Evolutionary algorithm in Forex trade strategy generation

Conference Paper · November 2010

DOI: 10.1109/IMCSIT.2010.5679921 · Source: IEEE Xplore

2 authors, including:

Paweł B. Myszkowski
Wroclaw University of Science and Technology
31 PUBLICATIONS **56** CITATIONS

SEE PROFILE

Proceedings of the International Multiconference on Computer Science and Information Technology pp. 81–88
ISBN 978-83-60810-27-9
ISSN 1896-7094

Evolutionary Algorithm in Forex trade strategy generation

Paweł B. Myszkowski
Wrocław University of Technology,
Wyb. Wyspiańskiego 27, 51-370 Wrocław, Poland
Email: pawel.myszkowski@pwr.wroc.pl

Adam Bicz
Wrocław University of Technology,
Wyb. Wyspiańskiego 27, 51-370 Wrocław, Poland
Email: cyklop@gmail.com

***Abstract*—This paper shows an evolutionary algorithm application to generate profitable strategies to trade futures contracts on foreign exchange market (Forex). Strategy model in approach is based on two decision trees, responsible for taking the decisions of opening long or short positions on Euro/US Dollar currency pair. Trees take into consideration only technical analysis indicators, which are connected by logic operators to identify border values of these indicators for taking profitable decision(s). We have tested the efficiency of presented approach on learning and test time-frames of various characteristics.**

***Keywords:* financial data mining, evolutionary algorithm, genetic programming, decision tree induction, trade strategy, Forex**

I. Introduction

IN THE literature we can find many approaches to the automatic trading and many attempts to use artificial intelligence methods to trade on stock markets. A good summation on large part of the work already done in this matter is [2], where the list of surveyed markets, potential input data and modeling methods were shown. On financial markets automatic trade systems are widely used (especially by great investment funds), but the rules of taking the decisions are not publicly known. We don't know how far they are human controlled, and how far autonomic they are in their decisions. There is a lot of artificial neural networks (ANN), evolutionary algorithms (EA) or genetic programming (GP) usages as appropriate methods to search rules and strategies that could use ineffectiveness of the markets to earn money. How far they are well developing on current, not historical data, is an open question.

We have decided to investigate the usability of EA to generate rules, which will trade on foreign exchange market (Forex). Forex is a very interesting market because of its liquidity and open hours. It's closed only on weekends, normally operating 24 hours per day [7]. It means that (sometimes huge) price gaps between trading days, what is a normal behavior in stock markets, are here rarely observed. Thanks to high liquidity we can always sell and buy any amounts of assets at actual price without the risk that we will strongly affect the price or run out of contrary orders (thus being unable to sell or buy at reasonable prices), at least if we are individual investors and not great investment funds. Liquidity means constant movements, which means constant opportunities to make profitable trades. Thanks to high leverage and very small transaction costs it's possible to generate high profits even on very small price movements (such as 0.1% changes). Unfortunately, the leverage means the risk of taking high losses or even losing whole invested capital.

A. Related works

In the literature there are many interesting approaches, which take into consideration an automating profitable trade strategy generation.

In paper [16] is described GP to build decision trees. There are two types of decision trees, based on moving averages values and filters – decision is being taken after stock value rising up over last maximum or falling down below last minimum for specified percent value. Decision trees evaluated by GP can use arithmetical operations, functions, logical operators (incl. negation), conditional operators (IF-THEN), numerical and logical constants. However, its search space is being very big. Also in [16] daily intervals are used, where we used 10 minutes intervals. It means that the transactions have a much longer investment time horizon. Such approach has successfully generated many profitable strategies on various currency pairs, at limited risk. The risk value is calculated with beta indicator[1], which was compared with its value for some benchmark portfolios based on various stock market indicators. What's interesting, rules generated for one currency pair were much worse, when used on others.

GP can be used also for trade strategies generation on Warsaw Stock Exchange were investigated [15]. Based on the decision trees stored in GP individual a portfolio up to 10 stocks is build. The content of the portfolio vary in time. After doing several experiments authors stated that the profits earned by automatically generated strategies exceed those generated by benchmark (WIG stock indicator as buy&hold strategy profit). On decreasing time-frames no tremendous profits were generated, but substantial losses were avoided too and according to decreasing character of used time-frame such results are acceptable.

In work [13] EA is used to optimize values for predefined filters. Authors optimize and test filters on data from Australian Stock Exchange. They find out that EA finds profitable strategies more effectively than greedy algorithm. This conclusion is important if we want to use this solution for real time usage.

Paper [18] shows that there is no direct correlation between the distance from the learning to testing time-frame

[1] measures the correlation between an investment's value and movements in the overall market

and the efficiency of gained strategy. There is another interesting conclusion that no positive results of removing strategies, which behave worse on some validating time-frames, where found. It means that if we will remove strategies, because they are not as good as the others on one validating frame, we probably can lose one of the best strategies for another time-frame.

