Multi-Agent Deep Reinforcement Learning for Liquidation Strategy Analysis

Reproduced by Janet Wang

December 5, 2020

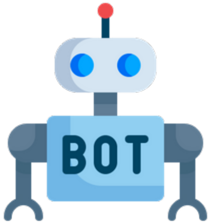
UNIVERSITY of VIRGINIA

Motivation

- Liquidation is the sale of stock shares by financial institutions to **minimize implementation shortfall** and **manage risk level** for investors wanting to cash out
- Implementation shortfall is the difference between decision price and the final execution price of the trade. Since market price drops during liquidation, a trading cost occurs.
- Liquidation of large number of stock shares would have **huge impact on the dynamic market**, making the environment difficult to predict
- **Multiple organizations** may want to liquidate their assets under certain market conditions **at the same time**

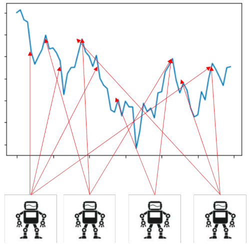
In the past, banks rely on experienced traders to minimize trading cost

Gomber et al., 2011
Brogaard et al., 2010
Sutton & Barto, 2018…



Single-agent RL is widely discussed but ignores the stochastic and interactive nature of the market

This paper



Multi-agent RL can adapt to high market complexity and derive more practical liquidation strategies

Background I: Liquidation Setup

- **Optimal Liquidation Problem:**
  - Agent$_j$ at risk aversion level λ_j aims to sell X_j shares within time frame T in N trades, implementation shortfall incurs as market price P drops
  - Our goal is to find a trajectory x_t of remaining shares $[x_{j,1}, x_{j,2}, \ldots, x_{j,T}]$at each time step $k = 1, .., T$ that minimizes implementation shortfall $E\left(X_j\right)$

- **Trading environment Simulation - Almgren and Chriss, 2001**
  - The model returns price information when agents sell at every step k
  - Price = unaffected price process, temporary, and permanent impact

$$P_k = P_{k-1} + \sigma\tau^{1/2}\xi_k - \tau g(\frac{n_k}{\tau}), k = 1, \ldots, N$$

Discrete arithmetic random walk

Market impact

  - σ: volatility of stock
  - $g(v)$: linear function of average rate of trading
  - $\tau = N/T$: Time interval
  - n_k: Number of shares to sell; N :Total number of trades

Background II: Liquidation as a MDP

- **MDP Problem definition:**
  - **State $\boldsymbol{s} = [\boldsymbol{r}, \boldsymbol{m}, \boldsymbol{l}]$**
    - k - current step
    - $r_k = \log(\frac{P_k}{P_{k-1}})$ – log return at time t_k
    - $m_k = \frac{N_k}{N}$ - number of trades remaining normalized
    - $l_{J,k} = \frac{x_{j,k}}{X_j}$ - number of shares remaining for agent J normalized
    - _Start state_: $s = [0, m, 1]$, _Terminal state_: $s = [r, 0, 0]$
  - **Action $\boldsymbol{a_k}$**: selling fraction remaining shares between 0 and 1
  - **Reward $\mathbf{R}(\boldsymbol{s}, \boldsymbol{a})$**: difference between two utility functions of the agent at t and $t+1$ since Markov decision process only dependent on current state
    - **Risk aversion** λ, **Trading trajectory** (vector of shares left) x at k

$$U(\boldsymbol{x}) = E(\boldsymbol{x}) + \lambda V(\boldsymbol{x}), \quad (1) \qquad R_t = U_t(\boldsymbol{x}_t^*) - U_{t+1}(\boldsymbol{x}_{t+1}^*).$$

$$E(\boldsymbol{x}) = \sum_{k=1}^{N} \tau x_k g\left(\frac{n_k}{\tau}\right) + \sum_{k=1}^{N} n_k h\left(\frac{n_k}{\tau}\right), \quad (2)$$

$$V(\boldsymbol{x}) = \sigma^2 \sum_{k=1}^{N} \tau x_k^2, \quad (3)$$

Expected implementation shortfall

Function of trading trajectory

Cost of trading factored in here

  - **Policy $\boldsymbol{\pi}(\boldsymbol{s})$**: the distribution of selling percentage a_k at state s

Background II (cont'd)

- **MDP goal:**
  - minimizes implementation shortfall $E(X_j)$
  - defined as cost function with fixed and variable proportions
  - We want an optimal sequence of actions $[a_{j,1}, a_{j,2}, \ldots, a_{j,T}]$, or an optimal sequence of policies $\pi(s)$ from MDP
  - This sequence of actions can translate into trading trajectory (vector of remaining shares) $[x_{j,1}, x_{j,2}, \ldots, x_{j,T}]$ that solves the optimal liquidation problem defined earlier

Related Work

- **Yang et al., 2018a**
  - Reinforcement learning algorithms
- **Almgren & Chriss, 2001**
  - Problem of optimal liquidation strategy is investigated by the impact model that agents liquidate assets completely in a given time frame
  - Impact of the stock market is divided into three components
    - Unaffected price process
    - Permanent impact
    - Temporary impact
- **Other researches related to deep reinforcement learning algo**
  - Omidshafiei at., 2017
  - Mnih et al., 2016; Lowe et al., 2017
  - Lillicrap et al., 2016
  - Wang et al., 2016

