Complex Stock Trading Strategy Based on Particle Swarm Optimization

Fei Wang, Philip L.H. Yu and David W. Cheung

Abstract— **Trading rules have been utilized in the stock market to make profit for more than a century. However, only using a single trading rule may not be sufficient to predict the stock price trend accurately. Although some complex trading strategies combining various classes of trading rules have been proposed in the literature, they often pick only one rule for each class, which may lose valuable information from other rules in the same class. In this paper, a complex stock trading strategy, namely weight reward strategy (WRS), is proposed. WRS combines the two most popular classes of trading rules−moving average (MA) and trading range break-out (TRB). For both MA and TRB, WRS includes different combinations of the rule parameters to get a universe of 140 component trading rules in all. Each component rule is assigned a start weight and a reward/penalty mechanism based on profit is proposed to update these rules' weights over time. To determine the best parameter values of WRS, we employ an improved time variant Particle Swarm Optimization (PSO) algorithm with the objective of maximizing the annual net profit generated by WRS. The experiments show that our proposed WRS optimized by PSO outperforms the best moving average and trading range break-out rules.**

I. Introduction

TRADING rules are widely used in financial market as a technical analysis tool for security trading. Typically, they predict the future price trend by analyzing historical price movement and initiates buy/sell signals accordingly. Trading rules have developed for more than a century and many empirical studies provided supporting evidence to the significant profitability of different trading rules [1][2][3][4][5]. Until nowadays trading rules are commonly used by practitioners to make trading decisions in many financial markets.

Pring [6], however, argued that no single trading rule can ever be expected to forecast all price trend and it is important to combine these simple rules together to get a complex trading strategy. Hsu and Kuan [7] first examined the profitability of three classes of complex trading strategies: learning strategies (LS), vote strategies (VS) and fractional position strategies (FPS). Their results showed that the three classes of complex trading strategies did not provide significant improvement as compared with simple trading rules. However, the failure of these complex trading strategies is because they are relatively primitive. For example, LS picked the best simple trading rule for trading decision making each time instead of combining all rules in an appropriate manner. For VS and FPS, both of them regarded all simple trading rules as equally important without considering their relative performances.

To address the above problems, Subramanian et al. [8] proposed a weighted combination of simple trading rules. In their study, each component rule associated with a given weight initiates its own signal and the signal to be implemented is eventually determined by the sum of these weighted signals. They created this combination by applying a Genetic Algorithm (GA) to optimize the best set of weight vectors. Thereafter Briza et al. [9] proposed a similar stock trading strategy whose weight vector was optimized by Particle Swarm Optimization (PSO). Both combined strategies are found to outperform the best component trading rules in terms of profit of the test set. However, they only considered a common used rule for each class of trading rule in their studies. This may not guarantee that the trading rules under consideration always perform better than those not considered. Therefore it is important to include various combinations of parameters for each class of rule as many as possible to get a comprehensive coverage of component rules. Note that the weights of component rules were held fixed during the whole trading period in their approaches. Given such a complex and dynamic market, a trading strategy with a static choice of component weights is hard to perform well consistently over time. In this regard, an objective of this paper is to consider a dynamic updating scheme for component weights.

In this paper, we present a complex stock trading s-trategy called weight reward strategy (WRS) which combines two classes of the simplest and most popular trading rules−moving average and trading range break-out [2][5]. For moving average, we get 119 rules by considering different values of its two parameters. For trading range break-out, we get 21 rules by taking 21 different values of its single parameter. Therefore there are 140 simple rules in all. All parameters are well selected to represent each class's performance in a wide range [10]. Each component rule is assigned a start weight, and a reward/penalty mechanism based on component rules' performance is proposed to update their weights over time. The trading signal of WRS is determined by the weighted sum of component rules' signals and two additional signal threshold parameters.

Together with component rules' start weight and other five parameters of WRS to be discussed later, there are altogether 145 parameters for WRS. We use an improved time variant

Fei Wang is with the Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong (email: fwang@cs.hku.hk).

Philip L.H. Yu is with the Department of Statistics and Actuarial Science, The University of Hong Kong, Pokfulam, Hong Kong (email: plhyu@hku.hk).

David W. Cheung is with the Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong (email: dcheung@cs.hku.hk).