In work [9] EA usefulness to generation set of rules is investigated, which decides about the moments to buy or sell stocks on the Paris Stock Exchange (France). The individual there is represented not as a tree, but a sequence of bits, where each bit responses for usage of technical analysis indicators. Such representation allows EA to search the solution space very effectively. As there were shown, strategies generated in such way give better results than benchmark buy&hold strategy. What's important, there wasn't found any proof that some stocks are easier in prediction. Also, that constant generating of new rules on current data can improve the results, which is contrary to the conclusion from [18]. Generated rules were profitable only on those stocks on which they were learned, and there weren't found any sign of some indicators being preferred over others.

Another EA usage were shown in [1], where task was to find technical trading rules on Standard and Poor's composite stock index. Rules generated by EA have form of decision trees, where the nodes can be logical operators, functions (avg/max/min), arithmetic operations, comparisons or constants. Training data were given from years 1963-69, and test data from the years 1970-1989 and other periods since 1929. The rules generated excess returns only on the first test period. On periods before 1963, including the transaction costs, they generated losses.

In [12] method searches for similarities in the history of stock value movement to predict its values. EA is used to find as good as possible analogies between some latest intervals and those taken from history of given stock. Such historical data are preprocessed, where outliers are removed to make the predictions more accurate. There was proven that filtering outliers and replacing them with moving average or just average values from the nearest neighbors gives positive results. Prediction results of volume values are much worse, mostly because of some huge random fluctuations.

The stock trading predictions system was build in [17] . As predictions there were used ANN, dynamic time-frames and case based reasoning technique. The ANN was used to predict buy/sell points for interesting stocks, previously chosen by some pre-defined rules, e.g. monthly sales, daily trading volumes, 5 day average volumes. Such method generated significant revenues on test data, even for stocks in downtrend.

In [20] there is another interesting ANN usage, where the task is to predict "next" values of Warsaw Index Futures. Technical analysis techniques – such as channel breakout, Bollinger Bands, Momentum Oscillators, Moving Averages – are used in combination of ANN predictions and without them, as competitive strategies. Technical analysis usage with ANN forecast makes possible to generate profits even for strategies, which normally were not profitable at all.

In [10] consists in Evolution Strategies usage to portfolio optimization problem. The main goal was defined to minimize the risk at fixed level of the expected return. Risk was measured as a semi-variance of returns (similar to variance, but emphasizing the "downside risk" - *actual return - expected return* but only if it's smaller than 0). Artificial "experts" were built by the EA, using some of predefined technical analysis rules (chosen by the EA algorithm). The portfolio of stocks was chosen at the beginning, and artificial "experts" were able to reduce or increase the share of each stock in the portfolio. There is stated, that the results were much better than random portfolio shares (buy&hold strategy), but worse than market index. Worth noticing, that buy&hold is here understood as keeping initial (random) portfolio shares unchanged, and market index is market value weighted (bigger companies has bigger share in the index). They stated that additional research is needed.

Work [8] uses a hybrid approach based on various types of ANN and EA to detect interesting patterns in stock market, where data came from the daily Korea Stock Price index 200. EA were applied to optimization of time delays and network architectural factors. The various types of ANN were connected into a multilayer feed-forward network, where time series were the input. Also there were tested and compared time delay ANN, adaptive time delay ANN, recurrent ANN and stated that their integrated approach gives better results than standard ANN.

Another interesting approach was used in [19]. A data mart was created and data mining techniques applied to predict future stock prices using historical values. A traditional grey prediction[2] model was expanded by fuzzy logic model. Data for each stock were pre-classified in groups using the "time" attribute. On the test data, the average deviation from the predicted highest stock price and the actual price was less than 9%, which authors interpreted as pretty accurate, as "predictions are feasible".

Another interesting approach presents [4], where Genetic Network Programming (GNP) was used for building stock trading model. Build network consists of judgment and processing nodes. Judgment nodes use technical analysis indicators and candlestick chart formations to decide which next node should be processed, and processing nodes take actions such as buying/selling stocks. Training and test data were taken from Tokyo Stock Exchange (Japan). As results are very promising, is suggested that it was achieved due to the representation of solutions as graph structures, which allows memorizing past actions sequences more effectively.

In our work we generate trade strategies as BUY/SELL decision trees, which are responsible for opening long or short positions on the EUR/USD currency pair on Forex market.

This paper is organized as follows. Section II describes details of proposed EA based approach: representation schema, genetic operators, selection method and the form of the fitness function. Research to find as good as possible EA parameters is shown in section III. Experiments are present-

[2] grey system theory requires limited amount of data to estimate the behavior of unknown systems [5]

ed in section IV, where are defined used dataset, research methodology and gained results. Section V shows possible further research and issues that have been skipped in this work. Section VI consists of conclusions and short summary of this work.

II. Proposed Approach

Our approach is based on GP [11] and is slightly inspired by [15]. In our work EA generates trade strategy based on BUY/SELL decision trees that consists in technical analysis indicators that are connected by logical operators. We decided to work with EUR/USD currency pair, because it's the most popular and liquid pair as it generates 27% of the total volume on Forex [3].

