Analysis of Dynamic Properties of Stock Market Trading Experts Optimized with an Evolutionary Algorithm

Krzysztof Michalak[1]

[1] Department of Information Technologies,
Institute of Business Informatics,
Wroclaw University of Economics, Wroclaw, Poland
krzysztof.michalak@ue.wroc.pl

Abstract. This paper concerns optimization of trading experts that are used for generating investment decisions. A population of trading experts is optimized using a dynamic evolutionary algorithm. In the paper a new method is proposed which allows analyzing and visualizing the behaviour of optimized trading experts over a period of time. The application of this method resulted in an observation that during certain intervals of time the behaviour of the optimized trading experts becomes more stable.

Keywords: trading rules, dynamic optimization, usage patterns, trading expert optimization, trading expert stability

1 Introduction

Softcomputing methods are commonly used in the context of financial problems. Neural networks are used for time series prediction [8] (sometimes combined with population-based methods [3]) and also for the detection of rare events [12, 15]. Evolutionary methods perform well on problems like portfolio optimization [2, 10, 13] and trading strategies optimization [1, 6]. Optimization of trading experts based on stock market trading rules is one of the approaches to developing trading strategies using computational methods [4, 14]. Apart from trading expert optimization one of the interesting topics is the analysis of trading rules interactions and usage patterns [9, 11].

In this paper dynamic evolutionary optimization approach is used for trading experts optimization. The variability of the experts optimized by the evolutionary algorithm over time is then analyzed and a visualization method is proposed. Results presented in this paper suggest that there exist time periods in which there is little change in the optimal set of trading rules.

2 Dynamic Optimization of Trading Experts

Algorithmic trading requires making decisions about buying and selling financial instruments based on information obtained from the market. Trading experts analyzed in this paper generate "buy" and "sell" signals at consecutive time instants for each stock separately based on technical analysis indicators, stock price, volume, etc. The working of these experts is based on a set of trading rules which generate individual "buy" and "sell" signals using various relationships between the observed values. An example of a trading rule based on two moving averages is presented in Algorithm 1. The "buy" signal is represented by a numeric value "1" and the "sell" signal is represented by a numeric value "-1". A rule can also output "0" which can be interpreted as "no decision". These numeric values are used for calculating composite signals which are based on averages of values returned by many trading rules.

Algorithm 1 An example of a trading rule based on two moving averages.

```
IN:
    $\tau_{fast} = 10$ - the period of the fast moving average
    $\tau_{slow} = 80$ - the period of the slow moving average
    $t$ - the time instant for which to generate the decision
OUT:
    A decision for the time instant $t$

if $MA_{\tau_{fast}}(t) < MA_{\tau_{slow}}(t)$ then
  return -1          // Sell
else
  if $MA_{\tau_{fast}}(t) > MA_{\tau_{slow}}(t)$ then
    return 1         // Buy
  else
    return 0         // No suggestion
  end if
end if
```

In this paper the following structure of a trading expert is used:

$$b_1, \ldots, b_{N_{rules}}, s_1, \ldots, s_{N_{rules}}, \varTheta_{buy}, \varTheta_{sell} \tag{1}$$

where:

N_{rules} - the number of trading rules,

b_i - a binary variable that determines if the i-th rule is used for generating "buy" signals,

s_i - a binary variable that determines if the i-th rule is used for generating "sell" signals,

$\varTheta_{buy}$, $\varTheta_{sell}$ - decision thresholds for "buy" and "sell" decisions respectively.

As presented in Equation (1), trading experts used in this paper turn individual rules on and off separately for generation of "buy" and "sell" signals. Also, the $\varTheta_{buy}$ and $\varTheta_{sell}$ decision thresholds are adjusted separately. The individual trading rules generate their own "buy" and "sell" signals represented by numbers −1, 0 or 1. The composite "buy" signal y_{buy} is calculated as an average from those rules for which $b_i = 1$:

$$y_{buy} = \frac{\sum_{i=1}^{N_{rules}} b_i y_i}{N_{rules}} \quad , \tag{2}$$

where:
y_i - the signal generated by the i-th rule.

