

# Hierarchical Reinforcement Learning for Optimal Agent Grouping in Cooperative Systems

Liyuan Hu

Department of Statistics, London School of Economics  
l.hu11@lse.ac.uk

## Abstract

This paper presents a hierarchical reinforcement learning (RL) approach to address the agent grouping or pairing problem in cooperative multi-agent systems. The goal is to simultaneously learn the optimal grouping and agent policy. By employing a hierarchical RL framework, we distinguish between high-level decisions of grouping and low-level agents' actions. Our approach utilizes the CTDE (Centralized Training with Decentralized Execution) paradigm, ensuring efficient learning and scalable execution. We incorporate permutation-invariant neural networks to handle the homogeneity and cooperation among agents, enabling effective coordination. The option-critic algorithm is adapted to manage the hierarchical decision-making process, allowing for dynamic and optimal policy adjustments.

## 1 Introduction

Sequential grouping or pairing problems are ubiquitous in many domains, from resource allocation and scheduling to team formation and matchmaking. These problems are challenging due to their inherent combinatorial complexity and the need for efficient decision-making under constraints.

In studies of natural [Jeanson et al., 2005, Wittemyer and Getz, 2007] and multi-agent systems [Phan et al., 2021], grouping has been recognized as a means to enhance efficient collaboration. However, devising a general criterion for grouping agents without specific domain knowledge remains a significant challenge for the research community. Previous works often focus on well-structured tasks with predefined roles or task decompositions [Cossentino et al., 2014, Lhaksmana et al., 2018, Phan et al., 2021, Jiang et al., 2021], which may not be practical in real-world settings and can restrict the transferability of these methods. Existing approaches to grouping and pairing policies in RL have limitations. For example, user pairing in the NOMO system [Wang et al., 2021, Lee and So, 2020] suffers from exponential growth in the state-action space when modeling  $Q(s, a)$  directly with actions as inputs. Other RL-based methods [Zhou et al., 2016] focus on grouping problems but do not address sequential decision-making. Grouping problems typically involve partitioning a set into mutually disjoint subsets according to specific criteria and constraints, yet these approaches lack scalability and flexibility for sequential applications. Recently, Saul et al. [2005] proposed an automatic agent clustering method based on the contribution weight of each agent to specific groups. Our method differ with their work in a sense that the policy of agent grouping is learnt under RL framework with the goal of maximizing the cumulative reward while they learn the grouping by a more indirect sparsity regulation.

The application of RL to multi-agent matching or pairing problems presents unique computational and modeling challenges, particularly when scaling to larger problem sizes. Traditional Q-learning or option-critic approaches face critical limitations in this context:

### 1. Challenges in Critic Network Design and Learning:

- **Expansive Option Space:** The number of ways to pair  $n$  items (teams) is given by the double factorial  $(n - 1)!!$ , leading to a massive growth in the option space. For example, when  $n = 20$ , the number of pairings exceeds 3.7 billion.
- **Inefficient Modeling:** Pairings such as  $(1, 2)$  and  $(3, 4)$  versus  $(3, 4)$  and  $(1, 2)$  represent identical team matchings but may be treated differently in conventional Q-function models, leading to redundancy and inefficiency.

### 2. Challenges in Policy Network Design and Learning:

- **Expansive Action Space:** As the number of teams and team sizes increase, the action space grows exponentially, complicating policy learning.
- **Scalability Issues:** Traditional option-critic architectures struggle to scale due to the exponential growth in policy parameter space with increasing option complexity.

To address these challenges, we propose the following contributions:

### 1. Solutions to Challenges in Critic Network Design:

- We redefine the critic Q-function by embedding pairing information directly within the network architecture rather than treating it as a separate input. This ensures permutation invariance of team pairings, enhancing modeling efficiency and reducing computational redundancy (see Figure 2).
- We decompose the joint Q-function into combinations of pair Q-values, transforming the problem of finding optimal team pairings into a linear optimization problem. This significantly reduces computational complexity by avoiding exhaustive enumeration (see Section 3.2).

