Stock Trading System based on the Multi-objective Particle Swarm Optimization of Technical Indicators on End-of-day Market Data

Antonio C. Briza

Dept. of Computer Science, University of the Philippines-Diliman, Diliman, Quezon City, Philippines

Prospero C. Naval, Jr. *

Dept. of Computer Science, University of the Philippines-Diliman, Diliman, Quezon Ciy, Philippines

Abstract

Stock traders consider several factors or objectives in making decisions. Moreover, they differ in the importance they attach to each of these objectives. This requires a tool that can provide an optimal tradeoff among different objectives, a problem aptly solved by a multi-objective optimization (MOO) system.

This paper aims to investigate the application of multi-objective optimization to end-of-day historical stock trading. We present a stock trading system that uses multi-objective particle swarm optimization (MOPSO) of financial technical indicators. Using end-of-day market data, the system optimizes the weights of several technical indicators over two objective functions, namely, percent profit and Sharpe ratio.

The performance of the system was compared to the performance of the technical indicators, the performance of the market, and the performance of another stock trading system which was optimized with the NSGA-II algorithm, a genetic algorithm-based MOO method. The results show that the system performed well on both training and out-of-sample data. In terms of percent profit, the system outperformed most, if not all, of the indicators under study, and, in some instances, it even outperformed the market itself. In terms of Sharpe ratio, the system consistently performed significantly better than all the technical indicators. The proposed MOPSO system also performed far better than the system optimized by NSGA-II.

The proposed system provided a diversity of solutions for the two objective functions and is found to be robust and fast. These results show the potential of the system as a tool for making stock trading decisions.

Key words: Multi-objective optimization, particle swarm optimization, stock trading systems, technical indicators

1. Introduction

Success in trading stocks depends on timing the trades well. For years, stock traders have depended on two major tools: fundamental analysis, which relies on company performance and growth projection, and technical analysis which analyzes the trade history of a security through charts and mathematical formulas called technical indicators.

In recent years, Artificial Intelligence(AI) techniques have also been employed in timing the trades in stock market. AI can be applied in stock trading in

* Corresponding author.

Email addresses: acbriza@up.edu.ph (Antonio C. Briza), pcnaval@up.edu.ph (Prospero C. Naval, Jr.).

two ways. First, as a tool to develop a trading system whose goal is to give trading signals using historical end-of-day market data. These trading systems are ordinarily validated by testing in an out-of-sample data. Second, AI can be used as an aid in developing trading agents whose objective is to post buy and sell orders which are processed by an artificial stock exchange. These tools usually utilize intraday (real time) data and are validated by the agent's participation in a simulated stock trading exchange.

An example of the first type of application is the work of Skabar and Cloete [33] who used genetic algorithm (GA) and neural networks (NN) to determine buy and sell points for commodities in the stock exchange. Other examples are researches which have focused on the optimization of the parameters of technical indicators. Among these are Fernandez-Rodriguez et al. [14] who optimized the parameters of the moving average indicator using GA; Lin et al. [25] who used GA to find the best parameter combination for the filter trading rule; and De la Fuente et al. [7] who likewise used GA to optimize the parameters of three technical indicators.

For the second type of application, we can cite the study of Subramanian et al. [34] who designed agents that are based on composite trading rules trained by GA and Genetic Programming (GP). The performance of the agents were evaluated by making them compete with other automated agents in the Penn-Lehman Automated Trading Project [18].

While the studies above have shown remarkable results, it should be noted that they optimize only one objective function – either profit or risk. However, a stock trader's decision usually depends on several factors. Moreover, traders give different importance to these factors, depending on their trading personalities, e.g. the level of risk they could take on. This requires a tool that can provide an optimal tradeoff among different objectives. This problem is aptly solved by a MOO system.

While several applications of MOO exist in literature in the areas of stock prediction and modeling [24,11,23,1], portfolio selection [3,32,29,13,19], and portfolio optimization [10,9,2], to the best of our knowledge, only Fukomoto and Kita's study [15] made use of MOO in timing the entry to the stock market.

Fukomoto and Kita used a GA-based multi-objective optimization approach to train intraday trading agents on two objective functions, namely profit ratio and variance of profit. The agents were tested by running them in U-Mart [21], an artificial market simulator. Using this approach, the simulation obtained several trading strategies which were better than others (or *non-dominated* in more technical terms [1]) in terms of the two objective functions.

The success enjoyed by MOO in training an intraday agent for an artificial market encourages its application to end-of-day market data. This is what this paper would like to contribute.

Specifically, we present a stock trading system that uses multi-objective particle swarm optimization (MOPSO) of financial technical indicators. Using end-of-day market data, the system optimizes the weights of several technical indicators over two objective functions, namely, percent profit and Sharpe ratio [30].

Our proposed MOO tool is based on Particle Swarm Optimization (PSO) rather than on GA for the two reasons. First, the optimization variables (weights) are continuous, and in this case, PSO is the more appropriate method since it does not require discretization of the decision variables as GA does. Second, in their study, Hassan et al. [16] claim that PSO and GA, on the average, yielded the same quality of solutions; however, PSO has the advantage of being more computationally efficient than GA. Thus, using PSO could be a great advantage since it could speed up computation time, an important factor in designing a stock trading system or agent. Subramanian et al. [34], for instance, mentioned that improving the computation time was a potential challenge in their paper.

We thus present one of the few real world applications of MOPSO in literature. This is significant especially if we consider that the growing number of variants of the MOPSO algorithm demands a validation of MOPSO's effectiveness in solving real life problems.

The remainder of this paper is organized as follows. In Section 2 we give a brief description of the PSO algorithm and the multi-objective optimization problem (MOP). The popular approaches to MOPs, including MOPSO and its variants are also described. This is followed by a discussion of the proposed stock trading system in Section 3. The results are presented and discussed in Section 4. The paper concludes with the Summary and Conclusion in Section 5.

[1] See Section 2.1 for an explanation of the concept of non-domination.

2. Multi-objective Optimization Using Particle Swarm Optimization

In this section, we give a description of the multi-objective optimization problem and the particle swarm optimization algorithm. This is followed by a brief discussion of the adaptation of PSO to solve MOPs and the use of MOPSO in real world applications. Finally, we introduce , Multi-objective Particle Swarm Optimization - Crowding Distance (MOPSO-CD), the PSO-based MOO method that we will use in the development of our trading system.

2.1. *Multi-objective Optimization*

Real world optimization problems are not just limited to single objectives. Many times, they require having a balance (or trade offs) among different interacting, and possibly conflicting objectives. Multi-objective optimization addresses this problem. Multi-objective optimization entails finding a set of solutions that optimizes several objectives. The notion of an optimum solution is different in multi-objective problems as compared to single objective ones since what is required is a set of tradeoff solutions rather than a single global optimum. This notion is commonly called Pareto optimality.

The multi-objective problem can be stated in the following manner:

$$minimize$$

$$y = f(x) = (f_1(x), \ldots, f_n(x))$$

$$\text{subject to}$$

$$g(x) = (g_1(x), \ldots, g_n(x)) \leq 0$$

$$h(x) = (h_1(x), \ldots, h_n(x)) = 0$$

$$where$$

$$x = (x_1, \ldots, x_m) \in X$$

$$y = (y_1, \ldots, y_n) \in Y$$

Here, x is known as a decision vector, X is the decision space, y is an objective vector, Y is the objective space, while $g(x)$ and $h(x)$ are constraints that must be satisfied during the optimization process. The solution to the above problem is the Pareto front which consists of one or more Pareto points, i.e. decision vectors x^* that optimize the different objectives.