The composite "sell" signal is calculated likewise. Buy and sell transactions are made when the respective signal exceeds a given threshold ($y_{buy} > \Theta_{buy}$ or $y_{sell} > \Theta_{sell}$). The parameters of an expert can be directly used as a chromosome in an evolutionary algorithm.

The situation in the economy and on the market changes continually. Thus, one can expect that the applicability of various trading rules may change over time. Therefore, the optimization of trading experts can be treated as a dynamic optimization problem. In this paper we assume that changes in the environment occur at certain time instants and the algorithm can evolve for a preset number of generations N_{gen} after each change. This scenario is depicted in Figure 1. Because the situation on the market changes quickly only a limited number of generations can be allowed within each time interval.

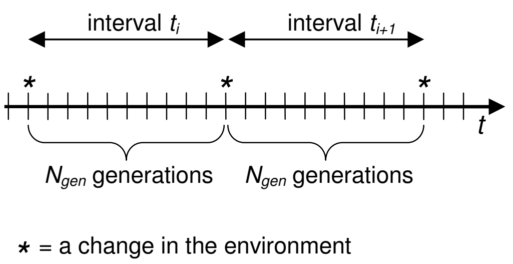

Fig. 1. An overview of dynamic optimization scenario. N_{gen} generations of the evolutionary algorithm are executed between every two changes in the environment.

A well-known fact concerning the dynamic optimization is that in an evolutionary algorithm the population may converge to an optimum for a given time instant which decreases variability in the population and makes it very hard to adapt to new conditions in the time instants that follow. One of the methods of preventing such loss of diversity is the addition of random immigrants [7]. In this paper three methods of introducing random immigrants to the population are presented.

Reinit - the population is initialized randomly for each new interval. This is equivalent to a static optimization - optimizing the trading experts separately for each interval without using any prior knowledge gathered from previous intervals.

Every Interval - random immigrants are added to the population when the optimization starts to handle a new time interval. The number of random immigrants was set to be equal to the population size.

Every Generation - random immigrants are added in the same way as in the **Every Interval** approach, plus some random immigrants are added at the beginning of each generation. The number of immigrants arriving at the beginning of each generation was set to 20% of the population size.

In all the proposed approaches a single-objective evolutionary algorithm with roulette-wheel selection is used. The genetic operators are bit-flip mutation with probability 0.02 for b_i and s_i, polynomial mutation [5] with distribution index 20 for Θ_{buy} and Θ_{sell} and a single-point crossover with crossover probability 0.9. An overview of the evolutionary algorithm used in this paper is given in Algorithm 2.

The $Evaluate(P, t_k)$ procedure evaluates all the specimens in a given population P on data from time interval t_k. The fitness of a specimen that represents trading expert parameters (cf. Equation (1)) is the return obtained when investing during the time interval t_k using the trading signals calculated according to Equation (2). The b_i, s_i, Θ_{buy} and Θ_{sell} parameters are taken from the specimen for which the fitness is calculated.

The optimization of trading experts described in this paper was performed using minute quotations of 50 stocks and ETF shares: AA, AAPL, AIG, ALU, AMD, ANR, BAC, BSX, C, CHK, CSCO, DELL, EEM, EFA, EMC, EWJ, EWZ, F, FAZ, FCX, FXI, GE, GLW, HPQ, INTC, IWM, JPM, MS, MSFT, MU, NOK, NWSA, ORCL, PBR, PFE, QQQ, RF, S, SDS, SIRI, SPY, T, TZA, VALE, VWO, VXX, WFC, XLF, XLI and YHOO. The range of available data contains quotation from the period from 2011.10.17 to 2013.05.20. The above-mentioned companies were selected as 50 companies with the largest total volume of transactions in the available data range.

The dynamic optimization was performed for a number of time intervals t_k, $k \in \{1, \ldots, N_{time}\}$. Each time interval t_k corresponded to an 8-week period starting at the week number k. Therefore, the population was first optimized with respect to the performance on an interval $week_1, \ldots, week_8$, then $week_2, \ldots, week_9$, and so on. The number of time intervals was set to $N_{time} = 76$ with the first day of $week_1$ on 2011.10.17 and the last day of $week_{76}$ on 2013.05.20. Specimen fitness for each time instant t_k was calculated as the overall return obtained by using the trading expert encoded in the specimen over the interval $week_k, \ldots, week_{k+7}$. In this paper intra-day trading was assumed: all the remaining stocks were sold at the end of the day. Commission was set to 0.4% per transaction. In every method a population of 50 specimens was evolved for 30 generations for each time interval t_k.