### 2. Solutions to Challenges in Policy Network Design:

- We implement a Permutation Invariant (PI) network architecture that leverages the homogeneity of agents for effective parameter sharing (see Figure 1).
- Instead of using independent policy networks for each option, we incorporate pairing information directly within the policy network architecture (see Figure 1).

Our critic and policy networks employ a shared-parameter embedding for individual states, which are aggregated to form team character embeddings. These embeddings are transformed to generate PI matching embeddings for team pairs, which are summed to produce a final Q-value through a single fully connected layer. This architecture avoids the curse of dimensionality by circumventing direct engagement with the expansive action space, ensuring scalability as the number of teams or individuals increases without a proportional rise in parameter count.

## 2 Related Work

### 2.1 Hierarchical Reinforcement Learning (HRL)

Hierarchical RL addresses the challenge of large state and action spaces in long-horizon tasks by decomposing the problem into a hierarchy of subtasks. A higher-level policy selects optimal subtasks, while lower-level policies solve them, improving exploration and scalability.

**HRL Foundations.** Pateria et al. [2021] highlight how HRL improves exploration through structured subtasks or subgoals. The Semi-Markov Decision Process (SMDP) framework [Sutton et al., 1999] underpins many HRL methods by allowing variable-length subtasks. The Option-Critic architecture [Bacon et al., 2016, Chunduru and Precup, 2022] automates the learning of options (subtasks) and their termination conditions, facilitating adaptive and autonomous decision-making. Extensions to multi-agent systems [Chakravorty et al., 2019] allow decentralized execution and cooperative behavior.

**Policy over Options.** An option is defined as a tuple  $(\mathcal{I}, \pi, \beta)$ , where  $\mathcal{I}$  specifies the initiation set,  $\pi$  is the intra-option policy, and  $\beta$  is the termination function. Our work builds on this foundation by considering recursive optimality, where sub-behaviors are individually optimal, and policy learning over large option spaces, where permutation invariance and parameter efficiency are key.

### 2.2 Multi-Agent Reinforcement Learning (MARL)

Multi-agent RL involves multiple interacting agents learning policies to optimize collective or individual objectives. Hierarchical approaches [Pateria et al., 2021] integrate HRL concepts to decompose complex team-based decision problems, enabling more efficient coordination and scalability.

### 2.3 Matching Problems

Matching problems, particularly in reinforcement learning, span static and dynamic contexts, bipartite and monopartite structures, and stochastic settings.

**Bipartite vs. Monopartite Matching.** Bipartite matching involves two distinct sets (e.g., tasks and workers), while monopartite matching operates within a single homogeneous set. Dynamic bipartite graph matching (DBGM) [Wang et al., 2019] addresses scenarios where tasks and workers arrive and leave dynamically. The proposed adaptive batch-based framework leverages RL for batch splitting, enabling theoretical guarantees and efficient real-time decision-making.

**Order Dispatch and Markov Matching Markets.** Order dispatch systems [Xu et al., 2018] apply policy iteration and Q-learning to optimize driver-task assignments, handling stochastic action spaces [Cohen et al., 2022]. Markov matching markets [Min et al., 2022] introduce stochastic agent sets and planner actions that affect environmental transitions, integrating RL to optimize matching policies.

**Competing Matching with Complementary Preferences.** Huang [2022] tackle many-to-one matching using Thompson Sampling for preference exploration and double matching for stability, demonstrating RL’s applicability in balancing exploration and exploitation.

### 2.4 The Option-Critic Architecture

The Option-Critic framework [Bacon et al., 2016] provides a foundation for hierarchical policy learning by automating the discovery of options and their termination conditions. However, traditional architectures face challenges in scaling to large option spaces.

#### Differences in Proposed Architecture.

1. **Option Space:** Our problem involves an extremely large, known option space, necessitating parameter-efficient critic networks and advanced optimization methods to avoid exhaustive enumeration.
2. **Termination Function:** Unlike traditional approaches requiring learned termination functions, our framework assumes predefined termination conditions, simplifying the learning process.