A. Individual representation

Each individual is represented as two decision trees, where one is responsible for BUY signals, and one for SELL signals recognition. Each of the decision trees is actually a trading rule. It consists of logical functions binding leafs, which are technical analysis indicators.

Used technical indicators are some variants of Simple Moving Averages (SMA), Weighted Moving Averages (WMA), Displaced Moving Averages (DMA), Relative Strength Index (RSI), Moving Average Convergence-Divergence (MACD), Rate Of Change indicators (ROC), volume value and volatility for specified period values. They are widely described in [14] as very useful tools for the investors to determine trends, reversal points, to identify interesting buy/sell points. They are very popular among investors of any kind, but can be used in sometimes very complex ways, where the same indicator value can be interpreted by various investors in a different way.

In decision tree each node can be logical (and/or) or terminal node in form *(Indicator [<, >] Value)*. Example tree can be presented as follows:

```
((ISMINOF<11) or ((ROC (5)<0.234) and (MACD (26,12)<-1.155))) and (VOLUME<833)
```

Example of decision tree is shown in Figure 1.

B. Selection method

We used a combination of tournament and roulette method. We select randomly 5 (value is set experimentally) individuals, whom we rate and sort in descending order. We chose one individual with probabilities 5/15, 4/15, 3/15, 2/15, 1/15. After selection mutation and crossover generates an individual.

C. Genetic operators

In our work mutation and crossover operators are used.

Mutation operator works on one individual and gives some changes in its genome. Mutation is stochastic based, some methods of decision trees:

- leaf mutation, (I) where a random value is chosen from the possible value set for given indicator (60% chance), (II) mutation changes inequality sign from ">" to "<" and in another direction (16% probability) and last one (III) changes the indicator type (24% chance)
- mutation for logical nodes, that changes "and" into "or" and in another direction (50% chance)
- mutation of tree size: (I) that reduces of given tree to left or right child (50% chance), (II) split given leaf into two leaves connected with „or" or „and" node. The second node is generated randomly (40% chance for each child, only if not at maximal height) and (III) switches left and right child (10% chance)

Crossover operator links two individuals and switches their buy/sell decision trees. The probability usage of crossover was experimentally set to 3%.

D. Fitness function

We have chosen a fitness function that is closely related to the real use of strategies in the stock market, which is the possibility of such strategy to generate gains. It must include transaction costs, because ignoring them can result in over reactive strategies, generating losses if transaction costs are taken into consideration.

```
Fitness =
```

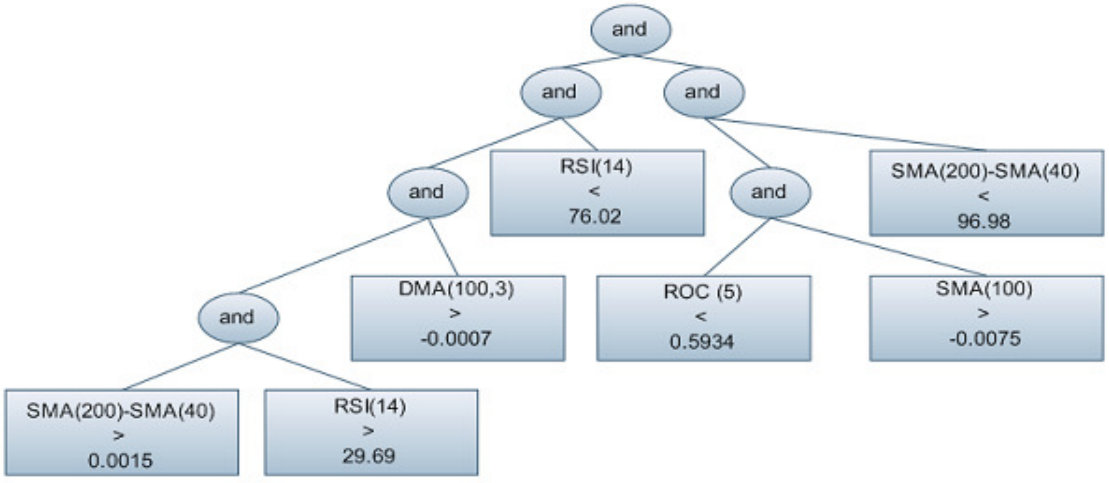

Fig 1: Example of buy decision tree. The presented decision tree as rule: *(*(((SMA(200)-SMA(40)>0.00150) and (RSI(14)>29.69711)) and (DMA(100,3)>-0.00073)) and (RSI(14)<76.02417)) and (((SMA(100)>-0.00757) and (ROC (5)<0.59348)) and (SMA(200)-SMA(40)<96.98830))

```
Sum of all profits/losses
+ Transaction count * Transaction cost
– MAX (0, 4 - Transaction count) * 8
```

Our fitness function form needs stock simulation, playing with the chosen strategy on the learning set data, and taking the result in points (adjusted by transaction costs as provision that equals 4 points). The last component in given function is responsible for faster learning to make more transactions rather than trying to buy/sell and hold. The 4 constant means, that if the strategy has made at least 4 transactions, there will be no punishment. For each transaction less than 4, it gets a punishment of 8 points (which is 2 times greater than transaction costs, which equals 4 points). It means that the punishment for the simplest buy&hold strategy is 24 points as the maximal value.

