MacMic: Executing Iceberg Orders via Hierarchical Reinforcement Learning

Hui Niu[*1], **Siyuan Li**[*2] and **Jian Li**[†1]
[1]Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China
[2]Faculty of Computing, Harbin Institute of Technology, Harbin, China
niuh17@mails.tsinghua.edu.cn, lijian83@mail.tsinghua.edu.cn, siyuanli@hit.edu.cn

Abstract

In recent years, there has been a growing interest in applying reinforcement learning (RL) techniques to order execution owing to RL's strong sequential decision-making ability. However, realistic order execution tasks usually involve a large fine-grained action space and a long trading duration. The former hinders the RL agents from efficient exploration. The latter increases the task complexity, since the agent must capture price advantages throughout the day as well as micro changes within a few seconds on the limited order books. In addressing these challenges, we propose MacMic, a novel Hierarchical RL-based order execution approach that captures market patterns and executes orders from different temporal scales. MacMic employs a high-level agent to split the parent order into smaller slices at coarse-grained time steps. Then a low-level agent is adopted to execute these slices by placing fixed-size sub-orders at a continuous time. Besides, to balance the multifaceted objectives of the two tasks, MacMic pretrains a causal stacking hidden Markov model (SHMM) to obtain both effective macro-level and micro-level market states. Comprehensive experimental results on 200 stocks across the US and China A-share markets validate the effectiveness of the proposed method.

1 Introduction

Optimal order execution is an essential financial problem that embodies the bridge between trading intentions and actual transactions. When executing a large order, brokers may encounter a significant negative slippage caused by market impacts. An effective solution is to divide the large order into smaller sub-orders and execute them sequentially over an extended period, typically hours or one day. Well known as *Iceberg Order* [Esser and Mönch, 2007], the strategy is widely adopted by financial institutions.

Compared to static traditional methods, Reinforcement Learning (RL) has become a promising approach for acquiring execution strategies that can continuously adapt to dynamic markets. However, existing RL approaches for optimal execution mainly focus on strategies executed for a few minutes, as shown in Table 1. While these studies ignore the challenging long-duration strategies, designing such strategies to beat the daily market volume-weighted average price (VWAP) is a critical concern for many realistic brokers. Besides, state-of-the-art RL-based execution strategies utilize limit orders to save transaction costs [Chen *et al.*, 2022; Pan *et al.*, 2022], resulting in a large two-dimensional action space (for deciding both the price and size of the limit order). Unfortunately, learning from scratch for tasks with large action spaces and long time horizons remains a significant challenge for RL algorithms [Bellemare *et al.*, 2017; Wang *et al.*, 2021].

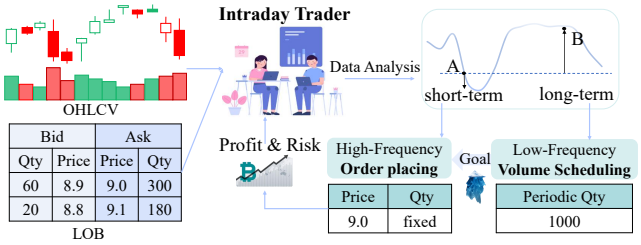


Figure 1: The efficient hierarchical workflow of trade execution.

In addressing these challenges, we formulate the optimal execution problem as a hierarchical Markov Decision Process (MDP), decomposing the (long-duration) task into two stages with simpler subgoals, during which only one-dimensional price or volume needs to be decided. As shown in Figure 1, during the **volume scheduling** stage, the large parent order is subdivided into smaller iceberg orders for execution within a time frame of several minutes; Subsequently, in the **order placement** stage, the segmented iceberg order is transformed into a series of fixed-size sub-orders submitted to a trading venue, with only the limit price need to be specified.

A key challenge to solve this hierarchical MDP is to balance the multifaceted objectives of different decision processes. The volume scheduling process is mainly concerned with longer-term profit, while order placement cares about the best execution within a short period but might not always be aware of long-term profit sources. Therefore, treating the two stages as distinct tasks and simply integrating their solutions provided by existing flat RL methods leads to unsatisfying performance. In contrast, Hierarchical RL (HRL) is

an effective alternative that can jointly address the two tasks without ignoring their mutual influence. Besides, the success of RL algorithms hinges on the quality of representations. Since the volume scheduling task necessitates long-term market representations, whereas the order placement task focuses more on short-term market fluctuation, there is a keen need to derive effective multi-granularity market representations. While relying on the reward function to learn features in complex systems can be severely limited, a number of recent works have significantly improved RL performance by introducing auxiliary losses [Srinivas *et al.*, 2020; Stooke *et al.*, 2020].

Motivated by the above inspirations, we propose MacMic, a hierarchical RL (HRL) framework incorporated with effective representation learning techniques to address the challenging long-duration order execution problem. The primary contributions of this paper can be summarized as follows:

- **Problem Formulation:** We formulate the optimal execution problem as a hierarchical MDP. This formulation decomposes the challenging task into two sub-tasks with one-dimensional action spaces, enabling continuous-time order placing while addressing the poor exploration caused by the large action space to decide limit orders.
- **Model Structure:** We introduce a novel stacking Hidden Markov Model (SHMM) to learn effective representations. As an unsupervised learning auxiliary task, SHMM learns both macro and micro-level market representations without defining labels.
- **Learning Algorithm:** We solve the proposed MDP within an HRL framework. To train the high-level policy, we design a hindsight expert dataset and integrate imitation learning (IL) techniques. Ablation studies validate the superiority of the HRL framework and the effectiveness of IL in Section 6.2.
- **Empirical Improvement:** Comprehensive experiments on 200 realistic stock data demonstrate the proposed MacMic outperforms current state-of-the-art baselines.