Pareto optimality is based on the concept of dominance. We say that candidate solution $x^{(i)}$ dominates another candidate solution $x^{(j)}$ when both of the following conditions are satisfied. First, $x^{(i)}$ is no worse than $x^{(j)}$ in all objective functions; and second, $x^{(i)}$ is strictly better than $x^{(j)}$ in at least one objective function.

In multi-objective problems, the optimum solution is the set of all non-dominated solutions. A non-dominated solution is called a Pareto point while the set of all Pareto points (the optimal set of tradeoff solutions) is called the Pareto front.

2.2. *Particle Swarm Optimization*

Particle Swarm Optimization is a popular computational technique developed by Kennedy and Eberhart [20] that is based on the social behavior of birds flocking to look for food. Reyes-Sierra and Coello Coello [31] note two reasons for PSO's popularity. First, since it is relatively simple so that its implementation is straightforward; and second, it has been found to be very effective in a variety of applications, producing very good results at very low computational cost. PSO has been found to be effective in optimization problems requiring real-valued decision variables [4,12]. Hassan et al. [16] report that while PSO's performance is comparable to GA, PSO is computationally more efficient than GA.

In PSO, a population of possible solutions (called particles) are first initialized. These particles are then allowed to explore (or fly) through a solution search space looking for the optimum solution. Each particle maintains the best solution it has found thus far (particle best) as well as the best solution that the group (called particle swarm) has found thus far (global best). The direction of the search is then updated based on the values of particle bests and the group's global best.

The position and velocity updates for PSO are described by the following:

Updating Particle i from iteration t to $t+1$:
(Velocity Equation)

$$V_i[t+1] = w * V_i[t] + C_1 * r_1(pbest_i - P_i[t]) + C_2 * r_2(gbest - P_i[t])$$

(Position Equation)

$$P_i[t+1] = P_i[t] + V_i[t+1]$$

The factors w and C_1 are the particle's inertia and self-confidence factors. The confidence factor for the entire swarm is expressed by C_2. The quantities r_1 and r_2 are positive random numbers drawn from a uniform distribution.

2.3. *Multi-objective Particle Swarm Optimization*

Several population-based evolutionary algorithms have been proposed to address MOPs. Among the popular ones are NSGA-II [8], PAES [22] and SPEA2 [35]. Other algorithms have extended PSO to solve MOP. The first proposal of such kind is MOPSO [5]. Less than a decade after MOPSO, several other variant MOPSO algorithms have already been proposed. In a survey of these algorithms, Reyes-Sierra and Coello Coello [31] cite two main algorithmic design aspects in adapting PSO to MOP. These are the selection and updating of leaders (global best) and the creation of new solutions via updating of positions or mutation (or turbulence). The authors also note that, applications using MOPSO are still very few compared with those using other multi-objective evolutionary algorithms. They think that this may be due to MOPSO's relative novelty as compared to more known multi-objective genetic algorithms. The success obtained by the few applications that used MOPSO (for instance in molecular docking [17] and in blind color image fusion [26]) encourages research for other applications of this method.

2.4. *Multi-objective Particle Swarm Optimization - Crowding Distance*

MOPSO's performance was compared with other multi-objective algorithms in [6]. In that study, MOPSO was the only algorithm which was able to cover the entire Pareto front for all the test functions that were presented. Citing the above study, Raquel and Naval [27] note that the success of MOPSO can be attributed to its use of an archive of non-dominated solutions as well as to its new mutation operator. They further observed that while MOPSO is superior to other MOAs in converging to the true Pareto front, NSGA-II was better than it in terms of promoting diversity. This prompted them to propose a new algorithm that makes use of the specific strengths of the two algorithms. From MOPSO, they adopted the former's mutation operator and use of an external archive; and from NSGA-II, they made use of its diversity and constraint-handling mechanisms. Their new proposed algorithm, called MOPSO-CD, incorporates the crowding distance density estimator introduced in NSGA-II in selecting the global best and in the deletion of the low ranking non-dominated solutions in the sorted archive. Fig. 1 shows a block diagram of the algorithm. The pseudocode of MOPSO-CD is given below and further details of about this algorithm are found in [27].

```
begin
  initialize swarm;
  evaluate_objfns;
  store pbests;
  store non-dominated particles in archive;
  t ← 0;
  while (t < t_max)
     compute crowding distances in archive and select guides;
     compute new positions;
     mutation;
     evaluate_objfns;
     impose_constraints;
     update archive;
     update pbests;
     t ← t + 1;
  endwhile
end
```

3. Multi-objective optimization of financial technical indicators

In this section, we describe the trading system and how it employs the multi-objective optimization of financial technical indicators.

3.1. *The Trading System*

Subramanian et al. [34] optimized a set of weights associated with selected indicators. They employed GA and GP to optimize a weighted combination of four indicators (Moving Average , Price Channel Breakout, Price Trend, Order Book Volume imbalance). A decision was taken based on the combined weight of the indicators. In one experiment, they used the Sharpe ratio as their objective function, then in another, they used the Sortino Ratio.

Our trading system also optimized a set of weights associated with selected indicators; however, they were optimized over two objective functions using a MOO method. We have chosen percent profit and Sharpe ratio as our objective functions.

The Sharpe ratio is defined as the ratio of average returns to risk. The percent profit and Sharpe ratio measure two different trading criteria – profit

