Evolutionary Computation and Trade Execution

Wei Cui[1,2], Anthony Brabazon[1,2] and Michael O'Neill[1]

[1] Natural Computing Research and Applications Group,
University College Dublin, Ireland.
wei.cui.1@ucdconnect.ie; anthony.brabazon@ucd.ie; m.oneill@ucd.ie
[2] School of Business, University College Dublin, Ireland.

Summary. 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 chapter 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 a Genetic Algorithm to evolve an efficient trade execution strategy. Finally, we suggest a number of opportunities for future research.

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 [21]. AT is also common in European financial markets with approximately 50% of trading volumes being accounted for by algorithmic trading programs [15]. Significant volumes in Asian markets are similarly traded [14]. Algorithmic trading is seen in multiple financial markets ranging from equities to FX (foreign exchange), to derivative (futures, options etc.) markets.

In this chapter 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. 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 $125 per share the transaction will only take place if the market price falls to $125 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 are limit order markets which 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 200 shares are currently available for sale at a price of 133.2 (or higher), and buyers are seeking 300 shares at a price of 132.9 (or lower). 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
300	132.9	133.2	200
200	132.8	133.3	300
400	132.7	133.4	100
500	132.6	133.5	300
300	132.5	133.6	200
100	132.4	133.7	400

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 400 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 minimize market impact is to break the order up into smaller lots and spread them out over time. While this may 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.

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), what type of order to use, when to submit the order, and how execution performance is to be measured. 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. In this chapter 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 any trade execution strategy, we need highly detailed transaction level data. An ordinary way is to obtain the data from the exchange. However, this only provides us with a single sample path of order book data over time. Another approach is to consider the output data from an *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). Novelly, this chapter evaluates the strategy employing the data from an ASM.

This chapter is organized as follows: Section 2 gives the necessary microstructure background relevant to the trade execution from two aspects: trading cost and price formation. Section 3 discusses trade execution strategy and corresponding performance evaluation. Section 4 provides concise introduction to the EC methodologies, and related work with application in trade execution. Section 5 explains agent-based modeling and simulates an artificial stock market. Section 6 demonstrates the use of GA to evolve a quality execution strategy. Section 7 conclude this chapter by giving a number of avenues for future work.

2 Background

The finance literature on market microstructure is vast and consequently, we only discuss a limited subset of concepts from it, trading cost and price formation, which are most relevant to this chapter.

2.1 Trading Cost

Trading cost can be decomposed into direct cost and indirect cost. Direct cost are observable directly, such as commissions, fees and taxes. Indirect costs are more difficult to observe. Indirect costs can be divided into three

main components: market impact cost, opportunity cost and bid-ask spread cost. Early studies of indirect costs focused on the bid-ask spread [20]. Lately, market impact cost and opportunity cost have received more attention.

Execution needs to balance all of these factors [1]. If trading costs are not properly managed throughout trading it could cause a superior opportunity to become only marginally profitable and a normally profitable opportunity to turn into a loss [22].

Factors which influence transaction cost are trade size, market capacitation, time windows, trading time, order imbalance, volume of time horizon etc.

Market Impact

As investors transact shares in the market they cause market impact (price impact) in the stock. Buy orders are associated with increasing prices and sell orders are associated with decreasing prices.

The market impact is typically decomposed into permanent and transitory components which provide estimates of the information and liquidity costs of the trade [20].

Due to its importance, there is much research on the causes of market impact [22]. Empirical evidence showed that block price impacts are a concave function of order size and a decreasing function of market capitalization (liquidity). Bikker [4] found that average market impact costs equal 20 basis points for buys and 30 basis points for sells, and market impact costs are influenced by timing of the trades, such as the day of the week, the period of the month and the month of the year at which the stock is traded. Stocks with high capitalization yield lower price impact cost than stocks with low capitalization [28]. Price impact costs increase as order imbalance increases [27].

Opportunity Cost

There are two reasons why opportunity cost can arise [20]. One reason is that an order is only partially filled or is not executed at all. This often happens using passive trading strategies, such as a limit order strategy which trades only limit order in the market. For example, a trader who anticipates that the market price will move down, sets the limit price below the best available bid price. If the market price actually moves up during that day, he will suffer a high cost due to unexecuted order. The other reason is that some orders traded in the market are executed with a delay, in times of adverse price movement.

