Efficient Trade Execution Using a Genetic Algorithm in an Order Book Based Artificial Stock Market

Wei Cui
Natural Computing Research & Applications Group,
University College Dublin, Ireland.
will.weicui@gmail.com

Anthony Brabazon
Natural Computing Research & Applications Group,
University College Dublin, Ireland.
anthony.brabazon@ucd.ie

Michael O'Neill
Natural Computing Research & Applications Group,
University College Dublin, Ireland.
m.oneill@ucd.ie

ABSTRACT

Although there is a plentiful literature on the use of evolutionary methodologies for the trading of financial assets, little attention has been paid to the issue of efficient trade execution. Trade execution is concerned with the actual mechanics of buying or selling the desired amount of a financial instrument of interest. This paper introduces the concept of trade execution and outlines the limited prior work applying evolutionary computing methods for this task. Furthermore, we build an Agent-based Artificial Stock Market and apply Genetic Algorithm to evolve an efficient trade execution strategy. At last we suggest a number of opportunities for future research.

Categories and Subject Descriptors

I.6 [**Simulation and Modeling**]: Miscellaneous; J.4 [**Social and Behavioral Sciences**]: Economics

General Terms

Economics, Experimentation, Performance

Keywords

Algorithmic Trading, Trade Execution, Volume Weighted Average Price, Artificial Stock Market, Genetic Algorithm, Evolutionary Computation

1. INTRODUCTION

Algorithmic trading (AT) can be broadly defined as the use of computers to automate aspects of the investment process. Hence, AT can encompass the automation of decisions ranging from stock selection for investment, to the management of the actual purchase or sale of that stock. A significant proportion of all financial asset trading is now undertaken by AT systems with this form of trading accounting for approximately 20-25% of total US market volume in 2005. Boston-based research firm Aite Group predicts that AT will account for more than half of all shares traded in the U.S. by the end of 2010 [12]. AT is also common in European financial markets with approximately 50% of trading volumes being accounted for by algorithmic trading programs [9]. Significant volumes in Asian markets are similarly traded [8]. Algorithmic trading is seen in multiple financial markets ranging from equities to FX (foreign exchange), to derivative (futures, options etc.) markets.

In this paper we restrict attention to one aspect of financial trading to which AT can be applied, namely efficient trade execution. A practical issue that arises for investors is how they can buy or sell large quantities of a share (or some other financial asset) as efficiently as possible in order to minimize market impact and information leakage. Typically, orders to buy or sell a share can be either *market orders* (the transaction is undertaken immediately in the market at current prices) or limit orders (the purchase (sale) must occur at a price which is no greater than (or less than) a pre-specified price). So for example, if a customer places a limit order to buy a stock at \$25 per share the transaction will only take place if the market price falls to \$25 or less. Hence, when a market order is placed, the customer does not have control over the final price(s) at which the order will be filled, and in a limit order, while the customer has some price control, there is no guarantee that the order will actually be executed.

Most major financial markets now operate based on an electronic order book, where participants can see the current unfilled buy and sell orders. Table 1 illustrates a sample order book, showing the quantities that investors are willing to buy (bid side) and sell (ask side) at each price. We can see that 187,400 shares are currently available for sale at a price of 133.1 (or better), and buyers are seeking 332,100 shares at a price of 132.9 (or less). The order book also illustrates that there are limits to the quantity of shares available for purchase / sale at each price point. Of course, the order book is highly dynamic, with the quantities of shares offered at each price changing constantly as trades are executed, as investors add new limit orders on the bid and ask sides, or as investors cancel limit orders they have previously placed.

Table 1: Sample order book for a share with volume and price information for bid and ask

Bid		Ask	
Vol	Price	Price	Vol
332,100	132.9	133.1	187,400
134,900	132.6	133.3	211,800
342,700	132.5	133.4	242,900
86,700	131.8	133.5	142,400
124,500	131.7	134.6	93,400
78,100	131.3	134.7	187,400

GECCO'09, July 8–12, 2009, Montréal Québec, Canada.
ACM 978-1-60558-505-5/09/07.

