Triple-BERT: Do We Really Need MARL for Order Dispatch on Ride-Sharing Platforms?

Zijian Zhao (zzhaock@connect.ust.hk), Sen Li*

Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology

Problem Background

- **Arbitrary Order Arrival:** Orders can arrive at any time without a fixed schedule.
- **Centralized Assignment:** A centralized platform efficiently assigns orders to vehicles, often bundling them with en-route orders.
- **Dynamic Route Updates:** Vehicles continuously update their routes to reflect the shortest possible path.
- **Order Management:** Unassigned orders return to the platform for reassignment but may be canceled if not confirmed within a specified time threshold.
- **Challenges:** The observation and action spaces are extremely large in ride-sharing scenarios. With 1000 vehicles and 10 orders, the number of combinations can approach 10^{30}.

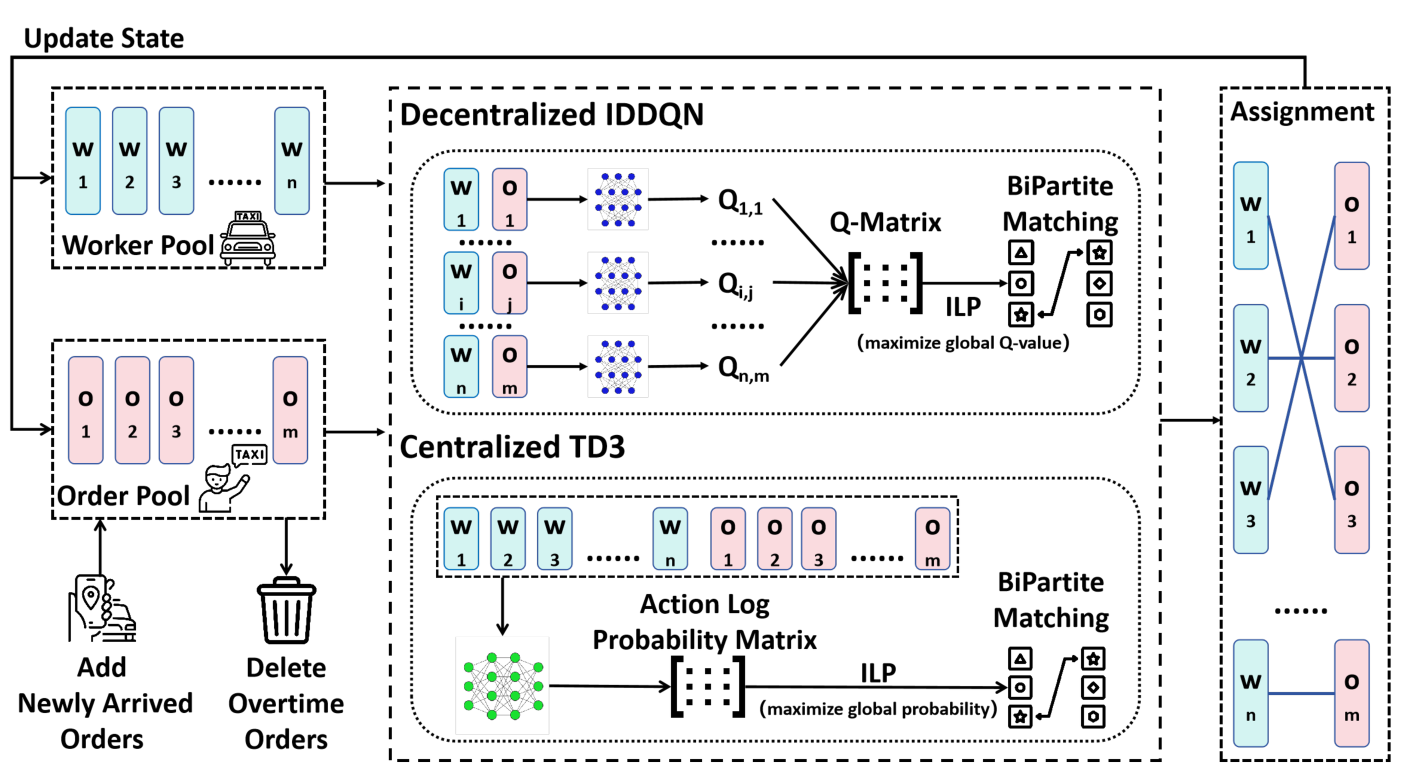

Fig. 1: Workflow

Previous MARL-based Method

- **Step 1:** Estimate the Q-value for each vehicle-order pair $y_{i,j,t}$ at time t.
- **Step 2:** Decide order assignment A_t by maximizing the global Q-value:

$$
\begin{aligned}
&\max_{A_t} \sum_{i\in\mathcal{I}} a_{i,j,t}\cdot y_{i,j,t}, \\
\text{s.t.}\quad &\sum_{i\in\mathcal{I}} a_{i,j,t} \le 1, \quad \forall j \in \mathcal{J}_t, \\
&\sum_{j\in\mathcal{J}_t} a_{i,j,t} \le 1, \quad \forall i \in \mathcal{I}, \\
&a_{i,j,t} \in \{0,1\}, \quad \forall i \in \mathcal{I}, j \in \mathcal{J}_t.
\end{aligned} \tag{1}
$$

  - $\mathcal{I}$: Vehicle set
  - $\mathcal{J}_t$: Order set at time t
  - $y_{i,j,t}$ = Q-Network(Vehicle-i, Order-j)
- **Step 3:** Update the estimator (policy) using TD-learning.
- **Shortcomings:**
  - Decentralized methods face challenges of unstable environments and poor cooperation.
  - Centralized methods encounter the Curse of Dimensionality (CoD).

Proposed SARL-based Method

We propose a centralized SARL solution based on a variant of TD3 for large-scale trip-vehicle assignment tasks. (i) To address the large observation space, we propose a BERT-based network, leveraging its self-attention and parameter reuse mechanisms. (ii) Regarding the large action space, we introduce a novel action decomposition mechanism that divides the joint action probability into the virtual action probabilities of each individual vehicle.