2.2 Price Formation

Many modern markets operate using an electronic limit order book as described above. 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. According to the first rule, the buy (or sell) limit order with higher (or lower) price get executed prior to others with lower (or higher) price. The second rule means that where two or more limit buy (or sell) orders have the same limit price, the buy (or sell) limit order which arrives at the market earlier get executed prior to the others. A simple price formation process is shown in Figure 1.

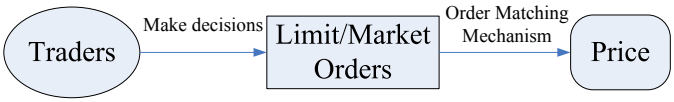

Fig. 1. Price Formation

3 Trade Execution Strategy

A dilemma facing traders is that trading too quickly reduces the risk of suffering high opportunity costs but is associated with high market impact cost. Trading too slowly minimizes market impact cost but can give rise to opportunity cost. These costs are balanced out in a trade execution strategy, by splitting a large trade into lots of small pieces and spreading them over several hours using market or limit orders. For example, an algorithmic trading strategy may equally divide a 100,000-share order into 100 small orders, where each small order has 1,000 shares. These orders then may be sent to the market over the course of the day, in order to minimize market impact. Another advantage of doing this is that these orders can be disguised in the market and prevented from being detected by other participants in the market.

This section presents important factors in a trade execution strategy and discusses how to evaluate performance of trading strategies.

3.1 Factors

Factors of a trading strategy include the number of orders, type of each order, size per order and submission time per order. Moreover, if a submitted order is a limit order, the strategy has to specify a limit price and a duration time.

If immediacy is desired, the market order is the appropriate instrument to use. However, market orders pay an implicit price for immediacy. Limit

orders avoid impact cost but bear the risks of non-execution. In practice, traders submit both types of orders, in order to balance the opportunity cost of delaying execution against the price impact associated with immediate execution.

Duration time or lifetime is another important factor. The lifetime can range from zero to entire trading time of the day. However, longer lifetime is not always the better choice, since transacting on longer period also faces more risk.

The limit price of a limit order plays a significant role in order execution, which always floats around the best bid/ask price. When placing a limit order, it is simpler to just consider the relative limit price, which is the difference between the limit price of buy/sell order and the current best bid/ask price. Choosing a relative limit price is a strategic decision that involves a trade-off between patience and cost. For example, if a trader wants to submit a limit buy order when the current best ask price is a, an impatient buyer will submit a limit order with a limit price p well above a, which will immediately result in a transaction. A buyer of intermediate patience will submit an order with price p a little smaller than a; this will not result in a immediate transaction, but will have high priority as new sell orders arrive. A very patient buyer will submit an order with p much smaller than a; this order is unlikely to be executed soon, but it will trade a good price if it does. A lower price is clearly desirable to the buyer, but it comes at the cost of lowering the probability of trading. Usually, the lower the price to buy or the higher the price to sell, the lower the probability there will be a trade. However, the choice of limit price is a complex decision that depends on market environment.

3.2 Types

Algorithmic Trading systems typically aim at achieving or beating a specified benchmark with their executions and may be distinguished by their underlying benchmark, their aggressiveness or trading style as well as their adaptation behavior [23].

In practice the most commonly used algorithms in the market place according to their benchmarks are: arrival price, time weighted average price (TWAP), volume weighted average price (VWAP), market-on-close (MOC), and implementation shortfall (the difference between the share-weighted average execution price and decision price for a trade). Arrival price is the midpoint of the bid-ask spread at order-receipt time. VWAP is calculated as the ratio of the value traded and the volume traded within a specified time horizon. MOC measures the last price obtained by a trader at the end of the day against the last price reported by the exchange. Implementation shortfall is a model that weighs the urgency of executing a trade against the risk of moving the stock.