Target Task

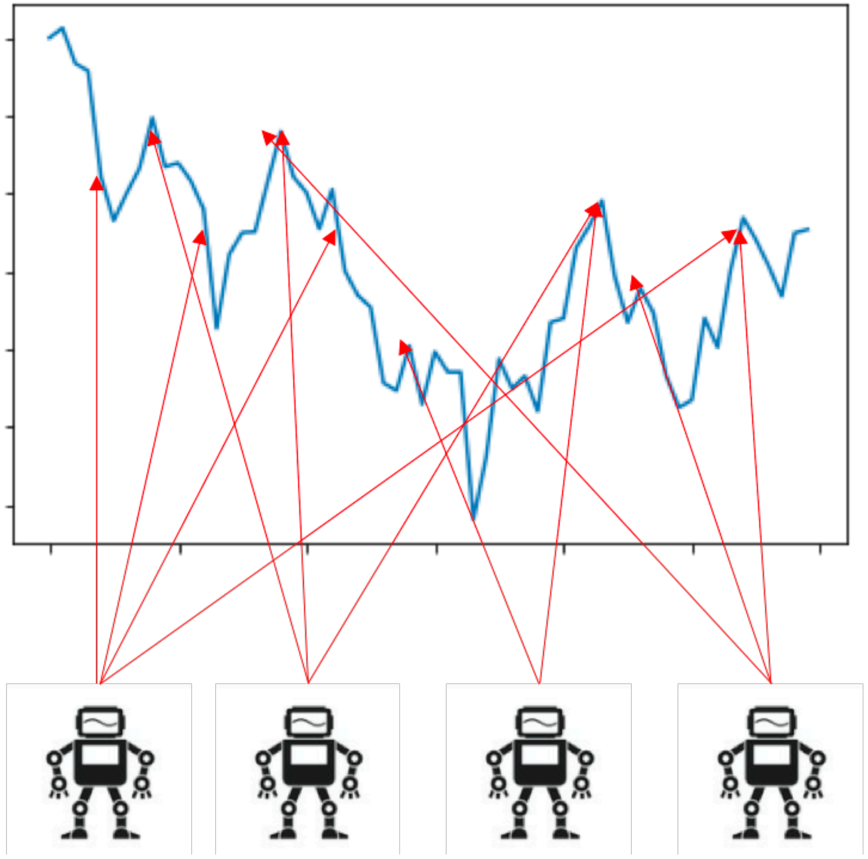
1. Extend to **multi-agent trading environment**

2. **Prove two theorems** to conclude with necessity of using multi-agent RL

3. Analyze how **cooperative and competitive relationship** between two trading agents would affect implementation shortfall

4. Derive an **optimal liquidation strategy** for a known agent

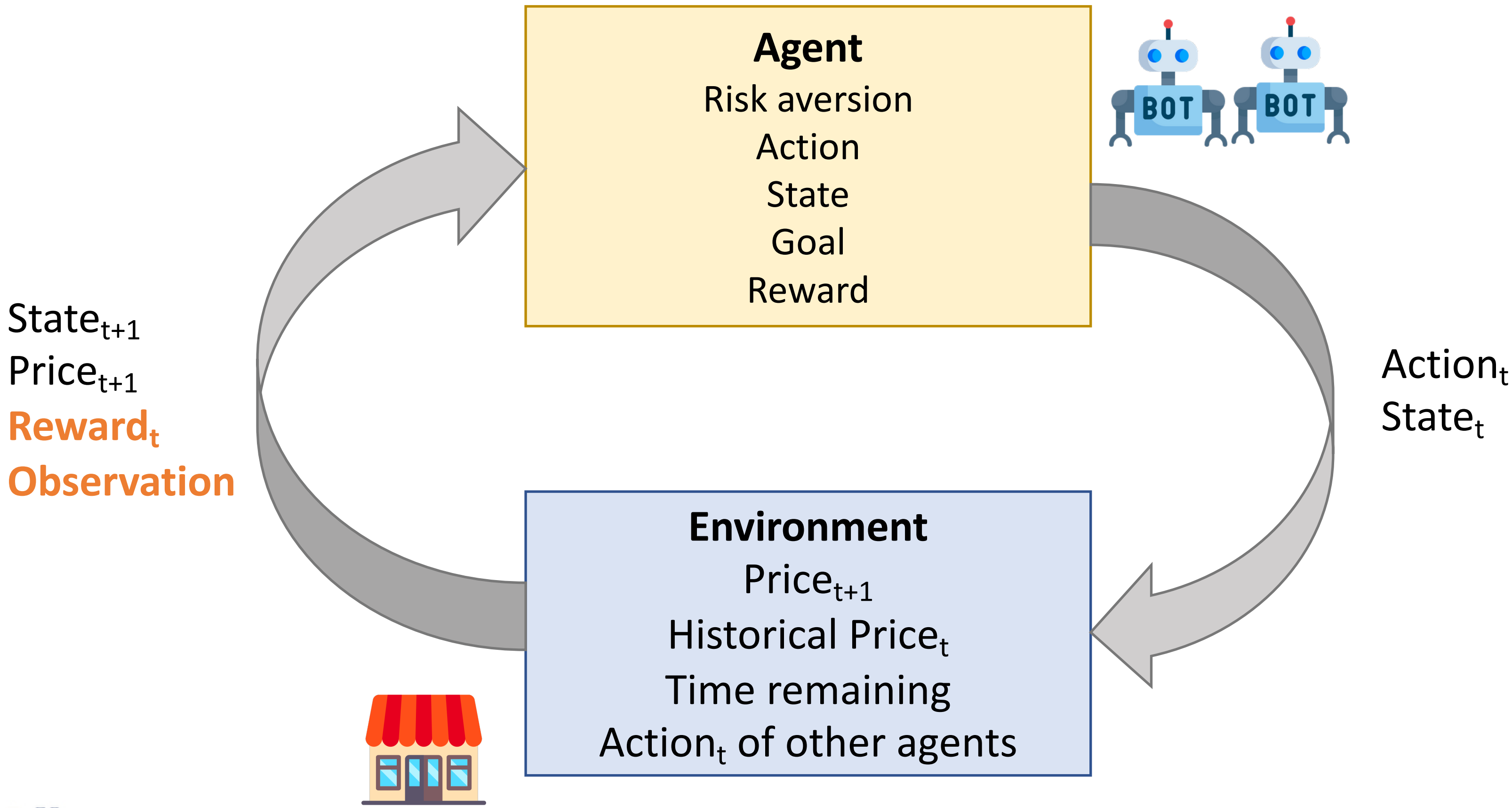
Multi-agent RL is necessary to derive trading strategy



