A Novel Strategy for the 2005 Penn–Lehman Automated Trading Competition

Rahul Savani and Ben Veal

Department of Mathematics, London School of Economics
Houghton St, London WC2A 2AE, United Kingdom
email: rahul@maths.lse.ac.uk, b.w.veal@lse.ac.uk

September 8, 2005

CDAM Research Report LSE-CDAM-2005-12

Abstract. The Penn-Lehman Automated Trading (PLAT) project is a broad investigation of algorithms and strategies for automated trading in financial markets. It is centered on the Penn Exchange Simulator, which simulates an electronic crossing network. The simulator realistically mixes bids from automated stock trading agents with bids from the real stock market. The participants in the project develop automated trading agents that are entered in formal automated trading competitions. This paper concerns the 2005 PLAT competition. Unlike previous years, this competition was run with automated agents only, and no real-time stock market data. This allowed the authors' winning strategy, Jump & Dump, to exploit a basic, but easily overlooked, flaw in the other agents. We outline Jump & Dump and explain how and why it won.

1 Introduction

The Penn–Lehman Automated Trading (PLAT) project [5] is based in the Computer and Information Science department at the University of Pennsylvania. It receives generous financial and professional support from the Equity Capital Management group of Lehman Brothers in New York City. The project is a broad investigation of algorithms and strategies for automated trading in financial markets and related environments. The project makes use of the Penn Exchange Simulator (PXS), a simulator for automated trading that uses real-world, real-time stock market data, which is available over modern electronic crossing networks (ECNs). During its first two years, the PLAT project had over 60 participants from both the university of Pennsylvania and many external institutions. These participants develop PXS automated trading agents that are entered in formal automated trading competitions. For more details see [4].

The Nasdaq is a well-known electronic stock exchange trading high-tech stocks such as Microsoft and Intel. The Island ECN[1] is an electronic broker on the Nasdaq, which trades about 25% of the Nasdaq-listed daily trading volume. PXS uses publicly available data from the Island ECN. In a simulation, PXS receives buy and sell orders for a single stock from automated trading agents, and deals with the matching and execution of these orders. Some agents are designed to simulate the underlying market, for example by

[1] Island ECN is now known as INET ATS, but continues to use the same website [3].

placing historical or real-time orders taken from the order books of the Island ECN. Other agents are supplied by the participants and are designed to generate trading profits.

Participants are given access to the PXS server in order to investigate, design, and test trading agents. Each academic year, agents are submitted for two formal competitions, a preliminary competition in December and then a final competition the following May. This paper concerns the May 2005 PLAT competition and in particular the authors' winning strategy, Jump & Dump. This year's competition used historic data, but did not use real-time data, which was necessary for the success of Jump & Dump.

In Section 2 we give an overview of PXS, which is important for understanding Jump & Dump. Section 3 explains the nature and rules of the competition. Section 4 briefly describes the agents that took part in the May 2005 competition. In Section 5 we describe Jump & Dump. We conclude in Section 6 and discuss further work in Section 7.

2 The Penn Exchange Simulator (PXS)

PXS is a computer program that runs on a Linux server at the University of Pennsylvania. It receives orders to buy or sell a single stock from automated trading agents, deals with the order-matching process, and provides feedback and market information to the agents. As in many securities markets, PXS maintains two lists that are collectively known as the *limit order books*. One is the *bid* (or buy) book, which is a list of orders to buy shares. The other is the *ask* (or sell) book, which is a list of orders to sell shares. Each order consists of two pieces of information, the volume of shares to be bought or sold and the price per share at which to buy or sell. The price of an order is the *worst* price at which the order will be executed. This means that if a buy order is executed, the buyer is guaranteed to pay no more than the price of the order, but may pay less. Similarly, if a sell order is executed, a seller is guaranteed to receive no less than the price of the order, but may receive more. Figure 1 gives an example of how the books are typically displayed, ordered by price. When a new order is received from an agent it is placed at the appropriate position in the limit order book. If there is an existing order with the same price already in the book, then the new order is placed above (in the case of sell orders) or below (for buy orders) the existing order. The top order of the bid book represents the best deal from a seller's point of view and the bottom order in the ask book represents the best deal from a buyer's point of view. These two orders are called respectively the *current bid* and *current ask*, or sometimes just the bid and ask.