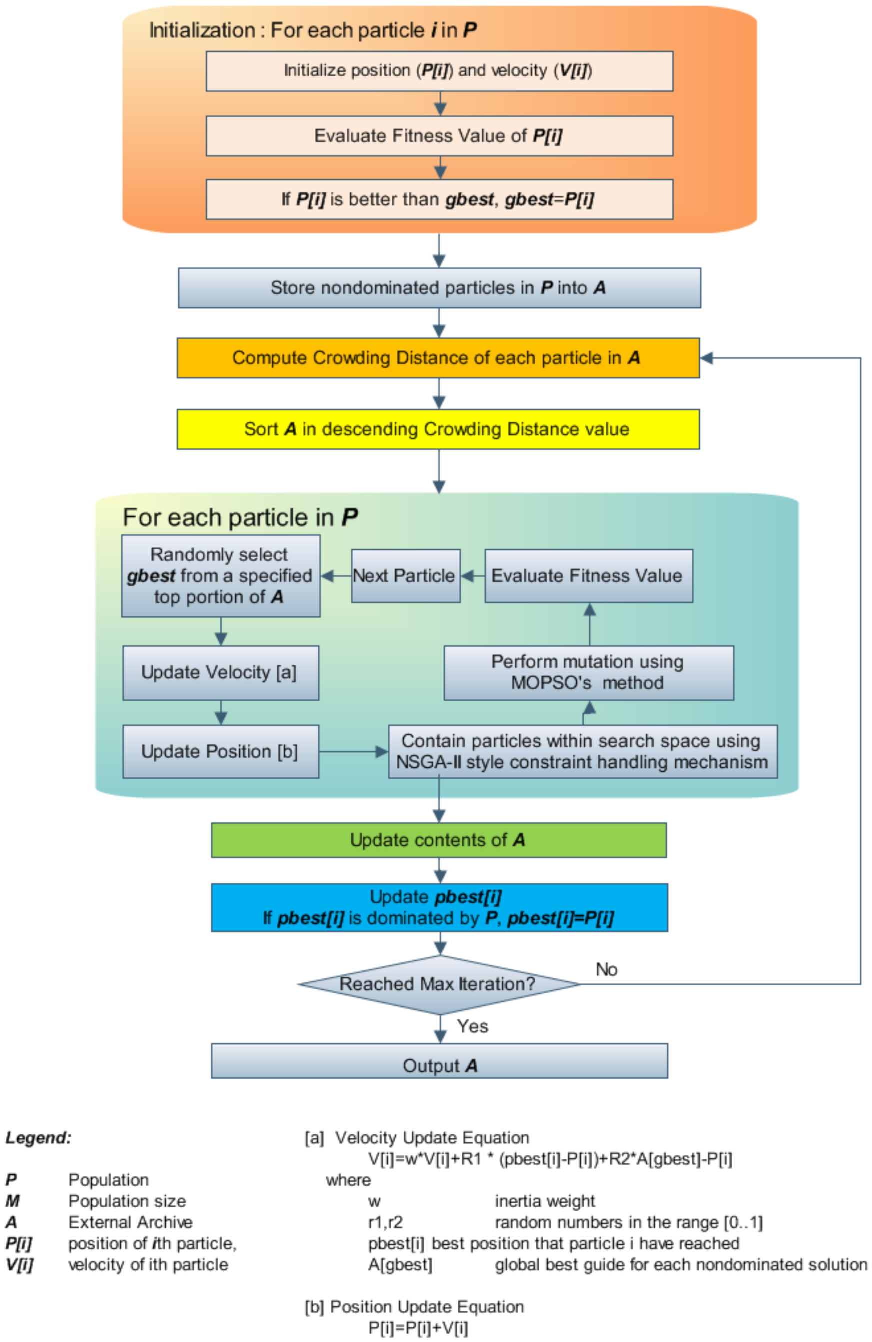


Fig. 1. The MOPSO-CD algorithm

and risk. A trading system with a high profitability may, at the same time, carry with it a huge risk. We formally define the Sharpe ratio, SR, as

$$SR = \frac{\mu}{\sigma}$$

where

- μ: the mean value of all returns ρ_i
- ρ_i: return (profit or loss) of a trading system at the ith trade
- σ: the standard deviation of all returns ρ_i

3.2. *Input Data*

The input data consisted of two items: the daily closing price of a stock index over a selected period and the values of some selected technical indicators over the same period.

We use a stock index to undertake the study since stock indices are more stable and are less susceptible to price speculations. Specifically, we focused on the Dow Jones Industrial Average (DJIA) index from February 5, 1976 up to March 25, 2002 (6,600 observation points). The data file contained daily closing prices.

We selected five popular technical indicators: Directional Movement Index (DMI), Linear Regression (LIN), Moving Average Convergence-Divergence (MAC), Moving Average (MAV), Parabolic Stop and Reverse (PSR). These indicators were evaluated over the same period as the security data. The parameters that were used were the prescribed parameters according to literature.

3.3. *Trading Strategies*

Each technical indicator was associated with the usual trading rule defined in literature. The trading rule associated with an indicator generated a signal value, S_i, of 1 if the indicator is in a Long (or Buy) position, and a value of -1 if the indicator is in a Short (or Sell) position. Moreover, a weight w_i was attached to each technical indicator. The trading decision then was made to depend on the weighted decision value $\sum S_i w_i$. A trade was executed if this value exceeded $(0.5 \sum S_i w_i)$; a trade was terminated when this value went below $(0.5 \sum S_i w_i)$. The weighted decision value also determined the amount to be used in the trades. An initial investment value was used to trade the stock. Profits from the executed trades were not used for reinvestment. Transaction costs were not included in the design of the system.

3.4. *Multi-objective optimization*

Applied to our system, we can summarize the multi-objective optimization problem in the following manner:

Given

- [DMI,LIN,MAC,MAV,PSR] = [1,2,3,4,5]
- $\vec{S}_i$
  signal vector associated with the Indicator i
  where
  $\vec{S}_{i_j} = 1$ if position=Buy at jth trading day
  $\vec{S}_{i_j} = -1$ if position=Sell at jth trading day
  $\vec{S}_{i_j} = \vec{S}_{i_{j-1}}$ if position=Hold at jth trading day
- $\vec{w} = (w_1, w_2, \ldots, w_5)$, $w_i \in [1..10]$
  weights associated with the 5 indicators
- $WDTR(\vec{w})$, the weighted decision trading rule defined by:
  Given
  $\vec{D}_j = \sum \vec{S}_{i_j} w_i$
  If current position is not Buy, change position to Buy if
  $\vec{D}_j > 0.5 \sum w_i$
  If current position is not Sell, change position to Sell if
  $\vec{D}_j < 0.5 \sum w_i$
  Else
  position= Hold

Optimization Problem : [2]

maximize
$y = f(\vec{w}) = (f_1(\vec{w}), f_2(\vec{w}))$

where
$f_1(\vec{w}) = PercentProfit(WDTR(\vec{w}))$
$f_2(\vec{w}) = SharpeRatio(WDTR(\vec{w}))$

We use the MOPSO-CD algorithm to obtain the solutions of this multi-objective optimization problem.

[2] A maximization problem can considered as a minimization problem of the negative values of the objective functions.

4. Experiments

In this section, we discuss the experiments we conducted to determine acceptable values for the proposed system. These are the training and testing window size, and the optimization parameters (population size and the number of generations to use). Finally, we compared the proposed system's performance with that of a similar stock trading system optimized using NSGA-II, a popular multi-objective genetic algorithm.

4.1. *Training and Testing Window Size*

For the first set of experiments, we investigate how the system performs with respect to different market conditions. We adapted the training and testing methodology followed by Schoreels et al. [29], who developed a GA agent which optimized technical indicators for stock selection using end-of-day equity market data. They selected 3 different increasing and overlapping time spans having varying market conditions for training the agent. Testing followed a similar pattern, with the testing time span being equal to that of the training. In our study, we chose 3 different increasing and overlapping time spans (or periods) having varying market conditions. The three periods had the same starting date, and had window sizes of 330, 660 and 1320 trading days (Table 1). We named these three periods as Period A, B and C.

Table 1
Training and testing periods for identifying an acceptable optimization window size.

Period	Window size	Training Range	Testing Range
A	330	4/28/1981 to 8/16/1982	8/17/1982 to 12/2/1983
B	660	4/28/1981 to 12/2/1983	12/5/1983 to 7/16/1986
C	1320	4/28/1981 to 7/16/1986	7/17/1986 to 10/3/1991

After determining which window size is most acceptable, we will validate its performance in other market conditions by conducting 2 more tests of the same window size in the two succeeding periods. We shall call these succeeding periods as Periods D and E. The experiments in this section were conducted using a population size of 200, archive size of 100 and were run for 100 generations.