Proposed Solution Multi-agent

- **Extend Single-agent reinforcement learning to multi-agent setting**

For 1 episode



Proposed Solution (cont'd)

- **Multi-agent reinforcement learning setting**
  - In a J-agents environment, the **State** vector $s = [r, m, l]$ at time t_k would be $[r_{k-D}, \ldots., r_{k-1}, r_k, m_k, l_{1,k}, \ldots l_{J,k}]$
  - **Action:** $n_{J,k}$ is number of shares to sell for each at each time step using:
    - $n_{J,k} = a_{J,k} \times x_{J,k}$
    - $x_{J,k}$ = shares remaining after sale at time t_k for agent j
  - **Reward** $R_{j,k}(s, a)$ denotes optimal trading trajectory for agent J:
    - $R_{J,k} = U_{J,t}(x *_{j,t}) - U_{J,t+1}(x *_{j,t+1})$
  - **Policy** $\pi(s)$ is the distribution of selling percentage a at state s
  - **Q-value** $Q_\pi(s, a)$ is the expected reward achieved by following policy π
  - **Observation** O: each agent only observes environment and itself
    - $O_{J,k} = [r_{k-D}, \ldots., r_{k-1}, r_k, m_k, l_{1,k}, \ldots l_{J,k}]$

Implementation: Actor-Critic in DDPG

- **Actor-Critic Model – Mnih and Lowe, 2017**
  - Uses neural network to approximate both Q-value and the action
  - The **critic** learns the Q-value function and updates policy parameters
  - The **actor** brings the advantage of computing continuous actions without the need of a Q-value function
  - The critic supplies actor knowledge of performance
  - The method has good convergence properties

Algorithm 1 DDPG-Based Multi-agent Training

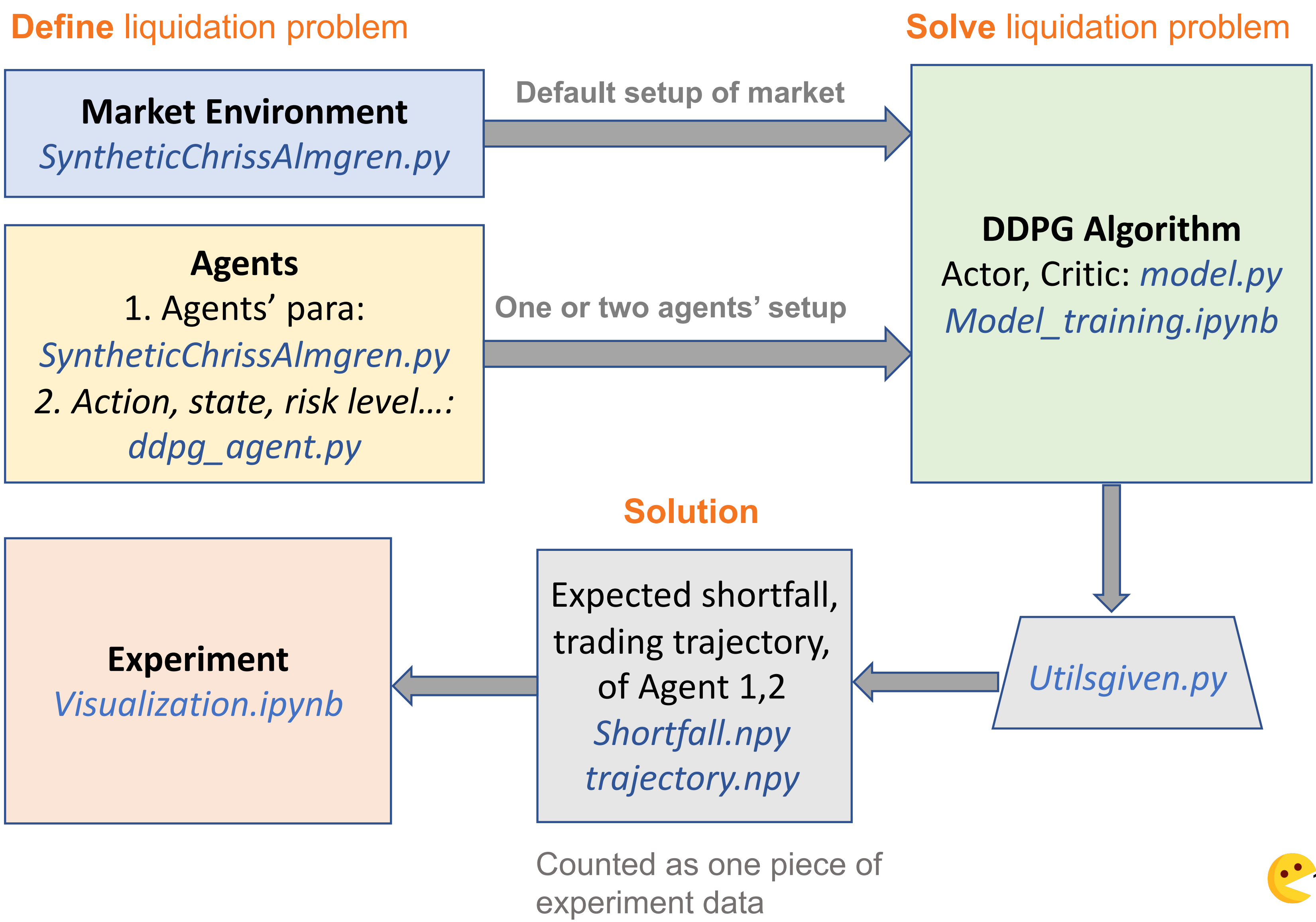
Input: number of episodes M, time frame T, minibatch size N, learning rate λ, and number of agents J

