Artificial Intelligence Modeling of Financial Profit and Fraud

D. Sawh, K. Ponnambalam, F. Karray

***Abstract*— The goal of financial engineering has largely been application of mathematical techniques to derivatives pricing and portfolio optimization. As the market is increasingly automated, artificial intelligence based methods are becoming relevant as they are adaptively trained to search networks for an optimal reward. This paper describes artificial intelligence methods in the context of two financial applications: 1) optimal trading strategy in the order book, and 2) detecting insider trading in transaction tables. Q-learning and Sarsa methods are compared for both applications. Q-learning revisits states thus emphasizing their rewards. Sarsa searches more states and discovers those with higher rewards. Further work will model input data using options theory.**

***Index Terms*—fraud, insider trading, intelligent systems, reinforcement learning**

I. Introduction

REINFORCEMENT learning (RL) is a form of goal-directed learning based on a numerical reward signal. It endeavours to optimize the rewards which translate into the "best behaviours" for that system. It can begin to learn the dynamics of an environment without any prior knowledge. This property is demonstrated through two opposing financial applications: profit-making, and fraud-discovery. The feature permitting this flexibility is that RL is an adaptive, model-free system. Reinforcement learning is particularly applicable to finance because of the need to continually create and update policies.

There are two approaches to learning in any artificially intelligent learning systems: unsupervised (or "off-policy" method) and supervised (or "on-policy" method). In an unsupervised method, the policy to generate the learning method behaviour is different from the estimation policy which is the operating policy during real-time decision-making. Q-learning is an example of an off-policy method. These methods tend to be exploitative and inflexible when new scenarios are introduced into the environment.

Manuscript received March 1, 2011; revised March 22, 2011. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada.

Deitra Sawh is a PhD student with the Department of Systems Design Engineering, University of Waterloo, Waterloo, ON N2L 3G1 Canada (519-888-4567 X36191; e-mail: dsawh@uwaterloo.ca).

Kumaraswamy Ponnambalam is a Professor with the Department of Systems Design Engineering, University of Waterloo, Waterloo, ON N2L 3G1 Canada (519-888-4567 X33282; e-mail: ponnu@uwaterloo.ca).

Fakhri Karray is a Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1 Canada (519-888-4567 X35584; e-mail: karray@uwaterloo.ca).

Unsupervised learning uses the same action policy in every iteration and updates the policy as the environment changes in real-time. Sarsa is an example of an on-policy method. These methods tend to be exploratory and time-consuming.

This paper presents two financial applications of RL. The first creates an optimal trading strategy based on expected trades in the market. It is predictive and constructed analogously to the grid-world example from Sutton's book [1].

The second approach searches for insider trades in the transaction book. It is a backward-looking model and the dataset is defined similarly to the data model described by Lu [2]. This method defines the dataset as the entire trade record (not just spreads) and defines actions by attributes of the trade.

Both experiments compare Q-learning (off-policy) and Sarsa (on-policy). Sutton states that Q-learning finds the optimal policy whereas Sarsa finds the safest policy [1]. This can be considered a risky trader (optimal) vs. a non-risky trader (safe). The RL algorithms are Q-learning and Sarsa. The main difference between them is that Q-learning is always maximizing the value function to select the action. Sarsa has the option to choose actions randomly during the learning process.

A. Markov Property

The application of RL is based on the Markov Property. This means that the next state is determined solely by the current state and current action of the agent. In financial terms this adheres to the efficient market hypothesis stating that all information in the market is known and captured in the current price, or in other words, current prices fully reflect all available information [3]. RL looks for states having attributes in common with the current state, not with any state encountered before the current state. On one hand this property is beneficial because if we are on the correct path then it will link together the trades with the most in common. However if we are on the wrong path, then the system will just link together what it encounters. This motivates the need to select the most suspicious transaction initially and follow it. Intuitively this property is commensurate with the concept of insider trading in that the information is held by one person for one trade. In fact, when the Markov property is broken, then there is fraud.

II. Trading Strategy Model

On the trading floor, the order book plays a fundamental role in identifying market sentiment. It reveals at what price there is interest in the market and the volume at that price.