If a buy order is received with a price that is at least the current ask then these two orders are matched, and a transaction is carried out. The transaction volume is the smaller volume of the matching pair. The transaction occurs at the price of the current ask. PXS transfers the appropriate (virtual) shares and cash between the accounts of the agents involved. After such a transaction there may still be remaining volume for the incoming order, and so several transactions can result from a single incoming order.

An example is illustrated in Figure 1 and is explained next. Transaction costs, which are explained in Section 3, are ignored in this example. The current ask is 200 shares at \$24.25 each. A buy order for 1250 shares at \$24.70 is received. First 200 shares are sold for \$24.25. Therefore, \$4850 is debited from the buyer's account and credited to

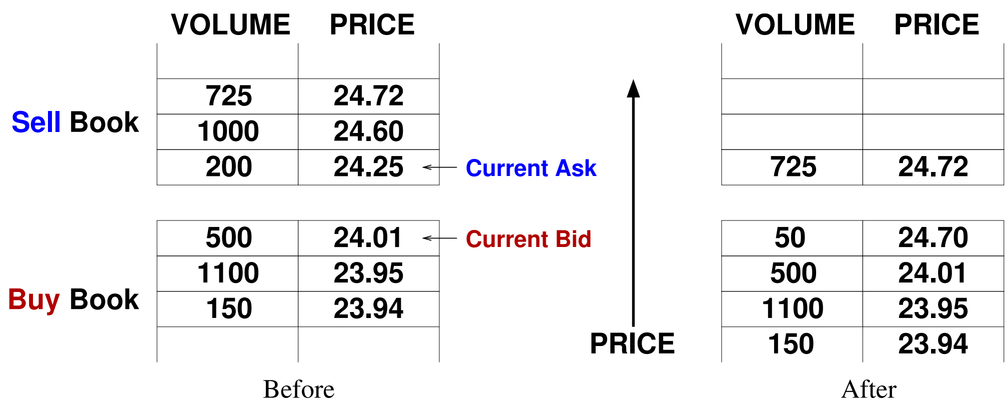

Figure 1: The order book before and after a buy order for 1250 shares at \$24.70

the appropriate seller. This transaction reduces the volume of the incoming order to 1050 shares. The new current ask becomes 1000 shares at \$24.60. Next 1000 shares are sold for \$24.60, with \$24600 debited from the buyer's account and credited to the appropriate seller. The new current ask becomes 725 shares at \$24.72. Since the price of the incoming buy order is less than \$24.72, no more transactions occur, and the new current bid becomes 50 $(= 1250 - 200 - 1000)$ shares at \$24.70.

Similarly, if a sell order is received with a price that is less than or equal to the current bid, one or more transactions occur. These transactions stop when either the total volume of the incoming order has been traded or the residual volume of the incoming order becomes the current ask (at the price of the incoming order).

Orders are of two types: limit orders and market orders. A limit order, which guarantees a price, goes onto one of the books, and is generally not immediately matched. A market order is an order that is immediately executed at the best prices available, i.e. the current ask price for buy orders or the current bid price for sell orders. With PXS a market order can be effected by placing either a bid at a very high price or an ask at a very low price, thereby guaranteeing matchings.

A PXS simulation consists of one day's trading on a single stock from 9am until 4pm (simulator time) with any number of agents. PXS can read in orders from a historic data file to simulate trading on a specific day in the past, or can be run in real-time, reading in live orders directly from the Island ECN order books. In both cases the real-life orders are mixed in with agent orders to create a realistic trading environment (see [4] for details). Unless in live mode, a simulation is run in accelerated time, and typically takes between 15 minutes and 1 hour, depending on the number of agents and their complexity. The state of the limit order books that PXS maintains is made available to all participating agents. PXS also provides agents with other basic statistical information, such as the quartiles of the books, total volume of the books, etc. One of the most important pieces of information is the 'last price', which is the last price at which a transaction occurred. Agents typically build up a history of last price values and use this to determine their trading actions.