for $j = 1, J$ % initialize each agent separately **do**
  Randomly initialize critic network $Q_j(O_j, a|\theta_j^Q)$ and actor network $\mu_j(O_j|\theta_j^\mu)$ with random weight θ_j^Q and θ_j^μ for agent j;
  Initialize target network Q_j' and μ_j' with weights $\theta_j^{Q'} \leftarrow \theta_j^Q$, $\theta_j^{\mu'} \leftarrow \theta_j^\mu$ for each agent j;
  Initialize replay buffer B_j for each agent j;
end for
for episode $= 1, M$ **do**
  Initialize a random process $\mathcal{N}$ for action exploration;
  Receive initial observation state s_0;
  for $t = 1, T$ **do**
    for $j = 1, J$ %train each agent separately **do**
      Select action $a_{j,t} = \mu_j(O_{j,t}|\theta_j^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise;
    end for
    Each agent executes action $a_{j,t}$;
    Market state changes to s_{t+1};
    Each agent observes reward $r_{j,t}$ and observation $O_{j,t+1}$;
    for $j = 1, J$ **do**
      Store transition $(O_{j,t}, a_{j,t}, r_{j,t}, O_{j,t+1})$ in B_j;
      Sample a random minibatch of N transitions $(O_{j,i}, a_{j,i}, r_{j,i}, O_{j,i+1})$ from B_j;
      Set $y_{j,i} = r_{j,i} + \gamma Q_j'(s_{t+1}, \mu_j'(O_{j,i+1}|\theta_j^{\mu'}|\theta_j^{Q'}))$ for $i = 1, \dots, N$;
      Update the critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_{j,i} - Q_j(O_{j,i}, a_{j,i}|\theta_j^Q))^2$;
      Update the actor policy by using the sampled policy gradient:

$$\nabla_{\theta^\mu} \pi \approx \frac{1}{N}\sum_i \nabla_a Q_j(O, a|\theta_j^Q)|_{O=O_{j,i}, a=\mu_j(O_{j,i})} \times \nabla_{\theta^\mu} \mu_j(O_j|\theta^\mu)|_{s_i};$$

      Update the target networks:
      $\theta_j^{Q'} \leftarrow \tau\theta_j^Q + (1-\tau)\theta_j^{Q'}$,
      $\theta_j^{\mu'} \leftarrow \tau\theta_j^\mu + (1-\tau)\theta_j^{\mu'}$.
    end for
  end for
end for

Implementation: Process Overview



Data Summary

- **Market setup**
  - Default setting for all experiments
  - Based on hypothetical assumption: total shares = 1 million
  - Initial stock price = 50, T = 60 days with N=60 trades, 1 trade/day

Financial Parameters

Annual Volatility:	12%	**Bid-Ask Spread:**	0.125
Daily Volatility:	0.8%	**Daily Trading Volume:**	5,000,000

- **Agents setup**
  - redefine the following parameters every time as we run a new experiment to get arrays of expected shortfalls and trajectories

Almgren and Chriss Model Parameters

Total Number of Shares for Agent1 to Sell:	500,000	**Fixed Cost of Selling per Share:**	$0.062
Total Number of Shares for Agent2 to Sell:	500,000	**Trader's Risk Aversion for Agent 1:**	1e-06
Starting Price per Share:	$50.00	**Trader's Risk Aversion for Agent 2:**	1e-06
Price Impact for Each 1% of Daily Volume Traded:	$2.5e-06	**Permanent Impact Constant:**	2.5e-07
Number of Days to Sell All the Shares:	60	**Single Step Variance:**	0.144
Number of Trades:	60	**Time Interval between trades:**	1.0