In terms of adaptation behavior, any algorithmic trading strategy can also be categorized into one of the two categories: static strategy and adaptive strategy. Static strategy pre-determines order trading schedule and will not

change during the process of trading. This strategy can not adapt to changing environment. For example, an aggressive strategy that places only market orders to buy can not change its aggressiveness if the market price keeps moving up. On the other hand, adaptive strategy adapts to changing market conditions such as market price movements, volume profiles due to special events such as new announcements or fed indicators, as well as changes in volatility, by altering their aggressiveness of trading adequately. Intuitively, a more aggressive action can be represented either as raising (or lowering) the buy (or sell) limit order price or as increasing (or decreasing) the order volume. For example, in times of favorable price movement adaptive strategies are likely to become more aggressive by submitting more market orders or raising (or lowering) buy (sell) price or increasing order volume, and in times of adverse price movement adaptive strategies are more passive in order to avoid unnecessary market impact cost by submitting more limit orders or lowering (or raising) buy (sell) price or decreasing order volume.

Several researchers have made contributions to adaptive trading strategy. Almgren [2] showed evidence that strategies that are adaptive to market developments, i.e. that can for example vary their aggressiveness, are superior to static strategies. Nevmyvaka [30] proposed dynamic price adjustment strategy, where limit order's price is revised every 30 seconds adapting to the changing market state. Wang [35] proposed a dynamic focus strategy, which dynamically adjusts volume according to real-time update of state variables such as inventory and order book imbalance, and showed that dynamic focus strategy can outperforms a static limit order strategy.

3.3 Performance Evaluation

Execution performance is assessed by comparing execution costs relative to a benchmark. The execution cost measure is a weighted sum of the difference between the transaction price and the benchmark price where the weights are simply the quantities traded [15]. The most used benchmark prices are VWAP, TWAP, arrival price, implementation shortfall, which have been introduced above.

The rationale here is that performance is considered good if the average execution price is more favorable than the benchmark price and bad if the average execution price is less favorable than the benchmark price. Take the VWAP as an example, which is an industry standard benchmark.

The VWAP benchmark is calculated across the time horizon during which the trade was executed and is calculated as

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

Hence, 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].

4 Evolutionary Computation in Trade Execution

Evolutionary computation is a subfield of artificial intelligence. The basic idea of an evolutionary algorithm is to mimic the evolutionary process, just as the name implies. The evolutionary process is operated on the solutions or the encodings of solutions. In financial markets, EC methodologies have been used for solving a broad selection of problems, ranging from predicting movements in current values to optimizing equity portfolio composition. An overview of EC applications in finance can be seen in [8].

GA is a kind of EC algorithm. The key steps in an GA are [7]:

1. Initialization. Construct an initial population of encodings to potential solutions to a problem;
2. Calculation. Calculate the fitness of each potential solution in the population;
3. Selection. Select a pair of encodings (parents) corresponding to potential solutions from the existing population according to the fitness;
4. Crossover. Perform a crossover process on the encodings of the selected parent solutions;
5. Mutation. Apply a mutation process on the encodings of the two child solutions and then store them in the next population;
6. Repeat. Repeat steps 3-5 until n encodings of potential solutions have been created in the new population, and the old population are discarded;
7. Repeat Again. Go to step 2 and repeat until the desired fitness level has been reached or until a predefined number of populations have elapsed.

Another kind of EC is Genetic Programming (GP), an extension of GA. In GP, the evolutionary operators are applied directly to the solutions, thus the evolutionary search is operated on a space of computer programs. In financial application, this space can be a society of option pricing formulas, trading rules, or forecasting models. GP offers the potential to generate human-readable rules.

4.1 Related Work

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 [29] who applied a genetic algorithm 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.

5 Agent-based Artificial Stock Market

In this chapter, the data used to test the execution strategies are derived from an artificial stock market. This section gives a brief introduction to the agent-based modeling technique.

5.1 Agent-based Modeling

Agent-based modeling is a simulation technique to model non-linear systems consisting of heterogeneous interacting agents. The emergent properties of an agent-based model are the results of "bottom-up" processes, where the decisions of agents at a microscopic level determine the macroscopic behavior of the system. An 'agent' refers to a bundle of data and behavioral methods representing an entity constituting part of a computationally constructed world. The agents can vary from simple random zero-intelligence (ZI) agents as in [18] to sophisticated inductive learning agents. 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 [5].