Particle Swarm Optimization (PSO) [11] to optimize the best set of the 145 parameters. In the literature, GA and PSO are the most popular optimization algorithms used to optimize trading rules [4][8][9][12]. We choose PSO rather than GA because PSO is not only easy to be implemented, but also it has higher computation efficiency and can achieve the same performance as compared with GA [13][14].

The rest of this paper is organized as follows: Section II briefly introduces PSO algorithm. Section III describes proposed WRS in detail and the optimization of WRS is presented in Section IV. Section V discusses the experimental results and conclusion is given in Section VI.

II. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a stochastic evolution algorithm based on swarm intelligence, which was first inrtoduced by Kennedy and Eberhart in 1995 [15]. Since its inception, PSO has shown great success in solving function optimization problems and has been widely applied in a variety of engineering applications [9][16].

PSO is motivated by the behavior of bird flocks in finding food. Suppose a flock of birds want to find food, but they do not know where the food is before they find it. However, this bird flock can always find their food at last. PSO uses a swarm of particles to simulate these birds. Each particle is a possible solution of the optimization problem and has a random initial position X and velocity V. The objective function targeted to be optimized is used to evaluate each particle position's fitness. Higher fitness means a better position. For each particle, PSO uses $pbest$ to record the best position this particle has arrived. For the whole swarm, $gbest$ is used to record the global best position achieved by all particles. At time t, PSO updates each particle's velocity using the following equation:

$$V_{t+1} = wV_t + c_1 r_1 (pbest - X_t) + c_2 r_2 (gbest - X_t) \quad (1)$$

where w is the inertia weight, c_1, c_2 are the acceleration coefficients and r_1, r_2 are two random numbers in the range between 0 and 1 [17].

The first term of equation (1) indicates an inertia for a particle wondering in the search space. The second term represents self-cognition of past experience of a particle, i.e., the particle tends to move towards its past best position. Similarly, the third term indicates that particles have social cognition to the whole swarm and are attracted by the global best position. For PSO, inertia weight w controls the influence of previous velocity on a particle. A large w allows particles to explore more search space for the optimal position, while small w helps swarm to search in a local area for the exact solution. Coefficients c_1 and c_2 control the influence of $pbest$ and $gbest$ on particles' movement. A higher value of c_1 means each particle is more likely to be attracted to a different position, so the whole swarm is more widespread in the search space. Its effect is similar to w. In contrast a high value of c_2 leads all particles converge to the current global best position.

After updating the velocity, each particle will move to a new position according to:

$$X_{t+1} = X_t + V_{t+1} \quad (2)$$

This particle movement will repeat iteratively until all particles converge to the optimal position at last, like when birds find the food at the end of searching.

III. Weight Reward Strategy

In this section, we describes how weight reward strategy (WRS) utilizes different simple trading rules to generate trading signals and how it rewards and penalizes these rules according to their trading performance.

A. Moving average and trading range break-out

WRS is based on moving average (MA) and trading range break-out (TRB) because they are the two of simplest and most popular simple trading rules [2].

In MA, there are two averages of stock prices over two moving windows of nl days and ns days respectively, where $nl > ns$ so the former average is long-period average and the later one is short-period average. Both averages are recalculated and updated each trading day. The signal generation of MA is simple. Consider a trading day, MA initiates buy (sell) signal if the short-period moving average is above (below) the long-period moving average.

The second trading rule is TRB which is simpler than MA. On trading day d, TRB initiates buy (sell) signal if current day's stock price is higher (lower) than the highest (lowest) stock price during the past n days. The past n-day prices form a trading range, and the buy or sell signal is invoked when one day's stock price breaks out the range.

For both MA and TRB, we can get different rules by taking different parameter values. It is important to add as many as rules to WRS's rule pool because we cannot guarantee the selected parameter value is always better than the others. Therefore, we combine 119 MA rules and 21 TRB rules together to get a comprehensive coverage of MA and TRB.

B. Signal generation

In WRS, each component rule r_i is assigned a start weight wt_i, which measures the influence of r_i to the signal generated by WRS. Consider a trading day d, each component rule initiates a signal $s_{i,d}$. This signal $s_{i,d}$ takes value 1, 0 and -1 if the signal is 'buy', 'do nothing' and 'sell' respectively. The signal of WRS on trading day d is given by:

$$s_d = \sum wt_i s_{i,d} \quad (3)$$

where the sum of the all weights ($\sum wt_i$) should be 1 so that s_d is between -1 and 1.