Method	Execution Time	Total Order Size
[Nevmyvaka *et al.*, 2006]	2∼8 min	5K;10K
[Hendricks and Wilcox, 2014]	20∼60 min	10K;1M
[Lin and Beling, 2020]	2 min	300 ∼ 7K
[Fang *et al.*, 2021]	30 min	<1K
[Wang *et al.*, 2021]	20min	10K∼40K
[Chen *et al.*, 2022]	16 sec	80
Ours	240 min	100K+

Table 1: Existing studies of RL for order execution.

2 Related Work

Traditional finance approaches for optimal execution are typically model-based [Almgren and Chriss, 2001; Huberman and Stanzl, 2005; Bertsimas and Lo, 1998]. These methods usually assume that market price movements follow stochastic processes and employ stochastic optimal control methods to analytically derive the volume trajectory. Nevertheless, these assumptions may not hold in real-world scenarios. In practice, rule-based strategies such as time-weighted average price (TWAP) strategy [Bertsimas and Lo, 1998] and volume-weighted average price (VWAP) strategy [Kakade *et al.*, 2004] are quite popular. Despite their simplicity, TWAP and VWAP remain widely favored today because their execution costs consistently align with the market.

Compared to static rule-based models, Reinforcement Learning (RL) has become a promising method to execute orders in dynamic markets. Existing RL approaches for optimal order execution mainly focus on strategies executed for a few minutes. [Nevmyvaka *et al.*, 2006] pioneered the application of model-free RL for order placement, employing Q-Learning [Watkins and Dayan, 1992] to guide the agent in selecting the limit price. [Ning *et al.*, 2018; Lin and Beling, 2019] proposed variations of DQN to choose discrete volumes of market orders. These adaptations aim to address the high-dimensional nature and complexity of financial markets using deep neural networks. There is a PPO-based optimal execution framework designed to make decisions based on raw level-2 market data [Lin and Beling, 2020]. Recently, [Pan *et al.*, 2022] introduced a hybrid framework in which the agent first scopes an action subset and then chooses a specific limit price to achieve accurate control. Besides, [Chen *et al.*, 2022] proposed a cost-efficient RL approach under a dynamic market environment. Despite these advances, when applied to long-duration tasks, such methods suffer from inefficiency caused by the complex action space and inadequate market representations.

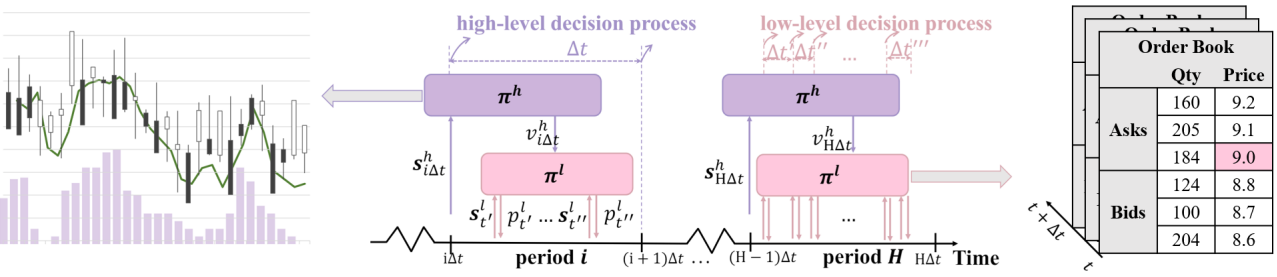
There are also studies that simplify the long-duration order execution problem to a coarse-grained volume-splitting problem. The policy distillation paradigm was employed in order execution [Fang *et al.*, 2021], utilizing a distilled PPO agent to determine optimal order-splitting volumes. While simplifying the order execution problem to either a coarse-grained volume splitting problem or a short-term order placement problem offers an easier partial solution, it is less realistic and leads to potential losses.

In contrast to these studies, we propose to balance the multifaceted objectives of the long-term price advantages and short-term execution opportunities leveraging the HRL framework.

3 A Hierarchical MDP Framework

Drawing inspiration from the way financial institutions employ iceberg orders [Esser and Mönch, 2007], we formulate the trade execution as two hierarchical MDPs, respectively for coarse-grained volume scheduling and fine-grained order placement, as shown in Figure 2. Without loss of generality, this paper considers sell-side order execution.

The volume scheduling process is mainly concerned with longer-term profit. In the volume scheduling MDP, the whole trading period T is equally segmented into H sub-periods, with the start points denoted as $t \in \{0, 1, ..., (H-1)\} \cdot \Delta t$, where the sub-period $\Delta t = \lfloor T/H \rfloor$. At the start of each sub-period, the high-level policy determines the desired volume of the iceberg order to be executed in this sub-period. At the end of the last sub-period, the remaining volume is traded with a market order.



(a) High-Level Volume Scheduling (b) Hierarchical MDP (c) Order Placement

Figure 2: The hierarchical MDP for order execution.

Order placement cares about the best execution within a short period. A way more in line with real scenarios to model order placement is a continuous-time decision process, rather than adhering to fixed discrete time points. Specifically, we divide the iceberg order allocated by the high-level agent into sub-orders of a fixed basic size b_i. b_i is calculated as the average size of orders sent to the market in the $(i-1)$-th sub-period. Therefore, the low-level policy only needs to set the limit price for each sub-order. Whenever the current sub-order is completed or canceled due to timeout (5 seconds in this work), the agent decides the price of the next sub-order and sends it to the market, until the target volume is executed. In this way, we achieve a continuous-time order placing.