The branch of agent-based modeling that deals with economic environments is sometimes referred to as agent-based computational economics (ACE), which naturally includes agent-based artificial markets [3]. They view financial markets as interacting groups of learning, boundedly-rational agents, by incorporating a well-defined price formation mechanism and a representation of market participants.

5.2 Artificial Market Models

Most artificial markets implement simplified market models, which omit some institutional details of trading, but serve their research needs sufficiently. One example is the clearing house model, where a number of agents only trade an asset at discrete time intervals. At the start of each time period, every agent forms his expectation for the future price in the next period, according to the available information, such as current market price and historical prices. Then, the trader can decide the proportion of the asset he will hold in the next period in order to maximize his profit. After collecting the accumulated buy and sell orders of all the agents, the market is cleared at a new market

price. The renowned Santa Fe artificial stock market [26] is a such market, based on clearing house model. For general reviews, see for example [10, 24, 26, 25, 32, 34].

Table 2. Artificial Model Comparison

	Market Mechanism	**Order Waiting Time**	**Order Size**	**Agent Type**
Chiarella:2002	Double Auction	Discrete time steps	One unit of the stock	Fundamentalist, chartist and noise trader
Chiarella:2009	Double Auction	Discrete time steps	Generated using a specified demand function	Fundamentalist, chartist and noise trader
Raberto:2005	Double Auction	Exponential distributed	Depend on cash endowment	ZI agent
Chan:2001	Double Auction	Discrete time steps	One share	Artificial -intelligent trader
Yang:2003	Double Auction	Discrete time steps	A fixed number of shares	Neural learning agent
Daniel:2006	Double Auction	Exponential distributed	Normal distributed	ZI agent

However, the clearing house is only an approximate description of the way stock exchanges operate around the world. Nowadays, most financial markets are electronic markets, operating on an order book. Several researchers have made contributions to the models which implement the realistic trading market model, e.g. a limit order market, moving from the more stylized earlier financial market models toward more models incorporating explicit market microstructure [9, 31, 36].

It is difficult to design a market that perfectly reflects all the details of a real stock market. Therefore several choices, simplifications and assumptions are needed in order to make attempts to represent market structures and traders' behavior. In Chan's market [9], a submitted limit order price has to be better than the current price, for instance, any subsequent bid must be higher than the current bid to be posted, and subsequent ask is lower than the current ask to be posted. Yang's market [36] is similar to Chan's [9]. In Chiarella's markets [11, 12], traders set bids and asks and post market or limit orders according to exogenously fixed rules. One major drawback of the Chiarella's model comes from the assumption that there exists a constant fundamental value of the asset that all agents know, which is not realistic [19]. A comparison of these models can be seen in Table 2.

5.3 Simulation

In this chapter, our model is based on the zero-intelligence (ZI) model [13], which aims to generate a realistic aggregate order flow using very simple assumptions. The ZI agents are responsible for generating the order flow, by placing random orders to buy or sell. In this model, only one stock is traded, and dividends are ignored. Traders trade orders 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.

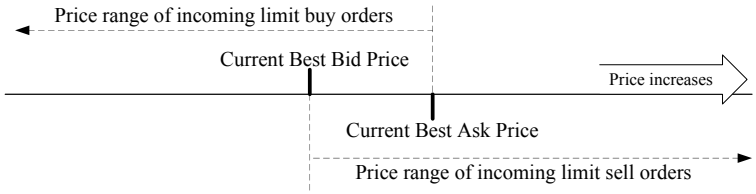

Fig. 2. Place Limit Price

Table 3. Initial Parameters for Order Book based ASM

Explanation	Value
Initial Price	$price^0 = 100$
Tick Price	$\delta = 0.01$
Probability of Cancellation Order	$\lambda_c = 0.07$
Probability of Market Order	$\lambda_m = 0.33$
Probability of Limit Order	$\lambda_l = 0.60$
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	$(\mu, \sigma) \sim (4.5, 0.8) * 100$ shares
Waiting Time	$\tau = 6, 90$

