Journal of Universal Computer Science, vol. 14, no. 7 (2008), 1136-1153
submitted: 1/10/07, accepted: 21/1/08, appeared: 1/4/08 © J.UCS

Reinforcement Learning on a Futures Market Simulator

Koichi Moriyama, Mitsuhiro Matsumoto, Ken-ichi Fukui,
Satoshi Kurihara, and Masayuki Numao
(I.S.I.R., Osaka University, Japan
{koichi, m_mit, fukui, kurihara, numao}@ai.sanken.osaka-u.ac.jp)

Abstract: In recent years, market forecasting by machine learning methods has been flourishing. Most existing works use a past market data set, because they assume that each trader's individual decisions do not affect market prices at all. Meanwhile, there have been attempts to analyze economic phenomena by constructing virtual market simulators, in which human and artificial traders really make trades. Since prices in a market are, in fact, determined by every trader's decisions, a virtual market is more realistic, and the above assumption does not apply. In this work, we design several reinforcement learners on the futures market simulator U-Mart (Unreal Market as an Artificial Research Testbed) and compare our learners with the previous champions of U-Mart competitions empirically.

Key Words: reinforcement learning, market simulation
Category: I.2.6, I.6.8

1 Introduction

In recent years, market forecasting by machine learning methods has been flourishing. In reinforcement learning domain, there are several works using real market data, e.g., [O et al. 2002, Nevmyvaka et al. 2006]. Most of them divide a *past* market data set into training and test sets, learn a strategy from the training set, and then verify the result by the test set. This is because each trader's individual decisions are assumed not to affect the market at all, since there are too many traders in the market. This assumption allows a learner to learn passively; that is, a learner can learn from only a data set. In such a passive learning domain, we can use offline, batch learning methods.

Some researchers in economics are interested in virtual markets. A virtual market, in which human and/or artificial traders trade virtual stocks, provides researchers with a tool for analyzing interesting market phenomena that current economic theories cannot explain, and for dissecting the market structure itself. Since the prices in a virtual market are determined by every trader's decisions, it is more realistic, and the assumption described above does not apply. Therefore, a learner has to learn actively because the learner's strategy changes the market and vice versa. Reinforcement learning is inherently suitable for such active learning.

In this work, we design several reinforcement learners on the futures market simulator U-Mart (Unreal Market as an Artificial Research Testbed) [U-Mart, Kita 2000, Sato et al. 2001] and compare our learners with the previous champions of U-Mart competitions empirically.

This paper consists of four sections after this Introduction. [Section 2] explains what futures are and what U-Mart is. In [Section 3], we design several reinforcement learners each of which has a different state space. In [Section 4], we compare our learners with the past champions empirically. Finally, we conclude this paper in [Section 5].

2 U-Mart: a Futures Market Simulator

Futures are agreements to trade something at some time in the future (the *due date*) at the price specified now. As an example, suppose that two people agree that the buyer will buy gold at $15 per gram from the seller one year later. Such futures make it easy to plan to do something using gold one year later because neither the buyer nor the seller must be concerned with changes in the price of gold. Therefore, futures are required to manage risks in market prices, called *spot prices*. If the spot price of gold reaches $20 per gram on the due date, however, the buyer gets a gain because he/she has to pay only $15 to get $20 worth of gold. On the other hand, the buyer suffers a loss if the spot price falls to $10. Hence, futures themselves have values depending on spot prices, and therefore, they are traded at *futures prices* in a *futures market*.

U-Mart (Unreal Market as an Artificial Research Testbed) [U-Mart, Kita 2000, Sato et al. 2001] is a futures market simulator dealing with the stock index J30. J30 represents the average of stock prices of thirty large companies traded on the Tokyo Stock Exchange and was reported by the Mainichi Newspapers until 2003. Although J30 had no real futures market, traders on U-Mart trade J30 futures (*J30F*) based on J30 spot (*J30S*) prices. In a real market, futures prices generally influence spot prices. In U-Mart, however, futures prices do not affect spot prices at all because U-Mart deals only with futures trades based on existing spot prices, which are unknown to traders in advance.

In U-Mart, each trader initially has a certain amount of money and tries to maximize it by trading J30F. The entire simulation time, which corresponds to the time until the due date, is divided into several *days* and each day is divided into several *intervals*. During each interval, traders send “buy” and/or “sell” orders with the price and volume to the exchange. The exchange records them in an *order book*. At the end of the interval, the exchange plots a price-volume graph with demand and supply curves from the order book. The crossing point of the curves indicates the price and volume of being executed; that is, sell orders to the left of this point and buy orders to the right of it are executed at the corresponding price, as shown in [Fig. 1]. On the other hand, unexecuted orders remain in the order book throughout the day. See [Kita 2000] for details. Every trader whose orders have been executed changes his/her *position*, which is the difference between the amounts of futures that he/she has bought and has sold. Traders whose positions are not zero have to deposit some margin, depending on their positions, to the exchange.

At the end of the day, the exchange calculates the unrealized profit (loss) of each trader from the J30F closing price and clears it by paying (receiving) money to (from)

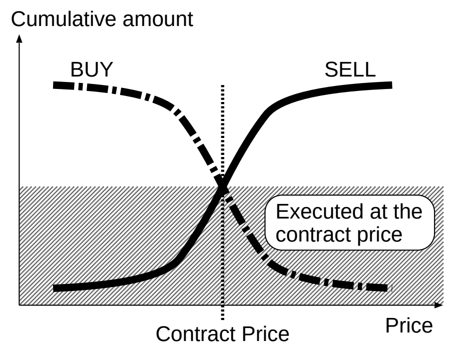


Figure 1: Demand-supply curves. Orders in the shaded area are executed at the contract price

the trader. Traders who cannot pay their unrealized losses become bankrupt and drop out of the simulation. At the end of the simulation, all positions are cleared according to the next day's first J30S price and all traders are ranked by several metrics.