4.1.1. *Population size and number of generations*

We investigate how the population size and number of generations affect the performance of the system. The archive size of the algorithm is always set to 50% of the population size since this the best size for MOPSO-CD. We used the following values of the population size and number of generations: 100, 200, 300, 400 and 500. We will use the results of the previous experiment to set the window size for the training and testing periods.

4.1.2. *Comparison with NSGA-II*

NSGA-II is one of the more popular multi-objective evolutionary algorithms known for its constraint-handling capability and its capacity to promote a diversity of solutions. We implemented our trading system using NSGA-II to optimize the weights of the indicators and then compare it with the results of the proposed system.

4.1.3. *Performance Evaluation*

We conducted 30 independent training and testing runs for each period; thus, we obtained 30 Pareto fronts. We then calculated PF_{best}, the average performance of the best points in the 30 Pareto fronts; and PF_{avg}, the average performance of all the points in the 30 Pareto fronts. A comparison of these two values (PF_{best} and PF_{avg}) is made with the performance of the indicators and the market, represented by the Buy-and-Hold (BH) strategy.

To help out in the analysis of the distribution of the Pareto points over the 30 independent runs, we employ a bivariate extension of the boxplot called the bagplot [28]. The bagplot visualizes a two-dimensional data's location, spread, correlation and skewness. Its main components are the depth median, the bag, the fence and the loop. The depth median is the bivariate median, the location in the graph with the greatest depth. It is indicated in the graph with a red asterisk. The bag, indicated by a dark blue region, contains 50% of the observations. The fence, created by expanding the bag 3 times, serves to separate the inliers from the outliers. The loop is the region inside the fence but outside the bag; it is indicated in the graph by a light blue region.

The bagplot represents in two dimensions several characteristics of a bivariate data. The data's general location is indicated by the depth median, its spread is indicated by the size of the bag, the correlation of the two datasets can be deduced from the orientation of the bag, and finally, the skewness can be observed from the shape of the bag and the loop.

5. Results and Discussion

5.1. *Training and Testing Window Size*

We used the depth median of the Pareto points' bagplot as the reference point in comparing their overall performance against the other indicators. The depth median, as we can recall, is the bivariate median of the spread. It is indicated in the graph with a red asterisk.

Figure 2 shows the bagplots of the performance of the Pareto points over the training periods of Periods A, B and C. The performance of the 5 technical indicators are also presented in the graphs, together with the performance of the market, which is indicated by BH (the Buy-and-hold trading strategy). As there is no Sharpe ratio associated with BH (since it only trades once, and thus have an undefined standard deviation), we have decided to plot it in the graph with a Sharpe ratio value of 0. With the depth median as the representative of the Pareto points, we observe the Pareto points outperformed all indicators in terms of Sharpe ratio and it outperformed all but Linear Regression in terms of percent profit.

Analyzing now the testing phase (results shown in Figure 3), we observe that the Pareto points did not perform equally well over the three periods: the performance improved from Periods A to C in Period A, the Pareto points were outperformed by four indicators; in period B it lies in the vicinity of the other indicators; finally, in Period C, it outperforms all indicators in terms of both objective functions. The reason for this could be that, with a bigger training window, the system is more exposed to different market price movements. We therefore choose an optimization window of 1320 trading days (the window size of Period C).

We now validate the window size of 1320 trading days over the two periods succeeding it, Periods D and E. The date ranges for these two additional tests are given in Table 2 while the results are presented in Figure 4. We observe that the Pareto points outperformed all technical indicators on both objective functions for both training and testing data. Furthermore, in Period E, some Pareto points outperformed the market's performance.

Table 2
Additional testing and training periods used in validating the performance of the selected window size.

Period	Training Range	Testing Range
D	7/17/1986 to 10/3/1991	10/4/1991 to 12/20/1996
E	10/4/1991 to 12/20/1996	12/23/1996 to 3/25/2002

5.2. *Population size and number of generations*

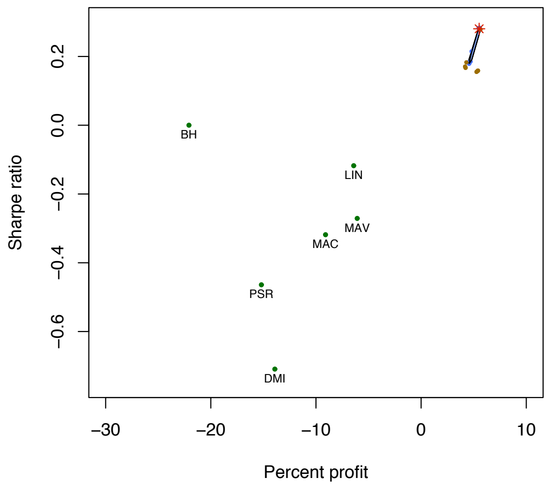
We varied the values of the population size and number of generations, both parameters ranging from 100 to 500, while keeping the other parameters constant. Tests were then conducted over Period C. We present the results in Figure 5.

From the figures we could see that the population size that yielded higher percent profits is 200, while the one which yielded higher Sharpe ratios is 500. Depending on our trading preference, we can choose either 200 or 500 for our population size. If we decide to have higher profits, then we select the population size of 200. After this choice, choosing the number of generations to use is a bit easier as we see that for the population size of 200, running the optimization at 400 generations yielded good results. Note that even if we have chosen to have higher percent profit over higher Sharpe ratio, the values of our the system's Sharpe ratios are still better than those of the technical indicators(see Figure 6(a)).

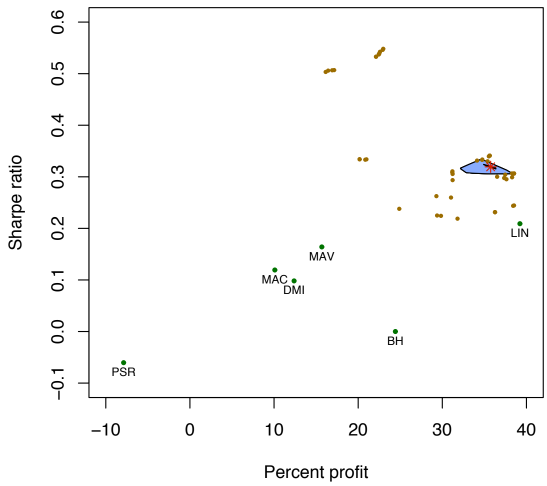
5.3. *Recomputation using the selected parameters*

We now present the tuned system with the parameters we have selected in the above experiments (i.e., setting the window size to 1320, population size to 200, archive size to 100, and number of generations to 400). The runs were made over Periods C, D and E. The results are shown as bagplots in Figure 6.

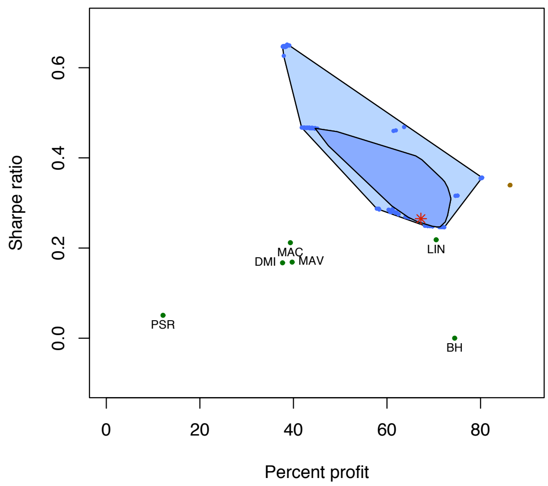
In Figure 7, we present a representative Pareto front from Training Periods C, D and E. Table 4 shows a summary of the performance of all the Pareto points over Periods C, D and E.