There are two order signs, buy or sell. The agents are equally probable to generate a buy order or a sell order. There are three types of orders in our model: market order, limit order and cancellation order (to delete an order from the order book). In London Stock Exchange, about 2/3 of the submitted order are limit orders and 1/3 are market orders [16], and roughly 10% of limits order in the order book are canceled before before being executed [6]. When an agent is active, she can try to issue a cancelation order with probability λ_c (oldest orders are canceled first), a market order with probability λ_m, a limit order with probability $\lambda_l = 1 - \lambda_c - \lambda_m$. Traders do not always place limit

order at best bid/ask prices or inside the bid-ask spread. About 1/3 of limit orders fall outside the bid-ask spread and the density of placement falls off as a power law as a function of the distance from the best bid/ask price [17]. In our 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. The parameters used in our simulation are presented in Table 3.

Algorithm 1 Behavior of ZI Agent

```
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;
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));
    else
        /* limit price to be out of the spread */
        generate LimitPrice price_i(Δ) ~ 1/Δ^(1+α);
        /* power-law distributed*/
    end if
else
    /* a market order to be submitted */
    generate OrderSize vol(t) = [vol(a(t))|vol(b(t))].
    /* same size as best counterpart*/
end if
```

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 [33], which is one of the most popular agent-based modeling platforms. The algorithm used in our simulation is described in Algorithm 1.

6 Experiments

This section describes how to use a GA to uncover a quality trade execution strategy and evaluate it using the data generated from the artificial market described above.

6.1 Data

This simulated market collects four kinds orders: market buy orders, limit buy orders, market sell orders and limit sell orders.

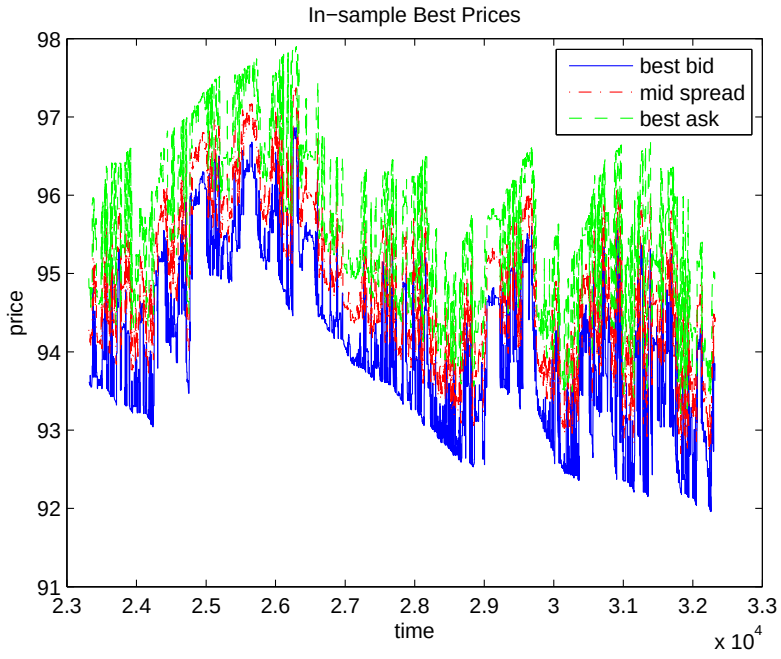

Fig. 3. In-sample Data

The ASM simulation uses a database to store the details of each incoming order and best prices at each time point, which are limit buy orders, limit sell orders, market buy orders, market sell orders and best buy/sell orders.

The limit buy/sell order record contains each limit order's index number, arrival time, volume, submitting limit price, time when canceled or traded. The market buy/sell order record contains each market order's index number, arrival time, traded volume, traded price and the index number of corresponding traded limit order. The best price record contains the best bid and ask orders' index number, volume, price, and mid-spread price at each time when new order comes.

The ASM simulation was run for 30 virtual days. Each record in our dataset includes the following order-specific variables: size (in number of shares), side (buy or sell), market or limit, limit price (if a limit order), starting time and ending time for the entire order.