3 Reinforcement Learning on U-Mart

In this section, we consider how a reinforcement learner, called an *agent*, gets gains in a U-Mart simulation. Generally, to get a gain, a trader either has to sell at a price higher than the price at which he/she bought, or has to buy at a price lower than the price at which he/she sold. Suppose that an agent sent a buy order in the t-th interval, i.e., between times $t - 1$ and t, and that the order has been executed at t. The agent gets a gain if the price at $t + 1$ becomes higher than that at t. Therefore, to get a gain, an agent has to predict the transitions of J30F price in the near future. In this work, the agent learns to predict transitions through Q-learning [Watkins and Dayan 1992], a representative reinforcement learning algorithm.

3.1 Q-learning

Suppose that an agent senses a state $s_t \in \mathcal{S}$ and selects an action $a_t \in \mathcal{A}(s_t)$ at a discrete time t. $\mathcal{S}$ is a set of possible states in the environment, and $\mathcal{A}(s_t)$ is a set of possible actions in the state s_t. After selecting an action, the agent receives a reward $r_{t+1} \in \mathbb{R}$ and senses a new state s_{t+1}. Q-learning updates an *action value function* Q by the following rules to make Q approach the true value under the optimal policy π^*, which is the expected sum of rewards discounted by $0 < \gamma < 1$ under π^*, i.e., $E_{\pi^*}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+1+k}\right)$.

$$Q_t(s, a) = \begin{cases} Q_{t-1}(s_t, a_t) + \alpha\, \delta_t & \text{if } (s, a) = (s_t, a_t), \\ Q_{t-1}(s, a) & \text{otherwise.} \end{cases} \tag{1}$$

α is a parameter called the learning rate, with $0 < \alpha \leq 1$, and δ_t is called the TD error that approaches 0 when $Q_t(s, a)$ approaches the true value of (s, a) under π^*:

$$\delta_t \triangleq r_{t+1} + \gamma \max_{a \in \mathcal{A}(s_{t+1})} Q_{t-1}(s_{t+1}, a) - Q_{t-1}(s_t, a_t). \tag{2}$$

For all s and a, $Q_t(s, a)$ is proved to converge to the true value under the optimal policy when (i) the environment has the Markov property, (ii) the agent visits all states and takes all actions infinitely, and (iii) α decreases properly [Watkins and Dayan 1992].

If the true value function under the optimal policy, Q^*, is known, the agent can choose an optimal action a^* in a state s from Q^* by

$$a^* = \arg\max_{a' \in \mathcal{A}(s)} Q^*(s, a'). \tag{3}$$

If the agent always chooses such actions during learning, however, Q_t can converge to a local optimum because the agent may not visit all states. To avoid this, the agent typically uses a stochastic method like *softmax* [Sutton and Barto 1998] to choose actions. Softmax calculates action choice probabilities $p_{s_t}(a)$, where $a \in \mathcal{A}(s_t)$, as

$$p_{s_t}(a) \triangleq \frac{\exp(Q_{t-1}(s_t, a)/T)}{\sum_{a' \in \mathcal{A}(s_t)} \exp(Q_{t-1}(s_t, a')/T)} . \tag{4}$$

T is called the temperature, with $T > 0$, and controls the effect of randomness.

3.2 Learning to Predict Price Transitions on U-Mart

As we saw in [Section 2], an agent's order, either to "sell" or "buy", consists of a price and volume. In this work, however, we only discuss how the learning agent determines whether to "sell", "buy", or "do nothing". For this purpose, the agent applies Q-learning to predict price transitions. The agent simply determines the order price from the price at $t - 1$ and the order volume from the softmax probability. Whenever an order would cause an agent's position to move across an upper or lower limit, the agent either does nothing or reduces the volume.

Here we design the following three types of state space of Q-learning:

- Transitions of J30F price as states,
- Transitions of J30S price as states, and
- Spread between J30F and J30S prices as states.

We see each type of state space in this order. Hereafter, $F(t)$ and $S(t)$ stand for "J30F price at *t*" and "J30S price at *t*", respectively. Price transition functions $\Delta F : \mathbb{N} \to \{up, same, down\}$ and $\Delta S : \mathbb{N} \to \{up, same, down\}$ are defined as follows.

$$\Delta F(t) \triangleq \begin{cases} up & \text{if } F(t) > F(t-1), \\ same & \text{if } F(t) = F(t-1), \\ down & \text{otherwise.} \end{cases} \tag{5}$$

$$\Delta S(t) \triangleq \begin{cases} \mathit{up} & \text{if } S(t) > S(t-1), \\ \mathit{same} & \text{if } S(t) = S(t-1), \\ \mathit{down} & \text{otherwise.} \end{cases} \tag{6}$$

Transitions of J30F price as states (Approaches 1–1 and 1–2)

The approaches described here use J30F price transitions up to the present (t-th interval). The agent predicts $\Delta F(t+1)$, i.e., the J30F price transition from t to $t+1$. The approaches are divided into two types by ways of prediction.

Approach 1–1:

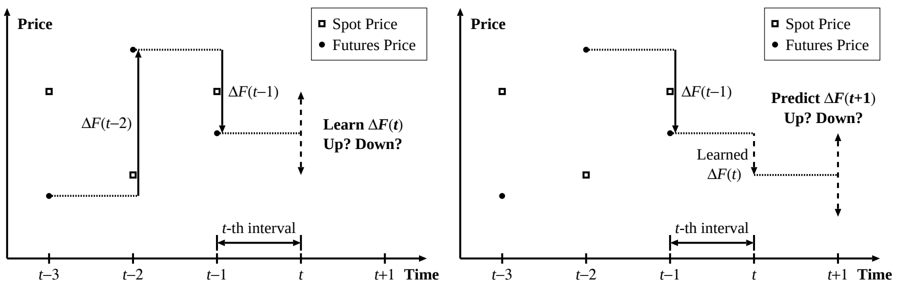
The agent learns to predict $\Delta F(t)$ from the past two J30F price transitions: $\Delta F(t-2)$ and $\Delta F(t-1)$ ([Fig. 2(a)], left). Q-function has nine *states*, each of which is a combination of two transitions — (*up*, *up*), (*up*, *same*), (*up*, *down*), etc. — and two *actions*, *up* and *down*, as a prediction of $\Delta F(t)$. To predict $\Delta F(t+1)$ through the Q-function, the agent regards the predicted $\Delta F(t)$ as a real transition and applies it with $\Delta F(t-1)$ to the Q-function ([Fig. 2(a)], right). If the predicted $\Delta F(t+1)$ is *up*, the order is "buy"; otherwise it is "sell". In the $(t+1)$-th interval, the agent knows $F(t)$ and calculates the true $\Delta F(t)$. If the true $\Delta F(t)$ is *same*, the *reward* is 0. If it is equal to the *action*, the *reward* is a positive value; otherwise, it is a negative value.