(a) Period A (window size=330; date range: 4/28/1981 to 8/16/1982), performance on training data

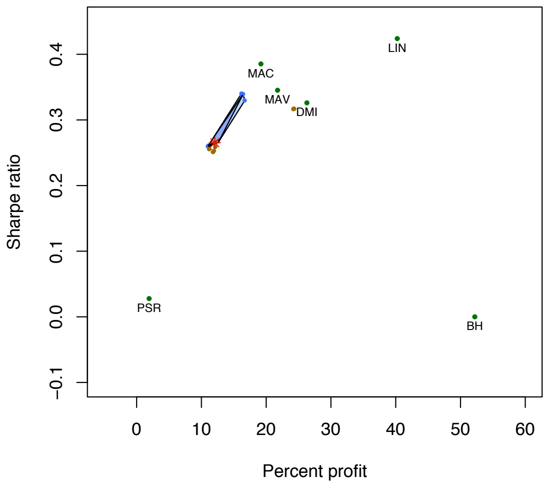


(b) Period B (window size=660; date range: 4/28/1981 to 12/2/1983), performance on training data

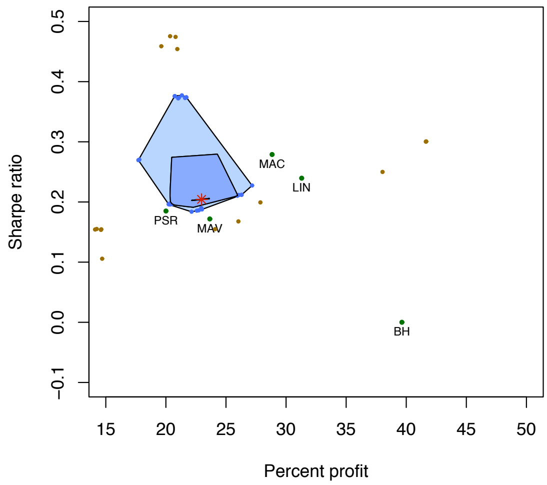


(c) Period C (window size=1320; date range: 4/28/1981 to 7/16/1986), performance on training data

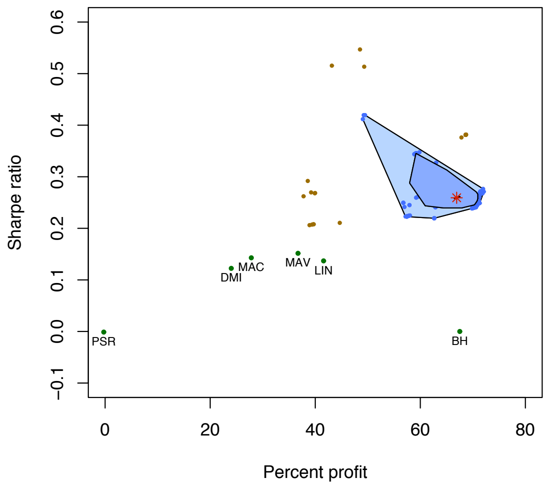
Fig. 2. Training performance of the Pareto points over Periods A, B, and C. BH (the Buy-and-hold trading strategy) represents the performance of the market. The red star in the figure indicates the location of the Depth Median- the bivariate median- of the Pareto points' bagplot. With the Depth Median as the representative of the Pareto points, we observe the Pareto points outperformed all indicators in terms of Sharpe ratio and it outperformed all but Linear Regression in terms of percent profit.



(a) Period A (window size=330; date range: 8/17/1982 to 12/2/1983), performance on testing data

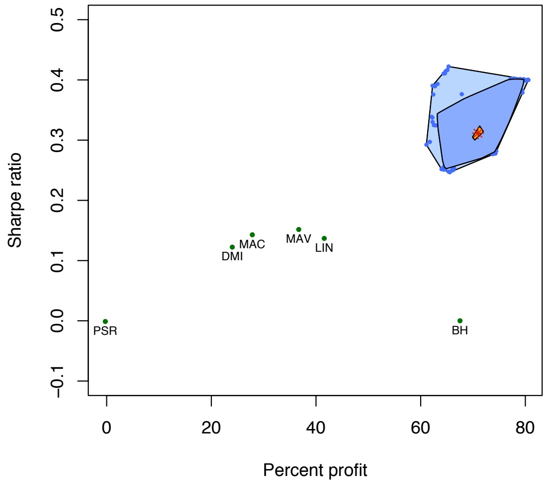


(b) Period B (window size=660; date range: 12/5/1983 to 7/16/1986), performance on testing data

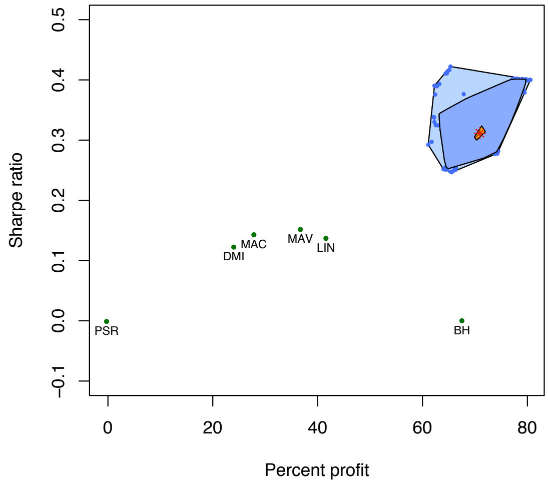


(c) Period C (window size=1320; date range: 7/17/1986 to 10/3/1991), performance on testing data

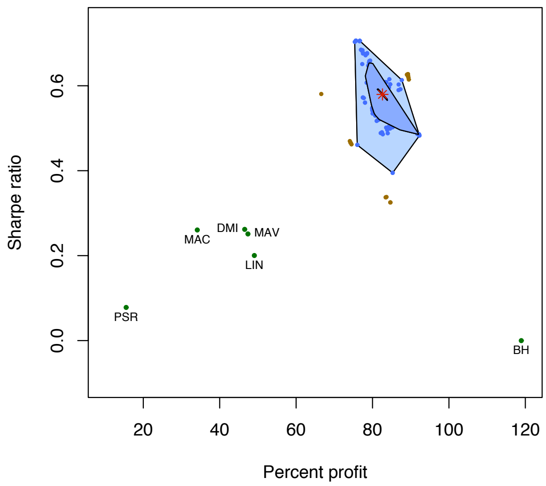
Fig. 3. Testing performance of the Pareto points over Periods A, B and C. The performance improved from Periods A to C: in Period A, the Pareto points were outperformed by four indicators; in period B it lies in the vicinity of the other indicators; finally, in Period C, it outperforms all indicators in terms of both objective functions.