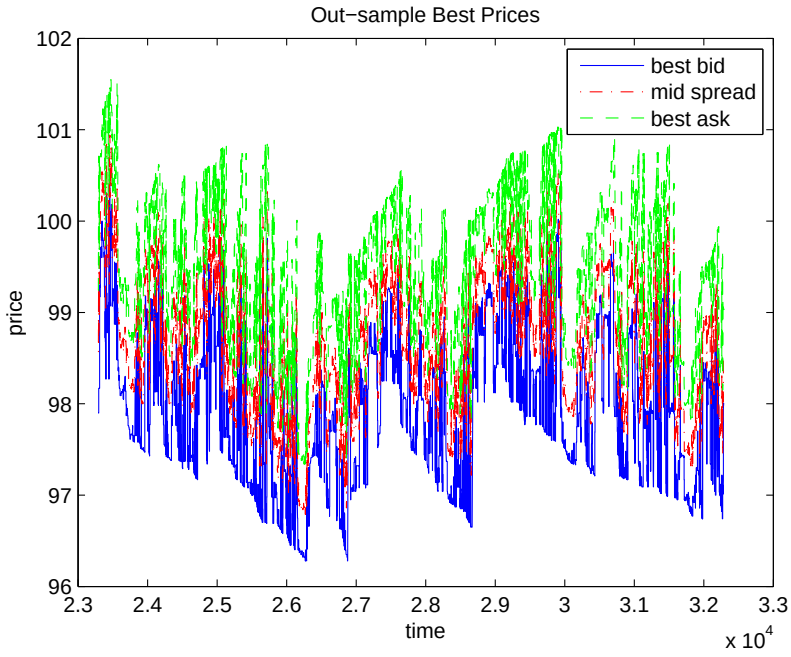

Fig. 4. Out-sample Data

6.2 GA Strategies

Commonly, a trader may wish to trade an order over a specified period, if the order can not be filled at once. For the special characteristic of our artificial market, we assume that trading period is two and a half hours.

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

How to divide a large trade depends on the order size and trading time. For simplicity, we divide our large trade into 30 smaller orders equally, and submit each smaller order into the market every 5 minutes (300 seconds).
Bear in mind that limit orders do not guarantee execution. When we are trading limit orders, we also need to consider how to deal with the unexecuted limit orders. They can either be executed as market orders at the end of their lifetimes to avoid unexecuted risk, or at the end of the whole trading period for better execution price.

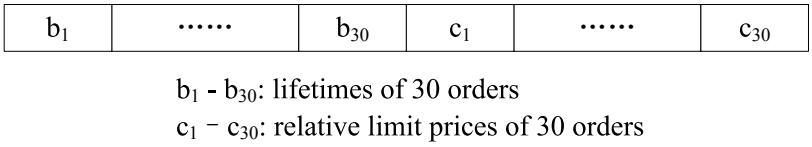

Fig. 5. Representation

In our experiment, we use both market and limit orders. The orders are submitted to the market every 5 minutes. 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, which will be determined by the GA methodology. Figure 5 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 to the entire executed orders' VWAP, which are $VWAP_{30}$ and $VWAP_{global}$ respectively. For both buy and sell orders, the smaller the VWAP Ratio, the better the strategy is.

$$VWAPRatio = \begin{cases} \frac{1000*(VWAP_{30}-VWAP_{global})}{VWAP_{global}} & \text{Buy Strategy} \\ \frac{1000*(VWAP_{global}-VWAP_{30})}{VWAP_{global}} & \text{Sell Strategy} \end{cases}$$

6.3 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 order flow generated from ASM. The new order flow is simulated as a market, where the 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. 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

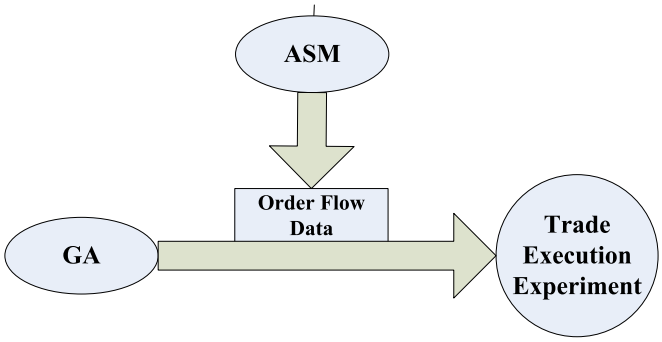

Fig. 6. Experiment

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.