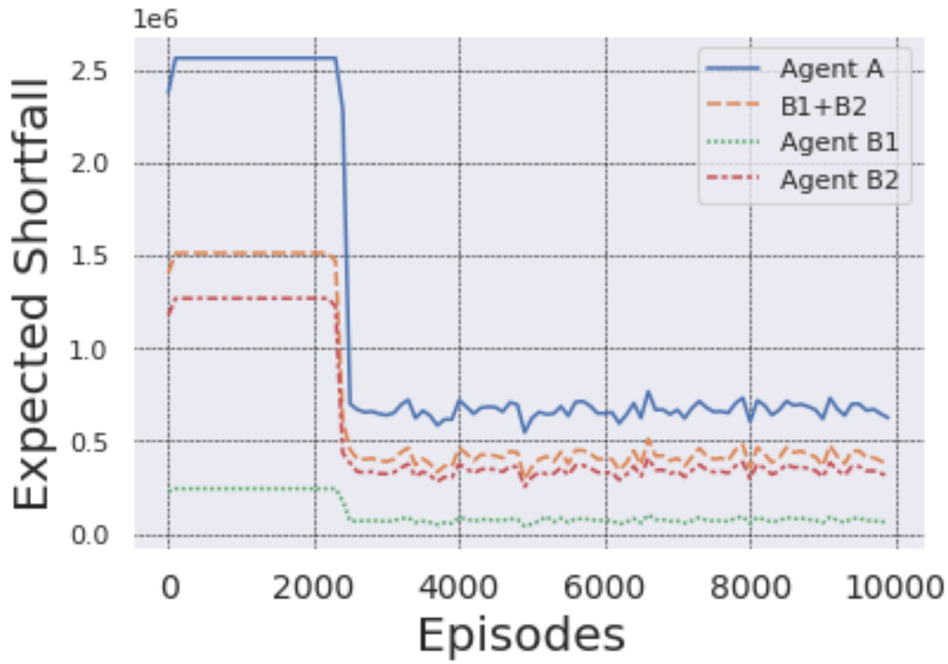
Experiment 1: Theorem I Verification

- **Theorem I definition:**
  - In a multi-agent environment with J agents where each agent has Xj shares to sell within a given time frame T, the total expected shortfall **is larger than or equal** to the sum of expected shortfall that these agents would obtain if they are in single-environment.

$$\sum_{j=1}^{J} E(X_j) \leq E(\sum_{j=1}^{J} X_j)$$

- **Experiment data used:**
  - Same risk level: $\lambda_A = \lambda_{B1} = \lambda_{B2} = 1^\wedge e^{-6}$
  - **Agent A:** 1 million shares in single-environment
  - **Agent B_1** and **Agent B_2** : liquidate 0.3 and 0.7 million shares respectively.
  - Compare average implementation shortfalls

Experimental Result 1:



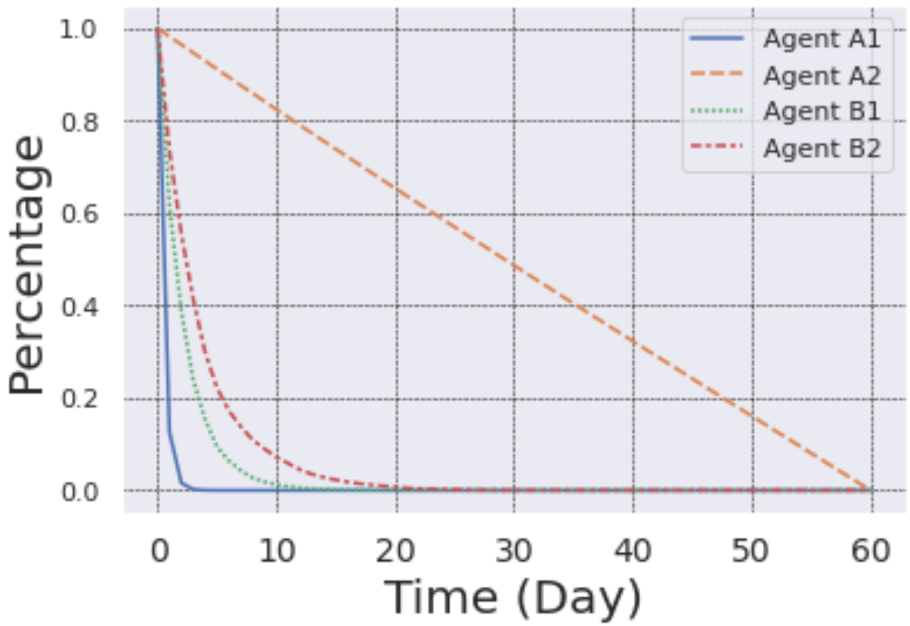
- The expected implementation shortfall $E(A)$ is larger than the sum of $E(B_1)$+$E(B_2)$
- Theorem 1 is proved.

Experiment 2: Theorem II Verification

- **Theorem II definition:**
  - In a two-agent environment where Agent 1 has risk aversion level λ_1 and Agent 2 has risk aversion level λ_2, where $\lambda_1 \neq \lambda_1$, and each of them has the same number of stocks to liquidate, the biased trajectories $x(\lambda_1)$ and $x(\lambda_2)$ would satisfy that $\boldsymbol{x^*(\lambda_1) \neq x(\lambda_1), x^*(\lambda_2) \neq x(\lambda_2)}$

- **Experiment data used:**
  - **Agent $\mathbf{A_1}$:** liquidate 1 million shares at $\lambda_{A1} = 1\text{^}e^{-4}$ in single environment
  - **Agent $\mathbf{A_2}$:** liquidate 1 million shares at $\lambda_{A2} = 1\text{^}e^{-9}$ in single environment
  - **Agent $\boldsymbol{B_1}$** and **Agent $\boldsymbol{B_2}$** liquidate 0.5 million shares each at $\lambda_{B1} = 1\text{^}e^{-4}$ and $\lambda_{B2} = 1\text{^}e^{-4}$.
  - Compare four agents' trajectories

Experimental Result 2:



- By comparing trajectories of A_1, A_2 with B_1, B_2 from the graph below, we notice that the trading trajectories of B_1, B_2 are biased
- Theorem 2 is proved.