III. Research on EA parameters

All parameters of EA were tested to find the best values. Searching for EA parameters is a multidimensional problem. Additionally best parameters vary for each of the learning frames, so our results are only some approximation and trade-off between many possibilities. Results were obtained from repeated 200 runs under the same conditions, which generated 200 strategies (1 for each run). Average profit inc. provisions was taken to decide which parameter value is most promising.

Example of such parameter is *maximal tree height.* Some tests (e.g. L1 time-frame, see Table I) have shown, that best results can be obtained with the maximal height equal 5 or 6. Too high trees mean probably too big search space, however too small trees are too simple to represent the complexity of price movements. Maximal height means, that tree can be lower, but never higher as specified height (if mutation could make the tree higher, it will be stopped). Our experiments have shown that the optimal values for transaction provision is 4 points, probability of mutation 72%, and probability of crossover 3%.

In Table II we have shown the averaged profits of strategies in population of *population size* parameter. We can see, that populations greater and smaller than 200 give worse results as the optimal 200, by 300 even much worse.

Strategy profits are measured on test data, not on training data. Stop condition (*generations*) amount in all tests is fixed and equal 150.

IV. Experiments

In our tests we have generated a very well developing strategy (verified on test data), which we will be describe in details in this section.

A. Used datasets

The EA operates on time-frames consisting of about 1400 of 10 minutes intervals, which is about 2 weeks of currency pair historical values [6]. To obtain the fitness of each rule pair (BUY/SELL decision) we virtually play stock with these rules, opening and closing positions on historical data (learning periods in the learning phase, testing periods in testing phase). Example of used datasets (time frames) in learning or validation process (cross-validation was applied for testing):

Falling:
L1 (2009-12-11 10:50, 2009-12-24 15:10) intervals: 1412
L2 (2010-01-12 19:40, 2010-01-26 00:00) intervals: 1412
...
L8 (2007-05-15 19:20, 2007-05-29 00:00) intervals: 1412
Neutral:
N1 (2009-09-24 00:00, 2009-10-07 00:00) intervals: 1386
N2 (2009-11-16 15:50, 2009-11-27 21:50) intervals: 1428
...
N6 (2009-11-02 15:50, 2009-11-13 21:50) intervals: 1428
Rising:
H1 (2009-08-30 23:10, 2009-09-11 00:00) intervals: 1397
H2 (2009-10-07 00:00, 2009-10-20 00:00) intervals: 1386
...
H9 (2009-06-23 04:20, 2009-07-07 00:00) intervals: 1412
Long time-frame:
LONG1 (2008-09-01 15:50, 2008-10-06 02:40) int.: 3593

There were used 24 time-frames for learning and testing process, 3 additional only for testing process. Charts for time-frames of selected L1, L2, L8 and H1 are shown in Fig 2. We can see, that in each of the time-frames a one directional price movement (so-called *trend*) could be noticed,

Table I. Results of testing of 200 strategies produced with various max tree height parameter

Max. tree height	2	3	4	5	6	7	8	9	10
Avg. profit	1028	933	1155,9	**1221,6**	**1290,7**	1073	1094,2	870	985,6
Avg. profit (incl. prov)	599,2	506,1	728,9	**806**	**866,8**	633,4	663	430	541

Table II. Results of experiments with various population size parameter

Population size (individuals)	25	50	100	150	200	250	300
Avg. profit	1000	800	1153	778	**1309**	1060	769
Avg. profit (incl. prov)	590	378	678	338	**878**	614	282

with various short term deviations from the trend line. If the time-frame has more such big-enough short term trends, it's more interesting for our algorithm, because it gives an opportunity to learn when to switch from long to short position. We can see, that time-frame L2 gave very few opportunities to earn more than a simple strategy "sell at the beginning and wait" (sell&hold strategy) (if we consider transaction costs). Time-frame L1 was much more interesting, because there were many rapid movements which resulted in corrective movements.

Simulating FOREX stock strategy can meet three types of situations in the market with long, short position or out of the market. We admitted to keep always 1 position open. The profits are measured in points (or so-called *pips*), which are the fourth number after the point, for example value 1,2548 means 8 pips, instead of percent difference. It is standard measure used in future markets, because each investor can use other leverage and so the percent difference varies, even if they open identical transactions.