Approach 1–2:

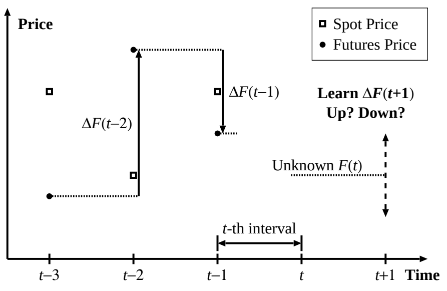
The agent learns to predict $\Delta F(t+1)$ directly from the past two J30F price transitions [Fig. 2(b)]. For this approach, Q-learning *states* and *actions* are identical with those of Approach 1–1, but an *action* gives a prediction of $\Delta F(t+1)$ instead of $\Delta F(t)$. If the *action* is *up*, the order is "buy"; otherwise, it is "sell". If the true $\Delta F(t+1)$ is *same*, the *reward* is 0. If it is equal to the *action*, the *reward* is a positive value; otherwise, it is a negative value. This approach differs from Approach 1–1 in that it does not consider $\Delta F(t)$ at all. Since the agent does not know the true $\Delta F(t+1)$ until the $(t+2)$-th interval, the *reward* is delayed.

Transitions of J30S price as states (Approach 2)

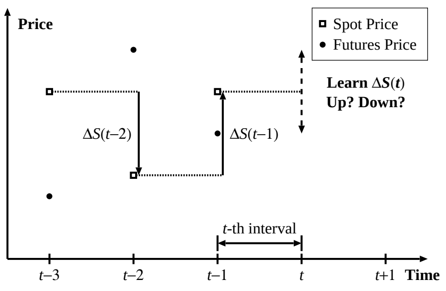
Generally, we can infer that a futures price will go up when the present spot price is high and the present futures price is lower than the present spot price. The approach described here uses J30S price transitions up to the present (t-th interval), together with this inference. The agent learns to predict $\Delta S(t)$ from the past two J30S price transitions: $\Delta S(t-2)$ and $\Delta S(t-1)$ [Fig. 2(c)]. Q-function has nine *states*, each of which is a combination of two transitions, and two *actions*, *up* and *down*, as a prediction of $\Delta S(t)$. The order is determined by rules in [Tab. 1], using the *action*, $S(t-1)$, and $F(t-1)$, according to the above inference. In the $(t+1)$-th interval, the agent knows $S(t)$ and calculates the true $\Delta S(t)$. If the true $\Delta S(t)$ is *same*, the *reward* is 0. If it is equal to the *action*, the *reward* is a positive value; otherwise, it is a negative value.



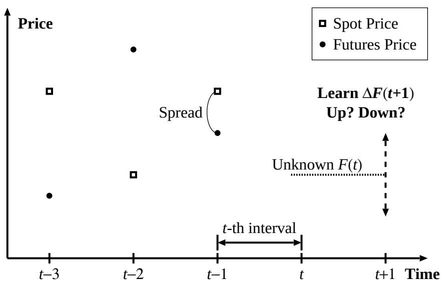
(a) Approach 1–1



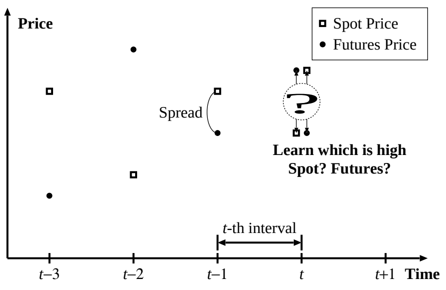
(b) Approach 1–2



(c) Approach 2



(d) Approach 3–1



(e) Approach 3–2

Figure 2: Proposed approaches. (a) Approach 1–1: Left: Learn $\Delta F(t)$ from $\Delta F(t-2)$ and $\Delta F(t-1)$; Right: Predict $\Delta F(t+1)$ from $\Delta F(t-1)$ and the learned $\Delta F(t)$. (b) Approach 1–2: Learn $\Delta F(t+1)$ directly from $\Delta F(t-2)$ and $\Delta F(t-1)$. (c) Approach 2: Learn $\Delta S(t)$ from $\Delta S(t-2)$ and $\Delta S(t-1)$. (d) Approach 3–1: Learn $\Delta F(t+1)$ from the spread between $F(t-1)$ and $S(t-1)$. (e) Approach 3–2: Learn whether $F(t)$ or $S(t)$ is high from the spread between $F(t-1)$ and $S(t-1)$

Action	$F(t-1)$	Order
up	$< S(t-1)$	Buy
up	$\geq S(t-1)$	Do nothing
down	$\leq S(t-1)$	Do nothing
down	$> S(t-1)$	Sell

Table 1: Rules for determining an order in Approach 2. The first, second, and third columns list the action, *the relation between $F(t-1)$ and $S(t-1)$, and the order, respectively*

Spread between J30F and J30S prices as states (Approaches 3–1 and 3–2)

The approaches described here use the spread between J30F and J30S prices. The spread, which is continuous, is discretized and used as the *states* of Q-learning. These approaches are divided into two types according to what the agent learns.