Experimental Analysis 1 & 2

- For same amount of shares, **shortfall implementation is greater if it is liquidated by one** agent than multiple agents
- Agents in **multi-agent environment factors in extra information** from observing other agents in the environment when deciding their trading trajectory
- The **selling patterns of other agents** would affect their liquidation strategy
- This demonstrates the **necessity of using multi-agents RL to derive trading strategy**

Experiment 3: Relationship analysis

- **Goal:**
  - Adjust reward to define competitive and cooperative relationships
  - Compare the sum of expected shortfalls with expected shortfalls trained in single-agent environment to evaluate relationship

Corporation

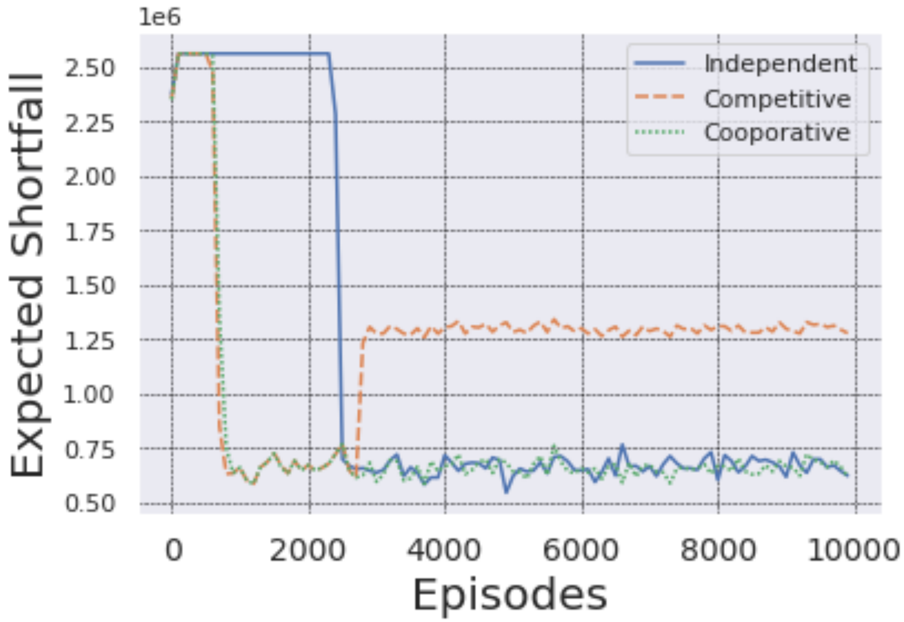
$$\tilde{R}^*_{1,t} = \tilde{R}^*_{2,t} = \frac{\tilde{R}_{1,t} + \tilde{R}_{2,t}}{2}$$

Competition

$$\begin{aligned}&\textbf{if } \tilde{R}_{1,t} > \tilde{R}_{2,t} \textbf{ then}\\ &\quad \tilde{R}^*_{1,t} = \tilde{R}_{1,t},\\ &\quad \tilde{R}^*_{2,t} = \tilde{R}_{2,t} - \tilde{R}_{1,t}\end{aligned}$$

- **Experiment data used:**
  - **Agent A:** liquidate 1 million shares at $\lambda_A = 1^\wedge e^{-6}$ in single environment
  - **Agent A$_2$**: liquidate 1 million shares at $\lambda_{A2} = 1^\wedge e^{-9}$ in single environment
  - **Agent B_1** and **Agent B_2** liquidate 0.5 million shares each at $\lambda_{B1} = \lambda_{B2} = 1^\wedge e^{-6}$ with competitive reward functions
  - Agent C$_1$ and Agent C$_1$ liquidate 0.5 million shares each at $\lambda_{C1} = \lambda_{C2} = 1^\wedge e^{-6}$ with cooperative reward functions.
  - Compare three shortfalls

Experimental Result 3:



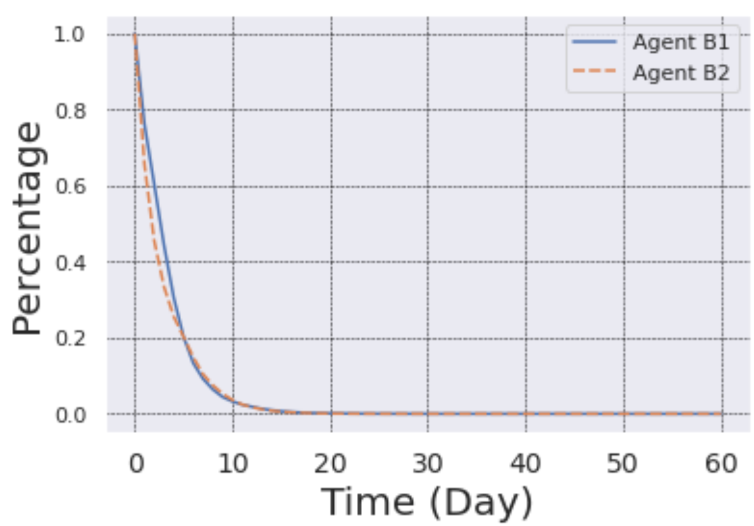
- **Two cooperative agents are not necessarily better** than independent agents training with independent rewards
- **Two competitive agents** learns to minimize expected shortfall faster than other types of agents, and then **malignant competition** leads to **significant increase in shortfall increment**