3 Analysis of Trading Rules Usage

This section analyzes the usage of trading rules in experts optimized using the evolutionary algorithm described in the previous section. In dynamic optimiza-

Algorithm 2 An overview of the evolutionary algorithm used for optimization of trading experts.

```
IN:
    variant - a method of introducing random immigrants to the population
    $N_{pop}$ - number of specimens in the population
    $N_{gen}$ - number of generations

// — In the "reinit" variant the population is initialized —
// — for every interval separately, not only at the beginning —
if variant $\neq$ "reinit" then
  $P$ = InitPopulation($N_{pop}$)
end if

// — Time interval —
for $k = 1 \to N_{time}$ do
  if variant = "reinit" then
    // — Initialize new population —
    $P$ = InitPopulation($N_{pop}$)
  else
    // — Random immigrants —
    $R$ = InitPopulation($N_{pop}$)
    $P = P \cup R$
  end if
  Evaluate($P$, $t_k$)

  // — Generation —
  for $g = 1 \to N_{gen}$ do
    if variant = "every generation" then
      // — Random immigrants —
      $R$ = InitPopulation($0.2 * N_{pop}$)
      $P = P \cup R$
    end if

    Evaluate($P$, $t_k$)
    $P$ = Select($P$, $N_{pop}$)
    Crossover($P$)
    Mutation($P$)
  end for
end for
```

tion context an interesting question is how often and in what way does the usage of individual trading rules change over time.

To address this question the following method is proposed. For each time interval t_k the population P_k optimized to give the highest return on the interval $week_k, \ldots, week_{k+7}$ is processed. A fraction $P_k^{(best)}$ of $q^{(best)}\%$ best specimens from the population P_k is extracted. For visualization of the usage of trading rules for generating the "buy" signals a matrix $A_{N_{time} \times N_{rules}}$ is calculated in which the k-th row contains average values of parameters b_i found in $P_k^{(best)}$ for $k = 1, \ldots, N_{time}$. For visualization of the usage of trading rules for generating the "sell" signals a similar, but separate matrix is built based on s_i parameters from the same fraction $P_k^{(best)}$ of the population P_k.

Visualization of trading rules usage

For visualization the columns of matrix A are clustered using an agglomerative hierarchical algorithm. In this clustering algorithm the center of a cluster is defined as the average value of those columns of the matrix A that belong to this cluster. The hierarchical clustering algorithm builds a tree structure in which nodes contain nested subsets of columns of matrix A. Each tree node T contains a list of columns $T.C$ and two references $T.left$ and $T.right$ to child nodes. The child nodes represent subsets that were clustered together to form the cluster $T.C$. The structure built by the clustering algorithm is shown in Figure 2.

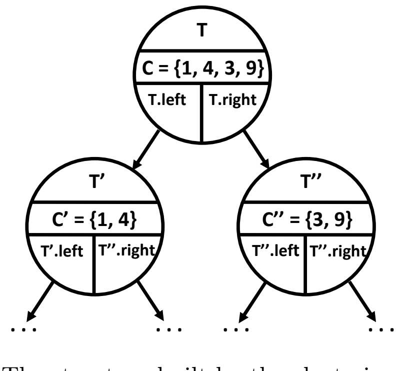

Fig. 2. The structure built by the clustering algorithm.

After clustering the *SortNode* procedure is used to change the ordering of columns in all clusters in such a way that there is minimal difference between adjacent columns from neighbouring clusters. The *SortNode* procedure works as follows when called for a tree node T. First, if the $T.left$ and $T.right$ subtrees are non-empty, the procedure is called recursively: *SortNode*$(T.left)$, *SortNode*$(T.right)$. Then, depending of which distance between columns is the smallest one, the *SortNode* procedure reorders columns in child nodes. Using notation shown in Figure 2 we have the $C' = T.left.C$ and $C'' = T.right.C$.