B. *Example of strategy*

Example of BUY decision tree is presented on Figure 1. It's a decision tree taken from a strategy which was best behaving on the test data. Transactions of this strategy on H5 dataset are shown in Table III. Such rule is easy to analyze, because for example it expects, that RSI indicator (for 14 intervals) has higher value than 30, and lower than 76, which are similar to values mentioned in the literature [14], 30 and 70. It means that the algorithm try to avoid extreme values (peaks). The important part of the rule is that SMA for 200 days must be at least 15 points higher than the SMA for 40 days, so it buys if the prices are relatively low, but at the same time not extremely falling down (already mentioned RSI > 30). Other parts of the rule are either only small optimizations or are of no meaning for the strategy, at least on the test data. Rule describing the SELL decision tree:

```
( (DMA(100,3)>-0.00082) or ((SMA(200)
>0.00238) or (SMA(200)>0.00349)))
  and (WMA(200)<-0.00067)
```

To understand how the SELL rule works, we can see, that it sells when either the DMA with parameters 100 and 3 (average on 100 intervals, delayed by 3 intervals) is higher than the actual price minus 8 points or the SMA of 200 intervals is higher than price +23 points and in both situations the WMA is lower that actual price minus 6 points. It's hard to judge if this rule is reasonable, because it's not so obvious. The last part using WMA means, that we will sell only if the price is a bit better that the average price in the past (200 intervals) – it ensures, that we will not sell if the price is too cheap.

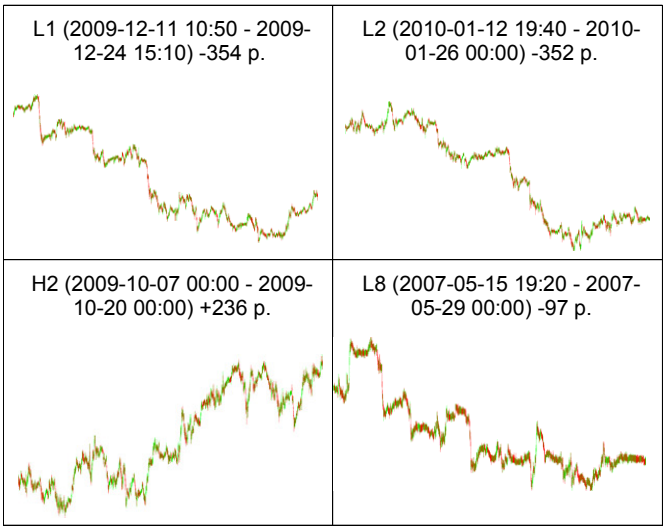

Fig 2. Example charts of the learning sets character. Time frames L1 and L8 are interesting for learning set than L2, which has fewer possibilities for position reversal.

C. *Tests*

In our research test (results are presented in Table IV and Table V) we are running our EA on all learning sets 200 times to get the best strategy for each run. After that we have generated 200 strategies to test their profit on 3 data sets.

To describe its genealogy, strategies learned on a specific time-frame are specified as the name of training time-frame and a name of the test data-set in brackets, for example "L1 (T1)" means strategies learned on time frame *L1* and tested on *T1* data set. Used data sets (*T1*, *T2*, *T3*) of time-frames are specified as follows (in brackets are the results of using *buy&hold* and *sell&hold* strategies, on each time frame from the dataset separately, and then adding the results):

T1: 24 time-frames (*sell&hold* strategy including provisions: 731 points, *buy&hold* strategy: -827 points)

T2: 23 time-frames (*sell&hold* strategy including provisions: -1051 points, *buy&hold* strategy: 959 points)

TABLE III. TABLE OF TRANSACTIONS ON H6 TIME-FRAME OF THE BEST FOUND STRATEGY (PROFITS ARE GIVEN WITHOUT PROVISION COST)

Transaction type	Opened	Closed	Opening price	Closing price	Profit [in points]
LONG	2009-05-28 11:50	2009-06-01 00:00	1.3855	1.4101	**246**
SHORT	2009-06-01 00:10	2009-06-02 07:30	1.4101	1.4114	**-12**
LONG	2009-06-02 07:40	2009-06-03 03:00	1.4121	1.4287	**166**
SHORT	2009-06-03 03:10	2009-06-03 13:30	1.4282	1.4202	**80**
LONG	2009-06-03 13:40	2009-06-04 18:40	1.4174	1.4181	**6**
SHORT	2009-06-04 19:50	2009-06-05 14:40	1.4181	1.4023	**158**
LONG	2009-06-05 14:50	2009-06-10 00:00	1.4006	1.4069	**63**

T3: 24 time-frames (*sell&hold* strategy including provisions: -1227 points, *buy&hold* strategy: 1131 points)

For reasons of clarity: we were testing our strategies on data sets, which consist of 23-24 time-frames. Total amount of used time-frames is 27. Time-frames are independent, short time periods (shown in Fig 2), and data sets *T1*, *T2* and *T3* are collections of time-periods of various kind (rising, falling), but adding the price changes, causes rising (*T2*, *T3*) or falling (*T1*) type. We have used 3 data-sets instead of one, to show how the generated strategies behave on rising and falling data-sets. Using only one data-set does not allow us to do this.