Approach 3–1:

The agent learns to predict $\Delta F(t+1)$ by using the spread between $F(t-1)$ and $S(t-1)$ [Fig. 2(d)]. Q-function has two *actions*, *up* and *down*, as a prediction of $\Delta F(t+1)$. If the *action* is *up*, the order is “buy”; otherwise the it is “sell”. The only difference between Approaches 1–2 and 3–1 is the state space.

Approach 3–2:

The agent learns to predict whether $F(t)$ or $S(t)$ will be high from the spread between $F(t-1)$ and $S(t-1)$ [Fig. 2(e)]. The *action* is a prediction of the higher price at t, either $F(t)$ or $S(t)$. If the *action* is $S(t)$, the order is “buy”; otherwise it is “sell”. In the $(t+1)$-th interval, the agent knows the true $F(t)$ and $S(t)$. If they are identical, the *reward* is 0. If the true relation is equal to the *action*, the *reward* is a positive value; otherwise, it is a negative value.

4 Experiments

U-Mart researchers conduct an annual competition of trading agents [U-Mart]. In this competition, called the U-Mart International Experiment, or UMIE, four J30S price patterns (ascending, descending, oscillating, and reversing series) are used to rank the contestants, each of which starts with one billion yen. The rank is determined by four metrics: (i) average profit, (ii) maximum profit, (iii) profit count, and (iv) bankruptcy count. The metrics are obtained from three experiments:

– One of the contestants with the standard agent set (4 patterns $\times$ 50 simulations),

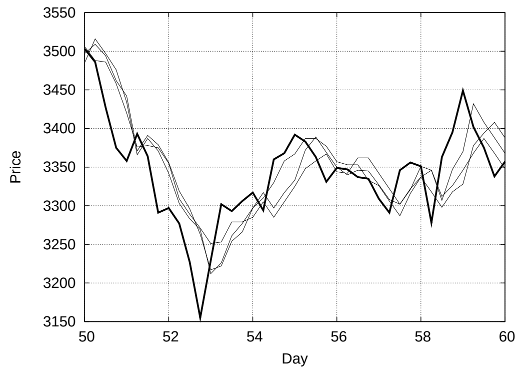


Figure 3: Example showing how futures prices differ in each simulation. The bold line is a spot price sequence, while the thin lines are the corresponding futures price sequences in different simulations

– All of the contestants with the standard agent set (4 patterns × 50 simulations), and

– Half of the agents randomly selected from all the agents (4 patterns × 250 simulations).

The duration of each simulation is sixty days, and each day has four intervals. The profit count and the bankruptcy count are the numbers of simulations that finished in the black and were terminated by bankruptcy, respectively. *Note that, even if spot prices are identical in different simulations, futures prices that we consider here become different in each simulation because of the randomness in the agents.* See [Fig. 3] as an example.

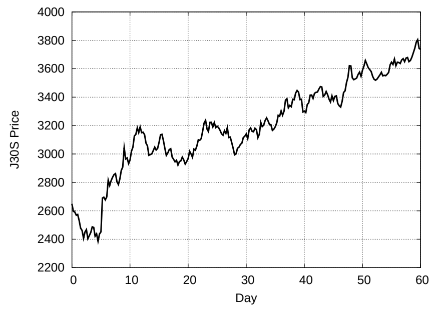
The standard agent set includes nineteen sample agents of ten types in the U-Mart developer's kit [U-Mart]. None of the sample agents learn anything; instead, they make decisions by applying fixed strategies. The distribution of sample agents and the strategies of each are described in [Appendix A].

We conducted three experiments corresponding to the above three experiments. This section describes the setup and result of each experiment.

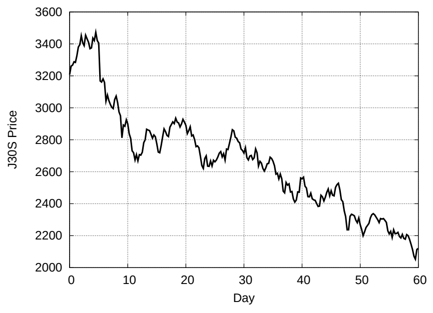
4.1 Experiment 1: One Contestant with Standard Set

4.1.1 Setup

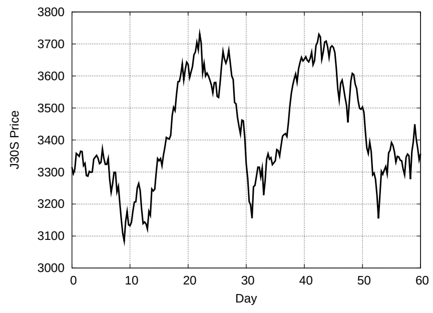
In this experiment, each contestant competed with the standard set of nineteen sample agents of ten types. We used the price patterns used in UMIE2005 [Fig. 4], which were not known to the contestants in advance. In each interval, every agent decided orders by means of the histories of J30F and J30S prices up to the previous interval. The contestants were the following:



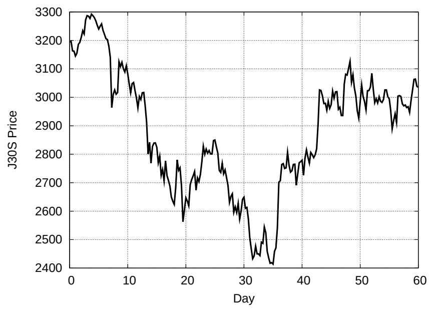
(a) Ascending series



(b) Descending series



(c) Oscillating series



(d) Reversing series

Figure 4: Four J30S price patterns used in UMIE2005

– The proposed agents (Approaches 1–1, 1–2, 2, 3–1, and 3–2), and

– Past UMIE champions, called Op, Ns, Ts, OFB, and TDP.

The champions are described briefly in [Appendix B].