The reordering of columns in child nodes is done according to which of the conditions is found to be true when comparing distances between the first and the last columns of C' and C'':

– if $d(C'.first, C''.first)$ is the smallest, C' is reversed,
– if $d(C'.last, C''.last)$ is the smallest, C'' is reversed,
– if $d(C'.first, C''.last)$ is the smallest, both C' and C'' are reversed,
– if $d(C'.last, C''.first)$ is the smallest, no reordering is performed,

The procedure of construction and clustering of the matrix A is performed separately for parameters b_i and s_i. The clustered matrix can be visualized as a checkerboard plot. Examples of checkerboard plots for the AAPL stock are presented in Figure 3. The plots are based on $q^{(best)} = 20\%$ of populations after 30 generations of the dynamic optimization evolutionary algorithm.

What can be easily seen from the clustered checkerboard plots is that there are many rules that are activated (black areas) or deactivated (white areas) together. Also, it can be observed that there are prolonged periods when no change to the best trading experts is introduced.

From the visual exploration two questions arise. First, how often are the rules used for generation of both "buy" and "sell" signals? Second, what is the stability of the best rule sets generated by the evolutionary algorithm?

The usage of the trading rules for generation of "buy" and "sell" signals

As shown in Equation (1) each trading expert contains separate variables for enabling the "buy" signals from individual rules (the b_i variables) and separate ones for the "sell" signals (the s_i variables). In order to get some insight on how the rules are used (for generating "buy" signals, "sell" signals, both or none) the percentage of specimens in which each of the four situations occurred for each of the rules $i = 1, \ldots, N_{rules}$ was calculated. The observed percentages calculated from 380000 specimens (50 stocks, 10 best specimens ($q^{(best)} = 20\%$), 76 intervals, 10 iterations) are presented in Tables 1-3.

Table 1. The percentages of specimens in which rules were used for generating "buy" signals, "sell" signals, both or none in the populations optimized using the "Every Generation" method.

Usage	Mean	Std. dev.
Not used	0.24387	0.01316
Buy	0.25827	0.013
Sell	0.24171	0.013111
Both	0.25614	0.013346

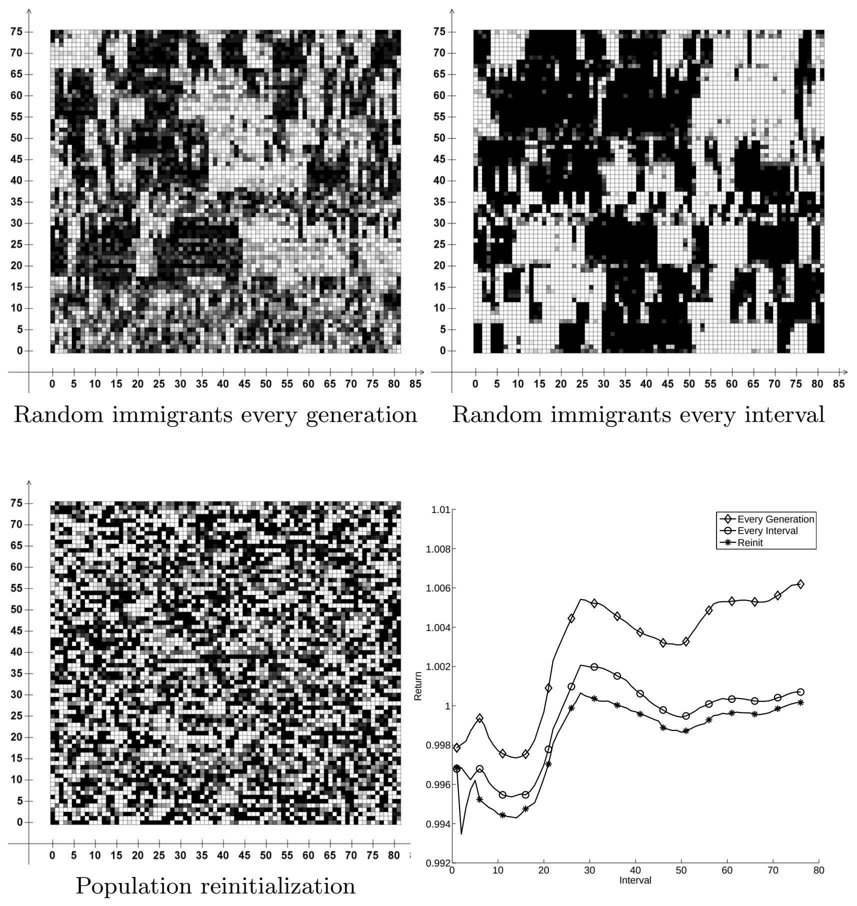

Fig. 3. Visualization of the usage of trading rules for generating "buy" signals for the AAPL stock. Black represents used rules, white represents unused rules.