Table 4. Parameters for Genetic Algorithm

Population size	30
Maximum number of generation	100
Generation gap	0.8
Crossover rate	0.75
Mutation rate	0.05
Selection method	Stochastic Universal Sampling
Crossover method	Single-Point

We used a population of 30 individuals, running for 100 generations, to evolve efficient GA strategies and we tested them with in-sample data and out-of-sample data separately. The parameters used in GA can be seen from Table 4. At the same time, we adopted another strategy to benchmark against our GA strategy, namely a pure market order strategy (MOS). It trades orders as market orders immediately on submission to the market.

6.4 Results and Discussion

Running both simulations for buy orders and sell orders over, we obtain the results shown in Tables 5 & 6.

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 5 & 6. 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.

Table 5. Results of Buy Order.

	MOS	GA Strategy			
	VWAP Ratio (10^{-3})	VWAP Ratio (10^{-3})	TradedOrderType		
			MLO	LO	MO
In-sample	4.4474	-2.5899	4	20	6
Out-of-sample	5.9748	0.5146	11	8	11

Table 6. Results of Sell Order.

	MOS	GA Strategy			
	VWAP Ratio (10^{-3})	VWAP Ratio (10^{-3})	TradedOrderType		
			MLO	LO	MO
In-sample	2.7389	-5.8376	6	23	1
Out-of-sample	3.2378	-1.8244	13	7	10

From Tables 5 & 6, the GA strategy outperforms the MOS strategy significantly, both in-sample and out-of-sample, which is consistent with the results in [29]. The two tables show that the GA strategy, which has more orders executed in the way of LO, has a smaller VWAP ratio, meaning better performance. All the GA strategies with negative VWAP ratios have more orders executed in the way of LO than those executed in the two other ways, except the best out-of-sample strategy in Table 6. 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. These results suggest the applicability and potential of GA for trade execution.

7 Conclusion and Future Work

In this chapter, we present a problem in trade execution and emphasize an evolutionary approach to this problem. Initially, we built an order book using agent-based modeling. Using the order flow produced by the ASM, we applied a Genetic Algorithm to optimize 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 back-testing strategies based on historical market data.

In future work, we intend to extend the application of EC to harder, 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 [23] 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 some of the assumptions in our ASM, such as adding market impact into the current model.

Acknowledgement

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant Number 08/SRC/FM1389.

References

1. Almgren, R. Execution costs. In *Encyclopedia of Quantitative Finance*. wiley, 2008.
2. Almgren, R., and Chriss, N. Optimal execution of portfolio transactions. *Journal of Risk*, 3(2):5–39, 2000.
3. Berseus, P. Creating an agent-based artificial market. Master's thesis, Lunds Tekniska Hogskola, 2007.
4. Bikker, J., Spierdijk, L., and Sluis, P. Market impact costs of institutional equity trades. Technical Report 27, Netherlands Central Bank, Research Department, 2007.
5. Bonabeau, E. Agent-based modeling: methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 93(3):7280–7287, 2002.
6. Bouchaud, J., Mezard, M., and Potters, M. Statistical properties of stock order books: empirical results and models. *Quantitative Finance*, 2(4):251–256, 2002.
7. Brabazon, A., and O'Neill, M. *Biologically Ispired Algorithms for Financial Modeling*. Spring-Verlag, Berlin Heidelberg, 2006.
8. Brabazon, A., O'Neill, M., and Dempsey, I. An introduction to evolutionary computation in finance. *Computational Intelligence Magazin*, 10(10):1–12, 2008.