Note that s_d summarizes all component rules' views on stock trend. If s_d is close to 1, this means most of influential component rules suggest buy signals. On the contrary, s_d nearing -1 means more influential rules suggest sell signals. So we propose that WRS initiates a buy (sell) signal if s_d is greater (smaller) than a positive buy (negative sell) threshold,

bth (sth); otherwise WRS do not initiates any signal and investors do nothing on that day. Higher threshold represents the strategy is more strict in buy or sell and lower threshold represents a more tolerant strategy. The signal generation of WRS is shown in Fig. 1.

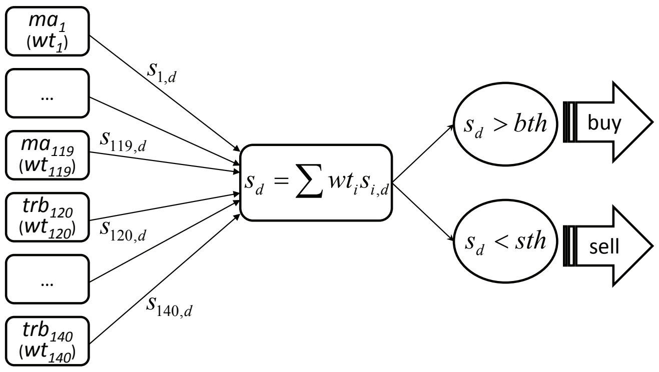

Fig. 1. Signal generation of WRS

C. Reward and penalty of component rules

In WRS, each component rule r_i is associated with a start weight wt_i. It represents how much we believe this rule. However, these rules' performance may change during trading, especially over a long time period. It is reasonable to reward a good rule by adding more weight to it and to penalize a bad rule by deducting some weight from it. The good or bad is measured by the profitability of rules in recent time, and the updating of weights should be conducted at regular intervals. As a result, two time spans−memory span ms and review span rs, which are introduced in the learning strategy (LS), are used here. Memory span is a historical period used for evaluating the rule performance. Review span is the time interval over which the weights of component rules should be updated. We set $ms \geq rs$ as suggested by [7].

Suppose on trading day d, we evaluate all rule's performance and update their weights accordingly. Let $profit_i$ denotes the profit of rule r_i from day $d - ms$ to day $d - 1$. For those nonprofitable rules, we deduct their weights by a constant:

$$wt_i = wt_i - \frac{rf}{ruleNum}, \quad \text{if } profit_i < 0 \tag{4}$$

where $ruleNum$ is the total number of component rules and rf is a parameter called reward factor controlling the degree for penalty and reward. It is noted that wt_i should not be negative, so wt_i is set to zero if it is smaller than $\frac{rf}{ruleNum}$.

All the weights deducted from the nonprofitable rules are summed to form a temporary variable $reward$. Then we increase the weight of those profitable rules using following equation:

$$wt_i = wt_i + \frac{reward}{profitNum}, \quad \text{if } profit_i > 0 \tag{5}$$

where $profitNum$ is the number of profitable rules found in the memory span.

Note that the sum of weights remains unchanged after the penalty and reward. However, if most of the rules are nonprofitable and only a few rules are profitable, above reward/penalty approach may add too much weight to those few profitable rules. Imagine that there are 100 rules in which only one rule is profitable in memory span, the weight increment of the only profitable rule is 99 times of the weight decrement of any other rule. The reward may be too much, especially when the rule is just profitable in a short period of time. To avoid a huge reward, we replace Equation (4) with:

$$wt_i = wt_i - \frac{rf}{ruleNum} \times \frac{profitNum}{ruleNum}, \quad \text{if } profit_i < 0 \tag{6}$$

where term $\frac{profitNum}{ruleNum}$ guarantees that the penalty weight of any nonprofitable rule and the reward weight of any profitable rule is capped at $\frac{rf}{ruleNum}$. It is noted that when all rules are profitable or all of them are nonprofitable, our reward/penalty mechanism would not be triggered as in the case there is no need to reward or penalize any rule.

IV. Optimization of WRS

For WRS, there are 140 start weights (wt_1 to wt_{140}), two time spans (ms, rs), two thresholds (bth, sth), and a reward factor (rf) to be determined. Identifying such a high dimensional parameter vector is a tough optimization problem. To tackle it, an improved time variant Particle Swarm Optimization (PSO) algorithm is used in this paper.