3 The competition

3.1 Financing of agents and liquidation

At the beginning of each simulation every agent starts with no cash and no shares. The agents are allowed to borrow cash (without interest) and 'short' shares, that is, sell shares they do not own which are to be bought back later. Agents can take arbitrarily long (positive) or short (negative) positions in both cash and shares. The final profit of an agent with a nonnegative end-of-day share position is equal to its final cash position, so the value of any remaining shares held is lost. The final profit of an agent with a negative end-of-day share position is equal to its final cash position minus twice the value of the shorted shares. In this way agents are encouraged to *liquidate* by the end of the day, that is return its share position to zero. At any point in a simulation, the sum of the share positions of all agents is zero. However, the cash positions of the agents will not sum to zero, since certain transaction costs are imposed. These are outlined next.

3.2 Transaction costs

A transaction occurs when an incoming order is matched with one that is already present in the order books. When such a pair of orders is matched, a fee of \$0.003 per share is charged to the incoming order, and a rebate of \$0.002 per share is given to the order that was already in the books. The setup is the same as on the Island ECN, and is designed to encourage liquidity.

3.3 Competition setup and ranking of agents

The final competition consisted of eleven separate simulations, each under different market conditions. In each simulation all competing agents were run simultaneously against each other, along with some other basic agents supplied by the organisers, which are briefly described in Section 4.

The daily profit of each agent was recorded. The *Sharpe ratio* is a standard measure of the performance of a trading agent over multiple trading days:

$$\text{Sharpe ratio} = \frac{\text{average profit}}{\text{standard deviation}} \tag{1}$$

(where the average is taken as the mean). The winning agent was the one with the highest Sharpe ratio, which encouraged large, but also consistent profits. This is discussed in Section 6.

4 Agents

Agents are computer programs that connect to the PXS server to place orders to buy or sell (virtual) shares. During the simulation they may request statistical information from PXS about the state of the order books, and previous transactions. Orders are always submitted as limit orders, but as mentioned previously a market order can be effected by placing a limit order at an extreme price. Typically, agents build up a history of last price values (prices at which transactions occurred) and use this to predict price movement. In the past, traders had to rely heavily on last price information and the current bid/ask to make trading decisions, as very little information about the order books was made available by brokers and exchanges. In recent years ECNs have started publishing detailed information about the state of their order books, and one of the challenges of the PLAT competition is to make good use of this information. However, few agents have actually used this extra information, and only basic methods have been employed.

Several basic agents are supplied by the organisers of the competition in order to test strategies. They can be divided into two types: background agents, and basic technical agents. These are briefly described next.

4.1 Background agents

The background agents are designed to provide liquidity to the market and create a more realistic simulation. There are three background agents - the symmetric, asymmetric, and real background agent. The first two periodically (typically every 10 seconds) produce a pair of buy and sell orders at fixed volume and random price. The prices are sampled from normal distributions whose means are relative to the current bid/ask. The real background agent produces orders according to files containing historic trading information. There is one file for buy orders and one for sell orders, each contains time, volume and *distance* information. The distance information tells the real background agent at what distance from the current bid/ask to place the order. For all three types of background agent, the order prices are generated relative to the current bid/ask. This allows more interaction with the trading behaviour of other agents, and so perhaps more realistic market behaviour. However, it also means any absolute price limits that may have been enforced by real traders are lost. This was a weakness that was exploited to great effect by the Jump & Dump agent.

4.2 Basic technical agents

Six basic technical agents were provided for testing. These agents were also used in the competitions. The agents were moving average, momentum, channel breakouts, market maker, relative strength index, and static order book imbalance. The first three agents, moving average, momentum, and channel breakouts, all maintain an array of historical last price values for some fixed period into the past (default setting is just 3 minutes). They base their buy and sell decisions on a comparison between the current last price value and some simple statistic derived from the history array (average value, first value, and extreme values respectively). In general, the market maker agent is the most profitable of these basic agents. It periodically places a pair of buy and sell orders close to the current bid/ask, obviously with the buy order priced lower than the sell order. If there is enough up and down movement in the price then both orders are matched and the agent makes profit from the difference in price of the orders. Thus the market maker exploits the volatility of the price and does best when this volatility is high. The relative strength index agent uses the sequence of last price values. It counts the length of runs of consecutive upward and downward movements and uses this information to predict when the price is going to change direction. It then buys or sells accordingly. The static order book imbalance agent compares volume weighted average prices of the bid and ask books, and buys or sells if one is much bigger than the other (if they are both roughly the same no action is taken). For each of these types of agent, buy and sell orders are placed at prices relative to the current bid/ask.