For Q-learning of the proposed agents, we set the *reward* as +1 when the *action* was right and as −1 when it was wrong. For Approaches 3–1 and 3–2, the spread was discretized into 20 sections, that is, there were 18 sections in increments of 5 from −45 to 45, a section for more than 45, and a section for less than −45. The learning rate α and the discount factor γ were 0.1 and 0.9, respectively. The temperature T in (4) was set to 1. When the proposed agent bought futures in the t-th interval, it ordered p_{s_t} (the softmax probability (4)) $\times$ 100 trading units, each of which consisted of 1000 shares, at $F(t-1)+20$ yen. When the agent sold, it ordered $p_{s_t} \times 100$ trading units at $F(t-1)-20$ yen. The lower and upper position limits were −300 and 300, respectively.

$F(t) - S(t)$	p_{s_t}(Futures)	p_{s_t}(Spot)
~ -45	13.0	87.0
$-45 \sim -40$	29.3	70.7
$-40 \sim -35$	15.3	84.7
$-35 \sim -30$	25.3	74.7
$-30 \sim -25$	20.4	79.6
$-25 \sim -20$	26.6	73.4
$-20 \sim -15$	40.1	59.8
$-15 \sim -10$	39.0	61.0
$-10 \sim -5$	49.0	51.0
$-5 \sim 0$	43.8	56.2
$0 \sim 5$	44.7	55.3
$5 \sim 10$	57.4	42.6
$10 \sim 15$	63.5	36.5
$15 \sim 20$	63.2	36.8
$20 \sim 25$	69.7	30.3
$25 \sim 30$	68.2	31.8
$30 \sim 35$	73.5	26.5
$35 \sim 40$	73.4	26.6
$40 \sim 45$	75.0	25.0
$45 \sim$	85.1	14.9

Table 2: Learned Q-table of Approach 3–2. For simplicity, this shows $p_{s_t}(a)$ in (4) with $T = 1$ (%)

Before starting the experiment, each proposed agent learned a policy through 100 simulations with the standard agent set. For example, [Tab. 2] shows the learned Q-table of Approach 3–2. The price patterns used for this learning phase were parts of the J30S price pattern in the U-Mart developer's kit [Fig. 5], with each part starting at a randomly chosen interval. The price pattern in the kit contained 2444 records between 1532 and 4778 yen and it did not match the UMIE2005 patterns at all. These agents continued to learn during the experiment, but at the beginning of each simulation, every agent's policy was initialized to that learned before the experiment. Two champions, OFB and TDP, also had to learn in advance. Hence, they learned through 100 simulations before the experiment, in the same way described above, but TDP did not continue learning during the experiment.

4.1.2 Result

[Tab. 3] shows the result of four metrics in UMIE and the precision of *actions* of the proposals. Among the proposed agents, while Approaches 1–1, 1–2, and 3–1 got losses in some (or all) series, Approaches 2 and 3–2 got gains in all of the series. Hence, Approaches 2 and 3–2 were better than the rest of the proposed agents. We also see that while Approach 2 was better than 3–2 when the price changed uniformly, i.e., in the ascending or the descending series, it was worse when the price undulated, i.e., in the oscillating or the reversing series. This is probably because, as indicated by the precision in [Tab. 3(e)], Q-learning of Approach 2 failed to learn *actions*. This means that the good result of Approach 2 strongly depended on the rules in [Tab. 1]. The rules are based only on the current spread to determine orders when the *action* is random, so

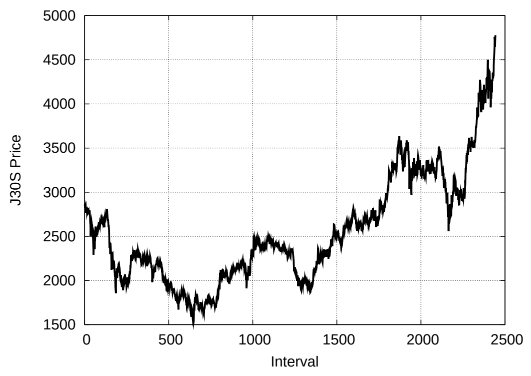


Figure 5: J30S price patterns in the U-Mart developer's kit used for learning

the resulting order was profitable when the price changed uniformly and useless when the price undulated. On the other hand, Approach 3–2 succeeded in learning the spread and got gains in all patterns. Therefore, we conclude that Approach 3–2 performed the best of all the proposed agents.

[Tab. 3(e)] also shows that Approach 1–1 succeeded in predicting $\Delta F(t)$ in all patterns to some degree. However, it failed to get gains in three patterns. This was because Approach 1–1 had to succeed in prediction two consecutive times (i.e., $\Delta F(t)$ and $\Delta F(t+1)$) in order to get gains. In fact, although the precision was around 60% in all series, this approach more often failed to predict $\Delta F(t+1)$ because $0.6^2 < 1/2$.

Approaches 1–2 and 3–1 failed to learn and to get gains. This was probably because the *reward* was delayed in these approaches. Since the environment had only stochastic agents, except for Approaches 1–2 and 3–1, it had the Markov property, and Q-learning could be applied to learn an optimal policy, in principle. The environment composed of nineteen stochastic opponents, however, was too complicated for these approaches to learn in only 100 simulations. Hence, the delayed reward failed to be transmitted to the corresponding state-action pair that brought the reward. As a consequence, the corresponding state-action pair was not reinforced appropriately and these approaches failed.

Surprisingly, some of the past UMIE champions (Op, OFB, and TDP) got bankrupted in some patterns. The huge gains that they got in some price patterns (i.e., Op and OFB in the descending series and TDP in the ascending series) probably compensated for the bankruptcies during the competitions. Meanwhile, Ns and Ts got gains in most of simulations in all of the patterns.