3.1 The High-Level MDP for Volume Scheduling

State. The state vector utilized by the high-level agent at the beginning of the i-th subperiod is defined as $\boldsymbol{s}_i^h = (\boldsymbol{m}_i^h, \boldsymbol{p}_i^h)$, where $\boldsymbol{m}_i^h = (\boldsymbol{h}_{i\Delta t-L\tau}, ..., \boldsymbol{h}_{i\Delta t-(L-1)\tau}, ..., \boldsymbol{h}_{i\Delta t})$ denotes the market state containing the market representations derived from the SHMM (Section 4.1), with τ denoting the sample step of the look-back window in SHMM. The private state $\boldsymbol{p}_i^h = (z_i^h, t^h)$ involves the remaining inventory ratio $z_i^h = I_i^h/Q$ and remaining time $t^h = 1 - i\Delta t/T$. Here Q is the total volume, and I_i^h is the remaining inventory. The reasons to including these private variables are: With ample time and less remaining volume, the agent is more inclined to employ a conservative strategy, awaiting superior trading opportunities. Conversely, in scenarios with less time and substantial remaining volume, the agent may opt for more aggressive strategies to ensure a faster turnover.

Action. The action of the high-level agent is the proportion of the total volume to be executed in this sub-period, defined as $a_i^h \in [0, \min(c, z_{i\Delta t}^h)]$, where $0 < c < 1$ represents the proportion of the maximum allowable execution rate in a sub-period. The value of c is prefixed according to the customer's preference. Furthermore, we impose a constraint that while $a_i^h < 0.001$, a_i^h is set to 0. This constraint helps prevent excessively small slices.

Reward. To gain long-term price advantages, we formulate the high-level reward as a volume-weighted price advantage over VWAP: $r_i = \sum_{j\in\mathcal{O}_i} \frac{q_j}{Q}\left(\frac{p_j - \tilde{p}}{\tilde{p}}\right)$, where $\mathcal{O}_i$ is the set of orders traded in the i-th sub-period $\left[i\Delta t, (i+1)\Delta t\right)$, p_j and q_j are the price and quantity of the j-th sub-order in $\mathcal{O}_i$ with $\sum q_j \le a_i^h$, and $\tilde{p}$ is the global VWAP.

3.2 The Low-Level MDP for Order Placing

State. The low-level agent adopts the market representations generated from SHMM at time t within the i-th sub-period as the market state $\boldsymbol{m}_t^l = (\boldsymbol{h}_{t-L\tau}, ..., \boldsymbol{h}_{t-(L-1)\tau}, ..., \boldsymbol{h}_t)$. The private states of the remaining time and volume can be defined as $(t - i\Delta t)/\Delta t$ and I_t^l/a_i^h respectively. Notably, since the total volume of the i-th sub-period is allocated by the high-level policy, we add a new task-aware private variable $u_t = a_i^h/b_i$ that implies the task urgency w.r.t. the target volume to learn order placing policies with different goals.

Action. Each low-level action corresponds to a limit order decision with the target price $a_t^l = p_t^{limit}$. It is a discrete variable $a_t^l \in \{-\frac{n^p-1}{2}, ..., -1, 0, 1, ..., \frac{n^p-1}{2}\}$ which represents a price level with a_t^l ticks below the best ask price.

Reward. The reward function of the low-level policy is a volume-weighted price advantage over sub-period VWAP $\tilde{p}^i$: $r_t = \frac{b_i}{a_i^h}\left(\frac{p_t - \tilde{p}^i}{\tilde{p}^i}\right)$, where p_t is the actual trading price of this executed order and $p_t \ge a_t^l$.

4 Methodology

In this section, we introduce an HRL framework named MacMic designed for optimal trade execution. An overview of MacMic is depicted in Figure 3. The section 4.1 elaborates on a stacking hidden Markov model (SHMM) to extract effective multi-granularity representations for downstream policy learning. Subsequently, an HRL approach incorporated with IL techniques is proposed in Section 4.2.

4.1 Multi-granularity State Modelling

Although pre-training effective market representations via supervised learning shows great promise in other quantitative trading tasks, it is not a good choice for optimal execution. Since the agent must capture price advantages throughout the day as well as micro changes within a few seconds on the limited order books, we have no idea about the suitable prediction window. In this subsection, we develop a stacking hidden Markov model (SHMM) to extract effective multi-granularity representations in an unsupervised learning manner.

The hidden Markov model (HMM) is famous for effective hidden state discovery and volatility handling. Most studies employ first-order HMM for financial time series, which constrains HMM's fitting capability in complex systems. To break this bottleneck, SHMM stacks the high-order HMMs parameterized by neural networks (NNs) in depth.

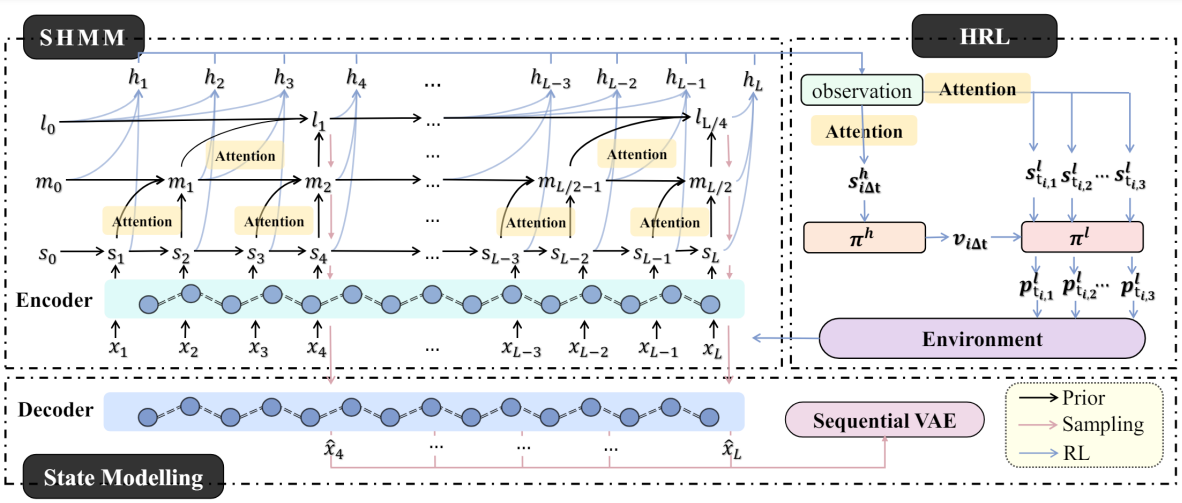


Figure 3: An overview of the proposed MacMic framework.