4.3 Other competitors

This year there were fewer competitors than in previous years, with eight in total. A preliminary competition was held in December 2004 which gave participants the opportunity to test out agents and get some idea of the sort of strategies that might be used in the main competition. Many of the competitors (including the authors) tried some variation of market making. The more successful ones applied different strategies depending on the market conditions, for example using market making in times of high volatility and using some trend following strategy, such as moving average, momentum, or channel breakouts, when there is general upward or downward price trend. Two of the agents (Sohn and Ricketts) tried using order book information to predict price movement (by comparing the bid book with the ask book). Other agents maintained an array of last price values and used this to predict volatility and price movement. The winner (and runner up for the main competition) was an entry by Viren Kumar and Mohit Mutreja which used a mixture of three different basic strategies: a market making variant, a trend following momentum strategy, and a 'contrarian' strategy, which buys if it detects downward price movement and sells if it detects upward movement. After this competition the authors noted that all these agents priced their orders relative to either the current bid/ask or the last price.

5 Jump & Dump

The authors' final entry for the May 2005 Penn–Lehman automated trading competition was a very simple but sneaky strategy, dubbed 'Jump & Dump'. Rather than passively trying to predict stock price movement and trade accordingly (as the other competitors did), Jump & Dump actively took control of the market to manipulate the trading price to its advantage. Two important factors enabled it to do this. Firstly, since no share position limits were imposed, Jump & Dump was able to take massive long positions in the traded stock. Secondly, none of the other competing agents had any reality checks built in to see whether prices were sensible.

One of the parameters of Jump & Dump is 'grossProfit', the desired level of profit (before transaction costs[2]). The aim was to ensure not only high average profit, but also consistent performance, and hence a high Sharpe ratio.

[2] It would be hard to account for transaction costs accurately, since PXS does not make this information available to agents. We chose to ignore them altogether, since they make little impact on our final profit.

Jump & Dump (without signalling)

1. Buy lots of shares
2. Buy all the shares in ask book. Figure out at what price we would need to sell all our shares at in order to make enough profit (call this the *jump price*). Place a pair of buy/sell orders for just one share each close to the *jump price*.
3. Continue to buy any shares in the ask book that are priced lower than the *jump price*, and keep placing small buy/sell orders close to the *jump price*. Continue this behaviour for 40 minutes.
4. Sell off all shares until cash level = grossProfit.

Figure 2: Outline of basic Jump & Dump agent without signalling

The results were spectacular: Jump & Dump completely dominated the competition, with profits at least ten times higher than our competitors in every simulation. In previous competitions the highest daily profit achieved had been $33,387 (Nevmyvaka 5/5/2003), whereas Jump & Dump achieved an *average* profit of $734,810,063, and a Sharpe ratio of 3.87, more than twice that of our nearest competitor (Kumar, with a Sharpe ratio of 1.33), and again higher than previous records. However, the results were not as good as we had hoped, as we had set the grossProfit parameter to $1,000,000,000 and were expecting a much higher Sharpe ratio. The reason this did not happen is explained later. Figure 2 gives a brief outline of the basic strategy.