A. Time variant PSO

In PSO, the tradeoff between global exploration and local exploitation of particles is the main influencing factor to PSO's performance [17]. Generally, exploration should be enhanced at the early stage of searching so that more search space can be explored by particles. While at the later stage the algorithm should focus on exploitation to find the exact and accurate optimum.

To achieve these goals, Shi and Eberhart [18] suggested to reduce PSO's inertia weight w linearly over the iterations so as to help particles to find the optimal position more efficiently. In later work, Ratnaweera et al. [11] proposed a time-varying acceleration coefficients PSO which linearly reduce the first acceleration coefficient c_1, and increase the second acceleration coefficient c_2 over the iterations. Based on these studies, in this paper we lineally updates w, c_1 and c_2 according to the following iterative equations [11][18]:

$$w_t = (w_F - w_I)\frac{t}{max\_t} + w_I, \tag{7}$$

$$c_{1t} = (c_{1F} - c_{1I})\frac{t}{max\_t} + c_{1I}, \tag{8}$$

$$c_{2t} = (c_{2F} - c_{2I})\frac{t}{max\_t} + c_{2I}, \tag{9}$$

where w_I, w_t and w_F denote the initial, current and final value of w respectively (similar for c_1 and c_2), t is the current iteration number and max_t is the maximum number of iterations.

B. Objective function

The ultimate goal of any stock trading strategy is to make profit from the stock market, so the parameter optimization of WRS is led by this goal. We use annual net profit generated by the WRS as the objective function of the optimization. At first, each stock in the market is assigned the same amount of initial equity. During the trading, all the equity available for a stock can only be used to invest in this stock. Because we do not allow short selling, the equity for each stock is always positive. On the last day of trading, we sell all stocks in hand and sum their equities together to obtain the final equity. Then the annual net profit (ANP) of WRS can be calculated as follows:

$$ANP = \frac{\sum e_{kF} - \sum e_{kI}}{y \times \sum e_{kI}} \tag{10}$$

where e_{kI} and e_{kF} is the initial and final equity of stock k respectively, and y is the length of trading period in years.

In (10), we have already included the consideration of transaction cost so that ANP represents the average annual profit net of the transaction cost. For each buy or sell, 0.1% of the total turnover is cut from the equity as transaction cost in our study.

C. Start weights optimization

Recall that the sum of component rules' start weights is restricted to be 1, i.e., $\sum wt_i = 1$. It is difficult to satisfy this constraint if the weights are optimized directly by PSO because of its stochastic nature. In this regard, a new parameter vector α is introduced here and a one-to-one transformation between wt and α is used in this study:

$$wt_i = \frac{e^{\alpha_i}}{\Sigma e^{\alpha_i}}. \tag{11}$$

As (11) guarantees that the sum of wt is 1 regardless of the value of α, the new parameter vector α is optimized to get the best set of start weights via (11).

V. Experiments

A. Data

The constituent stocks of NASDAQ100, which are 100 of the largest domestic and international nonfinancial stocks on the Nasdaq Stock Market, are considered in our study. The daily stock closing price data from 1994 to 2010 are collected from Reuters 3000Xtra. Because not all stocks were issued before 1994, only 52 stocks having data through the whole period are considered in our experiments. The data from 1995 to 2002 is used for optimizing, in other words, training the WRS. The data from 2003 to 2010 is used for testing the profitability of the WRS and the simple trading rules. It is noted that for some component rules such as MA with nl = 250, it needs data over the past 250 days to calculate current day's long-period moving average. Therefore, the data in 1994 has been reserved for data preparation in training.

B. Experiment setup

The swarm size is set to 250 and the maximum number of iterations is set to 500. There is also a stoping criterion, that is, if $gbest$ keeps unchanged for at least 50 iterations, the optimization will stop. Table I gives the parameter settings of PSO and the search space boundary of WRS optimization.