When trading financial assets, particularly when an investor is looking to buy or sell a large quantity of the asset, the problem of *market impact* arises. Market impact arises when the actions of an investor start to move the price adversely against themselves. Hence, market impact is the difference between a transaction price and what the market price would have been in the absence of the transaction. For example, if an investor wished to buy 300,000 shares given the above order book, he would end up driving up the price paid for some shares to 133.3. The obvious strategy to minimise market impact is to break up the order up into smaller lots and spread it over several purchases. While this will reduce the market impact, it incurs the risk of suffering *opportunity cost*, that market prices may start moving against you during the multiple purchases. Hence, the design of trade execution strategies when trading large blocks of financial assets is intended to balance out these factors.

A popular benchmark for evaluating execution performance is *volume-weighted average price* (VWAP) which is calculated across the time horizon during which the trade was executed and is calculated as

$$VWAP = \frac{\sum(Volume * Price)}{\sum(Volume)}$$

If the price of a buy trade is lower than VWAP, it is a good trade. If the price is higher, it is a bad trade. Although this is a simple metric, it largely filters out the effects of volatility, which composes market impact and price momentum during the trading period [1].

The task in devising an efficient execution strategy is complex as it entails multiple sub-decisions including how best to split up the large order, what *style* to adopt in executing each element of the order (aggressive or passive), how execution performance is to be measured, and what type of order to use. When an asset is traded simultaneously on multiple markets a further decision must be made as to how to split up the order amongst these markets. In addition, the electronic order book(s) faced by the investor are constantly changing.

In the past the task of designing an execution strategy was undertaken by human experts but it is amenable to automation due to the high-frequency market. This highly dynamic environment in which a trade is executed suggests that adaptive algorithms may have particular utility. In this paper we apply a Genetic Algorithm (GA) to evolve an efficient trade execution strategy and highlight other possible Evolutionary Computation (EC) applications for this issue.

To test the performance of the trade execution strategy, we also need to collect desired order flow data. An ordinary way is to obtain the data from the exchange. This only provides us with a single sample path of order book data over time. Another approach is to consider the output data from the currently popular *Artificial Stock Market* (ASM), a simulation of the real stock market. An advantage of a simulation-based approach is that many sample paths can be generated and utility of a trade execution strategy can be tested over all of these paths. Most ASM models are built by a computer technique called *Agent-based Modeling* (ABM) which is a hot research topic in fields of social sciences. It has been applied to finance for many years. ASM is modeled as multiple interacting agents with a price formation mechanism to investigate the emerging characteristics of financial markets. Many researchers have suggested ASM as a alternative market in place of real market to test trade strategies. Novelly, this paper evolves an efficient trade execution strategy using Genetic Algorithm and to evaluate the strategy employing an ASM.

1.1 Structure

This paper is organized as follows: the next section will give a brief description of application of EC in trade execution; section three will introduce the ASM model used and shows how to simulate the ASM to generate the desired order flow data; the followed section will demonstrate how the order flow data is used to evolve the efficient execution strategy deploying GA methodology; last section will conclude this paper by giving a number of avenues for future work.

2. EVOLUTIONARY COMPUTATION AND TRADE EXECUTION

Most published papers on trade execution adopt a dynamic programming approach [2, 5, 6, 16, 17], where given a fixed block of shares to be traded within a fixed number of time periods, and given a price-impact function that yields the execution price of an individual trade as a function of the shares traded and market conditions, the object is to determine the optimal sequence of trades as a function of market conditions (closed-form expressions in some cases) that minimize the expected cost of executing the entire transaction [2].

2.1 Evolutionary Computation

Evolutionary computation, includes GA, Genetic Programming (GP), Particle Swarm Optimization (PSO), etc., applied in two basic aspects of finance, which are optimization and model induction [4]. Optimization application are due to the complexity and high dimension of some financial problems, such as portfolio selection. The task of model induction is often to discover the unknown models of some financial processes from a dataset.

Genetic Algorithm is a heuristic function for optimization. A population of potential solutions to the optimization problem is evolved iteratively using Darwinist evolution or natural selection. According to the algorithm "survival of the fittest", the final result at the end of the evolution is the best solution at that time. It has already been used in many areas of finance.