PXS provides agents with the order book details that are required for Step 2. After emptying the ask book, any sell order placed will become the new current ask, and any buy order higher than the current bid will become the new current bid. Jump & Dump chooses a jump price such that if it sold its shares at that price it would be able to make at least the required profit level. In the following equation, the coefficient of 1.5 is used to account for the extra shares bought in Step 3.

$$\begin{aligned} jump\ price &= (1.5 \times grossProfit)/(shares\ held) + last\ price \\ \Rightarrow 1.5 \times required\ grossProfit &= (jump\ price - last\ price) \times (shares\ held) \\ &= approximate\ profit\ from\ selling\ all\ shares \end{aligned}$$

The current bid and ask are then set by placing a buy order for one share at the jump price, and a sell order for one share at jump price + $0.0001. The next incoming order that matches will match at one of these prices. This behaviour in maintained in Step 3 for 40 minutes in order to make sure the history arrays of the other agents fill up with these inflated prices. In Step 4 the profits are reaped by selling off the shares at this inflated price. Since the other agents have history arrays filled with inflated prices, and make orders relative to the current bid/ask, they are willing to trade at this inflated price level. Any downward price movement is likely to be minute compared to the price hike created in Step 2. The liquidation penalty ensures that the value of any positive share position is ignored when working out final profit. Hence, we only need to sell enough shares until the required profit level is reached. This ensures consistent profit levels, and hence a high Sharpe ratio. Figure 3 shows the cash value of Jump & Dump in a typical simulation. At 12:25 Jump & Dump buys all shares in the ask book causing the sudden drop in cash seen on the graph. Between 13:25 and 14:00 the shares are sold off until the cash level rises above $1,000,000,000.

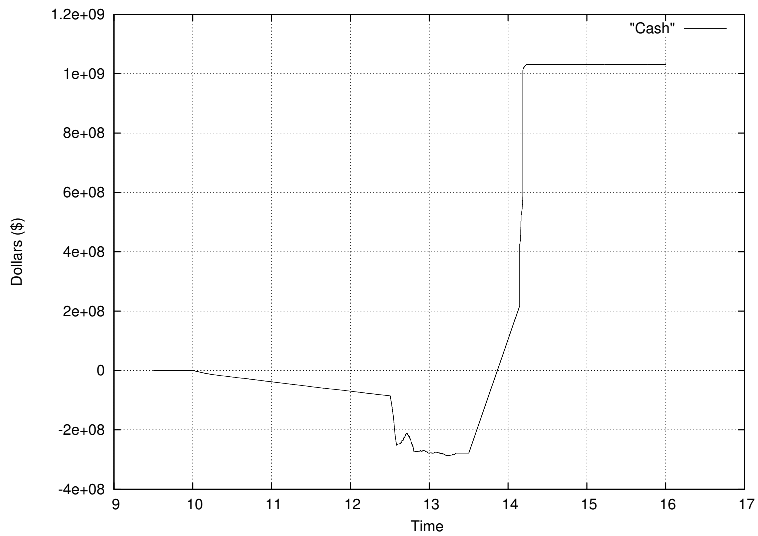

Figure 3: Cash position of Jump & Dump in a typical simulation

In practice the actual final cash value achieved is often out by a factor of 20% or more. A main reason for this is that often there is not enough volume in the bid book at the inflated price to match all of the sell orders placed by Jump & Dump at this stage. All the inflated price buy orders are used up until only the low priced ones (most likely placed by other agents before Step 2) are left, so the current bid jumps back down again. Any agents placing sell orders relative to the current bid will cause the current ask to also jump back down. Using a larger coefficient of grossProfit (set at 1.5 for the competition) would probably have fixed this problem since fewer shares would need to be sold. This problem was overlooked since it did not occur while we were testing the algorithm (time constraints meant we could not run a lot of tests). Figure 4 shows the price behaviour in a typical simulation. There is some fluctuation in price at around 12:30 while Jump & Dump tries to establish the new price level. Then just after 14:00 the price jumps back down due to all the high-priced bids getting matched, and subsequent matchings occur at low prices.