Unlike the first-order HMM, the high-order HMM developed in SHMM considers the dependency of observation not only on the present state but also on the preceding $n-1$ states. SHMM takes a L-step lookback window (with time interval τ) of the collected macro- and micro-level market information as input, denoted as $(\boldsymbol{x}_1, ..., \boldsymbol{x}_L)$, and outputs effective market representations $(\boldsymbol{h}_1, ..., \boldsymbol{h}_L)$. Let $\{p_\psi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn})\}_{j\le\lfloor N/n\rfloor}$, $q_\phi(\boldsymbol{i}_{j\le\lfloor N/n\rfloor}|\boldsymbol{o}_{1:n:N})$, and $p(\boldsymbol{o}_{jn}|\boldsymbol{i}_j)$ denote the prior models, posterior models, and generative models respectively. $\boldsymbol{i}$ represents the hidden state and $\boldsymbol{o}$ represents the observation for each layer; N represents the sequence length of the hidden states, and n refers to the order of temporal dependence within the HMM. Following Causal Markov Condition [Neuberg, 2003], the joint distribution of observations and states could be factorized as

$$p(\boldsymbol{i}_{1:\lfloor N/n\rfloor}, \boldsymbol{o}_{1:n:N}) = \prod_{j=1}^{\lfloor N/n\rfloor} p(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}) p(\boldsymbol{o}_{jn}|\boldsymbol{i}_j).$$

In SHMM, the prior model $p_\psi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn})$ for each step j is formulated as a Gaussian distribution $\mathcal{N}(\mu_\psi(\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}), \Sigma_\psi(\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}))$ parameterized by a Gated Recurrent Unit (GRU)[Chung *et al.*, 2014] network with a time-axis attention mechanism applied to capture the high-order temporal dependency.

The posterior is expected to simulate the prior p_ψ (and p). q_ϕ is reparameterized in a similar way, given by:

$$q_\phi(\boldsymbol{i}_{1:\lfloor N/n\rfloor}|\boldsymbol{o}_{1:n:N}) = \prod_{j=1}^{\lfloor N/n\rfloor} q_\phi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}), \quad (1)$$

where $q_\phi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}) \sim \mathcal{N}(\mu_\phi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}), \Sigma_\phi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}))$. Similarly, the posteriors are parameterized by attentive GRU and two FCs for encoding hidden.

For each step j, the generative models $p_\psi(\boldsymbol{o}_{jn}|\boldsymbol{i}_j)$ are probability density functions of Gaussian distributions to reconstruct the stock observations, with two FCs outputting the mean and log-variance vectors respectively.

To train the proposed SHMM, we reformulate the objective function of a sequential VAE framework[Higgins *et al.*, 2016]. The evidence lower bound (ELBO) with $q_\phi(\boldsymbol{i}_{1:\lfloor N/n\rfloor}|\boldsymbol{o}_{1:n:N})$ as variational distribution is:

$$\mathbb{E}_{p(\boldsymbol{i}_{1:\lfloor N/n\rfloor}, \boldsymbol{o}_{1:n:N})}(\mathcal{L}_{q_\phi, p_\psi}),$$
$$\mathcal{L}_{q_\phi, q_\psi} = \left[\mathbb{E}_{q_\phi(\boldsymbol{i}_{1:\lfloor N/n\rfloor}|\boldsymbol{o}_{1:n:N})} \log\left(\frac{p(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn})}{q_\phi(\boldsymbol{i}_{1:\lfloor N/n\rfloor}|\boldsymbol{o}_{1:n:N})}\right)\right]. \quad (2)$$

Substituting the posterior and prior formulated as above, the ELBO can be written as:

$$\mathbb{E}_{p(\boldsymbol{o}_{1:n:N})}\left[\sum_{j=1}^{\lfloor N/n\rfloor} \mathcal{L}^j_{q_\phi, p_\psi},\right] \quad (3)$$

$$\mathcal{L}^j_{q_\phi, p_\psi} = \mathbb{E}_{q_\phi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn})}\left[\log(p_\psi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn})\right]$$
$$- D_{KL}(q_\phi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn}), p_\psi(\boldsymbol{i}_j|\boldsymbol{i}_{j-1}, \boldsymbol{o}_{jn-n:jn})). \quad (4)$$

The reformulated ELBO in Eq. (3) serves as the maximization objective of the state modeling auxiliary task in an unsupervised learning manner.

As shown in Figure 2, in this paper, we pre-train a 3-layer SHMM with temporal order $n = 2$ to extract three hidden variables s_t, m_j and l_i which model the short-term, middle-term, and long-term factors of observed stock variables respectively. Finally, the three kinds of hidden states are combined to output the latent variables $\boldsymbol{h}_t$ of the current state, integrating both macro and micro market information.

4.2 Policy Learning via HRL

Since the hierarchical MDP proposed in Section 3 inherently aligns with the principles of HRL, in this section, we propose our solution algorithms respectively for the two levels of MDPs within an HRL framework.