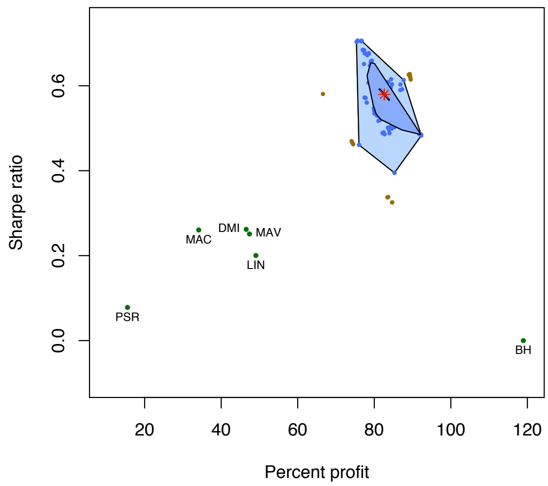
(a) Period D (date range: 7/17/1986 to 10/4/1991), performance on training data



(b) Period D (date range: 10/4/1991 to 12/20/1996), performance on testing data



(c) Period E (date range: 10/4/1991 to 12/20/1996), performance on training data



(d) Period E (date range: 12/23/1996 to 3/25/2002), performance on training data

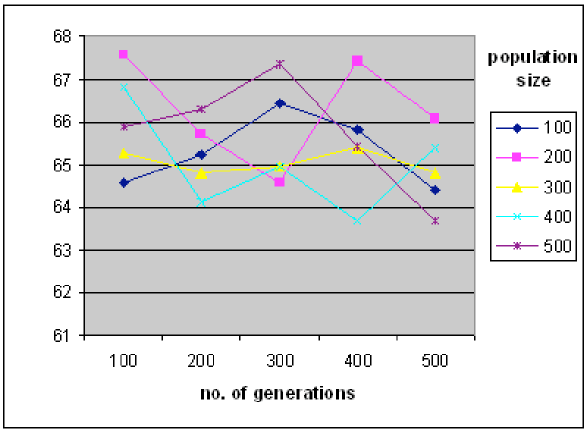
Fig. 4. Additional testing and training periods used in validating the performance window size= 1320. We observe that the Pareto points outperformed all technical indicators on both objective functions for both training and testing data. Furthermore, in Period E, some Pareto points outperformed the market's performance.

Table 3
Performance of the Pareto points over training data. A total of 30 Pareto fronts were obtained from 30 independent runs. A comparison is made with the performance of the technical indicators and the Buy-and-Hold.

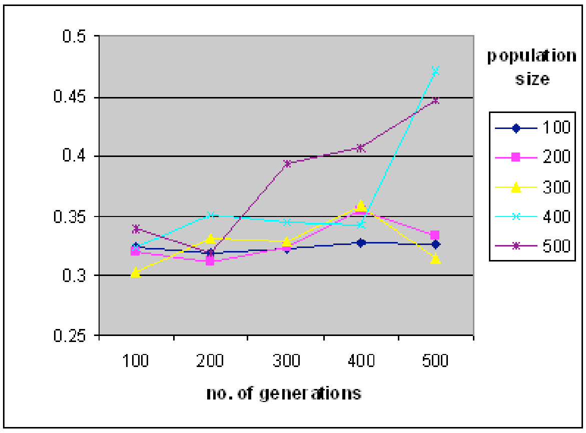
	Training Period A		Training Period B		Training Period C	
	% Profit	Sharpe	% Profit	Sharpe	% Profit	Sharpe
Pareto Points						
Depth Median	68.12	0.2718	76.8981	0.3768	83.33	0.6011
Best	86.33	2.5853	80.5756	0.5263	92.28	0.71
Average	67.41	0.3550	73.1537	0.3490	83.23	0.58
Std.Dev	12.60	0.2733	7.0627	0.0652	6.6729	0.0877
Indicators						
DMI	37.66	0.1672	24.01	0.1224	46.53	0.2619
LIN	70.52	0.2183	41.58	0.1368	49.08	0.2003
MAC	39.34	0.2119	27.82	0.1429	34.14	0.2604
MAV	39.73	0.1688	36.71	0.1517	47.40	0.2512
PSR	12.12	0.0508	-0.26	-0.0011	15.50	0.0780
Buy-and-Hold	74.46	-	67.52	-	118.94	-

Table 4
Performance of the Pareto points over testing data. A total of 30 Pareto fronts were obtained from 30 independent runs. A comparison is made with the performance of the technical indicators and the Buy-and-Hold.

	Testing Period A		Testing Period B		Testing Period C	
	% Profit	Sharpe	% Profit	Sharpe	% Profit	Sharpe
Pareto Points						
Depth Median	64.23	0.2881	76.09	0.5578	43.75	0.2120
Best	71.51	0.53	84.68	0.9067	60.57	0.49
Average	62.19	0.2857	71.86	0.5050	43.95	0.27
Std.Dev	9.24	0.0591	9.91	0.0901	6.62	0.13
Indicators						
DMI	24.01	0.1224	46.53	0.2619	11.35	0.0526
LIN	41.58	0.1368	49.08	0.2003	47.86	0.1374
MAC	27.82	0.1429	34.14	0.2604	17.65	0.0720
MAV	36.71	0.1517	47.40	0.2512	32.80	0.1249
PSR	-0.26	-0.0011	15.50	0.0780	2.11	0.0082
Buy-and-Hold	67.52	-	118.94	-	58.45	-



(a) Average value of the percent profit of the Pareto points



(b) Average value of the Sharpe ratio of the Pareto points

Fig. 5. Effect of size of population and number of generation on the objective functions percent profit and Sharpe ratio

5.4. *Distribution of solutions*

From the bagplots of Periods C , D and E (Figure 6), we could draw out the following observations. The general location of the data, represented by the depth median, suggest the good performance of the system. The depth median is generally located to the upper right of the other indicators, indicating higher values for both objective functions. The differing orientations of the bags across the different periods confirm our claim that the two objective functions are not necessarily correlated. The size and shape of the bags indicate the existence of a diversity of solutions.

5.5. *Performance of the stock trading system optimized with NSGA-II*

Tables 5- 5 show the results of the performance of the trading system as optimized by the NSGA-II algorithm. We observe that even after altering the different parameters of NSGA-II, there is little change in the quality of solutions.

This result is in line with Engelbrecht observation that PSO is found to be effective in optimization problems requiring real-valued decision variables [12] and it adds on the report of Hassan et al. [16] about the advantages of PSO over GA .