In the traditional stock market we can always compare results of our strategy with a simple *buy&hold* strategy, which means simply buying at the beginning and selling at the end, which is the simplest way to invest. Growth of the market is the expected direction, as we all expect that the economy will rise in a long term.

Differently from the stock market on FOREX we have no real benchmark strategy – *buy&hold* strategy is exactly as good as *sell&hold* (sell at the beginning, buy at the end, such strategy is possible when we consider futures market), because the expected market direction isn't known. Such situation occurs because we always consider pair of currencies, for example EUR/USD, which we could as well switch to USD/EUR to get the ideal opposite of all price movements. That's the reason why we are showing the results of both strategies – *buy&hold* and *sell&hold*. Of course, provision costs are included, which arise when we open a transaction at the beginning of each time frame and close it at the end. The transaction costs are 2 point per transaction (1 for opening and 1 for closing), which is reasonable value on EUR/USD pair. We used transaction costs equal 4 only in the learning approach, to get more steady strategies.

In Table IV and Table V are given some experiments results. Benchmark *(buy&hold)* and *(sell&hold)* columns are the same values as those for datasets, adjusted by the learning time frame which is always taken out of the consideration. Positive *buy&hold* means that the prices on the test data was mainly raising, negative means the opposite. It's including provisions. Profits are presented in points rather than percentage, because we always have one position

TABLE IV. RESULTS OF TESTING OF 200 STRATEGIES PRODUCED ON 5 RISING TIME-FRAMES AND ONE LONG TIME-FRAME ON 3 DATA SETS

Learned on (tested on)	H1 (T1)	H1 (T3)	H9 (H1)	H2 (T1)	H2 (T3)	LONG1 (H1)	SUM
Benchmark (buy&hold) incl. provision	-1115	908	-1028	-1126	897	1131	-333
Benchmark (sell&hold) incl. provision	1023	-1000	936	1034	-989	-1227	-223
Avg. profit **not** incl. provision	217	172	-289	-824	1238	-220	294
Avg. profit incl. provision	-67	-70	-680	-1341	763	-451	-1846
Avg. profit with transaction balancing incl. provision	-30	-29	-604	-1267	825	-413	-1518
Combined strategy including provision	**-960**	17	-259	**-403**	1398	-455	-662
Standard deviation (from profit incl. provision)	1013	934	796	890	716	705	-
Max (from profit incl. provision)	2574	1993	2074	1856	2758	1814	-
Min (from profit incl. provision)	-2938	-2644	-2896	-3256	-1827	-2532	-
Median (from profit incl. provision)	-161	-217,5	-777	-1112	843	-386	-

TABLE V. RESULTS OF TESTING OF 200 STRATEGIES PRODUCED ON 5 FALLING AND 3 NEUTRAL TIME-FRAMES ON 3 DATA SETS

Learned on (tested on)	L1 (T1)	L1 (T2)	L2 (H1)	L8 (T1)	L8 (T3)	N1 (H1)	N2 (H1)	N6 (H1)	SUM
Benchmark (buy&hold) incl. provision	-536	**959**	-538	-793	1230	-816	-920	-1032	-2446
Benchmark (sell&hold) incl. provision	**444**	-1051	446	701	-1322	724	828	940	1710
Avg. profit **not** incl. provision	2246	2037	-156	**1547**	1585	189	409	142	7999
Avg. profit incl. provision	**1381**	**1295**	**-688**	**752**	790	-565	135	40	3140
Avg. profit with trans. balancing incl. provision	1493	1381	-542	**847**	919	-462	184	51	3871
Combined strategy including provision	1533	1125	**592**	1354	1458	-95	-157	**379**	**6189**
Standard deviation (from profit incl. provision)	1245	972	1102	1500	999	1019	968	729	-
Max (from profit incl. provision)	2457	2843	3215	2596	3062	2062	3312	2968	-
Min (from profit incl. provision)	-4425	-3196	-3022	-4323	-3660	-3761	-2556	-2391	-
Median (from profit incl. provision)	**1810**	**1362**	-851	1174	946	-563	99	-7	-

opened, so earned points mean always the same real profit (10 dollars per point). That means, that consecutive profits add linearly, not exponentially. To get exponential results we have to open variable amount of contracts. It's difficult to give the results in percentage, because we need to know the leverage of the potential investor.

Average profit is the profit earned by 200 strategies generated on present learning set, ignoring transaction costs. *Avg. profit incl. provisions* are average results adjusted by transaction costs (2 points per transaction). *Avg. profit with transaction balancing (incl. provision)* is an average profit per strategy generated by playing with all strategies simultaneously, and balancing together buy and sell transactions opened on the same interval. This value is always between avg. profit and avg. profit including provision. *Combined strategy (incl. provision)* is a result of a strategy being built from all 200 strategies generated by EA. We run all strategies simultaneously, but open a transaction only if at least 60% want to open it and agree for the same direction. We are always playing with maximally 1 opened position. *Standard deviation, max, min* and *median* are some statistical functions run on all profits including provision for all 200 strategies.