Optimize High-Level Policy

Below we start by describing the policy network utilized by the high-level agent. As illustrated in Figure 3, to focus on

useful features helpful for gaining global price advantage, we apply a temporal attention layer on the market representation $\boldsymbol{m}_i^h = (\boldsymbol{h}_{i\Delta t - L\tau}, ..., \boldsymbol{h}_{i\Delta t-(L-1)\tau}, ..., \boldsymbol{h}_{i\Delta t})$ generated by SHMM. For simplification, we use t to represent $i\Delta t$, and l to represent $l\tau$. We consider the high-level problem as a continuous control problem, and the action of the high-level policy network is formulated as:

$$
\begin{aligned}
&e_{t-l}^h = \boldsymbol{V}_e^t \tanh(\boldsymbol{W}_1[\boldsymbol{h}_{t-l}; \boldsymbol{h}_t]), \forall l \in \{1:L\},\\
&\alpha_{t-l}^h = \frac{\exp(e_{t-l}^h)}{\sum_{j=0}^{L}\exp(e_{t-j}^h)},\\
&a_t^h = c \cdot \text{sigmoid}\Big(W_2 \cdot \text{concat}\big(\sum_{l=0}^{L}\alpha_{t-l}^h \cdot \boldsymbol{h}_{t-l}, \boldsymbol{p}_t^h\big) + \boldsymbol{b}_t^h\Big),
\end{aligned}
\tag{5}
$$

where $\boldsymbol{V}_e$, $\boldsymbol{W}_1$, $\boldsymbol{W}_2$ and $\boldsymbol{b}_t$ are free parameters, α_t is the normalized attention weight output by the temporal attention module, $\boldsymbol{p}_t^h$ is the high-level private state, and c is the predefined action limitation.

We utilized the actor-critic RL framework [Konda and Tsitsiklis, 1999], where the critic evaluates the action taken by the actor by computing the value function, and the actor (policy) is optimized to maximize the value output by the critic. To improve the sample efficiency, we use the off-policy actor-critic method TD3 [Fujimoto *et al.*, 2018] as the base learner, and the policy $\pi : a = \mu_\theta(s)$ is updated with the deterministic policy gradient [Silver *et al.*, 2014]:

$$
\nabla_\theta J(\theta) = \mathbb{E}_{s\sim\rho^\mu}\big[\nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s,a)\big|_{a=\mu_\theta(s)}\big], \tag{6}
$$

where Q^μ is a value function approximating the expected cumulative reward of the policy $\mu_\theta(s)$.

In Equation (6), D denotes the replay buffer collected by a behavior policy, which is generated by adding some noise to the learned policy π. Following the TD3 method [Fujimoto *et al.*, 2018], the value function Q is optimized in a twin-delayed manner with the data sampled from the replay buffer D.

Incorporation with IL techniques. Since the stochastic trading environment induces a hard exploration problem, learning with a pure RL objective in Equation (6) is extremely difficult. While human traders could usually identify a subset of feasible trading actions in specific market conditions, RL agents may engage in suboptimal actions, leading to inefficiencies and potential losses. Leveraging expert data is a favorable approach to improve the exploration efficiency of RL algorithms[Sun *et al.*, 2018; Ding *et al.*, 2018]. In hindsight, we can create a trading expert who always assigns more sell volumes at higher prices. We propose to augment the RL method with the objective of imitating the quoting behavior in an expert dataset D_E as:

$$
\pi = \arg\max_\pi \mathbb{E}_{(\boldsymbol{s},a)\sim D}\big[Q(\boldsymbol{s}, \pi(\boldsymbol{s}))\big] - \lambda \mathbb{E}_{(\boldsymbol{s},\hat{a})\sim D_E}\big[(\pi(\boldsymbol{s}) - \hat{a})^2\big], \tag{7}
$$

where λ is a scaling coefficient that balances maximizing the Q values and minimizing the behavior cloning (BC) loss. We set λ to decrease with the growth of the training steps.

Optimize Low-Level Policy

The low-level order placement is considered a discrete control problem with $n^p = 7$ discrete relative price levels. Similarly, we employ a temporal attention layer to extract useful features for low-level policy. Here we utilize the Dueling Q-Network [Wang *et al.*, 2015] to decide the price levels of the limit orders. Formally, The action value $Q^l(s^l, a^l)$ at state $s^l \in \mathcal{S}^l$ and the action $a^l \in \mathcal{A}^l$ are expressed in terms of the common state value $V^l(s^l)$ and the corresponding (state-dependent) action advantage $Adv(s^l, a^l)$:

$$
Q(s^l, a^l) = V(s^l) + \bigg(Adv(s^l, a^l) - \frac{1}{n^p}\sum_{a^{l'}\in\mathcal{A}^l} Adv(s^l, a^{l'})\bigg). \tag{8}
$$

The Q function is trained based on the one-step temporal-difference error:

$$
\hat{Q} = r + \gamma Q^-\big(s^{l'}, \arg\max_{a^{l'}} Q(s^{l'}, a^{l'})\big), \tag{9}
$$

$$
L = \mathbb{E}_{(s,a,s',r)\sim\mathcal{D}}\bigg[\frac{1}{N}\big(\hat{Q} - Q(s^l, a^l)\big)^2\bigg], \tag{10}
$$

where $\mathcal{D}$ denotes the prioritized experience replay buffer.

Training Scheme

In the general setting of HRL, the high-level policy and the low-level policy are only trained together in a single environment. However, this training scheme suffers from data insufficiency. In this work, we pre-train the low-level policy in the LOB environment of each asset with various target volumes, deriving multiple low-level policy parameters. By introducing the pre-training scheme, the low-level policy would be trained with more diverse interactions, thereby having better generalization and robustness. Following the iterative training scheme in [Nevmyvaka *et al.*, 2006], we augment the task urgency and private state repeatedly in the low-level pre-training. Specifically, we traverse the target quantity from 0 to a max target quantity q^{max}. In this way, the low-level policy is trained with the augmented private states, and thus can generalize to different subtasks assigned by the high-level policy. We also pre-train the high-level order-scheduling policy with a TWAP order-placement strategy. Finally, the high-level policy and the low-level policy are trained together.