**Comparison to Multi-Agent Cooperative Option-Critic.** Chakravorty et al. [2019] address partially observed state spaces and decentralized execution in multi-agent settings. Our work contrasts by leveraging fully observed state spaces, homogeneity assumptions, and parameter-efficient critic designs to manage large option spaces effectively.

In summary, our approach integrates and extends hierarchical RL, multi-agent RL, and matching theory to address challenges in large-scale, dynamic decision-making tasks. By leveraging permutation-invariant architectures and efficient policy optimization, we provide scalable solutions for complex pairing and grouping problems.

## 3 Hierarchical Multi-Agent Reinforcement Learning for Agent Grouping/Pairing

We first introduce the motivating example. In the Intern Health Study [Wang et al., 2023], medical interns from diverse backgrounds—different specialties and universities—are weekly grouped into teams to compete on health metrics including daily step counts, mood scores, and sleep hours, all managed through a dedicated app. This app is responsible for determining the optimal team pairings each week and deciding whether to send health-promoting messages to individuals daily, with the goal of maximizing interns’ health throughout their internship period. This setup presents a challenging multi-agent problem, where high-level decisions on team formations and low-level decisions on individual interventions must be dynamically optimized. The hierarchical nature of these decisions—where team pairings influence daily interactions and messaging—motivates the application of a hierarchical multi-agent reinforcement learning approach, capable of efficiently coordinating both levels to achieve the best health outcomes.

This problem is addressed within a hierarchical Reinforcement Learning (RL) framework, involving homogeneous and fully cooperative agents. This approach is well-suited for scenarios where agents need to dynamically form groups or pairs to optimize collective performance.

- **Homogeneous and Fully Cooperative Agents.** In this setting, agents are homogeneous, meaning they share the same structure, skills, and learning models. This homogeneity allows for efficient

learning and scalability, as agents can share experiences and model parameters. Despite sharing the same policy network, agents can exhibit diverse behaviors due to their unique observations at any given time. Additionally, agents can leverage centralized value functions, incorporating mutual information and joint actions, to enhance learning efficiency [Gronauer and Diepold, 2021]. We adopt the CTDE (Centralized Training with Decentralized Execution) paradigm, which is state-of-the-art for multi-agent learning [Kraemer and Banerjee, 2016, Oliehoek et al., 2008].

- **Hierarchical Reinforcement Learning** The hierarchical RL framework involves two levels of decision-making:

1. **Option Policy:** This higher-level policy determines the groupings or pairings of agents (e.g., weekly team pairings), effectively selecting "options" that define the structure of interactions.
2. **Intra-Option Policy:** Within the context of a selected grouping (option), this lower-level policy governs the day-to-day interactions (e.g., daily messaging) aimed at optimizing health outcomes.

Given the availability of a centralized controller in our application (e.g., an IHS APP), option decisions can be based on the full state space, unlike scenarios where options are based on partial observations [Chakravorty et al., 2019]. The homogeneity of agents suggests the use of permutation-invariant neural networks to reduce the parameter space of the joint Q-function or policy network.

### 3.1 The Options Framework

Define the state for subject  $j$  of team  $i$  observed at time  $t$  as  $s_{i,j,t}$ , the action as  $a_{i,j,t}$  and the reward as  $r_{i,j,t}$ . We consider a discrete and finite individual action space  $\mathcal{A} = \{a_1, \dots, a_{|\mathcal{A}|}\}$ . Define the number of teams as  $M$ , the number of days requiring individual action decision each week as  $H$  and the weeks are indicated by  $w$ . We model the underlying system as an MDP and employ the options framework to structure hierarchical learning. An option  $\omega$  is defined as a triple  $(\mathcal{I}_\omega, \pi_\omega, \beta)$ :