	Asc.	Desc.	Oscil.	Rev.
1–1	51.2	−44.7	−102.0	−68.7
1–2	−53.6	−53.1	−84.9	−31.3
2	224.6	227.1	24.6	133.6
3–1	−126.1	−169.3	−121.0	−110.3
3–2	209.0	197.2	158.7	177.8
Op	−682.4	575.3	−44.0	−648.9
Ns	337.7	347.8	144.4	**268.8**
Ts	232.3	211.8	**228.3**	190.5
OFB	−89.4	**1803.3**	−428.5	−150.6
TDP	**1463.9**	−674.2	−84.5	58.0

(a) Average profit (million yen)

	Asc.	Desc.	Oscil.	Rev.
1–1	264.2	220.7	120.0	134.3
1–2	197.6	210.4	63.3	248.6
2	341.8	342.5	103.6	250.0
3–1	139.5	127.0	146.7	98.2
3–2	324.2	287.1	259.6	310.1
Op	36.1	1510.3	154.4	−67.5
Ns	550.1	461.3	240.0	339.7
Ts	291.4	300.8	**305.4**	287.3
OFB	979.4	**2373.8**	−63.4	212.2
TDP	**1736.1**	−326.8	239.6	**462.0**

(b) Maximum profit (million yen)

	Asc.	Desc.	Oscil.	Rev.
1–1	30	17	6	9
1–2	19	21	9	20
2	49	**50**	36	**50**
3–1	6	3	6	6
3–2	**50**	**50**	**50**	**50**
Op	7	31	18	0
Ns	**50**	**50**	49	**50**
Ts	**50**	**50**	**50**	**50**
OFB	21	**50**	0	10
TDP	**50**	0	8	38

(c) Profit count

	Asc.	Desc.	Oscil.	Rev.
1–1	**0**	**0**	**0**	**0**
1–2	**0**	**0**	**0**	**0**
2	**0**	**0**	**0**	**0**
3–1	**0**	**0**	**0**	**0**
3–2	**0**	**0**	**0**	**0**
Op	5	**0**	**0**	8
Ns	**0**	**0**	**0**	**0**
Ts	**0**	**0**	**0**	**0**
OFB	22	**0**	**0**	2
TDP	**0**	19	**0**	3

(d) Bankruptcy count

	Asc.	Desc.	Oscil.	Rev.
1–1	60.1	60.7	59.0	60.2
1–2	46.9	47.1	47.9	47.7
2	50.1	50.3	49.2	50.4
3–1	41.6	41.0	42.0	41.4
3–2	65.9	68.5	64.5	64.6

(e) Precision (%)

Table 3: Result of Experiment 1, with bold face indicating the best: (a) Average profit of each approach. (b) Maximum profit of each approach. (c) Profit count, indicating the number of simulations that finished in the black. (d) Bankruptcy count, indicating the number of simulations terminated by bankruptcy. (e) Precision of the actions of the proposed approaches

4.2 Experiment 2: All Contestants with Standard Set

4.2.1 Setup

Most of the experimental settings were identical with those of Experiment 1, but all of the proposed agents, all of the champions, and the standard agent set participated in every simulation.

4.2.2 Result

[Tab. 4] shows the result of four metrics in UMIE and the precision of *actions* of the proposed agents. The average profit and the precision of *actions* of Approaches 2 and 3–2 were less than those obtained in Experiment 1. This is not surprising because these approaches learned their policies in the environment of Experiment 1, which had only the standard agent set as opponents. This is also because the environment contained multiple Q-learning agents, i.e., it did not have the Markov property. Although Q-learning can converge to an optimal policy in an environment with the Markov property (see [Section 3.1]), it does not guarantee anything when the environment does not have the Markov property.

4.3 Experiment 3: Randomly Selected Agents

4.3.1 Setup

Most of the experimental settings were identical with those of Experiment 1, but the participants were randomly selected from among all of the proposed agents, all of the champions, and the standard agent set. Hence, the number of simulations was different for each approach (strategy).

4.3.2 Result

[Tab. 5] shows the result of four metrics in UMIE and the precision of *actions* of the proposals. Since the number of simulations was different for each approach (strategy), the profit count and bankruptcy count are shown as ratios. The result was not so different from that of Experiment 2.

5 Conclusion

In this work, we have proposed several learners using Q-learning on the futures market simulator U-Mart. First, in [Section 2], we saw what futures are and what U-Mart is. In [Section 3], we proposed three types of learners, each of which had a different state space of Q-learning: the transitions of the futures price, those of the spot price, and the spread between the futures and spot prices. [Section 4] gave the results of experiments

	Asc.	Desc.	Oscil.	Rev.
1–1	54.7	−63.2	−126.9	−102.5
1–2	−84.0	−103.5	−99.3	−38.4
2	206.2	173.2	−24.3	48.3
3–1	−120.0	−55.6	−108.3	−51.6
3–2	117.0	−57.4	23.1	75.5
Op	−38.5	170.9	−52.3	−139.7
Ns	47.2	−335.1	131.7	**159.9**
Ts	213.3	114.6	**303.8**	158.7
OFB	−639.2	**3084.2**	−344.8	−109.2
TDP	**1757.9**	−552.5	73.7	−559.5

(a) Average profit (million yen)

	Asc.	Desc.	Oscil.	Rev.
1–1	251.0	209.4	54.1	78.2
1–2	169.3	251.0	93.7	186.1
2	309.9	326.8	103.6	211.6
3–1	77.2	300.6	81.4	128.4
3–2	349.0	135.7	182.0	322.5
Op	125.3	1466.2	230.8	46.7
Ns	717.1	−85.2	252.8	288.6
Ts	314.6	212.1	**373.0**	233.8
OFB	−57.8	**4273.4**	67.2	**494.2**
TDP	**2159.4**	−424.1	350.3	385.0

(b) Maximum profit (million yen)

	Asc.	Desc.	Oscil.	Rev.
1–1	35	11	3	8
1–2	11	12	10	22
2	**50**	48	21	37
3–1	8	14	9	17
3–2	42	15	30	39
Op	39	15	5	11
Ns	29	0	48	**50**
Ts	**50**	**50**	**50**	**50**
OFB	0	**50**	1	17
TDP	**50**	0	40	1