5 Experimental Setup

We conduct experiments to investigate the following research questions. **Q1:** Does MacMic succeed in finding an order execution strategy that beats the long-duration VWAP strategy? **Q2:** Does the high-level agent in the hierarchical framework achieve effective price discovery? **Q3:** Does the low-level agent aid in further improving the performance? **Q4:** What is the effectiveness of the proposed SHMM module?

5.1 Dataset

To conduct a comprehensive evaluation of the proposed MacMic, we collected data from 200 constituent stocks within two real-world stock indexes spanning both the Chinese and US markets to construct our datasets. The **CSI100** dataset comprises 100 component stocks, providing insights into the primary state of the China A-share market. The **NASDAQ100** dataset includes the 100 largest non-financial international companies listed on NASDAQ. For our experiments, 80% of the trading days are utilized as the training dataset,

while the remaining 20% constitute the test dataset. The trading task is to sell the shares of 5% of the entire market trading volume of each asset during a 4-hour period following the market's opening every day, i.e., $T = 240$ minutes. The episode length of the high-level policy is $H = 240$, and the execution time for the low-level policy is $\Delta t = 1$ minute. Besides, we set $n_p = 7$ for the low-level policy. More implementation details are provided in supplementary materials.

5.2 Benchmarks

We compare the proposed approach with two categories of methods:

Rule-Based and Traditional Methods

- **TWAP** strategy evenly divides an order into T segments and executes an equal share quantity at each time step.
- **VWAP** strategy aims to closely align the execution price with the true market average price, allocating orders based on the empirically estimated market transaction volume from the previous 10 trading days.
- **AC** (Almgren-Chriss) model analytically determines the efficient frontier for optimal execution. We only focus on the temporary price impact to ensure a fair comparison.

RL-Based Methods

- **DDQN** (Double Deep Q-network) is a value-based RL method that adopts state engineering, proposed in [Ning *et al.*, 2018].
- **PPO** is a policy-based RL method [Lin and Beling, 2020] that utilizes PPO algorithm with a sparse reward to train an agent with a recurrent neural network for feature extraction.
- **OPD** leverages RL with policy distillation to determine the size of each sub-order and place market orders[Fang *et al.*, 2021].
- **HALOP** first automatically scopes an action subset according to the market status and subsequently chooses a specific discrete limit price from the action subset using a discrete-control agent [Pan *et al.*, 2022].

5.3 Evaluation Metrics

Every trading strategy has been evaluated by the following financial metrics in both return and risk criteria:

- **Price Advantage over VWAP (PA).** We use the average excess return over VWAP-with-market-order as the return metric, displayed in basis points (bps, 1 bp = 0.01%). Note that here VWAP is calculated with true trading volume instead of its estimation adopted in the VWAP baseline.
- **Win Ratio (WR).** WR measures the ratio of the days that the agent beats VWAP, defined as:

$$WR_T = \frac{\sum_{i\in\mathcal{O}_\mathcal{T}} \mathbb{I}(PA_i > 0)}{\sum_{i\in\mathcal{O}_\mathcal{T}} \big(\mathbb{I}(PA_i > 0) + \mathbb{I}(PA_i \leq 0)\big)}. \quad (11)$$

Methods	PA (bps)↑	PA-std (bps)↓	WR↑	GLR↑	AFI↓
TWAP	-0.12	**3.12**	0.49	0.97	**0**
VWAP	-3.29	4.87	0.39	0.78	0.04
AC	-1.24	4.21	0.45	1.02	0
DDQN	-1.21	7.09	0.47	0.99	0.05
PPO	-0.98	7.01	0.52	0.92	0.03
ODP	0.32	6.78	0.53	1.07	0.10
HALOP	2.89	6.12	0.65	1.12	0.07
MacMic	**3.31**	6.23	**0.72**	**1.37**	0.02

Table 2: The comparison results of the proposed method and the benchmarks on CSI100.

Methods	PA (bps)↑	PA-std (bps)↓	WR↑	GLR↑	AFI↓
TWAP	-0.19	4.31	0.49	1.01	**0**
VWAP	-2.89	4.72	0.35	0.79	0.02
AC	-2.31	4.39	0.39	0.96	0
DDQN	-1.98	6.89	0.45	0.97	0.03
PPO	-1.20	6.79	0.46	0.96	0.04
ODP	1.03	6.82	0.54	1.05	0.12
HALOP	2.75	5.87	0.63	1.13	0.05
MacMic	**3.14**	5.69	**0.74**	**1.28**	0.01

Table 3: The comparison results of the proposed method and the benchmarks on NASDAQ100.

- **Gain-Loss Ration (GLR).** GLR reports the gain-loss ratio, defined as:

$$GLR_T = \frac{\mathbb{E}(PA|PA > 0)}{\mathbb{E}(PA|PA < 0)}. \quad (12)$$

- **Averaged Final Inventory (AFI).** AFI refers to the averaged remaining inventory at the end of the trading period, evaluating the non-fulfillment risk.

6 Results and Analysis

6.1 Comparison Results

To answer the question **Q1**, we compare the proposed MacMic with seven baselines on five meaningful financial metrics. The comparison results on the US market and China A-share markets are listed in Table 2 and 3 respectively. For a fair comparison, we tune the hyper-parameters of these methods for the maximum PA value on the validation dataset.