9. Chan, N., LeBaron, B., Lo, A., and Poggio, T. Agent-based models of financial markets: A comparison with experimental markets. Technical Report 4195-01, Massachusetts Institute of Technology, 2001.
10. Chen, S. Computationally intelligent agents in economics and finance. *Information Sciences*, 177(5):1153–1168, 2007.
11. Chiarella, C., and Iori, G. A simulation analysis of the microstructure of double auction markets. *Quantitative Finance*, 2:246–253, 2002.
12. Chiarella, C., Iori, G., and Perello, J. The impact of heterogeneous trading rules on the limit order book and order flows. *Journal of Economic Dynamics and Control*, 33(3):525–537, 2009.
13. Daniel, G. *Asynchronous simulations of a limit order book*. PhD thesis, University of Manchester, U.K., 2006.
14. Decovny, S. Asian-pacific gears up for algorithmic trading. *Market View*, 1:13, 2008.
15. Engle, R., Ferstenberg, R., and Russell, J. Measuring and modeling execution cost and risk. Technical Report 08-09, Chicago GSB Research Paper, 2008.
16. Farmer, J., Gerig, A., Lillo, F., and Mike, S. Market efficiency and the long-memory of supply and demand: Is price impact variable and permanent or fixed and temporary? *Quantitative Finance*, 6(2):107–112, 2006.
17. Farmer, J. D., Patelli, P., and Zovko, I. Supplementary material for 'the predictive power of zero intelligence in financial markets', available at: www.santafe.edu/ jdf/papers/zerosuppl.pdf, 2005.
18. Gode, D., and Sunder, S. Allocative efficiency of markets with zero-intelligence traders. *Journal of political economy*, 101:119–137, 1993.
19. Guo, T. An agent-based simulation of double-auction markets. Master's thesis, University of Toronto, 2005.
20. Keim, D., and Madhavan, A. The cost of institutional equity trades. *Financial Analysts Journal*, 54(4):50–52, 1998.
21. Kim, K. *Electronic and Algorithmic Trading Technology*. Academic Press, U.S.A., 2007.
22. Kissell, R. *Algorithmic Trading Strategies*. PhD thesis, Fordham University, 2006.
23. Kissell, R., and Malanut, R. Algorithmic decision-making framework. *Journal of Trading*, 1(1):12–21, 2006.
24. LeBaron, B. Agent-based computational finance: suggested readings and early research. *J. Econom. Dynam. Control*, 24:679–702, 2000.
25. LeBaron, B. Agent-based computational finance. In Tesfatsion, L., and Judd, K., Eds., *Handbook of Computational Economics, Volume 2: Agent-based Computational Economics*, 134–151. Elsevier, 2005.
26. LeBaron, B. A builder's guide to agent based financial markets. *Quantitative Finance*, 1(2):254–261, 2001.
27. Lim, M., and Coggins, R. Price impact of trades on the asx. In *Australasian Finance and Banking Conference*, 2003.
28. Lim, M., and Coggins, R. The immediate price impact of trades on the australian stock exchange. *Quantitative Finance*, 5(4):365–377, 2005.
29. Lim, M., and Coggins, R. Optimal trade execution: an evolutionary approach. In *Proc. IEEE Congress on Evolutionary Computation*, volume 2, pages 1045–1052, 2005.

30. Nevmyvaka, Y., Feng, Y., and Kearns, M. Reinforcement learning for optimized trade execution. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 673–680, New York, USA, 2006. ACM. 1143929 673-680.
31. Raberto, M., and Cincotti, S. Modeling and simulation of a double auction artificial financial market. *Physica A: Statistical Mechanics and its applications*, 355(1):34–45, 2005.
32. Samanidou, E., Zschischang, E., Stauffer, D., and Lux, T. Agent-based models of financial markets. *Reports on Progress in Physics*, 70(3):409–450, 2007.
33. Swarm. Swarm package can be obtained from http://www.swarm.org.
34. Tesfatsion, L. Agent-based computational economics: A constructive approach to economic theory. In Tesfatsion, L., and Judd, K., Eds., *Handbook of computational economics: agent-based computaional economics*, 269–277. Elsevier, North-Holland, Amsterdam, 2006.
35. Wang, J., and Zhang, C. Dynamic focus strategies for electronic trade execution in limit order markets. In *CEC-EEE '06: Proceedings of the The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services*, 26–35, Washington, DC, USA, 2006. IEEE Computer Society.
36. Yang, J. The efficiency of an artificial double auction stock market with neural learning agents. In Chen, S., Ed., *Evolutionary Computation in Economics and Finance*, 85–106. Springer-Verlag, 2002.