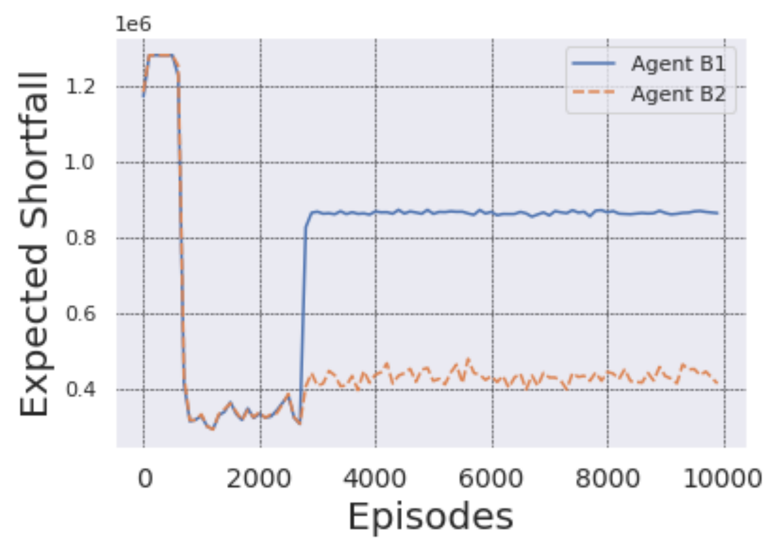
University of Virginia

Experimental Analysis 3:

We look deeper into how agents would behave in **competitive relationships**:

- **Two competitive agents** learns to sell all shares at similar pace
- Both agents perform well and roughly have same expected shortfalls but one starts to **outperform significantly at the cost of other**
- **E(X) for both agent increases**; none of them is winning





UNIVERSITY of VIRGINIA

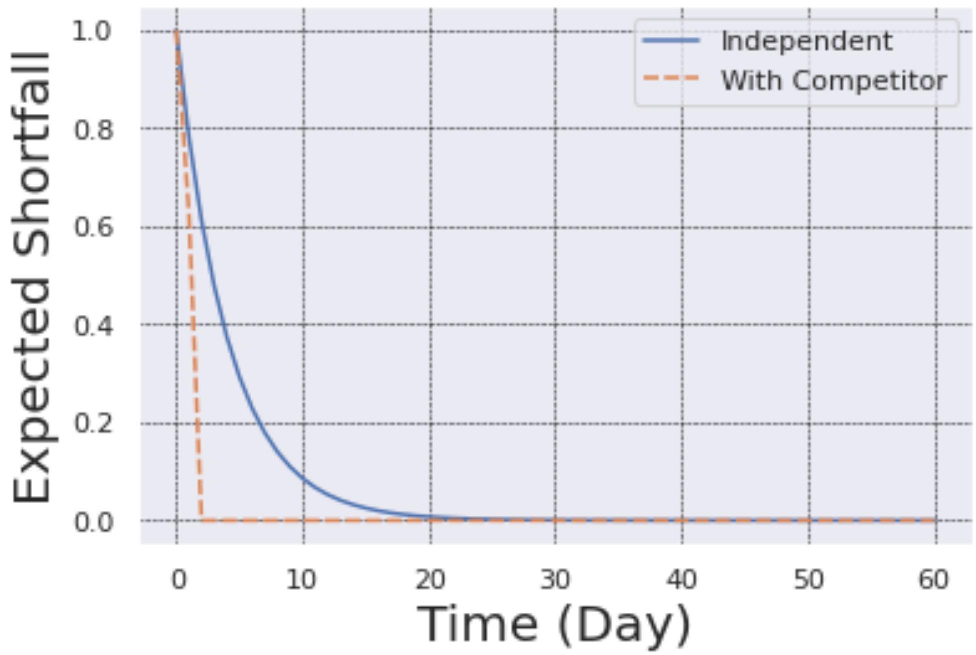
Experiment 4: Strategy Development

- **Goal:**
    - Find a trading trajectory that optimizes the expected shortfall given there are competitors in the environment

- **Experiment data used:**
    - **Agent A_1 and Agent A_2** each liquidate 0.5 million shares each at $\lambda_{A1} = 1\hat{}e^{-6}$ and $\lambda_{A2} = 1\hat{}e^{-9}$
    - **Agent A:** 1 million shares in single-environment from Experiment 1
    - Get trading trajectory of Agent A_1 and have Agent A's trading trajectory as benchmark for comparison

Experimental Result & Analysis 4:

- Without competitors, Agent A normally completes in 20 days
- With competitors, Agent A_1 normally sells in 2 days



Agents learn to **avoid taking unnecessary risk** by selling all shares in quite **a short time.**

Conclusion & Future Work

Conclusion

1. Single-agent environment **over-simplifies the dynamic** as well as the interactive nature of the stock market.
2. We extend the Almgren-Chriss model with reinforcement learning to set up the **basis of using multi-agent trading environment** to have a better analysis of expected shortfalls

Future work

1. Development of more **realistic trading environment**
2. Include more dynamic factors such as **news, general strategy and legal complaints**

Conclusion & Future Work (cont'd)

Conclusion

1. Cooperative relationship is not better than independent one
2. Competitive relationship would hurt overall and individual performance
3. Agents with competitor **liquidate faster to avoid risk**

Future work

1. Consider **optimistic bull** or **pessimistic bear**
2. Use this as an application to **predict stock price movements** after liquidation

UNIVERSITY of VIRGINIA

References

Almgren, R. and Chriss, N. Optimal execution of portfolio transactions. Journal of Risk, 3:5–40, 2001.