TABLE I
Parameter settings of PSO and the boundaries of WRS parameters

PSO	Value	WRS	Boundary
w_I	0.9	α_i[1]	$[-1,1]$
w_F	0.4	ms	$[150,300]$
c_{1I}	2.5	rs	$[20,150]$
c_{1F}	0.5	rf	$[0,1]$
c_{2I}	0.5	bth	$[0,0.9]$
c_{2F}	2.5	sth	$[-0.9,0]$

[1] (i = 1..140)

In Table I, the parameter settings of PSO are based on the suggestions of [11][18]. In order to avoid the data snooping bias to any component rule in the training period, the range of α is set as $[-1, 1]$ so that the range of wt is approximately $[0.001, 0.05]$. There are about 21 trading days in a month and about 252 trading days in a year, so the minimum review span rs is set to be shortly less than one month and the maximum memory span ms is set to be a little bit longer than one year in terms of trading days. Because $ms \geq rs$, the minimum value of ms and the maximum value of rs are both set to be 150 trading days. Reward and penalty for component rules should not be too big each time, so the maximum value of reward factor rf is set to be 1. The minimum value of rf is 0 means that there is no reward and penalty in this case. For buy and sell threshold, [$-$0.9,0.9] is wide enough to cover the threshold range, so they are set as the lower and upper bound of sth and bth respectively.

C. Experimental results

After training, WRS is compared with the best MA (ns = 125, nl = 150) and the best TRB (n = 125) in terms of the annual net profit (ANP) in the testing period. In addition to the annual net profit, the average return per trade ($Avg.return$) is also compared. Suppose an investor buys N shares of a stock at price p_0 and sells them at the price p_1. The transaction cost per buy or sell is c. The investor pays $N \times p_0 \times (1 + c)$ to buy the N shares and collects $N \times p_1 \times (1 - c)$ by selling them, so the return of this trade is given by:

$$return = \frac{N \times p_1 \times (1-c)}{N \times p_0 \times (1+c)} - 1 = \frac{p_1 \times (1-c)}{p_0 \times (1+c)} - 1. \tag{12}$$

Therefore the return per trade is independent of the investment. The results are shown in Table II. The number of trades ($No.trades$) and the average holding day per trade

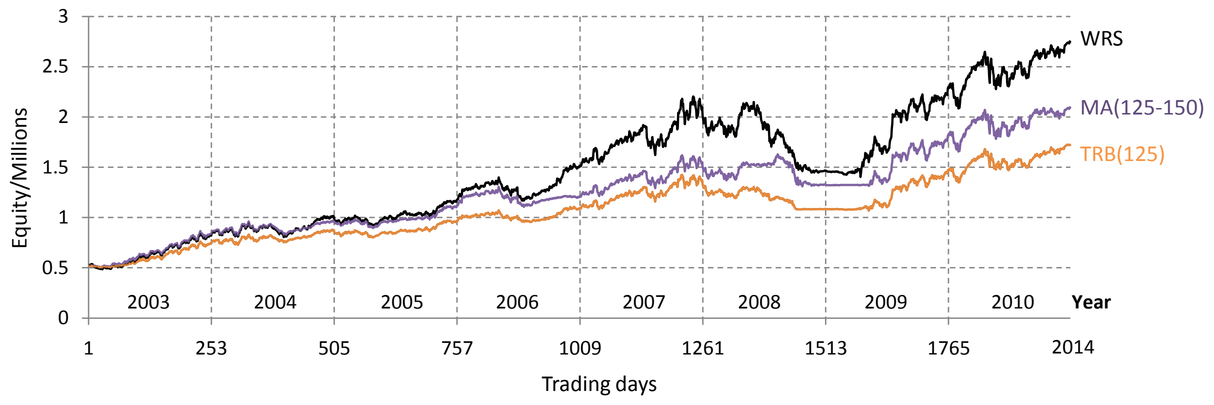

Fig. 2. Equity curves of WRS, MA(125-150) and TRB(125) in the testing period. Suppose the initial equity for each stock is \$0.01 million, so the initial equity for the market is \$0.52 million. There are altogether 2014 trading days from 2003 to 2010.

($Avg.hold\_day$) for each trading rule in the testing period are also given for statistic purpose. From Table II, we can find that both the annual net profit and the average return per trade of WRS is much higher than the best MA (ns = 125, nl = 150) and the best TRB (n = 125) in the testing period. This means WRS can take the advantage of component rules and outperform all of them. Table III gives more details about the 52 stocks' profits. It shows that WRS can generate significant profit from those profitable stocks and recover from those few nonprofitable stocks. To get a more comprehensive understanding to the performance of these three trading strategies, their equity curves are given in Fig. 2. It can be seen that the WRS keeps the highest cumulative equity during the eight years of testing period.