2.2 Literature Review

Despite the importance of optimizing trade execution, there has been relatively little attention paid in the literature to the application of evolutionary methodologies for this task. One notable exception is Lim and Coggins [15] who applied a genetic algorithm (GA) to evolve a dynamic time strategy to optimize the trade execution performance using order book data from a fully electronic limit order market, the Australian Stock Exchange (ASX). In their study, the total volume of the order was divided into 10 slices and was traded within one day using limit orders. Each evolved chromosome had N genes where each gene encoded the maximum lifetime that an individual order ($1 \rightarrow N$) would remain on the order book (if it had not already been executed) before it was automatically ticked over the spread to close out the trade. The fitness function was the VWAP performance of that strategy relative to the benchmark daily VWAP. Each strategy was trained on three months' worth of transaction-level data using a market simulator. The results were tested out of sample on three highly liquid stocks and tested separately for sell side and buy side. The in sample and out of sample performances were better than pure limit / market order strategies.

3. AGENT-BASED ARTIFICIAL STOCK MARKET

Agent-based modeling is one of the most exciting practical developments in modeling complex systems. In ABM, the system is modeled as a collection of autonomous, interacting agents. Each agent makes decisions based on a set of rules. Even a simple agent-based model can exhibit complex behavior patterns and provide valuable information about the dynamics of the real-world system that it emulates [3].

Financial markets have been widely recognized as complex systems comprised of interacting agents and price formation mechanisms. A wide range of financial markets, like stock markets, foreign exchange markets and option markets, have been modeled with ABMs. Artificial stock market simulation is the hottest topic in the realm of agent-based computational economic/finace (ACE). A review of ASM can be found in [14].

This section initially introduces the limit order book mechanism and then describes the Swarm platform used in our experiments.

3.1 Limit Order Book Mechanism

Here we want to build a continuous double auction artificial stock market with a limit order book mechanism. The output data flow (a simulated order book) from the ASM will be used to test trade execution strategies, which are required to include detailed information of all orders, such as each order's arrival time, executing time, trade size and order price.

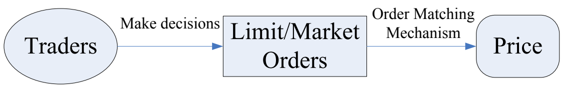

Figure 1: Price Formation

In a limit order market, orders arrive randomly in time. The price limit of a newly arrived order is compared to those of orders already held in the system to ascertain if there is a match. If so, the trade occurs at the price set by the first order. The set of unexecuted limit orders held by the system constitutes the dynamic order book, where limit orders can be cancelled or modified at any time or executed in price priority and time priority sequence [11]. These markets are usually transparent, with state of the book being widely visible to actual and potential market participants. These markets also provide hidden and reserve orders which are entirely or partially invisible. A simple price formation is illustrated in Figure 1.

3.2 Simulation

In this paper, the zero-intelligence (ZI) model in [7] is adopted, because this model is aiming to generate a realistic aggregate order flow using simplest assumptions. The ZI model models an asynchronous double auction market, which is entirely defined by the characteristics of the agents responsible for generating the order flow. It features a realistic microstructure populated with ZI agents, which originating from Gode and Sunder [10] mean that the agents place random orders to buy or sell. In this model, only one stock is traded, and dividends are ignored. Traders trade order via a centralized limit order book, without the intermediacy of a market maker, aiming to focus on the dynamics of a pure double auction. There are four aspects to design the ZI model, which are order sign, order type, limit order price and order size.

Table 2: Initial Parameters for Order Book based ASM

Explanation	Value
Initial Price	$price^0 = 100$
Tick Price	$\delta = 0.001$
Probability of Cancellation Order	$\lambda_c = 0.5$
Probability of Market Order	$\lambda_m = 0.15$
Probability of Limit Order	$\lambda_l = 0.35$
Probability of Limit Order in Spread	$\lambda_{in} = 0.35$
Probability of Limit Order Out of Spread	$\lambda_{out} = 0.65$
Limit Price Tail Index	$1 + \alpha = 1.3$
Order Size	$(\mu, \sigma) \sim (4.5, 0.8)$ shares
Waiting Time	$\tau = 44$ seconds

There are only two order signs, buy or sell. It is simple to consider that ZI agents have the equal probability to make a buy order and a sell order. When an agent is active, she can try to issue a cancellation order with probability λ_c (oldest orders are cancelled first), a market order with probability λ_m, a non-marketable limit order with probability $\lambda_l = 1 - \lambda_c - \lambda_m$. In this model, limit order price will be uniformly distributed in the spread with probability λ_{in}, and power-law distributed outside the spread with probability $1 - \lambda_{in}$. Limit order price ranges are illustrated in Figure 2. Here, market orders will simply have the same volume as the best counterpart order that they are about to remove, and non-marketable limit orders will follow a log-normal distribution. The parameters used in our simulation are presented in Table 2.