We can see that strategies learned on time-frames *L1* and *L8* (see Table V) give extraordinary profits on both rising as falling data sets. For example if one has been very bullish on all time frames from second data set he would have earned 959 points, but average strategy (learned on *L1*) has gained 1295 points. The median of all strategies is high: 1362 points. Results on first data set are even better – bearish *sell&hold* strategy would have earned 444 points, where average generated (on *L1*) strategy would have earned 1381 points. Each point is valued as 10 dollars gain for the investor with one position opened, so it means $13.810 gain per one standard contract (so called *lot*), which is a very good result.

Other interesting result is the value of avg. profit with transaction balancing vs. without balancing. If we run all strategies simultaneously and balance trades which are contrary to each other we can spare 12-13% on transactional costs. It means for example that instead of earning 752 points we can earn 847 points (see Table V, it would be 1547 points without transactional costs) on one data set using strategies generated on *L8*. Unfortunately we must be ready to have so many opened contracts as many strategies we use, which limits the use of this strategy to either a limited choice of strategies or investors with great amounts of cash allowing opening 100-200 contracts.

The ultimate solution to the already mentioned problem is a combined strategy. We take our 200 EA generated strategies and build one strategy from it. The rule is easy, if 60% of all strategies show buy signal, we buy. If 60% of all strategies generate sell signal, we sell. In this case uncertain strategies (what happens pretty often) act against opening transactions. Interesting results can be obtained too, if we decide to leave the market if the uncertainty rate is higher than 90-98%.

Such combined strategies have one main advantage: they are single strategies. We can open just one contract, and still get the knowledge from all strategies and get the same or even better results as the average of all these strategies. In fact the results of such combined strategy are nearly always better as avg. profit for all strategies (for *L1* tested on second dataset the result was a little bit worse – 1125 vs. 1295 points, but still comparable, the same for *LONG1* (see Table IV), only on *H1* (*T1*) the results for combined strategy were much worse than average, -960 vs. -67 points).

In the Figure 3 we can see a schematic representation of our combined strategy mechanism. In the lower part we can see how many strategies were bullish (light bars) and bearish (dark bars). They are placed on each other. If the amount of bullish/bearish transactions crosses the decision line, a new position is opened or the old one closed. We can move this line higher or lower, so we can possibly ignore some signals or not. Studies have shown that optimal values depend on the learning set and vary between 42-62%.

We can see that strategies generated on *L2*, and *H2* (*T1*) were behaving worse, than even a wrong direction of buy/sell and hold strategy. It's alarming, because it means that if we choose wrong learning set, the results can be really bad, although not much worse than keeping a wrong position opened. We cannot forget that it's an expected result. There are some time-frames which give a great opportunity for our EA algorithm to extract some reasonable rules and some that are not. We have to choose interesting time-frames by hand and by using some additional testing.

Fig 3: Combined strategy as an additional indicator. Chart below the actual price chart shows the distribution of bullish and bearish strategies (at each interval of time).

Such interesting time-frames are definitely *L1* and *L8* (shown in Table 2). Frames, that are generating some strategies which are very neutral are, not surprisingly, "neutral" time-frames *N1*, *N2*, *N6*. They would be profitable if we would ignore transactional costs. Interesting time-frame is *H2* – it produces profitable strategies but only for growing market. Obviously we wouldn't use such strategies because of heavy (but a bit lower than wrong *buy&hold* strategy) losses generated by those on falling market.

We have run a "long term" test on a 05-01-2007 - 01-05-2007 time-frame. Results for *buy&hold* on this time-frame would be 549 points, and for *sell&hold* gives -553 points.

Results for a strategy learned on *L8* are: the average profit equals 324 points (including provision: 178 points). The profit gained by transaction balancing equals 191 points and combined strategy gained 280.

Results are not as good as one could expect, but there are some important things to mention - we have learned our EA on a falling time-frame, and it's a rising one. If we would just sell at the beginning and buy at the end, we would lose 553 points and we have earned 280 points which is over a half of 549 points earned by a *buy&hold* strategy.

Also, we have tested *H2* combined strategy on the same time-frame. *H2* has gained 650 points, which is better than *buy&hold*. It's not surprising, it's a rising time-frame and it was thought on a rising time-frame too. What's interesting it's the fact, that it has made some profitable short trades (3 out of 4). It has proven to detect some local heights.

To summarize the results, our approach allows generating profitable strategies if we consider profits from more than 20 test time-frames. It was impossible to obtain a strategy, that was always gaining money independent of the actual time-frame, but when adding profits and losses from many time-frames, the results are definitely positive. It's important to choose a right learning time-frame – not each of them contains enough information to learn. Choosing combined strategy is nearly always a good idea, allowing us to open just one contract and still profit as an average strategy, without the risk connected with arbitrarily chosen strategy from all generated, which could possibly result in a very poor behavior (see *Min* column in Table IV or Table V). The most important – it's possible to trade with our strategies without human intervention and generate gains.