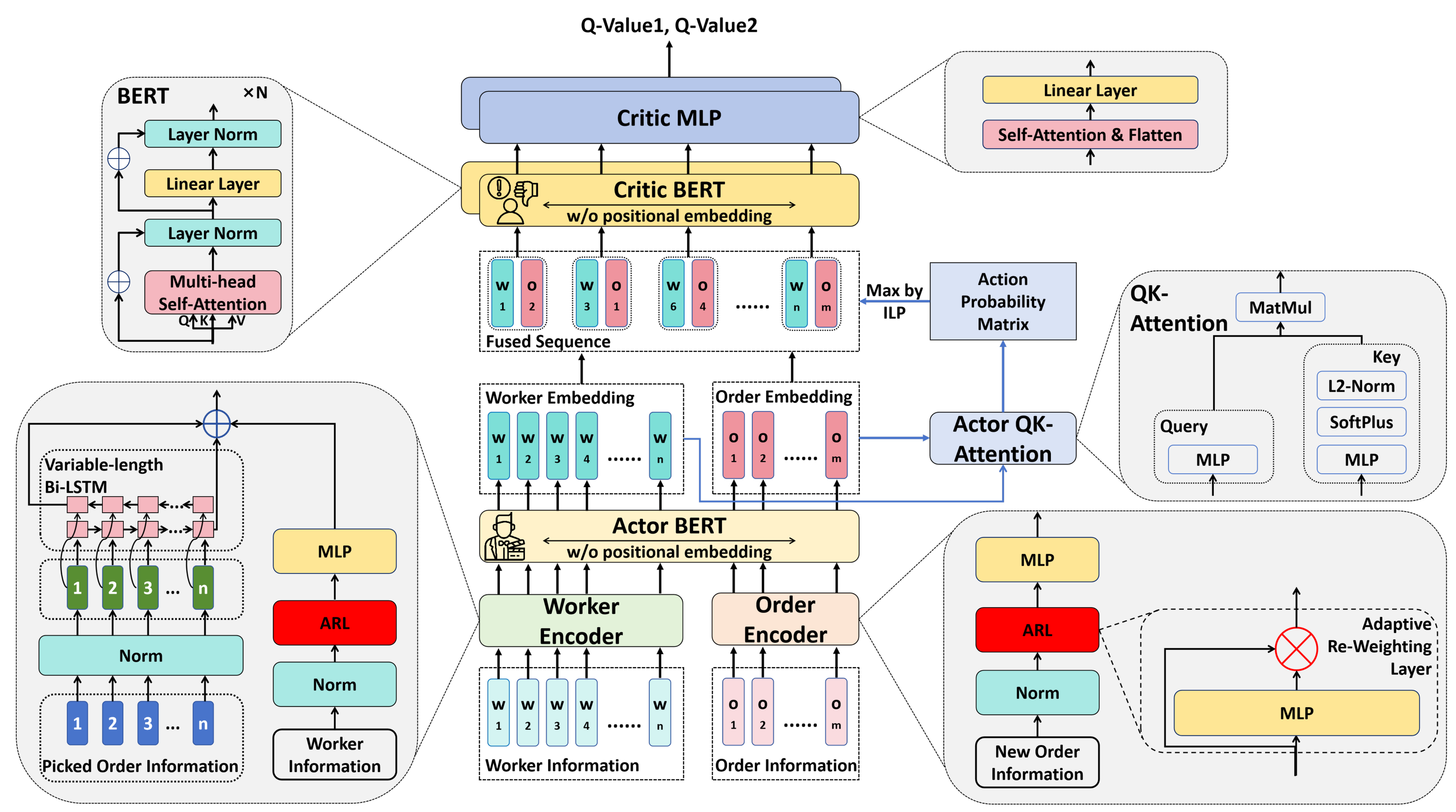

Fig. 2: Network Architecture

A. Network Architecture

- **Actor (Updated by Policy Gradient):**
  - Each vehicle and order information is treated as a token, from which features and relationships are extracted using Actor-BERT.
  - Generate a virtual matching probability between vehicle i and order j at time t, denoted as $\mathscr{P}_{i,j,t}$.
- **Critic (Updated by TD-Learning):**
  - Each matching vehicle-order pair is treated as a token, and features and relationships are extracted using Critic-BERT.
  - Estimate the Q-value based on the output of Critic-BERT.

B. Action Decomposition

- **Basic Principle:** Construct a structural policy space:

$$
\pi(A_t|S_t) = \mathsf{z}\left(\prod_{i,j\in \mathsf{h}(A_t)} \mathscr{P}_{i,j,t}\right) \tag{2}
$$

  - $\mathsf{z}(\cdot)$: A virtual increasing mapping function.
  - $\mathsf{h}(A_t)$: Defined as $\mathsf{h}(A_t) = \{(i,j)|a_{i,j,t}=1\}$.
- **Action Sampling:** Solve Equation 1 by replacing $y_{i,j,t}$ with $\log \mathscr{P}_{i,j,t}$:

$$
\arg\max_{A_t} \pi(A_t|S_t) = \arg\max_{A_t} \mathsf{z}\left(\prod_{i,j\in \mathsf{h}(A_t)} \mathscr{P}_{i,j,t}\right) = \arg\max_{A_t} \sum_{i,j\in \mathsf{h}(A_t)} \log \mathscr{P}_{i,j,t}\,. \tag{3}
$$

- **Policy Updating:**

$$
\nabla_\Theta \mathsf{J}(\Theta) \propto \mathbb{E}_{\pi_\Theta}\left[\mathsf{Q}(S_t, A_t)\nabla_\Theta \sum_{i,j\in \mathsf{h}(A_t)} \log \mathscr{P}_{i,j,t}\right] \tag{4}
$$

Experiment Results

- **Dataset:** A real-world ride-hailing dataset from Manhattan, New York [6].
- **Training Process:**
  - First, pre-train the encoder component using a decentralized IDDQN approach.
  - Then, train the entire network using a centralized TD3 approach.
- **Performance:** Triple-BERT outperforms other MARL methods by optimizing pickup time, which leads to a higher order service rate and total reward.