- Initiation Set  $\mathcal{I}_\omega$ : Specifies the states where option  $\omega$  can be initiated.
- Intra-Option Policy

$$\pi_\omega(\mathbf{a}_t | \mathbf{s}_t; \theta) = \prod_{k=1}^{M/2} \prod_{i \in m_k(\omega)} \prod_j \pi(a_{i,j,t} | \mathbf{s}_{m_k(\omega),t}; \theta)$$

where  $\mathbf{s}_{m_k(\omega)} = \{s_{i,j,t}\}_{i \in m_k(\omega), j}$ . Define the set of team matching pairs induced by  $\omega$  as  $\mathcal{S}_\omega = \{(m_1(\omega)^1, m_1(\omega)^2), (m_2(\omega)^1, m_2(\omega)^2), \dots, (m_{M/2}(\omega)^1, m_{M/2}(\omega)^2) : \text{where } \cup (m_i(\omega)^1, m_i(\omega)^2) = [M] \text{ and } (m_i(\omega)^1, m_i(\omega)^2) \cap (m_j(\omega)^1, m_j(\omega)^2) = \emptyset, \forall i \neq j\}$ . Define  $m_i(\omega) = (m_i(\omega)^1, m_i(\omega)^2)$ . The parameter  $\theta$  is shared across all teams and options to ensure consistency and reduce complexity.

- Termination Condition  $\beta$ : Determines the probability of terminating the option in a given state:

$$\beta(S_{\cdot,\cdot,t}) = \begin{cases} 0 & \text{if } wH \leq t < (w+1)H \\ 1 & \text{otherwise} \end{cases}$$

Agents select an option  $\omega$  according to the policy over options  $\pi_\Omega(\omega | s)$ , execute the intra-option policy  $\pi_\omega$  until termination (as dictated by  $\beta$ ), and then repeat the process. This setup transforms the MDP into a Semi-Markov Decision Process (SMDP), enabling the use of optimal value functions over options  $V_\Omega(s)$  and  $Q_\Omega(s, \omega)$ .

We adopt the option-critic frame work to learn the joint Q network and intra-option policy. We define several important functions in the option-critic framework under our agent grouping settings. The option-value function  $Q_{\Omega,\psi}(s, \omega)$  is defined as:

$$Q_{\Omega,\psi}(s, \omega) = \sum_{\mathbf{a}} \pi_{\omega,\theta}(\mathbf{a} | \mathbf{s}) Q_{U,\psi}(\mathbf{s}, \omega, \mathbf{a})$$

where  $Q_U$  represents the value of executing an action in the context of a state-option pair:

$$Q_{U,\psi}(\mathbf{s}, \omega, a) = \sum_i r(s_i, a_i) + \gamma \sum_{\mathbf{s}'} P(\mathbf{s}' | \mathbf{s}, \mathbf{a}, \omega) U(\omega, \mathbf{s}')$$

The option-value function upon arrival  $U_\psi(\omega, \mathbf{s}')$  is:

$$U_\psi(\omega, \mathbf{s}') = (1 - \beta(\mathbf{s}'))Q_{\Omega, \psi}(\mathbf{s}', \omega) + \beta(\mathbf{s}')V_{\Omega, \psi}(\mathbf{s}')$$

with  $V_{\Omega, \psi}(\mathbf{s}) = \sum_\omega \pi_\Omega(\omega)Q_{\Omega, \psi}(\mathbf{s}, \omega)$ .

The gradient of the expected discounted return with respect to  $\theta$  and initial condition  $(\mathbf{s}_0, \omega_0)$  is:

$$\sum_{\mathbf{s}, \omega} \mu_\Omega(\mathbf{s}, \omega \mid \mathbf{s}_0, \omega_0) \sum_{\mathbf{a}} \frac{\partial \pi_{\omega, \theta}(\mathbf{a} \mid \mathbf{s})}{\partial \theta} Q_U(\mathbf{s}, \omega, \mathbf{a})$$

where  $\mu_\Omega(\mathbf{s}, \omega \mid \mathbf{s}_0, \omega_0)$  is the discounted weighting of state-option pairs along trajectories starting from  $(\mathbf{s}_0, \omega_0)$ .