V. Further work

The matter of risk of our automatically generated strategies was ignored in this work, so as the capital management. In further studies they need additional research, because without those it cannot be used in any long term trading system, were not only random, high capital losses are not allowed, but also the amount of opened positions must vary in a way that uses the invested capital optimally. In future market such as Forex without proper capital management a so called "margin call" can happen, which means that in case of losses an additional deposit of cash must be made, or the position will be closed and (nearly) all the money lost.

Another case that needs more attention is so called "take-profit" (*tp*) and "stop-loss" (*sl*) orders. They were ignored in this work, but they are widely used by most of the investors, especially on Forex market. They allow us to profit from very short moments of euphoria on the market, where the price peaks only for seconds (*tp*) or to save our capital if we want to limit our losses on single transaction (*sl*). We could possibly allow the genetic algorithm to set this orders to either fixed values (for example 100 points from the transaction price) or values constantly adjusted by decision trees.

VI. Conclusion

We think that although our automating generated strategies are still too unstable and unreliable to use them in real trading systems, they can be used as an additional help for an experienced investor. Especially in the form of additional indicator, which shows in real time how many strategies are for and how many against our decisions. If we combine strategies from rising and falling learning time-frames our indicator is even more interesting, revealing us the knowledge of hundreds of automatically generated strategies in a very accessible form (which needs some practice though). However, reliability of such an indicator still needs further studies. We are planing to verify our method results comparing with another methods on the same data.

References

[1] Allen F., Karjalainen R., "Using genetic algorithms to find technical trading rules", Journal of Financial Econ. 51(2), pp.245-271, 1999.

[2] Atsalakis G. S., Valavanis K. P., "Surveying stock market forecasting techniques - Part II: Soft computing methods", Expert Systems with Applications: An Inter. Journal Arch 36 (3), pp. 5932-5941, 2009.

[3] Bank for international settlements, "Foreign exchange and derivatives market activity in 2007", Triennial Central Bank Survey of Foreign Exchange and Derivatives Market Activity, 2007.

[4] Chen Y. , Mabu S., Shimada K., Hirasawa K., "A genetic network programming with learning approach for enhanced stock trading model", Expert Systems with App., 36(10), pp. 12537-12546, 2009

[5] Deng J.-L.,"Control problems of grey systems", Systems & Control Letters, 1(5), pp. 288-294, 1982

[6] FOREX historical data for EUR/USD pair http://www.forexrate.co.uk/forexhistoricaldata.php

[7] FOREX trading hours http://www.gftforex.com/forex/forex-trading-hours.asp

[8] Kim H., Shin K., "A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets", Applied Soft Computing, v.7 n.2, p.569-576, 2007

[9] Korczak J., Roger P., "Stock Timing using Genetic Algorithms", [in:] Jour. of Stoch. Models in Business and Indust., 18, pp.121-134, 2002.

[10] Korczak, J., Lipinski, P., Roger, P., "Evolution Strategy in Portfolio Optimization", Artificial Evolution, ed. P. Collet, LNCS 2310, Springer, 2002, pp.156-167.

[11] Koza J. R., "Genetic Programming: On the Programming of Computers by Means of Natural Selection", MIT Press, 1992

[12] Kucharski A., „Algorytmy genetyczne w prognozowaniu danych giełdowych - usuwanie obserwacji nietypowych", Badania Operacyjne i Decyzje, pp. 35-45, 1/2008 (*in polish*).

[13] Lin L., Cao L., Wang J., Zhang C., "The Applications of Genetic Algorithms in Stock Market Data Mining Optimisation", Zanasi, A., Ebecken, N. F. F. (eds.) Data Mining V. WIT Press (2004).

[14] Murphy J., "Technical Analysis of the Financial Markets" , 1999.

[15] Myszkowski P. B., Rachwalski Ł., „Trading rule discovery in Warsaw Stock Exchange using coevolutionary algorithms", 4th Inter. Symp. Advances in AI and App., Mrągowo (Poland), pp.81-88, 2009.

[16] Neely C., Weller P., Dittmar R., "Is Technical Analysis in the Foreign Exchange Market Profitable? A Genetic Programming Approach", Jour. of Finan. and Quantitative Analysis 32 (4), pp. 405–426, 1997.

[17] Pei-Chann C., Chen-Hao L., Jun-Lin L., Chin-Yuan F., Celeste S.P. Ng, "A neural network with a case based dynamic window for stock trading prediction", Expert Syst. with App. 36, pp.6889-6898, 2009.

[18] Thomas J., Sycara K., "The Importance of Simplicity and Validation in GP for Data Mining in Financial Data", AAAI Press, 1999.

[19] Wang Y.-F., "Predicting stock price using fuzzy grey prediction system". Experts Systems with Applications. V22. 33-39, 2002

[20] Witkowska D., Marcinkiewicz E., "Construction and Evaluation of Trading Systems: Warsaw Index Futures", Inter. Advances in Econ.Research 11 (1), pp. 83-92, 2005.