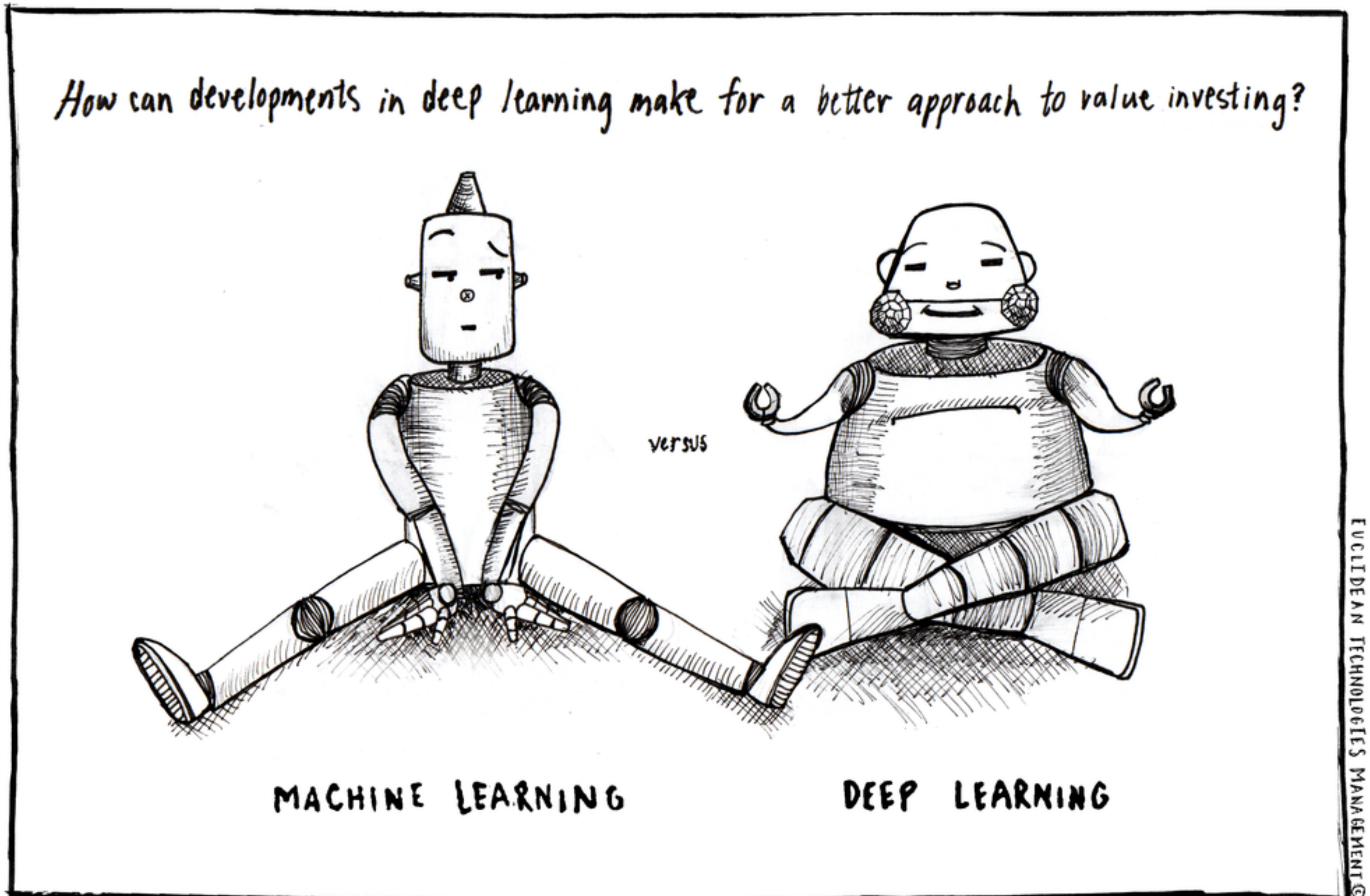
Bansal, T., Pachocki, J., Sidor, S., Sutskever, I., and Mordatch, I. Emergent complexity via multi-agent competition. arXiv preprint arXiv:1710.03748, 2017.

Brogaard, J. A., Brennan, T., Korajczyk, R., Mcdonald, R., and Vissing-jorgensen, A. High frequency trading and its impact on market quality, 2010. Buehler, H., Gonon, L., Teichmann, J., and Wood, B. Deep hedging. Quantitative Finance, pp. 1–21, 2019.

Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P. H., Kohli, P., and Whiteson, S. Stabilising experience replay for deep multi-agent reinforcement learning. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 1146–1155. JMLR. org, 2017

...

Thank you!



How can developments in deep learning make for a better approach to value investing?
versus
MACHINE LEARNING
DEEP LEARNING
EUCLIDEAN TECHNOLOGIES MANAGEMENT©