(c) Profit count

	Asc.	Desc.	Oscil.	Rev.
1–1	**0**	**0**	**0**	**0**
1–2	**0**	**0**	**0**	**0**
2	**0**	**0**	**0**	**0**
3–1	**0**	**0**	**0**	**0**
3–2	**0**	**0**	**0**	**0**
Op	6	**0**	**0**	11
Ns	**0**	**0**	**0**	**0**
Ts	**0**	**0**	**0**	**0**
OFB	40	1	19	20
TDP	**0**	50	3	49

(d) Bankruptcy count

	Asc.	Desc.	Oscil.	Rev.
1–1	53.0	54.6	53.6	51.9
1–2	47.0	48.1	46.9	47.3
2	50.5	50.4	49.8	47.3
3–1	46.1	47.3	46.8	47.1
3–2	58.1	56.6	58.0	53.5

(e) Precision (%)

Table 4: Result of Experiment 2, with bold face indicating the best: (a) Average profit of each approach. (b) Maximum profit of each approach. (c) Profit count, indicating the number of simulations that finished in the black. (d) Bankruptcy count, indicating the number of simulations terminated by bankruptcy. (e) Precision of the actions of the proposed approaches

	Asc.	Desc.	Oscil.	Rev.
1–1	−44.8	−106.6	−177.9	−132.9
1–2	−115.7	−78.7	−112.2	−67.3
2	167.3	138.7	−12.8	55.9
3–1	−100.4	−100.6	−104.2	−63.1
3–2	82.9	−20.7	31.6	54.6
Op	−476.6	727.1	−73.6	−342.3
Ns	55.3	−71.3	123.5	**184.7**
Ts	182.8	99.6	**230.7**	144.9
OFB	−187.6	**1471.3**	−295.3	−12.6
TDP	**1650.5**	−592.0	10.5	−305.6

(a) Average profit (million yen)

	Asc.	Desc.	Oscil.	Rev.
1–1	285.7	216.0	63.9	200.8
1–2	290.1	268.7	205.1	201.9
2	354.0	367.1	163.6	382.7
3–1	488.7	200.4	138.4	187.1
3–2	376.1	376.8	289.0	295.9
Op	117.3	1734.8	403.6	51.4
Ns	1489.8	1489.0	**510.4**	682.4
Ts	343.6	323.8	370.5	364.2
OFB	1663.6	**3400.8**	308.2	**1032.0**
TDP	**2608.0**	−249.9	392.8	625.0

(b) Maximum profit (million yen)

	Asc.	Desc.	Oscil.	Rev.
1–1	41.1	17.8	4.7	16.3
1–2	22.6	32.3	8.9	25.0
2	90.4	87.4	43.0	74.1
3–1	23.4	24.2	14.5	34.7
3–2	72.0	42.4	64.0	69.6
Op	22.9	82.4	16.0	7.6
Ns	45.7	40.6	92.0	96.4
Ts	98.5	78.0	**100.0**	**97.7**
OFB	31.2	**93.5**	13.8	47.1
TDP	**100.0**	0.0	61.2	31.9

(c) Profit count ratio (%)

	Asc.	Desc.	Oscil.	Rev.
1–1	**0.0**	**0.0**	**0.0**	**0.0**
1–2	**0.0**	**0.0**	**0.0**	**0.0**
2	**0.0**	**0.0**	**0.0**	**0.0**
3–1	**0.0**	**0.0**	**0.0**	**0.0**
3–2	**0.0**	**0.0**	**0.0**	**0.0**
Op	48.9	**0.0**	5.3	37.4
Ns	**0.0**	**0.0**	**0.0**	**0.0**
Ts	**0.0**	**0.0**	**0.0**	**0.0**
OFB	48.6	1.5	27.5	18.1
TDP	**0.0**	87.9	5.2	64.7

(d) Bankruptcy count ratio (%)

	Asc.	Desc.	Oscil.	Rev.
1–1	53.4	54.0	54.7	52.4
1–2	46.5	46.8	46.2	45.4
2	50.2	49.7	50.1	47.8
3–1	44.4	45.2	45.2	44.8
3–2	59.4	58.8	59.1	56.2

(e) Precision (%)

Table 5: Result of Experiment 3, with bold face indicating the best: (a) Average profit of each approach. (b) Maximum profit of each approach. (c) Profit count ratio, indicating the percentage of simulations that finished in the black. (d) Bankruptcy count ratio, indicating the percentage of simulations terminated by bankruptcy. (e) Precision of the actions of the proposed approaches

with the proposed agents, the standard agent set from the U-Mart developer's kit, and the champions of past U-Mart competitions.

The proposed approach predicting whether spot or futures price is high from the spread at the previous interval (Approach 3–2) obtained gains in all price patterns with good precision in Experiment 1. Although it suffered losses in the descending series in Experiments 2 and 3, it obtained good results overall in simulations with the champions of past U-Mart competitions. The results showed unexpected success of the proposed approaches since they were very simple state-action-reward definitions in a general reinforcement learning algorithm and were not designed so deliberately as compared with the champions. For example, the rules implemented in advance were only [Tab. 1] for Approach 2. Note that we intentionally ignore the multiagent perspective in this work, because nobody can perceive opponents' actions in a market that are usually required in multiagent reinforcement learning.

There are several future directions. First, we plan to participate in the U-Mart competition. The champions used here were *past* champions. We do not know if our proposed approaches will behave properly and get profits when competing with new entrants in the competition. Second, we have to tune the proposed agents. This is a challenge in terms of combining reinforcement learning and various types of knowledge in economics, investment, etc. Third, we have to check whether our approaches are applicable in a massive simulation containing an immense number of traders.

References

[Kita 2000] Kita, H.: "An Introduction to U-Mart" (2000) (In the U-Mart developer's kit downloadable from [U-Mart]).