As demonstrated in Table 2, on the CSI100 dataset, the proposed method significantly outperforms the seven commonly used baselines, including the strongest state-of-the-art RL-based baseline HALOP. It achieves the highest price advantage (3.31 bps) as well as a fascinating winning ratio (0.72), albeit with the smallest AFI (0.02) in comparison to the other four RL-based methods. Notice that in contrast to the three flat RL benchmarks, the strongest baseline, HALOP, employs a continuous policy to initially determine the range of price levels and subsequently employs a discrete agent to select the price level. Benefiting from the price trend prediction module, HALOP achieves significant improvements in these metrics. Unlike HALOP, whose two agents follow the same decision frequency, the proposed MacMic operates on multiple levels of granularity. Consequently, MacMic is more capable of identifying favorable trading opportunities from a longer-term perspective.

	High-level	Low-Level	SHMM	Imitation	AP (bps) ↑	WR ↑
(1)		✓			-8.57	0.23
(2)	✓				-7.29	0.28
(3)	✓	✓			-5.26 (+2.03)	0.35 (+0.07)
(4)		✓	✓		-2.96 (+4.33)	0.43 (+0.15)
(5)	✓	✓	✓		1.36 (+8.65)	0.43 (+0.15)
(6)	✓		✓	✓	2.29 (+9.58)	0.65 (+0.37)
(7)	✓	✓	✓	✓	**3.31 (+10.60)**	**0.72 (+0.34)**

Table 4: Ablation studies over different MacMic components.

Similarly, on the NASDAQ100 dataset in Table 3, MacMic significantly improved the PA, WR, and AFI metrics (14%, 10%, and 70% respectively) compared to the strongest baseline, HALOP. The comparison results across these 200 stocks with various baselines validate the effectiveness of the proposed multi-granularity HRL framework.

6.2 Ablation Study

To further investigate the effectiveness of the proposed model components, we compare the proposed approach with its variations summarized in Table 4. In Table 4, a "✓" symbol in the "High-level" column represents that for volume scheduling MDP, whether the proposed high-level RL policy is employed; a "✓" symbol in the "Low-level" column represents whether to use the proposed low-level policy or a TWAP-with-market-order strategy for order placement MDP; the "SHMM" symbol represents whether to adopt the SHMM to extract multi-granularity representations; and the "Imitation" column indicates whether IL techniques are incorporated into high-level policy learning.

Superiority of HRL against flat RL. By comparing the first three rows of the results listed in Table 4, we can find that directly applying an HRL method brings a slight improvement against the two basic baselines that only leverage flat RL methods. However, without the help of effective representations and IL techniques, all these strategies fail to beat the daily VWAP.

The effectiveness of SHMM. Comparing the results of the third row with those of the fifth row, we can observe significant improvements (6.62 bps on PA and 0.08 on WR) brought by the SHMM. Therefore, answering **Q4**, we can conclude that the SHMM plays a vital role in providing effective multi-granularity representations.

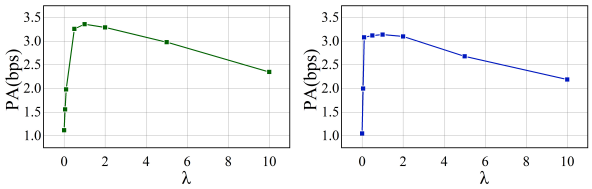


Figure 4: Sensitivity of λ on CSI100 (left) and NASDAQ100 (right).

The Effectiveness of IL. Comparing the results listed in the fourth and sixth rows in Table 4, we can find that the high-level policy is quite important in improving the price advantages of the HRL strategy. To further investigate the effectiveness of IL in high-level policy learning, we compare the results in the fifth row and the seventh row of Table 4. We can find the incorporation of IL techniques brings an improvement of 1.59 bps on PA and 0.22 on WR. The IL techniques greatly empower the price discovery ability of the high-level policy.

Furthermore, we depict the parameter sensitivity curve of the IL coefficient λ in Figure 4. Compared to MacMic trained with λ values significantly below 0.1 (near no-imitation) or above 3 (imitation-dominant), MacMic trained with $\lambda = 1$ achieves more favorable performance. This phenomenon implies the superiority of the mixture of the RL objective and the IL objective. Due to the mutual influence between volume scheduling and order placement, directly imitating the hindsight expert in high-policy learning may lead to sub-optimal trade decisions.

6.3 Discussion

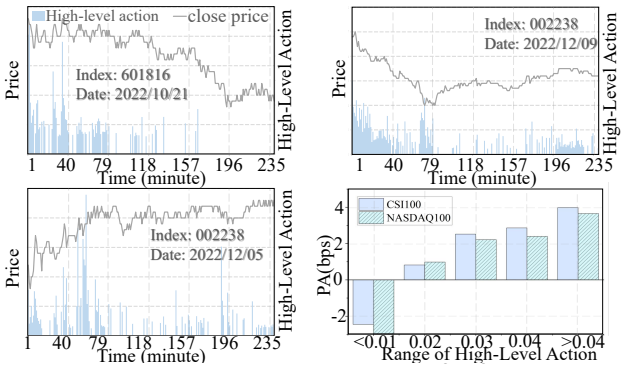


Figure 5: The price advantages of different ranges of high-level actions and some visualization results of the high-level policy.

Performance of High-level Policy. To further examine whether the high-level policy possesses good price discovery ability, we divided the action space into five bins and calculated the average PA for each bin relative to the entire market. As depicted in Figure 5, with the increasing value of the high-level action (the allocated volume), the PA of the corresponding sub-period is increasingly Improved. Besides, high-level policy performs well under different market conditions. Consequently, we can conclude that the high-level policy has effectively acquired the skill of price discovery, answering **Q2**.

Performance of Low-level Policy. Comparing the results in the last two rows of Table 4, we observe that the low-level policy further enhances the profitability of the high-level policy, answering **Q3**.