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

An example of how it is used is that the trader will encourage orders from clients at the likely prices in the market. Generally, the levels at which market players have the most invested will occur in the market.

In this example, the order book is modeled as a grid of expected trades. Each column represents a slice of time in which orders are stacked according to risk-neutrality. The height of the stack represents the depth of the market at that time. We would like to know at what levels to place our trades given the potential levels dictated by the order book. Each potential trade is considered to be a "state". The state is represented by a "spread", the distance from the current market price. A risk-neutral trader will place orders close to the market as there is a high probability of these levels being achieved. As a trader becomes riskier, she will place orders further away from the market. The risk is that the levels further away are not achieved in the desired time-period. It is assumed that the trader holds a position in the asset at each time-step and is closing her position at the highest profit. The path discovered by the RL search tells the trader where to place these orders.

Trade data can be considered as a grid of sequential spreads stacked in order of depth of the market (lowest spread to highest spread). An example of a grid is shown in Fig. 1. The x-axis is time while the y-axis is indicative of risk level.

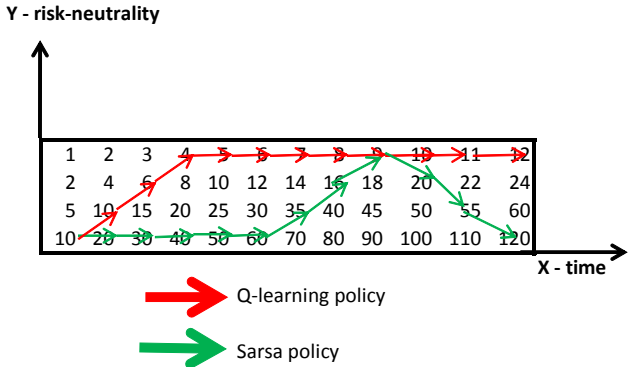

Figure 1: Grid of States and Path Example

The main actions in Sutton's [1] grid-world model are *up, down, left, right*. The actions in this model are *cross_up, cross_down, right*. These choices are motivated by the rewards of "returns". "Cross" implies that the algorithm cannot select a price in the same column. The financial reasoning behind this is that a trader can only execute one trade in each time period. Each column in the grid is considered to be a slice of the sequence of trades. The objective of this model is to search the grid for the optimal policy that achieves the goal return.

A. *Parameters*

This problem is cast as an episodic task. This implies that we can run many simulations of the same algorithm to look for pattern convergence. The grid is a 4 by 12 grid; 4 potential trades by 12 columns (time-steps). Potential trades are described by the spread made on the trade, which is also the return for the state. The main variables are as follows:

States : $\{S\} = \{s, spreads\}$

Actions: $\{A\} = \{a, cross\_up, cross\_down, right\}$

Reward : $\{R\} = \{r, spreads\}$

The start state in the 4 by 12 grid is at (1,1). Both learning methods use γ=1and α=0.1.

The goal state in this application deviates from the grid-world example. Sutton's algorithm specifies a state as the goal state. In this example, the goal is to achieve a specified profit which acts as a restriction on $\{R\}$. This is not necessarily associated with one spread (state). Therefore, this system does not have a goal STATE, but rather a goal REWARD value on each iteration.

B. *Experiment*

Tests are performed on several grids. The Q-learning method has the exact same policy independent of the location of the larger rewards. Sarsa changes the orientation of the path towards the largest rewards in the grid.

Across many simulations, Q-learning weights are the same. This occurs because of the restriction on the rewards. Q-learning always hits the goal reward quickly by taking the same actions. Sarsa weights increase in order of magnitude and distribute actions evenly. These results are summarized in Table I.

TABLE I
OPTIMAL TRADING STRATEGY RESULTS

	Q_learn	SARSA
When simulations are increased, optimal policy weights:	constant	increase
% of states visited	31.25	87.50
Action values	>90% convergence on action 1	Proportionately distributed across actions

III. INSIDER TRADING MODEL

The main drawback of existing fraud detection systems is the inability to adapt to new scenarios. For example, the ADS detection programs used to detect insider trading in the Nasdaq Stock Market are based on templates i.e. known methods or patterns of fraud [4]. Reinforcement learning provides a solution to this.