Table 2. The percentages of specimens in which rules were used for generating "buy" signals, "sell" signals, both or none in the populations optimized using the "Every Interval" method.

Usage	Mean	Std. dev.
Not used	0.24205	0.026379
Buy	0.26194	0.026503
Sell	0.2378	0.02493
Both	0.25821	0.026473

Table 3. The percentages of specimens in which rules were used for generating "buy" signals, "sell" signals, both or none in the populations optimized using the "Reinit" method.

Usage	Mean	Std. dev.
Not used	0.24395	0.0091361
Buy	0.25763	0.0099813
Sell	0.24265	0.0096086
Both	0.25577	0.0090758

The results shown in Tables 1-3 motivate using separate variables b_i and s_i for determining whether to use individual rules for generating "buy" and "sell" signals. In about 25% of specimens the rules are used for generating both the "buy" and "sell" signals, but equally often the trading expert is optimized towards using distinct rules for generating each of the signals.

The stability of the results produced by the optimal rule sets

Based on the visual exploration of the plots representing the usage of the trading rules with respect to time it can be observed that there exist certain periods of time in which the rules used by the best specimens remain the same.

Figures 4 and 5 present the average return (calculated over 10 iterations) obtained by the best specimens in the population after 30 generations of the evolution. As can be seen in the graphs, the best results are obtained when random immigrants are added to every generation of the evolutionary algorithm. On the other hand, totally reinitializing the entire population produces the worst results. This suggests that while high diversity is required for the algorithm to be able to track changing optimal rule set, some useful information can be obtained from previous time intervals. This information is lost when the population is reinitialized and the results deteriorate.

Figure 6 presents the dependence between the return obtained in time intervals t_k (x axis) and t_{k+1} (y axis), where $k = 1, \ldots, N_{time} - 1$ by specimens that had not changed for at least 3 intervals. Clearly, adding immigrants to every generation causes the optimized rule set to produce similar returns repetitively over several time intervals. In most cases the returns obtained in time intervals t_k and t_{k+1} are very similar (most of the points are placed near the $y = x$ line). Conversely, when immigrants are only added at the beginning of every interval there is much more difference between the return obtained in interval t_k and interval t_{k+1}. This suggests that when using the Every Interval method the algorithm does not adapt too well and the optimal rule set may remain unchanged even though the obtained returns vary significantly. Note, that in the case of immigrants added to every generation the diversity of the population may be expected to increase, but the returns given by optimized rule sets are less diverse. This can be indicative of a fact, that in the case of immigrants added to every generation the algorithm is not only able to find good solutions more easily than in the case of immigrants added once every interval only, but also that the