5.1 Signalling

In the preliminary competition agents were run against themselves in some of the simulations. Figure 5 shows the cash value of two Jump & Dump agents run in the same simulation (with a few other basic agents). One of the agents does as intended, with a

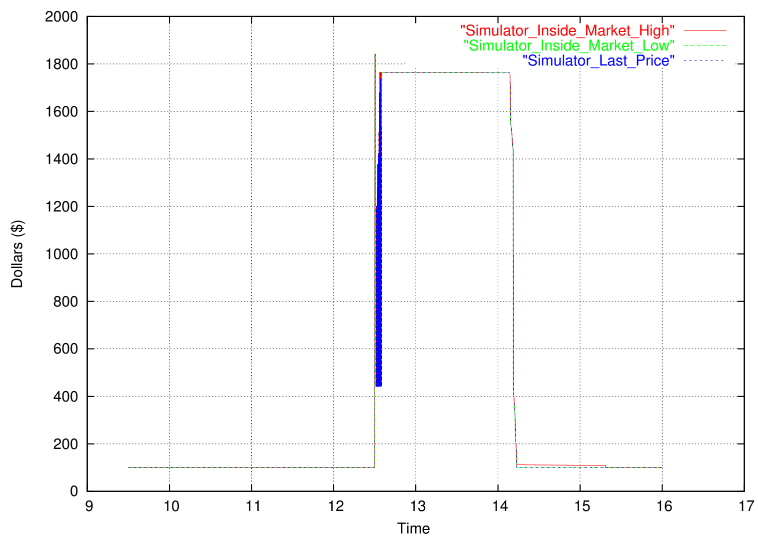

Figure 4: Simulator price in a typical simulation involving Jump & Dump

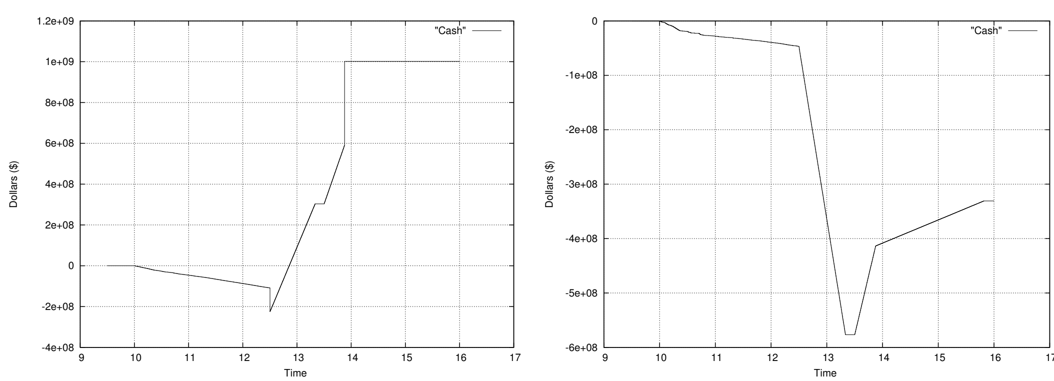

Figure 5: Cash value of two Jump & Dump agents (without signalling) run in the same simulation

final cash value close to \$1,000,000,000, but the other does very poorly. The reason for this is that both agents reach Step 2 at the same time, and both try to buy all shares in the ask book in a single order. PXS processes the orders one at a time, and so whichever agent has their order processed first will get the shares, whereas the other agent will have their order left on the bid book unmatched. When they do get matched it will be at the price the order was placed at, i.e. the jump price. Also, with two agents simultaneously dumping shares in Step 4, the high-priced bids get used up before the second agent gets a chance to recuperate its losses. To overcome this problem a signalling protocol is used: each Jump & Dump agent makes its presence and share position known to all other Jump & Dump agents. They can then divide the ask book between themselves at Step 2. Although the rules of the competition did not prevent us using external communication channels, constraints in time and programming skills led us to use the bid book for passing signals (this idea has been applied before in real life telecommunications auctions - see Section 6). In the modified strategy, the agent places a special buy order to signal its share position to other possible Jump & Dump agents. Figure 6 outlines the modified algorithm.

Jump & Dump with signalling

1. Buy lots of shares.
2. Place signal order indicating share position.
3. Retrieve signal orders from bid book. Work out common *jump price* appropriate for all Jump & Dump agents, and individual *jump volume* = amount of ask book that this agent should buy (details below).
4. Buy *jump volume* from ask book.
5. Maintain inflated price for 40 minutes by buying anything in the ask book that is priced lower than *jump price*, and periodically place high buy and sell orders.
6. Sell off shares until cash level = grossProfit.