[Kitano et al. 2005] Kitano, H., Nakashima, T., and Ishibuchi, H.: "Behavior Analysis of Futures Trading Agents Using Fuzzy Rule Extraction"; Proc. 2005 IEEE International Conference on Systems, Man and Cybernetics, IEEE Press, Piscataway, NJ (2005), 1477–1481.

[Nevmyvaka et al. 2006] Nevmyvaka, Y., Feng, Y., and Kearns, M.: "Reinforcement Learning for Optimized Trade Execution"; Proc. 23rd International Conference on Machine Learning, Omnipress, Madison, WI (2006), 673–680.

[O et al. 2002] O, J., Lee, J. W., and Zhang, B.-T.: "Stock Trading System Using Reinforcement Learning with Cooperative Agents"; Proc. 19th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA (2002), 451–458.

[Sato et al. 2001] Sato, H., Koyama, Y., Kurumatani, K., Shiozawa, Y., and Deguchi, H.: "U-Mart: A Test Bed for Interdisciplinary Research in Agent Based Artificial Market"; Aruka, Y., ed., "Evolutionary Controversies in Economics: A New Transdisciplinary Approach", Springer, Tokyo (2001), 179–190.

[Sutton and Barto 1998] Sutton, R. S. and Barto, A. G.: "Reinforcement Learning: An Introduction"; MIT Press, Cambridge, MA (1998).

[U-Mart] "U-Mart Project"; `http://www.u-mart.org/`.

[Watkins and Dayan 1992] Watkins, C. J. C. H. and Dayan, P.: "Technical Note: Q-learning"; Machine Learning, 8 (1992), 279–292.

A U-Mart Standard Sample Agent Set

This appendix describes the U-Mart sample agents in the standard agent set [U-Mart]. The number after the agent name indicates how many of these agents were included in the set. If not otherwise specified, the order price is $F(t-1)+20x$ yen, where $x \sim N(0,1)$, and the order volume is randomly determined between two parameters. See [U-Mart] for details.

Trend (1):

Buy if $F(t-1) > F(t-2)$, sell if $F(t-1) < F(t-2)$, otherwise do nothing.

AntiTrend (2):

Swap the conditions of Trend.

Random (1) and SRandom (3):

An order is randomly determined. SRandom uses $S(t-1)$ instead of $F(t-1)$ to calculate the order price.

Rsi (1) and SRsi (3):

Rsi uses the relative strength index (RSI) of J30F in t-th interval:

$$\frac{\sum_{\tau<t, F(\tau)>F(\tau-1)} F(\tau) - F(\tau-1)}{\sum_{\tau<t} \mid F(\tau) - F(\tau-1) \mid} .$$

Buy if RSI is less than 0.3, sell if it is more than 0.7, otherwise do nothing. SRsi uses RSI with J30S instead of with J30F.

MovingAverage (1) and SMovingAverage (3):

Calculate the moving average of J30F in ten intervals. Buy if it is more than $F(t-1)$, sell if it is less than $F(t-1)$, otherwise do nothing. SMovingAverage uses J30S instead of J30F.

SFSpread (2):

Calculate a spread ratio as $\{F(t-1) - S(t-1)\}/S(t-1)$. Buy if the ratio is less than or equal to -0.01, sell if it is more than or equal to 0.01, otherwise do nothing. The order price is $\left\{F(t-1) + S(t-1) + x\left|F(t-1) - S(t-1)\right|\right\}/2$, where $x \sim N(0,1)$.

DayTrade (2):

Always buy and sell simultaneously. The buying and selling prices are $0.99F(t-1)$ and $1.01F(t-1)$, respectively.

B Past UMIE Champions

This appendix describes the past UMIE champions. Op, Ns, and Ts were the champions of UMIE 2005, while OFB and TDP were those of UMIE 2004.

Osako_pivot (Op):

It calculates the average of the maximum price of the previous day, the minimum price of that day, and the final price of that day as a pivot. Then, it sets two support lines and resistance lines from the pivot. It determines an order from the relation among the futures price, the support lines, and the resistance lines. The volume is always 100 units, and the prices are $F(t-1)+50$ yen for buying and $F(t-1)-50$ yen for selling.

Nakamura_spread (Ns):

It simultaneously sends three orders, which are an arbitrage order, an order preparing for a bulge, and one preparing for collapse. The arbitrage order is determined by the spread between $F(t-1)$ and $S(t-1)$. The volume is also determined by the spread, and the price is the average of $F(t-1)$ and $S(t-1)$. For the bulge/collapse orders, it sends a sell/buy order at a price 200 yen higher/lower than $S(t-1)$. The volume is 400 units in both cases.

Trend_swift (Ts):

If $S(t-1) > S(t-2)$ and $S(t-1) - F(t-1) > 10$ yen, it buys at $F(t-1)+40$ yen. If $S(t-1) < S(t-2)$ and $F(t-1) - S(t-1) > 10$ yen, it sells at $F(t-1)-40$ yen. The volume depends on the spread.

OPUFuzzyB (OFB) [Kitano et al. 2005]:

It is a fuzzy-rule-based online learning agent with a table containing weights for rules. First, it calculates the spreads $S(t-1)-S(t-2)$, $S(t-1)-S(t-4)$, and $S(t-1)-S(t-6)$. Then, it calculates fuzzy reliabilities from three membership functions, each of which is a function of one of the spreads. Next, it determines an order from the spreads and the fuzzy reliabilities. The volume is always 200 units, and the price is $S(t-1)-5$ yen for buying and $S(t-1)+5$ yen for selling. In the next interval, it updates the weights according to the fuzzy reliabilities.

TriDiceP (TDP):

It has a state-action table, like that used in Q-learning, for determining orders. The states in the table are determined by quadratically approximated past spot price sequences. Orders are chosen by probabilities calculated from weights in the table. For a chosen order, the weight in the table is decreased to change behaviors. The volume is calculated from the order probabilities, and the prices are $S(t-1)+10$ yen for buying and $S(t-1)-10$ yen for selling.