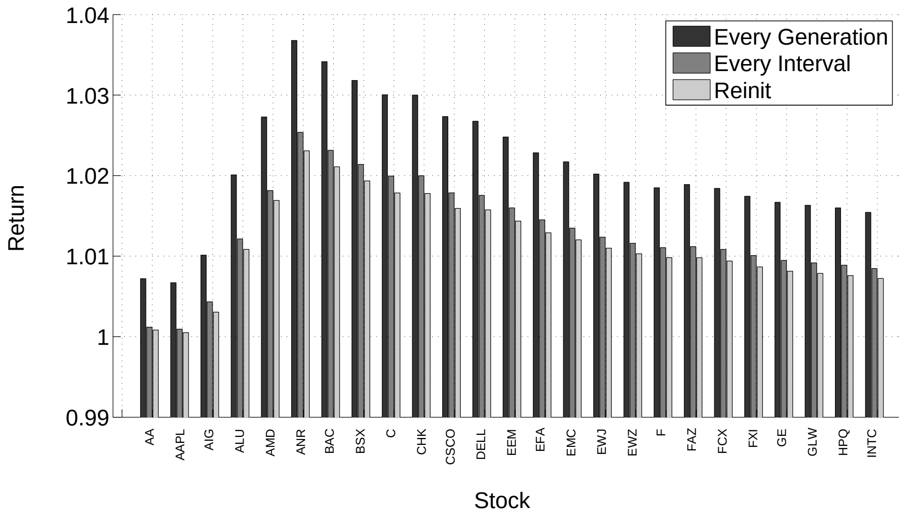

Fig. 4. The average return achieved by the best specimens after 30 generations (stocks AA to INTC).

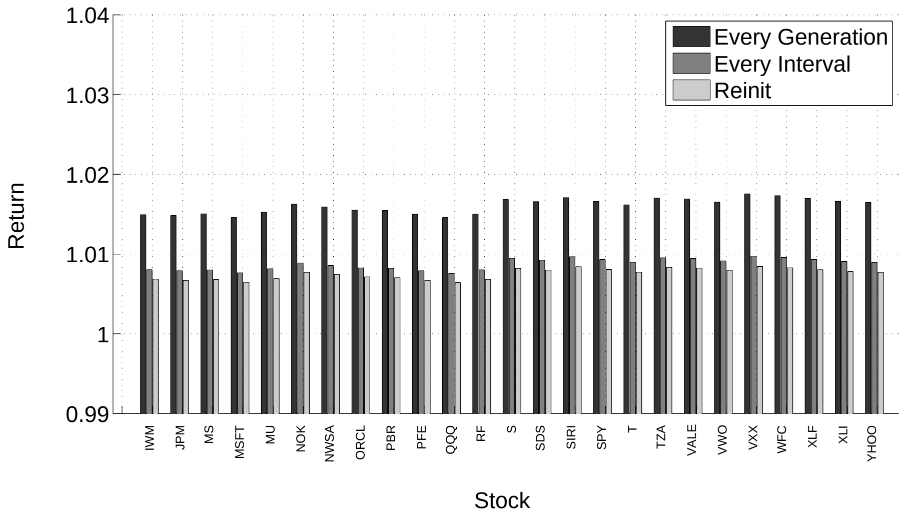

Fig. 5. The average return achieved by the best specimens after 30 generations (stocks IWM to YHOO).

optimized rule sets seem to be more universal i.e. they can produce good results for several consecutive time intervals.

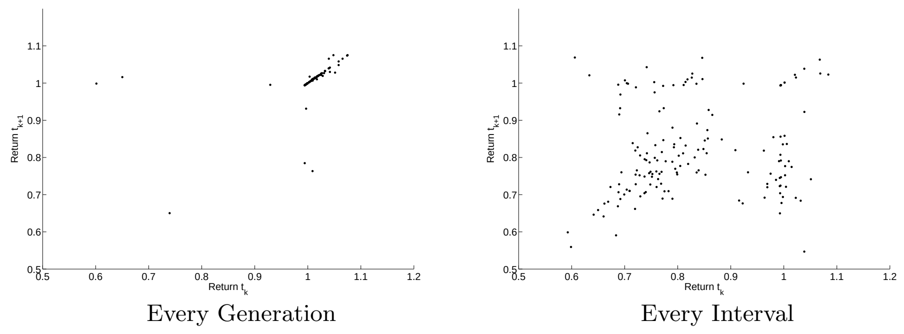

Fig. 6. The dependence between the return obtained in interval t_k (x axis) and interval t_{k+1} (y axis). The specimens that had not changed for at least 3 intervals were used.

4 Conclusion

In this paper dynamic properties of trading experts optimized using a dynamic evolutionary algorithm were analyzed. A new method of analyzing and visualizing the behaviour of trading experts over a period of time was proposed. The application of this method to trading experts generating decisions for 50 stocks and ETF shares resulted in several observations:

- Introducing additional random immigrants at the beginning of every generation allows the evolutionary algorithm to produce specimens with higher fitness values compared to the algorithm where random immigrants are added only once for every interval at the beginning of the evolution.
- On the other hand, reinitializing the entire population for each new time interval deteriorates the results significantly. This means that despite the large variability of trading rules usage, useful information can be extracted from trading experts optimized for past time intervals.
- Visualization of trading rules usage shows prolonged periods of time when the same set of rules produces the best investment returns.
- The evolutionary algorithm can produce good rule sets that are relatively stable - they produce similar return values for consecutive time intervals.
- Among the fittest specimens the fraction of rules used at the same time for generating "buy" and "sell" signals is approximately 1/4. This result suggests, that in order to build good trading experts one should allow using trading rules separately for generating "buy" and "sell" signals.