Figure 6: Outline of modified Jump & Dump agent with signalling

The signal order is for just one share at a price between zero and one. This is an unusual bid and is very unlikely to get matched (since the price is so low). Also, since most orders are typically for 300 or more shares, it is unlikely that non Jump & Dump agents will place a similar bid. The price of the bid indicates the agents share position: $signal\ price = (share\ position)/(some\ fixed\ large\ number)$. In Step 3, the number of signal orders found in the bid book indicates the number of Jump & Dump agents in this simulation, say α. Summing the share positions gives the total share position of all Jump & Dump agents, call this β. Let *prevShare* equal the number of shares held by an agent before emptying the ask book. For a particular agent, the jump volume and price are set to:

$$\begin{aligned} jump\ volume &= \frac{total\ volume\ of\ ask\ book + \beta}{\alpha} - prevShare \\ jump\ price &= \frac{1.5 \times grossProfit \times \alpha}{total\ volume\ of\ ask\ book + \beta} + last\ price \end{aligned} \tag{2}$$

Then if all goes to plan we have:

$$\begin{aligned} & approximate\ profit\ from\ selling\ all\ shares \\ \approx\ & (jump\ price - last\ price) \times (total\ shares\ held) \\ \approx\ & (jump\ price - last\ price) \times (jump\ volume + prevShare) \\ =\ & 1.5 \times grossProfit \end{aligned}$$

With these values all Jump & Dump agents should be able to achieve the required cash level in Step 6 by selling off just two-thirds of their shares. However, the problem of running out of high-priced buy orders still persists, and so the agents do not always make the required cash level. As Figure 7 shows, the signalling method works. In fact it should work with any number of Jump & Dump agents, although the most we have tried is four. As it turned out, duplicate agents were not used in the final competition, and so the signalling procedure was redundant.

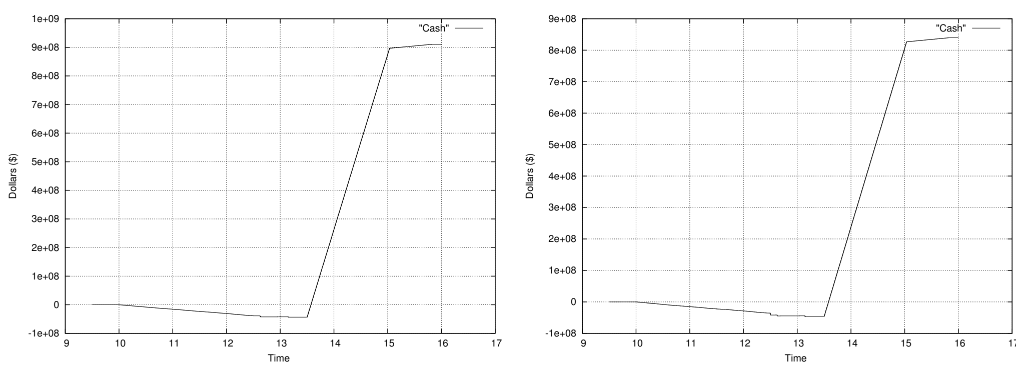

Figure 7: Cash value of two Jump & Dump agents run in the same simulation

In the final competition the only other agents to make any profit were those of Viren Kumar and Mohit Mutreja, winners of the preliminary December 2004 competition. All other competitors ended up with large losses. All opponents tried some variation of market making combined with some other simple strategies, but as in the preliminary competition they all priced their orders relative to the current bid/ask or last price. The full results can be found online at [6].

6 Conclusions

The success of Jump & Dump relied on the correct anticipation of the reactions of other competing agents to the unusual circumstances that Jump & Dump creates. Almost all agents submitted relative prices without checking whether the absolute price was sensible. This serves as a wake up call, not just to participants in future competitions, but also to those designing "real-world" automated trading strategies. The competition illustrates very clearly how much trouble an automated strategy can get in if it does not have sufficient "reality" checks built in. Simple checks on the price could have prevented the poor performance of most of the agents. Of course, it was very important that the competition was run with PXS in historic, but not live mode; it was known in advance that this would be the case.