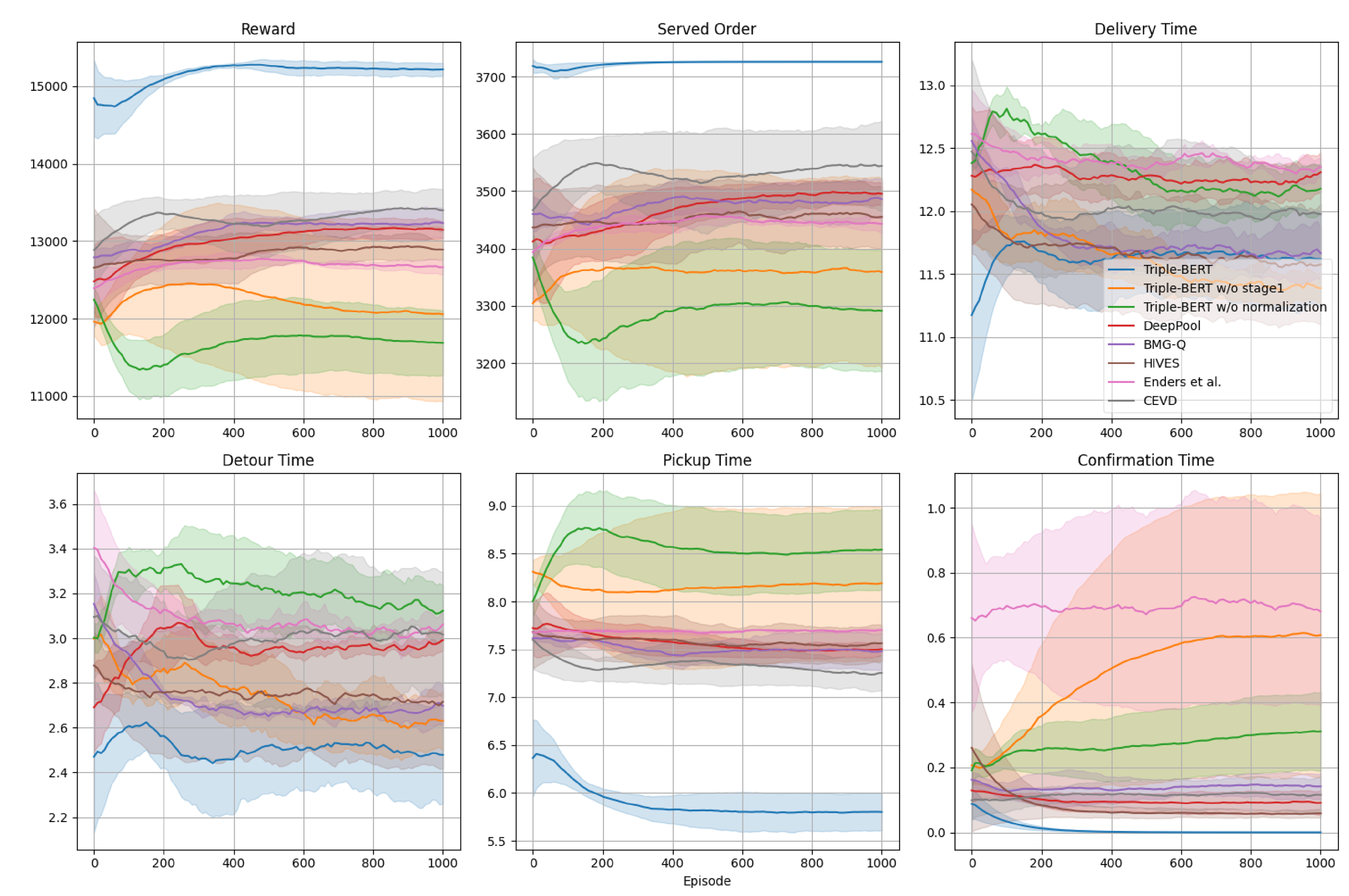

Fig. 3: Method Comparison

Method	Reward	Service-Rate	Delivery	Detour	Pickup	Confirmation
DeepPool [1]	12723.85	0.91	11.53	2.47	7.77	0.06
BMG-Q [5]	13036.29	0.92	**10.57**	**1.90**	7.61	0.10
HIVES [4]	12365.11	0.89	11.04	2.28	7.99	**0.03**
Enders et al. [3]	12041.62	0.90	12.28	2.90	7.94	0.80
CEVD [2]	13157.96	0.94	11.36	2.31	7.37	0.06
Triple-BERT	**14730.48**	**0.98**	11.53	2.52	**5.73**	0.13

Tab. 1: Average performance across multiple periods. The last four columns denote the time for each metric (unit: minute).

References

[1] Abubakr O Al-Abbasi, Arnob Ghosh, and Vaneet Aggarwal. "Deeppool: Distributed model-free algorithm for ride-sharing using deep reinforcement learning". In: *IEEE Transactions on Intelligent Transportation Systems* 20.12 (2019), pp. 4714–4727.

[2] Avinandan Bose et al. "On Sustainable Ride Pooling Through Conditional Expected Value Decomposition". In: *ECAI 2023*. IOS Press, 2023, pp. 295–302.

[3] Tobias Enders et al. "Hybrid multi-agent deep reinforcement learning for autonomous mobility on demand systems". In: *Learning for Dynamics and Control Conference*. PMLR. 2023, pp. 1284–1296.

[4] Jiang Hao and Pradeep Varakantham. "Hierarchical value decomposition for effective on-demand ride-pooling". In: *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*. 2022, pp. 580–587.

[5] Yulong Hu, Siyuan Feng, and Sen Li. "BMG-Q: Localized Bipartite Match Graph Attention Q-Learning for Ride-Pooling Order Dispatch". In: *arXiv preprint arXiv:2501.13448* (2025).

[6] New York City Taxi and Limousine Commission. *Nyc taxi and limousine commission-trip*