The first step in building the fraud detection system for insider trading is to determine which trades could be considered to be "fraudulent". The regulators are permitted to see almost everything at the financial institutions. It is therefore assumed in this study that auditors are able to view all trades executed by the financial institution. As with

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

State	Asset	Client	Buy/Sell	# shares	Salesperson	Trade Price	Client Price	Spread	Profit	%spread
1	Equity A	Client 12	B	52	SP1	12.50	12.85	0.35	18.20	42%
2	Equity B	Client 11	B	46	SP2	6.85	6.90	0.05	2.30	6%
3	Equity A	Client 17	B	13	SP1	12.52	12.90	0.38	4.94	45%
4	Equity C	Client 11	B	45	SP2	7.00	7.05	0.05	2.25	6%

Figure 2: Transaction Table

other detection systems, we are searching for an unjustified gain or loss. At the initial anomaly detection stage, we are looking for a large gain. This is expressed in the trading book as a large spread taken on a trade. In the vein of Lu [2], this system is a "fraud case builder", a screening tool that is the first step of fraud detection. It links together suspicious records and builds policies that dictate the behaviour of the data.

A. *Parameters*

The transaction table model defines actions by transaction table attributes. We search a table and select actions based on the attributes of the trade. The transaction table tested has 1000 trades, 4 salespersons, 19 clients and 1 asset. A sample of the transaction table is in Fig. 2. This problem is also cast as an episodic task. The RL parameters are:

States : $\{S\} = \{s, trades\}$

Actions: $\{A\} = \{a, salesperson, asset, client\}$

Reward : $\{R\} = \{spread/total\_spread\}$

The reward is defined as the percentage of total spread for each transaction. This system rewards those states where the spread contributes to the total spread made on all trades for that day. The algorithm begins searching at the first record in the transaction table. The search is complete at the last transaction. Both learning methods use γ=1and α=0.1.

B. *Experiment*

This experiment is performed on two datasets: 1) sorted: the first record is a suspicious record, 2) unsorted: in the order of transaction entry. The results are summarized in Table II for the action order of: Salesperson, Asset, Client.

The results are similar to that which was obtained in the grid-world model. While Q-learning is greater than 90% in action 1 (regardless of what the action is), Sarsa distributes the weights across three actions. Q-learning policy values are larger, emphasizing the fact that a few specific states of the system are stronger and revisited more often.

	% accuracy of fraudulent states	optimal policy weights (total)	action selection
Sorted SARSA	60%	2.19	Proportional across actions
Sorted Q_learn	90%	325.19	>90% Convergence on action 1
Unsorted SARSA	70%	1.72	Proportional across actions
Unsorted Q_learn	60%	4.51	>90% Convergence on action 1

IV. DISCUSSION AND CONCLUSIONS

In this paper we have illustrated how reinforcement learning can be used to discover an optimal trading strategy and detect fraud in the trading book. By using two learning algorithms, we demonstrated why Q-learning is considered to be an "exploitative" method while Sarsa is "exploratory". Q-learning proves particularly effective when given sufficient information. When the dataset is large and scenarios are constantly changing, Sarsa is a better method.

Moving forward, we wish to incorporate macroeconomic fundamentals into the action selection method. Additionally, we want to apply financial options theory and stochastic models to better describe the input data in both examples.

REFERENCES

[1] R.S. Sutton and A.G. Barto, *Reinforcement Learning: An introduction*. Cambridge, Massachusetts: MIT Press, 2006, ch. 1-8.

[2] F. Lu, "Uncovering Fraud in direct marketing data with a fraud auditing case builder," in *Knowledge Discovery in Databases: PKDD 2007, Proceedings*, vol. 4702, 2007, pp. 540–547.

[3] S.A. Ross, R.W. Westerfield, J.F. Jaffe, and G.S. Roberts, *First Canadian Edition Corporate Finance.* Toronto, ON: McGraw-Hill Ryerson, 1995, pp. 366–385.

[4] T.E. Senator, "Ongoing management and application of discovered knowledge in a large regulatory organization: A case study of the use and impact of NASD Regulation's Advanced Detection System (ADS)" in *Proc. Of SIGDKK00*, 2004.

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)