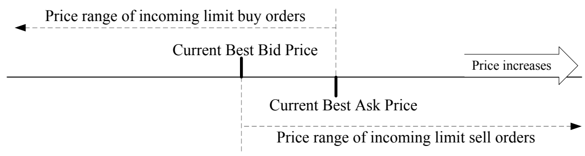

Figure 2: Place Limit Price

A striking phenomenon prevalent in ASM is the drastic simplification made about time. In this model, the order generation is modeled as a poisson process, which means that the time between orders follows an exponential distribution. In our simulation, we adopt a Swarm platform in JAVA, which is one of the most popular agent-based modeling platforms, instead of discrete-event simulation in [7]. The Swarm algorithm used in our simulation is described in Figure 3.

4. EXPERIMENT

As we have built an order book based ASM to generate order flow in last section, this section will describe how to use a GA to uncover a quality trade execution strategy.

```
Simulator:generate t from EXPONENTIAL(τ);
current_time = current_time + t.

Agent:generate P_sign from BERNOULLI(0.5);
generate independent P_type from UNIFORM(0,1]

if P_type <= λ_c then
    /* a cancel order to be submitted */
    Cancel oldest outstanding order
end
else
    if P_type > (λ_c + λ_m) then
        /* a limit order to be submitted */
        generate OrderSize log(vol) ~NORMAL(μ,σ);
        generate independent P_spread from UNIFORM(0,1];
        if P_spread <= λ_in then
            /* limit price to be in the spread */
            generate LimitPrice price(t) ∈ UNIFORM(b(t),a(t))
        end
        else
            /* limit price to be out of the spread */
            generate LimitPrice price_i(Δ) ~ 1/Δ^(1+α);
            /* power-law distributed*/
        end
    end
    else
        /* a market order to be submitted */
        generate OrderSize vol(t) = [vol(a(t))|vol(b(t))];
        /* same size as best counterpart*/
    end
end
```

Figure 3: Behavior of ZI Agent

4.1 Representation

The design of an execution strategy can be considered of consisting of two step. First is to divide a big block of shares into multiple small orders. Second step is to determine the parameters of each order, including order type (limit/market order), submitted time, limit price (limit order) and lifetime (the time length when an order appears in the order book before it is cancelled).

a_1	b_1	......	b_{30}	c_1	......	c_{30}

a_1: submitting time of first order
b_1 - b_{30}: lifetimes of 30 orders
c_1 – c_{30}: submitting limit prices of 30 orders

Figure 4: Representation

In our experiment, each large trade is divided into 30 orders. Because limit orders are preferable to market orders (as they achieve better execution prices), we only adopt limit orders. The orders are submitted to the market one by one, with each order being submitted at the end of the lifetime of the previous order. As in the real market, divided orders can always be fully traded if they are small enough. We assume that every market buy/sell order has the same size as the best limit sell/buy order in the order book, which is in accordance with the assumption in the ASM simulation. This means that one market order will cause only one limit order to be traded. So the parameters left for the 30 limit orders include limit order's price, lifetime in the order book before canceled if not executed by other market orders, and the submitted time of the first order, which will be determined by the GA methodology. Figure 4 shows the representation of each GA individual or chromosome. These parameters of every GA individual form a GA strategy. The purpose of this experiment is to evolve an efficient execution strategy which has the best average execution price.

