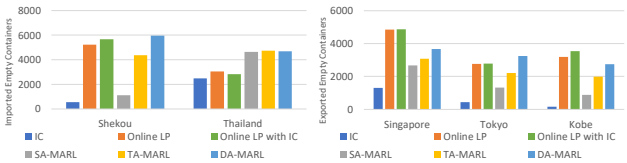
In Diplomatic Awareness MARL, α is an important parameter to control the proportion between territorial reward and diplomatic reward. We train the model with different α and the results are shown in Figure 5b. Every model is trained for 5 times due to time limitation, and every trained model is tested for 100 times. The result shows that neither r_I alone ($\alpha = 1$) nor r_C alone ($\alpha = 0$) performs well alone, and a combination ($\alpha = 0.4$ in our case) of them is essential to achieve better performance.

Communication is a crucial part to build up cooperation in MAS, and in our Diplomatic Awareness MARL design, shared information $\Phi_r(\cdot)$ and $\Phi_n(\cdot)$ about neighboring routes and transshipment routes is required to achieve high performance. However, it is possible that these information cannot be transferred in real-time in realistic scenario, i.e., agents can only have access to an outdated version of these information. Table 2 shows the fulfillment ratio when all agents can only access these information of k days ago. The result shows that our proposed method performs robustly without significant loss when the delay is in a reasonable range, i.e., $k \leq 20$.

5.5 Cooperation Ability Analysis

The major objective of ECR is to balance the SnD so that the shortage costs of deficit ports are minimized. Figure 6a shows the amount of imported empty containers of Shekou and Thailand, two major ports that are deficient of empty containers, by different methods. From Figure 3, Thailand is the next ports of a surplus port Singapore on route R2, which means it is not hard to obtain empty containers without complicated cooperative mechanism. For Shekou, the sitiation is much more severe as it need more containers than Thailand (shown in Figure 4) while the only supply port, Singapore, in route R4 doesn't have enough containers to supply Shekou. The only way that demand of Shekou can be sufficiently fulfilled is to use Tokyo and Kobe as transshipment ports to transport empty containers from America regions, which requires strong ability of cooperation between regions. Figure 6a shows that all the three MARL methods performs well on Thailand, while Diplomatic Awareness MARL outperforms all other methods on Shekou, indicating that our design is capable to fulfill the demand that requires inter-route cooperation.

For inter-route cooperation, the amount of exported empty containers at transshipment port is essential, since it is the source from which deficient ports such as Shekou obtain empty containers. Figure 6b shows the amount of exported empty containers of Singapore, Tokyo and Kobe, which are three major transshipment ports between different routes in our setting. It shows that the amount of exported empty containers at transshipment ports significantly increases with more cooperative awareness of MARL agent, which indicates that our cooperative design is effective. Online LP method with its IC version can also perform well on transshipment ports since they are globally optimized. However, the gap between LP models and environment confines their overall performance.



(a) (b)

Figure 6: (a) Imported empty containers of Shekou and Thailand, two major ports that are deficient of empty containers, by different methods. (b) Exported empty containers of Singapore, Tokyo and Kobe, three major transshipment ports between different routes, by different methods. "No Reposition" method is omitted since it won't import or export any empty containers.

6 CONCLUSION

In this paper, we first formulate the resource balancing problem in logistics networks as a stochastic game. Given this setting, we propose a cooperative multi-agent reinforcement learning framework, in which three levels of cooperative metrics are identified based on the scope of agents' awareness of cooperation, which promote efficient and cost-effective transportation. Extensive experiments on a simulated ocean transportation service demonstrate that our new approach can stimulate the cooperation among agents and give rise to a significant improvement in terms of both performance and stability. In future, we will integrate more types of cost, such as transport cost and inventory cost in real logistic scenarios, into a unified objective to optimize. Moreover, we will investigate more advanced RL techniques to achieve a more precise control of actions.

ACKNOWLEDGEMENT

We sincerely appreciate Ryan Ho, Johnson Lui, Karab Sze, Jeffrey Ko, Simon Choi, Tony Y Li, Apple Ng, Terry Tam and Wyatt Lei from Orient Overseas Container Line for their great support on this work.

REFERENCES

[1] Regina Asariotis, Hassiba Benamara, Hannes Finkenbrink, Jan Hoffmann, Jennifer Lavelle, Maria Misovicova, Vincent Valentine, and Frida Youssef. 2011. *Review of Maritime Transport, 2011.* Technical Report.

[2] Teodor Gabriel Crainic and Gilbert Laporte. 1997. Planning models for freight transportation. *European journal of operational research* 97, 3 (1997), 409–438.