The gradient of the expected discounted return with respect to  $\theta$  and initial condition  $(\mathbf{s}_0, \omega_0)$  is:

$$\sum_{\mathbf{s}, \omega} \mu_\Omega(\mathbf{s}, \omega \mid \mathbf{s}_0, \omega_0) \sum_{\mathbf{a}} \frac{\partial \pi_{\omega, \theta}(\mathbf{a} \mid \mathbf{s})}{\partial \theta} Q_U(\mathbf{s}, \omega, \mathbf{a})$$

where  $\mu_\Omega(\mathbf{s}, \omega \mid \mathbf{s}_0, \omega_0)$  is the discounted weighting of state-option pairs along trajectories starting from  $(\mathbf{s}_0, \omega_0)$ .

With those definitions, the option-critic framework in Bacon et al. [2016] can be adopted to solve the hierarchical policy learning in theory. However, the large option/action space involved in the Q functions above make it practically challenging to solve the optimization.

### 3.2 Network Architecture for Dimension Reduction

To handle the permutation invariance and scalability required for our multi-agent setting, we employ the Deep Set architecture [Zaheer et al., 2017] for both the policy and critic networks. This architecture ensures that the network is invariant to the order of input agents and can handle varying numbers of agents. The permutation invariant network enjoys several advantages:

- Permutation Invariance: The network’s output remains unchanged regardless of the order of input agents.
- Permutation Invariance: The network’s output remains unchanged regardless of the order of input agents.
- Input Size Irrelevance: The network can handle any number of agents without modification
- Privacy Consideration: Individual agent information is encoded into embeddings before being aggregated, ensuring privacy.
- Generalization: The trained network can be applied to new cohorts of agents with different sizes.

Without loss of generality, assume team  $i = m_1(\omega)^1$  at time  $t$ . The decomposed individual policy  $\pi(a_{i,j,t} \mid s_{m_1(\omega),t}) = \pi(a_{i,j,t} \mid s_{i,j,t}, s_{m_1(\omega)^1} \setminus s_{i,j,t}, s_{m_1(\omega)^2})$  is designed in this way. First the individual’s state  $s_{i,j,t}$ , the state of each of the rest individual from team  $i$ , and the state of each of the individuals from team  $m_i(\omega)^2$  are separately passed through three encoders. Then, the embeddings of the remaining individuals from team  $i$  and team  $m_1(\omega)^2$  are pooled together separately by some permutation invariant operators such as averaging to obtain the team member’s characters and the rival characters. Finally, the individual embedding, the team member’s characters and the rival characters are passed through some layers to output the final action for this individual. The policy network designed is illustrated in Figure 1.

The joint Q-network is designed to avoid explicit enumeration of the action space when combined with a greedy policy. It uses a shared encoder to generate embeddings for individual agents, which are then aggregated using permutation-invariant operators to form team and then group embeddings. This design significantly reduces the number of parameters and accelerates learning. The joint Q-network is illustrated in Figure 2. Given the Q-network architecture in Figure 2, the greedy policy over options can

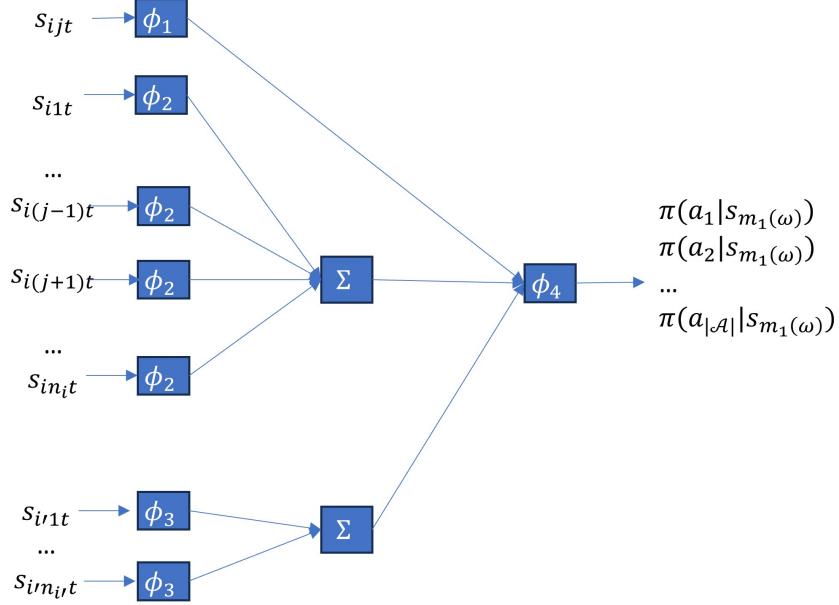


Figure 1: Permutation-Invariant Policy Network for subject  $j$  from time  $i$  at time  $t$ . Assume team  $i$  is matched with team  $i'$ .