The objective function we used here is ratio of the difference between the VWAPs of the 30 orders and the entire executed orders generated from the ASM simulation, which are $VWAP_{30}$ and $VWAP_{global}$ respectively, to the the entire executed orders' VWAP. For both buy and sell orders, the smaller the VWAP Ratio is, the better the strategy is.

$$VWAPRatio = \begin{cases} \frac{VWAP_{30} - VWAP_{global}}{VWAP_{global}} & \text{Buy} \\ \frac{VWAP_{global} - VWAP_{30}}{VWAP_{global}} & \text{Sell} \end{cases}$$

4.2 Data

The data used in this paper is generated from the ASM in last section. The ASM simulation uses a database to store the details of each incoming order and best orders at each time point. It consists of five tables shown in Figure 5, recording all kinds of orders, which are limit buy orders, limit sell orders, market buy orders, market sell orders and best buy/sell orders. The limit buy/sell order tables record each limit order's index number, arrival time, volume, submitting limit price, time when canceled or traded. The market buy/sell order tables record each market order's index number, arrival time, traded volume, traded price and the index number of corresponding traded limit order. The best order table records the best bid and ask orders' index number, volume, price, and mid-spread price at each time when new order comes.

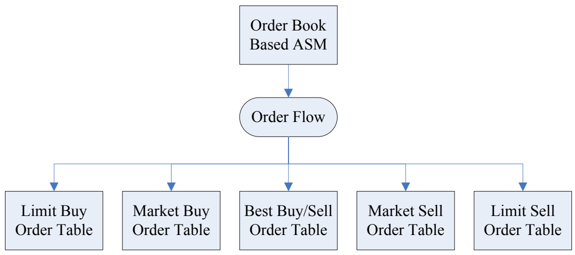

Figure 5: Data

In the ASM, all the data generated are stored in a database. We put them in different datasheets. These data contain the best orders (the best bid/ask prices, volumes and each recording time), limit orders (each limit order's size, limit price, submission time, execution/cancelation time) and market orders (each market order's size, arrival time, execution price). The ASM simulation was run for 10,081 seconds, corresponding to one day (8 hours).

4.3 Parameter Settings

In each generation of GA computation, several new individuals are produced, each being a strategy which defines how to send the 30 orders into market. To test the performance of each strategy, we incorporate the 30 new orders into the existing order flow generated from ASM in last section. The new order flow is simulated as a market, where new order will be traded. And we also assume that orders executed do not impact on the orders which arrive in the order book later. This is not real in the high frequency financial market. Figure 6 illustrates how the experiment works. In our experiment, orders can be executed in three different ways. The GA generates the limit price for each limit order. At the time when limit order is submitted to the order book, if the limit price crosses the best price in the opposite side of the order book, it will be executed immediately at the current best price as a marketable limit order (MLO). For instance, if the limit price of a buy order generated from GA is higher than the best ask price, this limit buy order is traded at the best ask price. If an order can not be executed during its lifetime, it will be automatically traded as a market order (MO) at the best price at the end of its lifetime. The last possibility is that the limit order (LO) is traded during its lifetime.

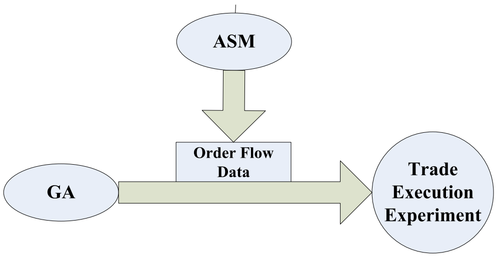

Figure 6: Experiment

We used a population of 25 individuals, running for 450 generations, to evolve an efficient GA strategy and test it with in-sample data and out-of-sample data separately. The parameters used in GA can be seen from Table 3. At the same time, we adopted another strategy to be compare with our GA strategy, which is pure market order strategy (MOS). It trades orders as market orders immediately after submission to the market.

Table 3: Parameters for Genetic Algorithm

Population size	25
Maximum number of generation	450
Generation gap	0.8
Crossover rate	0.8
Mutation rate	0.0005
Selection method	Stochastic Universal Sampling
Crossover method	Single-Point

4.4 Results and Discussion

Running both simulations for buy orders and sell orders over 30 trials, we obtain the results shown in Tables 4 & 5. The 'Best' provides the best result for objective value VWAP Ratio. The best results averaged over 30 trials are reported in the 'Mean'. The VWAP Ratio reveals the difference between the volume weighted execution price of GA orders and the average traded price of all orders during the whole simulation time. The better strategies have smaller VWAP ratios. The VWAP ratio of pure market order strategy, namely MOS, is also shown in Tables 4 & 5. In order to analyze the GA strategy, the execution types of the 30 orders are also calculated in our experiment. The three types are MLO, LO and MO.