5.6. *Running Time*

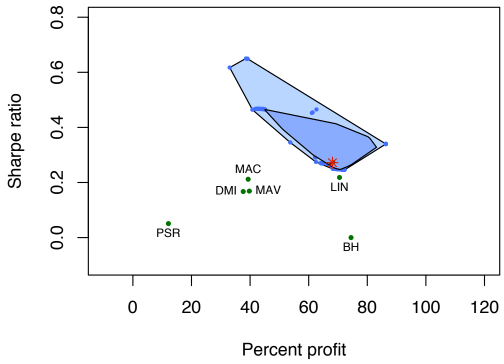
The system was fast enough to be used in actual decision making. An optimization of 1320 trading days only took 3 seconds in a computer with 1.6 GHz processor and 768 MB memory. To present a rough comparison, Subramanian et al.'s experiment, optimized their system over 6 months of intraday data, took 2 hours and 30 minutes to train for 12 generations. We note that our optimization was run for 100 generations, and without the discretization

Table 5
Effect of changing the values of the probability of crossover (pcross) and probability of mutation (pmut) on the performance the trading system as optimized by NSGA-II

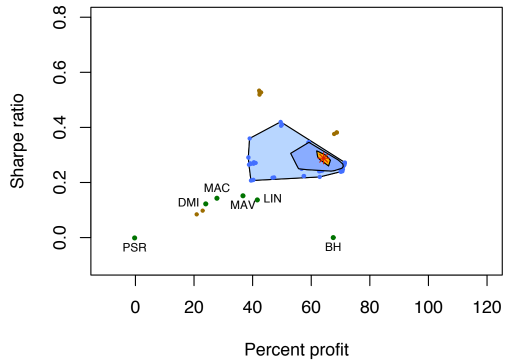
pcross	pmut	avg %Profit	avg Sharpe
0.9	0.2	3.17	0.0127
0.9	0.5	3.18	0.0126
0.9	0.033	3.12	0.0124
0.9	0.33	3.18	0.0129
0.9	0.1667	3.29	0.0134

Table 6
Effect of changing the values of the distribution index for real variable SBX crossover (eta-c) and the distribution index for real variable polynomial mutation (eta-m) on the performance the trading system as optimized by NSGA-II.

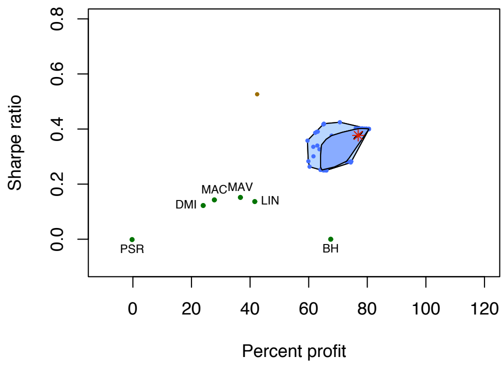
eta-c	eta-m	avg %Profit	avg Sharpe
5	5	3.19	0.0129
5	10	3.25	0.0130
10	10	3.13	0.0125
10	20	3.18	0.0126
15	20	3.07	0.0123
25	25	3.12	0.0124



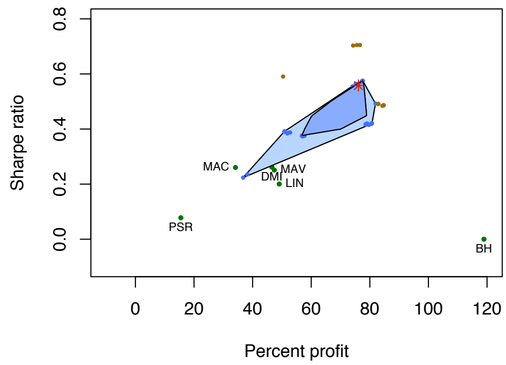
(a) Period C (date range: 2/5/1976 to 4/27/1981), performance on training data



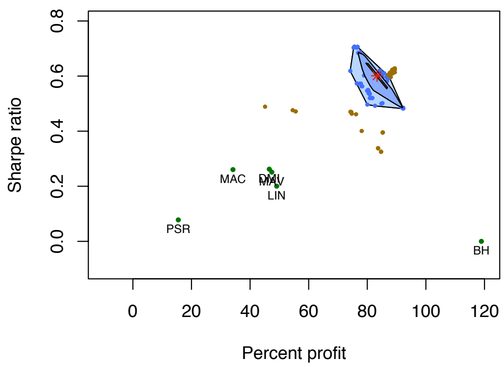
(b) Period C (date range: 4/28/1981 to 7/16/1986), performance on testing data



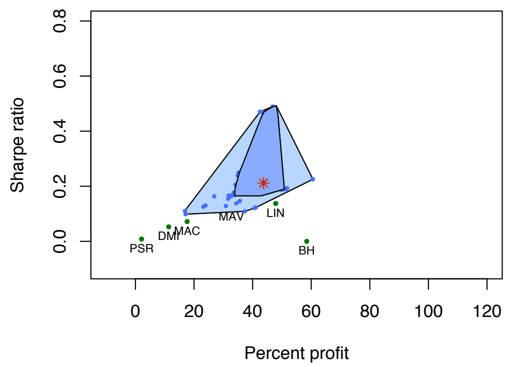
(c) Period D (date range: 2/5/1976 to 4/27/1981), performance on training data



(d) Period D (date range: 4/28/1981 to 7/16/1986), performance on testing data

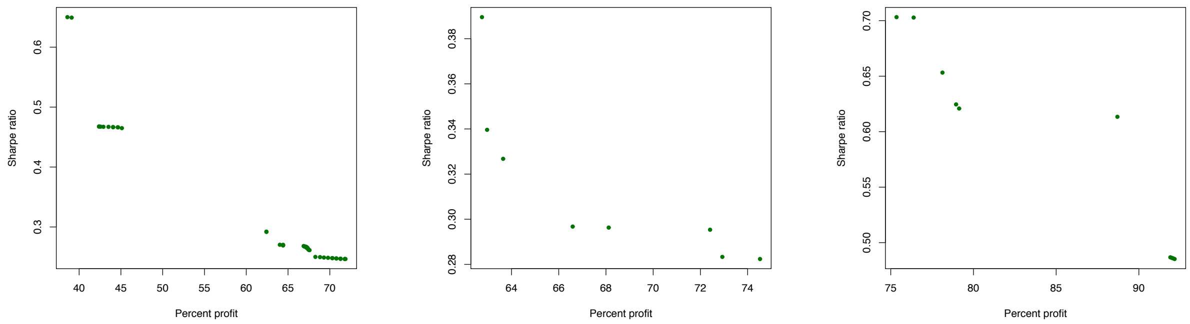


(e) Period E (date range: 7/17/1986 to 10/4/1991), performance on training data



(f) Period E (date range: 10/4/1991 to 12/20/1996), performance on testing data

Fig. 6. Performance of the Pareto points using the selected parameters: window size=1320, population size=200 and number of generations = 400.



(a) Period C (b) Period D (c) Period E

Fig. 7. Pareto fronts obtained from a representative run from Training Periods C, D and E. The representative run was chosen as the median run.

Table 7
Effect of changing the value of the population size on the performance the trading system as optimized by NSGA-II. The tests were run with 100 generations.

pop size	avg pct profit	avg sharpe
100	3.18	0.0126
200	-1.46	-0.0101
300	2.90	0.0116
400	-1.88	-0.0135

Table 8
Effect of changing the value of the number of generations on the performance the trading system as optimized by NSGA-II. The tests were run with population size of 100.

num gen	avg pct profit	avg sharpe
100	3.18	0.0126
200	-1.46	-0.0101
300	2.93	0.0117
400	2.93	0.0117

of the decision variables.

6. Conclusion

This paper has presented a stock trading system based on multi-objective particle swarm optimization. Using historical end-of-day market data, the system utilized the trading signals from a set of financial technical indicators in order to develop a trading rule which is optimized for two objective functions, namely, Sharpe ratio and percent profit.

The system performed well on both training and out-of-sample data. In terms of percent profit, the system outperformed most, if not all, of the indicators under study, and, in some instances, it even outperformed the market itself. In terms of Sharpe ratio, the system consistently performed significantly better than all the technical indicators. The proposed system also performed far better than the system optimized by NSGA-II.