be derived by solving:

$$\begin{aligned} \max & \sum_{t_1=1}^M \sum_{t_2=1}^M a_{t_1,t_2} \psi_4(t_1, t_2) \\ \text{s.t. } & a_{t_1,t_2} \in \{0, 1\} \\ & a_{t_1,t_2} = a_{t_2,t_1} \\ & \sum_{t_1=1, t_2 \neq t_1}^M a_{t_1,t_2} = 1 \quad \text{for } t_2 = 1, 2, \dots, M. \end{aligned}$$

where  $\psi_4(t_1, t_2) = \psi_4(\psi_3(\psi_2(\sum_{i \in t_1} \psi_1(s_i)) + \psi_2(\sum_{i \in t_2} \psi_1(s_i))))$ .

This formulation allows for efficient computation of the optimal groupings without explicitly enumerating all possible pairings.

## 4 Simulation Study

We design a simulated environment similar to the Intern Health Study. The state variable for each individual is the daily the square root of step count and the reward is the square root of next day's step count. The action is a binary variable indicating whether to send a text message or not. We also allow some teams not enter competition in some weeks. We would view this case as a pair  $(m_k(\omega)^1, m_k(\omega)^2)$  where  $m_k(\omega)^1 = m_k(\omega)^2$ . The transition function is defined as:

$$S_{m_k(\omega)^q, j, t+1} = \begin{cases} -0.0005 + 0.3289 S_{m_k(\omega)^q, j, t} + 0.0672 \sum_{j'} S_{m_k(\omega)^{q'}, [j'], j, t} + 0.0103 \mathbb{I}(q = q') & \text{if } A_{i,j,t} = 0; \\ -0.0009 + 0.32450 S_{m_k(\omega)^q, j, t} + 0.0746 \sum_{j'} S_{m_k(\omega)^{q'}, [j'], j, t} + 0.0136 \mathbb{I}(q = q') & \text{if } A_{i,j,t} = 1; \end{cases}$$

All encoders in the critic model and the policy network is a two-layer perceptron with each layer containing two nodes followed by a ReLu activation function. We initialize 10 teams each containing 10 subjects with initial state sampled from a standard normal distribution. We train the policy with 10000 iterations and a learning rate of 0.001. After training, the obtained policy is evaluated in the same simulated environment by executing it for 1000 steps and calculated the cumulative reward. The gamma discounted factor is set ot be 0.9. We compare the policy obtained by the proposed method with two