From Tables 4 & 5, the GA strategy outperforms MOS strategy significantly, both in-sample and out-of-sample tests, which is consistent with the results in [15]. And the two tables both shows that the GA strategy, which has more orders executed in the way of LO, has smaller VWAP ratio, meaning better performance. All the GA strategies with negative VWAP ratios have more orders executed in the way of LO than that executed in two other ways, except the best out-of-sample strategy in Table 5.

Table 4: Results of Buy Order.

	MOS	GA Strategy			
	VWAP Ratio (10^{-3})	VWAP Ratio (10^{-3})	TradedOrderType		
			MLO	LO	MO
In-sample					
Best	5.1029	-2.1741	7	15	8
Mean	5.4580	-1.7998	8	13	9
Out-of-sample					
Best	2.5061	1.2779	12	3	15
Mean	3.4146	2.4061	13	3	14

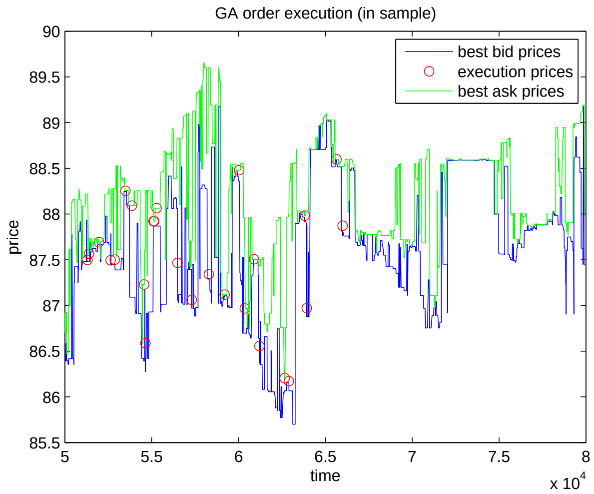

Figure 7: In-sample Results for Buy Order

Table 5: Results of Sell Order.

	MOS	GA Strategy			
	VWAP Ratio (10^{-3})	VWAP Ratio (10^{-3})	TradedOrderType		
			MLO	LO	MO
In-sample					
Best	2.1644	-4.2316	6	13	11
Mean	2.6200	-3.3232	9	12	9
Out-of-sample					
Best	1.9005	-0.4931	9	6	15
Mean	1.5803	0.6830	10	5	15

Also, GA strategies have achieved better VWAP than that of the whole simulation time for buy and sell in in-sample test, which is showed by the negative values of VWAP ratios. This is more significant for the sell order. It is easy to see this from Figures 7 & 8. Most of the execution prices, red circles in these figures, are below the average price in Figure 7 and are above the average price in Figure 8. But it is not significant for out-of-sample tests. However, these results suggest the applicability and potential of GA in trade execution.

5. CONCLUSION AND FUTURE WORK

In this paper, we present an evolutionary approach to the trade execution problem. Initially, we built an order book based artificial stock market (ASM) using agent-based modeling, which is a powerful computer technique for model-

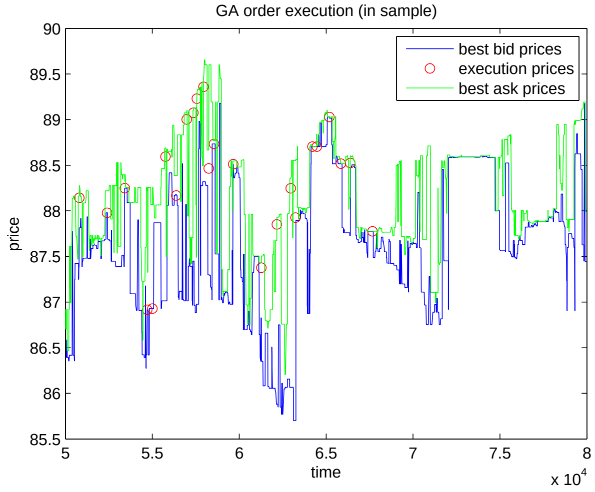

Figure 8: In-sample Results for Sell Order

ing complex systems. The platform used here is a Swarm platform in JAVA, which is open source. Using the order flow produced by the ASM, we applied a Genetic Algorithm for optimizing the parameters of efficient trade execution strategies, in order to achieve a better execution price than the currently popular benchmark Volume Weighted Average Price (VWAP). In our experiments, GA evolved strategies provide satisfactory results for this trade execution problem, indicating Evolutionary Computation methodologies have potential applications in the domain of trade execution. The success of applying order book based ASM for trade execution experiment suggests an alternative way for testing trade execution strategies, instead of using backtesting strategies based on historical market data.