TABLE II

PERFORMANCE OF WRS, THE BEST MA(125-150) AND THE BEST TRB(125) IN THE TESTING PERIOD

	WRS	MA(125-150)	TRB(125)
ANP	**53.52%**	37.86%	28.88%
$No.trades$	264	532	277
$Avg.hold\_day$	320	120	228
$Avg.return$	**48.38%**	11.30%	22.22%

TABLE III

STOCK PROFIT SUMMARY OF WRS, THE BEST MA(125-150) AND THE BEST TRB(125) IN THE TESTING PERIOD

Strategy	Summary	Profitable stocks[1]	Nonprofitable stocks[1]	Total stocks
WRS	$Number$	35	17	52
	ANP	80.49%	-2.00%	53.52%
MA(125-150)	$Number$	37	15	52
	ANP	54.61%	-3.46%	37.86%
TRB(125)	$Number$	41	11	52
	ANP	37.37%	-2.77%	28.88%

[1] Profitable stock means the stock whose final equity is more than its initial equity and vice versa.

In the training period, the MA rule with ns = 200 and nl = 250 generates the highest annual net profit among all of the 119 MA rules, and the TRB rule with n = 200 generates the highest annual net profit among all of the 21 TRB rules. To demonstrate that the performance of simple trading rules may fluctuate over time, we also test the performance of these two rules in the testing period. The results are shown in Table IV and Table V. The MA(200-250) makes the highest profit among all MA rules in the training period, but its performance drops a lot in the testing period, even is below the average performance of MA. Although the TRB(200) performs well in both of the training and testing period, it is not the best TRB any more in the testing period. This result gives support to our complex trading strategy with adequate combination of simple trading rules.

TABLE IV

PERFORMANCE OF THE MA RULES IN THE TESTING PERIOD

	MA (200-250)	Worst MA (1-5)	Best MA (125-150)	MA $Average$
ANP	19.03%	2.78%	37.86%	23.05%
$No.trades$	305	14331	532	1933
$Avg.hold\_day$	210	4	120	79

TABLE V

PERFORMANCE OF THE TRB RULES IN THE TESTING PERIOD

	TRB (200)	Worst TRB (10)	Best TRB (125)	TRB $Average$
ANP	27,21%	6.02%	28.88%	20.84%
$No.trades$	186	3434	277	1116
$Avg.hold\_day$	344	17	228	137

If there is no reward and penalty to component rules during trading, WRS becomes Weight Strategy (WS). We also compare WRS and WS in the testing period to see the influence of our reward/penalty mechanism. The results are given in Table VI. Together with Table II, we can find that WS also generates higher annual net profit than any of the simple trading rules. However, both the highest ANP and $Avg.return$ are generated by WRS indicates that it worths to conduct the reward/penalty mechanism. For WRS, because there are reward and penalty, component rules' weights may fluctuate according to their recent performance as evidenced by the profile plots of two selected weights in the testing period as shown in Fig. 3.

TABLE VI
PERFORMANCE OF WRS AND WS IN THE TESTING PERIOD

	WRS	WS
ANP	**53.52%**	39.34%
$No.trades$	264	350
$Avg.hold\_day$	320	239
$Avg.return$	**48.38%**	22.56%

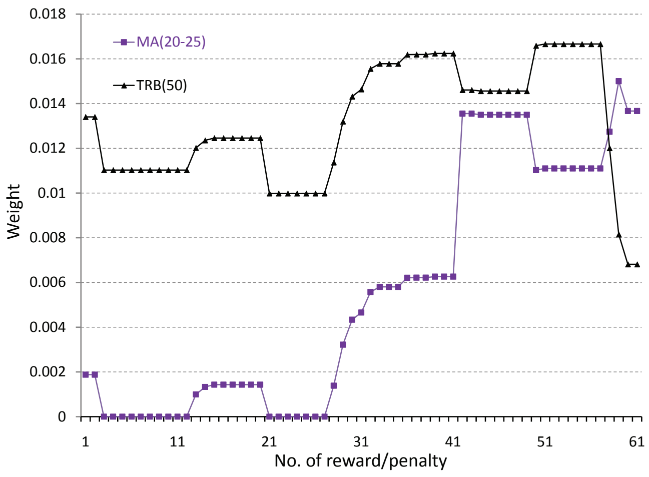

Fig. 3. The profile plots of the weights of MA (ns = 20, nl = 25) and TRB (n = 50) in the testing period. WRS updates the weights for 61 times during 8 years trading from 2003 to 2010. We observe that the weight of MA(20-25) keeps low at the beginning and becomes high eventually. The weight of TRB(50) keeps high for a long time and drops a lot at the end of trading.