7 Conclusion

This paper formulates optimal execution as a hierarchical MDP and proposes an HRL framework termed MacMic to tackle challenging long-duration execution with a dual-stage framework. It first employs a SHMM to distill multi-granularity market representations. It then leverages a low-level agent to determine the price for order placement, and a high-level policy to split volume in coarse granularity on top of the low-level policy. Extensive experiments have demonstrated the effectiveness of the proposed method.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. This work was supported by the National Natural Science Foundation of China (Grant No.62306088 and Grant No.62161146004) and Songjiang Lab (Grant No.SL20230309).

Contribution Statement

Author 1 (First Author): Conceptualization, Methodology, Software, Investigation, Formal Analysis, Writing (Original Draft); **Author 2 (First Author)**: Data Curation, Methodology, Software, Visualization, Writing (Original Draft); **Author 3 (Corresponding Author)**: Methodology, Funding Acquisition, Resources, Supervision, Writing (Review & Editing).

References

[Almgren and Chriss, 2001] Robert Almgren and Neil Chriss. Optimal execution of portfolio transactions. *Journal of Risk*, 3:5–40, 2001.

[Bellemare *et al.*, 2017] Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 2017.

[Bertsimas and Lo, 1998] Dimitris Bertsimas and Andrew W Lo. Optimal control of execution costs. *Journal of financial markets*, 1(1):1–50, 1998.

[Chen *et al.*, 2022] Di Chen, Yada Zhu, Miao Liu, and Jianbo Li. Cost-efficient reinforcement learning for optimal trade execution on dynamic market environment. In *Proceedings of the Third ACM International Conference on AI in Finance*, pages 386–393, 2022.

[Chung *et al.*, 2014] Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv*, abs/1412.3555, 2014.

[Ding *et al.*, 2018] Yi Ding, Weiqing Liu, Jiang Bian, Daoqiang Zhang, and Tie-Yan Liu. Investor-imitator: A framework for trading knowledge extraction. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[Esser and Mönch, 2007] Angelika Esser and Burkart Mönch. The navigation of an iceberg: The optimal use of hidden orders. *Finance Research Letters*, 4(2):68–81, 2007.

[Fang *et al.*, 2021] Yuchen Fang, Kan Ren, Weiqing Liu, Dong Zhou, Weinan Zhang, Jiang Bian, Yong Yu, and Tie-Yan Liu. Universal trading for order execution with oracle policy distillation. In *AAAI Conference on Artificial Intelligence*, 2021.

[Fujimoto *et al.*, 2018] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[Hendricks and Wilcox, 2014] Dieter Hendricks and Diane Wilcox. A reinforcement learning extension to the almgren-chriss framework for optimal trade execution. *2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*, pages 457–464, 2014.

[Higgins *et al.*, 2016] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2016.

[Huberman and Stanzl, 2005] Gur Huberman and Werner Stanzl. Optimal liquidity trading. *Review of finance*, 9(2):165–200, 2005.

[Kakade *et al.*, 2004] Sham M Kakade, Michael Kearns, Yishay Mansour, and Luis E Ortiz. Competitive algorithms for vwap and limit order trading. In *Proceedings of the 5th ACM conference on Electronic commerce*, pages 189–198, 2004.

[Konda and Tsitsiklis, 1999] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

[Lin and Beling, 2019] Siyu Lin and Peter A Beling. Optimal liquidation with deep reinforcement learning. In *Proceedings of the 33rd Conference on Neural Information Processing Systems, Deep Reinforcement Learning Workshop, Vancouver, Canada*, 2019.

[Lin and Beling, 2020] Siyu Lin and Peter A. Beling. An end-to-end optimal trade execution framework based on proximal policy optimization. In *International Joint Conference on Artificial Intelligence*, 2020.

[Neuberg, 2003] Leland Gerson Neuberg. Causality: Models, reasoning, and inference, by judea pearl, cambridge university press, 2000. *Econometric Theory*, 19:675 – 685, 2003.

[Nevmyvaka *et al.*, 2006] Yuriy Nevmyvaka, Yi Feng, and Michael Kearns. Reinforcement learning for optimized trade execution. *Proceedings of the 23rd international conference on Machine learning*, 2006.

[Ning *et al.*, 2018] Brian Ning, Franco Ho Ting Ling, and Sebastian Jaimungal. Double deep q-learning for optimal execution. *Applied Mathematical Finance*, 28:361 – 380, 2018.

[Pan *et al.*, 2022] Feiyang Pan, Tongzhe Zhang, Ling Luo, Jia He, and Shuoli Liu. Learn continuously, act discretely: Hybrid action-space reinforcement learning for optimal execution. In *International Joint Conference on Artificial Intelligence*, 2022.

[Silver *et al.*, 2014] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

[Srinivas *et al.*, 2020] A. Srinivas, Michael Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *ArXiv*, abs/2004.04136, 2020.

[Stooke *et al.*, 2020] Adam Stooke, Kimin Lee, P. Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. *ArXiv*, abs/2009.08319, 2020.

[Sun *et al.*, 2018] Wen Sun, J. Andrew Bagnell, and Byron Boots. Truncated horizon policy search: Combining reinforcement learning & imitation learning. *ArXiv*, abs/1805.11240, 2018.

[Wang *et al.*, 2015] Ziyun Wang, Tom Schaul, Matteo Hessel, H. V. Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *International Conference on Machine Learning*, 2015.

[Wang *et al.*, 2021] Rundong Wang, Hongxin Wei, Bo An, Zhouyan Feng, and Jun Yao. Commission fee is not enough: A hierarchical reinforced framework for portfolio management. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 626–633, 2021.

[Watkins and Dayan, 1992] Christopher Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.