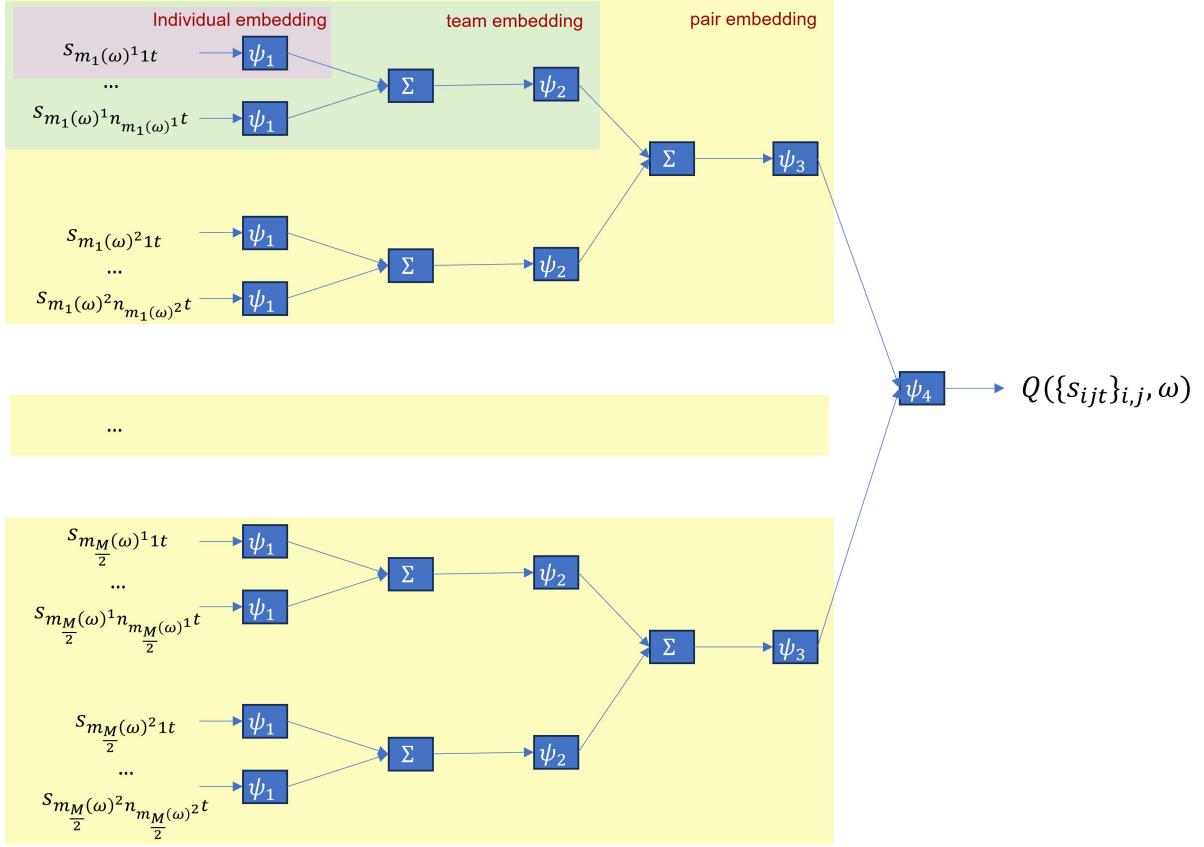


Figure 2: Permutation-Invariant Critic Network Design.

fixed policy (executing action 0 or 1) and a random policy which randomly match teams each week and choose action 0 or 1 for each individual with equal probabilities. The evaluated reward is reported in Table where the results are averaged over 20 repetitions.

Method	Average reward	0.9-discounted cumulated reward
Proposed	-143.04	-1436.97
Action 0	-259.87	2601.24
Action 1	-238.22	-2391.28
Random	-248.91	-2495.50

Table 1: The estimated value of different methods.

## References

- Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. *ArXiv*, abs/1609.05140, 2016. URL <https://api.semanticscholar.org/CorpusID:6627476>.
- Jhelum Chakravorty, Nadeem Ward, Julien Roy, Maxime Chevalier-Boisvert, Sumana Basu, Andrei-Stefan Lupu, and Doina Precup. Option-critic in cooperative multi-agent systems. *ArXiv*, abs/1911.12825, 2019. URL <https://api.semanticscholar.org/CorpusID:208512835>.
- Raviteja Chunduru and Doina Precup. Attention option-critic. *ArXiv*, abs/2201.02628, 2022. URL <https://api.semanticscholar.org/CorpusID:221093638>.
- Deborah Cohen, Moonkyung Ryu, Yinlam Chow, Orgad Keller, Ido Greenberg, Avinatan Hassidim, Michael Fink, Yossi Matias, Idan Szpektor, Craig Boutilier, and Gal Elidan. Dynamic planning in open-ended dialogue using reinforcement learning, 2022.

Massimo Cossentino, Vincent Hilaire, Ambra Molesini, Valeria Seidita, et al. *Handbook on agent-oriented design processes*. Springer, 2014.

Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55:895 – 943, 2021. URL <https://api.semanticscholar.org/CorpusID:234833859>.

Chao Huang. Two-sided matching with firms' complementary preferences. *arXiv preprint arXiv:2205.05599*, 2022.

Raphaël Jeanson, Penelope F Kukuk, and Jennifer H Fewell. Emergence of division of labour in halictine bees: contributions of social interactions and behavioural variance. *Animal behaviour*, 70(5):1183–1193, 2005.

Hao Jiang, Dianxi Shi, Chao Xue, Yajie Wang, Gongju Wang, and Yongjun Zhang. Multi-agent deep reinforcement learning with type-based hierarchical group communication. *Applied Intelligence*, 51: 5793–5808, 2021.

Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

Jaehee Lee and Jaewoo So. Reinforcement learning-based joint user pairing and power allocation in mimo-noma systems. *Sensors*, 20(24), 2020. ISSN 1424-8220. doi: 10.3390/s20247094. URL <https://www.mdpi.com/1424-8220/20/24/7094>.

Kemas M Lhaksmana, Yohei Murakami, and Toru Ishida. Role-based modeling for designing agent behavior in self-organizing multi-agent systems. *International Journal of Software Engineering and Knowledge Engineering*, 28(01):79–96, 2018.

Yifei Min, Tianhao Wang, Ruitu Xu, Zhaoran Wang, Michael I. Jordan, and Zhuoran Yang. Learn to match with no regret: Reinforcement learning in markov matching markets, 2022.

Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys (CSUR)*, 54(5):1–35, 2021.

Thomy Phan, Fabian Ritz, Lenz Belzner, Philipp Altmann, Thomas Gabor, and Claudia Linnhoff-Popien. Vast: Value function factorization with variable agent sub-teams. *Advances in Neural Information Processing Systems*, 34:24018–24032, 2021.

Lawrence K Saul, Yair Weiss, and Léon Bottou. *Advances in neural information processing systems 17: proceedings of the 2004 conference*, volume 17. MIT Press, 2005.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211, 1999. ISSN 0004-3702. doi: [https://doi.org/10.1016/S0004-3702\(99\)00052-1](https://doi.org/10.1016/S0004-3702(99)00052-1). URL <https://www.sciencedirect.com/science/article/pii/S0004370299000521>.

Jitao Wang, Yu Fang, Elena Frank, Maureen A Walton, Margit Burmeister, Ambuj Tewari, Walter Dempsey, Timothy NeCamp, Srijan Sen, and Zhenke Wu. Effectiveness of gamified team competition as mhealth intervention for medical interns: a cluster micro-randomized trial. *NPJ Digital Medicine*, 6(1):4, 2023.

Shaoyang Wang, Tiejun Lv, Wei Ni, Norman C. Beaulieu, and Y. Jay Guo. Joint resource management for mc-noma: A deep reinforcement learning approach. *IEEE Transactions on Wireless Communications*, 20:5672–5688, 2021. URL <https://api.semanticscholar.org/CorpusID:232404915>.

Yansheng Wang, Yongxin Tong, Cheng Long, Pan Xu, Ke Xu, and Weifeng Lv. Adaptive dynamic bipartite graph matching: A reinforcement learning approach. In *2019 IEEE 35th international conference on data engineering (ICDE)*, pages 1478–1489. IEEE, 2019.

G Wittemyer and Wayne M Getz. Hierarchical dominance structure and social organization in african elephants, loxodonta africana. *Animal Behaviour*, 73(4):671–681, 2007.

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: A learning and planning approach. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 905–913, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. *ArXiv*, abs/1703.06114, 2017. URL <https://api.semanticscholar.org/CorpusID:263878920>.

Yangming Zhou, Jin-Kao Hao, and Béatrice Duval. Reinforcement learning based local search for grouping problems. *Expert Syst. Appl.*, 64(C):412–422, dec 2016. ISSN 0957-4174. doi: 10.1016/j.eswa.2016.07.047. URL <https://doi.org/10.1016/j.eswa.2016.07.047>.