[3] Sam Devlin and Daniel Kudenko. 2011. Theoretical Considerations of Potential-based Reward Shaping for Multi-agent Systems. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 1 (AAMAS '11)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 225–232. http://dl.acm.org/citation.cfm?id=2030470.2030503

[4] Sam Devlin, Logan Yliniemi, Daniel Kudenko, and Kagan Tumer. 2014. Potential-based Difference Rewards for Multiagent Reinforcement Learning. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems (AAMAS '14)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 165–172. http://dl.acm.org/citation.cfm?id=2615731.2615761

[5] Rafael Epstein, Andres Neely, Andres Weintraub, Fernando Valenzuela, Sergio Hurtado, Guillermo Gonzalez, Alex Beiza, Mauricio Naveas, Florencio Infante, Fernando Alarcon, Gustavo Angulo, Cristian Berner, Jaime Catalan, Cristian Gonzalez, and Daniel Yung. 2012. A Strategic Empty Container Logistics Optimization in a Major Shipping Company. *Interfaces* 42, 1 (Feb. 2012), 5–16. https://doi.org/10.1287/inte.1110.0611

[6] Amy Greenwald and Keith Hall. 2003. Correlated-Q Learning. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML'03)*. AAAI Press, 242–249. http://dl.acm.org/citation.cfm?id=3041838.3041869

[7] Junling Hu and Michael P. Wellman. 2003. Nash Q-learning for General-sum Stochastic Games. *J. Mach. Learn. Res.* 4 (Dec. 2003), 1039–1069. http://dl.acm.org/citation.cfm?id=945365.964288

[8] Marc Lanctot, Vinicius Zambaldi, Audrūnas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. 2017. A Unified Game-theoretic Approach to Multiagent Reinforcement Learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., USA, 4193–4206. http://dl.acm.org/citation.cfm?id=3294996.3295174

[9] Jing-An Li, Stephen CH Leung, Yue Wu, and Ke Liu. 2007. Allocation of empty containers between multi-ports. *European Journal of Operational Research* 182, 1 (2007), 400–412.

[10] Xihan Li, Jia Zhang, Jiang Bian, Yunhai Tong, and Tie-Yan Liu. 2019. A Cooperative Multi-Agent Reinforcement Learning Framework for Resource Balancing in Complex Logistics Network. *arXiv preprint arXiv:1903.00714* (2019).

[11] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 2018. Efficient Large-Scale Fleet Management via Multi-Agent Deep Reinforcement Learning. ACM Press, 1774–1783. https://doi.org/10.1145/3219819.3219993

[12] Michael L. Littman. 2001. Friend-or-Foe Q-learning in General-Sum Games. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 322–328. http://dl.acm.org/citation.cfm?id=645530.655661

[13] Yin Long, Loo Hay Lee, and Ek Peng Chew. 2012. The sample average approximation method for empty container repositioning with uncertainties. *European Journal of Operational Research* 222, 1 (Oct. 2012), 65–75. https://doi.org/10.1016/j.ejor.2012.04.018

[14] Patrick Mannion, Jim Duggan, and Enda Howley. 2016. Generating multi-agent potential functions using counterfactual estimates. *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)* (2016).

[15] Kunal Menda, Yi-Chun Chen, Justin Grana, James W. Bono, Brendan D. Tracey, Mykel J. Kochenderfer, and David Wolpert. 2017. Deep Reinforcement Learning for Event-Driven Multi-Agent Decision Processes. *arXiv:1709.06656 [cs]* (Sept. 2017). http://arxiv.org/abs/1709.06656

[16] 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, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (Feb. 2015), 529–533. https://doi.org/10.1038/nature14236

[17] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. 1999. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML '99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 278–287. http://dl.acm.org/citation.cfm?id=645528.657613

[18] Ling Pan, Qingpeng Cai, Zhixuan Fang, Pingzhong Tang, and Longbo Huang. 2018. Rebalancing Dockless Bike Sharing Systems. *arXiv:1802.04592 [cs]* (Feb. 2018). http://arxiv.org/abs/1802.04592 arXiv: 1802.04592.

[19] Warren B Powell. 1996. Toward a Unified Modeling Framework for Real-Time Logistics Control. *Military Operations Research* (1996), 69–79.

[20] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 7587 (Jan. 2016), 484–489. https://doi.org/10.1038/nature16961

[21] Dong-Ping Song and Jing-Xin Dong. 2015. Empty Container Repositioning. In *Handbook of Ocean Container Transport Logistics*. Springer, Cham, 163–208. https://doi.org/10.1007/978-3-319-11891-8_6

[22] UNCTAD. 2017. *Review of maritime transport 2017.* OCLC: 1022725798.

[23] Xiaofeng Wang and Tuomas Sandholm. 2002. Reinforcement Learning to Play an Optimal Nash Equilibrium in Team Markov Games. In *Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS'02)*. MIT Press, Cambridge, MA, USA, 1603–1610. http://dl.acm.org/citation.cfm?id=2968618.2968817

[24] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. 2018. Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. ACM, New York, NY, USA, 905–913. https://doi.org/10.1145/3219819.3219824