VI. CONCLUSION

This paper has proposed a complex stock trading strategy, namely weight reward strategy (WRS), generated from different combinations of moving average and trading range break-out with their weights updated by a reward/penalty mechanism. A time variant Particle Swarm Optimization is used to optimize WRS. WRS outperforms the best moving average and trading range break-out rules in NASDAQ100 market from 2003 to 2010. For our future research, more simple trading rules could be included in the rule pool of WRS. Besides, it is often required to strike a balance between return and risk in investment, so multi-objective optimization in terms of profit and some risk measures such as Sharpe ratio and maximum drawdown could be studied in the future.

VII. APPENDIX

A. The parameter values of moving average

nl (number of days in a long-period moving average) = 5, 10, 15, 20, 25, 30, 40, 50, 75, 100, 125, 150, 200, 250 (14 values);

ns (number of days in a short-period moving average) = 1, 2, 5, 10, 15, 20, 25, 30, 40, 50, 75, 100, 125, 150, 200 (15 values).

Because ns should be less than nl, the total number of MA rules generated is 119.

B. The parameter values of trading range break-out

n (number of days for a trading range) = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 75, 80, 90, 100, 125, 150, 175, 200, 250 (21 values).

Because there is only one parameter, the total number of TRB rules generated is 21.

REFERENCES

[1] E. Fama and M. Blume, "Filter rules and stock-market trading," *The Journal of Business*, vol. 39, no. 1, pp. 226–241, 1966.
[2] W. Brock, J. Lakonishok, and B. LeBaron, "Simple technical trading rules and the stochastic properties of stock returns," *Journal of Finance*, pp. 1731–1764, 1992.
[3] R. Gencay, "The predictability of security returns with simple technical trading rules," *Journal of Empirical Finance*, vol. 5, no. 4, pp. 347–359, 1998.
[4] F. Allen and R. Karjalainen, "Using genetic algorithms to find technical trading rules," *Journal of Financial Economics*, vol. 51, pp. 245–271, 1999.
[5] L. Kestner, *Quantitative trading strategies: harnessing the power of quantitative techniques to create a winning trading program.* McGraw-Hill Professional, 2003.
[6] M. Pring, *Technical analysis explained: The successful investor's guide to spotting investment trends and turning points.* McGraw-Hill, 1991.
[7] P. Hsu and C. Kuan, "Reexamining the profitability of technical analysis with data snooping checks," *Journal of Financial Econometrics*, vol. 3, no. 4, pp. 606–628, 2005.
[8] H. Subramanian, S. Ramamoorthy, P. Stone, and B. Kuipers, "Designing safe, profitable automated stock trading agents using evolutionary algorithms," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation.* ACM, 2006, pp. 1777–1784.
[9] A. Briza and P. Naval Jr, "Stock trading system based on the multi-objective particle swarm optimization of technical indicators on end-of-day market data," *Applied Soft Computing*, vol. 11, no. 1, pp. 1191–1201, 2011.
[10] R. Sullivan, A. Timmermann, and H. White, "Data-snooping, technical trading rule performance, and the bootstrap," *Journal of Finance*, pp. 1647–1691, 1999.
[11] A. Ratnaweera, S. Halgamuge, and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.
[12] J. Nenortaite and R. Simutis, "Stocks' trading system based on the particle swarm optimization algorithm," *Computational Science-ICCS 2004*, pp. 843–850, 2004.
[13] R. Hassan, B. Cohanim, O. De Weck, and G. Venter, "A comparison of particle swarm optimization and the genetic algorithm," in *Proceedings of the 1st AIAA Multidisciplinary Design Optimization Specialist Conference*, 2005.
[14] J. Lee, S. Lee, S. Chang, and B. Ahn, "A comparison of ga and pso for excess return evaluation in stock markets," *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, pp. 45–55, 2005.
[15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ*, vol. 4. IEEE, 1995, pp. 1942–1948.
[16] J. Robinson and Y. Rahmat-Samii, "Particle swarm optimization in electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 52, no. 2, pp. 397–407, 2004.
[17] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of 1998 IEEE International Conference on Evolutionary Computation*, vol. 1. IEEE, 1998, pp. 69–73.
[18] ——, "Empirical study of particle swarm optimization," in *Proceedings of 1999 IEEE International Conference on Evolutionary Computation*, vol. 3. IEEE, 1999, pp. 101–106.