In future work, we intend to extend the application of the EC to harder and dynamic optimization problems in trade execution. For instance, if the price in market moves up or moves down, how should the trader change the limit price of limit order to get a better execution price? Kissell [13] proposed three adaptation tactics, which are Target Cost, Aggressive in the Money (AIM) and Passive in the Money (PIM), based on price adjustments to be consistent with investor's implementation goal during execution. Genetic Programming can be applied to this problem. Also, Agent-based Artificial Stock Market can be combined with GP. An agent with GP evolved strategy can be represented as an Algorithmic Trader in ASM, whose purpose is to evolve best execution strategy using GP. We also plan to relax more assumptions in the ASM, such as adding market impact into the current model.

6. REFERENCES

[1] Almgren, R. (2008). 'Execution costs', Available at http://www.courant.nyu.edu/ almgren/papers/eqf.pdf.

[2] Bertsimas, D. and Lo, A. (1998). 'Optimal control of execution costs', *Journal of Financial Markets*, 1(1):1-50.

[3] Bonabeau, E. (2002). 'Agent-based modeling: methods and techniques for simulating human systems', *Proceedings of the National Academy of Sciences*, 93(3):7280-7287.

[4] Brabazon, A., O'Neill, M. and Dempsey, I. (2008). 'An introduction to evolutionary computation in finance', *Computational Intelligence Magazine*, 10(10):1-12.

[5] Coggins, R., Blazejewski, A. and Aitken, M. (2003). 'Optimal trade execution of equities in a limit order market', *Proc. 2003 IEEE International Conference on Computational Intelligence for Financial Engineering*, pp 371-378.

[6] Coggins, R., Lim, M. and Lo, K. (2004). 'Algorithmic trade execution and market impact', Available at http://www.ballarat.edu.au/ard/it-ms/CIAO/IWIF/iwif1papers/CogginsLimLoIWIF1.pdf.

[7] Daniel, G. (2006). 'Asynchronous Simulations of A Limit Order Book', PhD's thesis, University of Manchester, U.K.

[8] Decovny, S. (2008). 'Asian-pacific gears up for algorithmic trading', *Market View*, 1(13):347-358.

[9] Engle, R.F., Ferstenberg, R. and Russell, J. (2008). 'Measuring and modeling execution cost and risk', NYU Working Paper No. FIN-06-044, Available at SSRN: http://ssrn.com/abstract=1293666.

[10] Gode, D. and Sunder, S. (1993). 'Allocative Efficiency of Markets with Zero-intelligence Traders', *Journal of Political Economy*, 101:119-137.

[11] Hasbrouck, J. (1997). *Empirical Market Microstructure: The Institutions, Economics, and Econometrics of Securities Trading*, Oxford University Press, New York.

[12] Kim, K. (2007). *Electronic and Algorithmic Trading Technology*, Academic Press, San Diego, U.S.A.

[13] Kissell, R. and Malanut, R. (2006). 'Algorithmic decision-making framework', *Journal of Trading*, 1(1):12-21.

[14] Lebaron, B. (2005). 'Agent-based Computational Finance', in *Handbook of Computational Economics, Volume 2: Agent-based Computational Economics*, ed. L.Tesfatsion and K.L.Judd, Elsevier, North-Holland.

[15] Lim, M. and Coggins, R. (2005). 'Optimal trade execution: an evolutionary approach', *Proc. IEEE Congress on Evolutionary Computation*, 2:1045-1052.

[16] Lorenz, J. and Almgren, R. (2008). 'Continuously Adaptive Arrival Price', Available at http://www.courant.nyu.edu/ almgren/papers/arrival1.pdf

[17] Obizhaeva, A. and Wang, J. (2006). 'Optimal Trading Strategy and Supply/Demand Dynamics', AFA 2006 Boston Meetings Paper, Available at SSRN: http://ssrn.com/abstract=686168.