How "realistic" is Jump & Dump? Could it be implemented in the real world? It is unlikely for such a strategy to so completely manipulate the market, since the numerous non-automated market participants would recognise the price anomaly. For a liquid stock it would require both deep pockets and a high tolerance for risk, because it would require taking a very long position. Furthermore, most stock exchanges have built in safety checks and will close the market if the price changes dramatically. However, for a low-liquidity stock with small book volume, it may be feasible to effect a small jump in price and stay within the limits of the exchange.

The possibility of having multiple Jump & Dump agents compete against each other necessitated the use of signalling. The bid books were used as the communication channel and this approach has been used before. For example, in the auction of mobile communication bandwidth by the FCC in the 1990's it was in the interest of the bidders to signal their preferences to each other. However, strict rules prohibited communication between bidding parties. To overcome this, some bidders used the last few digits of their bid prices to communicate their preferences. For more details see [1][2].

The Sharpe ratio, which is defined in (1), was used as the single measure of performance in the May 2005 competition. It is worth noting that the Sharpe ratio tends to infinity as the denominator, the standard deviation, tends to zero. Jump & Dump tried to achieve a large Sharpe ratio by limiting its profits to ensure that they were consistent. This highlights a problem with the Sharpe ratio: Any agent that can guarantee a positive average profit (no matter how small) may be able to engineer an enormous Sharpe ratio, by limiting its profits to ensure a small standard deviation.

7 Further work

Jump & Dump did not perform as well as we had hoped due to the problem of running out of high bids in Step 4 of Figure 6. This could have been remedied by setting the coefficient of grossProfit in (2) higher (than 1.5) when calculating the jump price, so that fewer shares would need to be sold to meet the required profit. Time constraints meant that we did not run enough tests beforehand to notice this problem.

What would be a good strategy against Jump & Dump? The only other agent to make a profit in the competition was that of Kumar and Mutreja (see [6]). This is probably because it correctly spotted and followed the price trends caused by Jump & Dumps excessive trading in Step 1 and Step 6 of Figure 6 (up and down respectively). However, Step 1 is not really necessary for Jump & Dump since the amount of shares bought in Step 4 is enough. A simple price check to prevent an agent from bidding too high would have prevented the excessive losses of many of the agents. If no agent is willing to buy shares at the inflated price level then Jump & Dump cannot succeed. It is easy for other agents to sell any amount of shares at any price lower than the jump price before Step 6 since Jump & Dump will buy them in order to maintain the inflated price level. This could lead to very high profits for other agents. This behaviour could also hinder Jump & Dump since it would need to sell more shares. If other agents employ price checks, a more effective Jump & Dump strategy might be to jump the price up (or down) more slowly, in several smaller steps. This strategy is also more likely to cause a rally amongst agents that detect the trend, making it easier to sell (or buy) back shares. These issues could be investigated further. It must be stressed however that Jump & Dump will not work if PXS is run in real-time mode with orders coming in from a real market (since the real traders are unaware of PXS internal trades and will not base their order prices around them).

Acknowledgements

The authors thank Michael Kearns and Elliot Yi Feng from the University of Pennsylvania for all their support, and Mark Baltovic for technical support at LSE.

References

[1] Cramton, P., and J. Schwartz (2000), Collusive Bidding: Lessons from the FCC Spectrum Auctions. *Journal of Regulatory Economics* **17 (3)** 229–52.

[2] Cramton, P., and J. Schwartz (2002), Collusive Bidding in the FCC Spectrum Auctions. *Contributions to Economic Analysis and Policy* **1** 1078–1078.

[3] `http://www.island.com`

[4] Kearns, M., and L. Ortiz (2003), The Penn–Lehman Automated Trading Project. *IEEE Intelligent Systems* **Nov/Dec 2003** 22–31.

[5] `http://www.cis.upenn.edu/~mkearns/projects/plat.html`

[6] `http://www.cis.upenn.edu/~mkearns/projects/newsandnotes04.html`