These results show the potential of the proposed system as a tool for making stock trading decisions and encourage further refinement in the system. Among the improvements could be explored in the future are the study of other technical indicators, the study of other objective functions such as length of trades and maximum drawdown and the addition of more objective functions.

References

[1] A. Abraham, C. Grosan, S. Y. Han, A. Gelbukh, Evolutionary multiobjective optimization approach for evolving ensemble of intelligent paradigms for stock market modeling, in: Lecture Notes in Computer Science, 4th Mexican International Conference on Artificial Intelligence (MICAI'05), vol. 3789, Springer Verlag, Germany, 2005, pp. 673–681.

[2] R. Armananzas, J. A. Lozano, A multiobjective approach to the portfolio optimization problem, in: Proceedings of the 2005 IEEE Congress on Evolutionary Computation (CEC 2005), vol. 2, 2005, IEEE Press.

[3] S. Chou, H. Hsu, C. Yang, F. Lai, A stock selection DSS combining ai and technical analysis, Annals of Operations Research 75 (0) (1997) 335–353, springer, Netherlands.

[4] M. Clerc, J. Kennedy, The particle swarm explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 5873.

[5] C. Coello, M.Lechuga, MOPSO: A proposal for multiple objective particle swarm optimization, in: Congress on Evolutionary Computation (CEC2002), vol. 2, Piscataway, New Jersey, 2002, IEEE Service Center.

[6] C. Coello, G. Pulido, M. Salazar, Handling multiple objectives with particle swarm optimization, IEEE Transactions on Evolutionary Computation 8(3) (2004) 256279.

[7] D. de la Fuente, A. Garrido, J. Laviada, A. Gómez, Genetic algorithms to optimise the time to make stock market investment, in: GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM Press, New York, NY, USA, 2006.

[8] K. Deb, S. Agrawal, A. Pratab, T. Meyarivan, A fast elitist nondominated sorting genetic algorithm for multiobjective optimization: NSGA-II, in: Parallel Problem Solving from Nature VI Conference, 2000.

[9] L. Diosan, A multi-objective evolutionary approach to the portfolio optimization problem, in: CIMCA '05: Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce Vol-2 (CIMCA-IAWTIC'06), IEEE Computer Society, Washington, DC, USA, 2005.

[10] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss, C. Stummer, Ant colony optimization in multiobjective portfolio selection, in: 44th Metaheuristics Intl. Conf., MIC'2001, Porto,Portugal, 2001.

[11] B. Egeli, M. Ozturan, B. Badur, Stock market prediction using artificial neural networks, in: Hawaii International Conference on Business, 2003.

[12] A. Engelbrecht, Fundamentals of Computational Swarm Intelligence, John Wiley & Sons, 2005.

[13] A. Fernandez, S. Gomez, Portfolio selection using neural networks, Computers and Operations Research 34 (2007) 1177.

[14] F. Fernández-Rodríguez, C. González-Martel, S. Sosvilla-Rivero, Optimisation of technical rules by genetic algorithms: Evidence from the madrid stock market, Fundacíon de Estudíos de Economía Aplicada (August 2001).

[15] R. Fukumoto, H. Kita, Designing trading agents for an artificial market with a multi-objective genetic algorithm, in: ICCIMA '01: Proceedings of the Fourth International Conference on Computational Intelligence and Multimedia Applications, IEEE Computer Society, Washington, DC, USA, 2001.

[16] R. Hassan, B. Cohanim, O. de Weck, G. Venter, A comparison of particle swarm optimization and the genetic algorithm, in: 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, No. AIAA-2005-1897, Austin, Texas, 2005.

[17] S. Janson, D. Merkle, M. Middendorf, Molecular docking with multi-objective particle swarm optimization, Appl. Soft Comput. 8 (1) (2008) 666–675.

[18] M. Kearns, L. Ortiz, The Penn-Lehman automated trading project (2003).

[19] G. Kendall, Y. Su, A particle swarm optimization approach in the construction of optimal risky portfolios, in: 23rd IASTED International Multiconference, Artificial Intelligence and Applications, 2001.

[20] J. Kennedy, R. Eberhart, Particle swarm optimization, in: 1995 IEEE International Conference on Neural Networks, Piscataway, New Jersey, 1995, IEEE Service Center.

[21] K.Kurumatani, et al., U-mart: A virtual stock market as a forum for market structure analysis and engineering, in: 5th joint Conderence on Information Science (JCIS'00), 1st Int'l Workshop on Computational Intelligence in Economics and Finance, 2000.

[22] J. D. Knowles, D. Corne, Approximating the nondominated front using the pareto archived evolution strategy, Evolutionary Computation 8 (2) (2000) 149–172.

[23] J. Li, S. Taiwo, Enhancing financial decision making using multi-objective financial genetic programming, in: Proc.of 2006 IEEE Congress on Evolutionary Computation (CEC'06), Sheraton Vancouver Wall Centre, Vancouver, BC, Canada, 2006.

[24] J. Li, E. P. K. Tsang, Improving technical analysis predictions: An application of genetic programming, in: Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference, AAAI Press, 1999.

[25] L. Lin, L. Cao, J. Wang, C. Zhang, The applications of genetic algorithms in stock market data mining optimisation, in: A. Zanasi, N. Ebecken, C. Brebbia (eds.), Data Mining V: Data Mining, Text Mining and their Business Applications (DATA MINING 2004), WIT Press, Southampton, Boston, 2004, p. 448.

[26] Y. Niu, L. Shen, The optimal multi-objective optimization using pso in blind color image fusion, mue 00 (2007) 970–975.

[27] C. R. Raquel, P. C. Naval, An effective use of crowding distance in multiobjective particle swarm optimization, in: GECCO '05: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, ACM Press, New York, NY, USA, 2005.

[28] P. J. Rousseeuw, I. Ruts, J. W. Tukey, The bagplot: A bivariate boxplot, The American Statistician 53 (4) (1999) 382–387.

[29] C. Schoreels, B. Logan, J. M. Garibaldi, Agent based genetic algorithm employing financial technical analysis for making trading decisions using historical equity market data, in: IAT '04: Proceedings of the Intelligent Agent Technology, IEEE/WIC/ACM International Conference on (IAT'04), IEEE Computer Society, Washington, DC, USA, 2004.

[30] W. F. Sharpe, The sharpe ratio, The Journal of Portfolio Management (1994) 49–58.

[31] M. R. Sierra, C. C. Coello, Multi-objective particle swarm optimizers: a survey of the state-of-the-art, Int. J. Comput. Intell. Res. 2 3 (2006) 287308.

[32] R. Simutis, S. Masteika, Intelligent stock trading systems using fuzzy-neural networks and evolutionary programming methods, Solid State Phenomena 97-98 (2004) 59–64.

[33] A. Skabar, I. Cloete, Neural networks, financial trading and the efficient markets hypothesis, in: ACSC '02: Proceedings of the twenty-fifth Australasian conference

on Computer science, Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 2002.

[34] H. Subramanian, S. Ramamoorthy, P. Stone, B. J. Kuipers, Designing safe, profitable automated stock trading agents using evolutionary algorithms, in: GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM Press, New York, NY, USA, 2006.

[35] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the Strength Pareto Evolutionary Algorithm, Tech. Rep. 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland (2001).