Further work may include development of methods that allow early detection of time intervals during which the optimal set of rules remains the same or very similar. It is worth investigating if the analysis of the stability of optimal trading rule sets could be useful for deciding if a given trading expert should be used for making decision at a given time or not.

References

1. Bauer, R.: Genetic Algorithms and Investment Strategies. Wiley, Chichester (1994)
2. Best, M. J.: Portfolio Optimization. Chapman&Hall/CRC (2010)
3. Cai, X., Zhang, N., Venayagamoorthya, G. K., Wunsch II, D. C.: Time series prediction with recurrent neural networks trained by a hybrid PSOEA algorithm. Neurocomputing, vol. 70, no. 13-15, pp. 2342-2353, Elsevier (2007)
4. Chiam, S. C., Tan, K. C., Al Mamun, A.: Investigating technical trading strategy via an multi-objective evolutionary platform. Expert Systems with Applications, vol. 36, no. 7, pp. 10408-10423, Elsevier (2009)
5. Deb, K., Goyal, M.: A Combined Genetic Adaptive Search (GeneAS) for Engineering Design. Computer Science and Informatics, vol. 26, pp. 30–45, (1996)
6. Dempster, M., Jones, C.: A Real-Time Adaptive Trading System using Genetic Programming. Quantitative Finance 1, pp. 397–413, (2001)
7. Grefenstette, J.J.: Genetic algorithms for changing environments. In: Manner, R., Manderick, B. (eds.) Parallel Problem Solving from Nature, pp. 137-144, Elsevier Science Publisher (1992)
8. Inoussa, G., Peng, H., Wu, J.: Nonlinear time series modeling and prediction using functional weights wavelet neural network-based state-dependent AR model. Neurocomputing, vol. 86, no. 1, pp. 59–74, Elsevier (2012)
9. Lipinski, P.: Dependency Mining in Large Sets of Stock Market Trading Rules. In: Enhanced Methods in Computer Security, Biometric and Intelligent Systems, ed. J. Pejas, A. Piegat, pp. 329–336, Kluwer Academic Publishers (2005)
10. Michalak, K., Filipiak, P., Lipinski, P.: Evolutionary Approach to Multiobjective Optimization of Portfolios That Reflect the Behaviour of Investment Funds. In: Artificial Intelligence: Methodology, Systems, and Applications - 15th International Conference, AIMSA 2012, Varna, Bulgaria, LNCS, vol. 7557, pp. 202-211, Springer (2012)
11. Michalak, K., Filipiak, P., Lipinski, P.: Usage Patterns of Trading Rules in Stock Market Trading Strategies Optimized with Evolutionary Methods. In: 16th European Conference, EvoApplications 2013, Vienna, Austria, April 3-5, 2013, Lecture Notes in Computer Science, vol. 7835, pp. 234–243, Springer (2013)
12. Michalak, K., Lipinski, P.: Prediction of high increases in stock prices using neural networks. Neural Network World, 15, vol. 4, pp. 359-366, Springer (2005)
13. Radziukyniene, I., Zilinskas, A.: Evolutionary Methods for Multi-Objective Portfolio Optimization. In: Proceedings of the World Congress on Engineering 2008, pp. 1155–1159, Newswood Limited (2008)
14. Wang, F., Yu, P. L. H., Cheung, D. W.: Combining technical trading rules using particle swarm optimization. Expert Systems with Applications (in press), Available online 24 October 2013, http://dx.doi.org/10.1016/j.eswa.2013.10.032.
15. Yu, L., Wang, S., Lai, K. K., Wen, F.: A multiscale neural network learning paradigm for financial crisis forecasting. Neurocomputing, vol. 73, no. 4-6, pp. 716–725, Elsevier (2010)