Generating Robust Investment Strategies with Agent-Based Co-evolutionary System

Rafał Dreżewski, Jan Sepielak, and Leszek Siwik

Department of Computer Science, AGH University of Science and Technology, Kraków, Poland
{drezew,siwik}@agh.edu.pl, sapielak@gmail.com

Abstract. Agent-based co-evolutionary systems seem to be convenient for modeling market-related behaviors, since they allow for defining competing and/or co-operating agents which can interact and communicate with each other and influence the environment and other agents. In the course of this paper the idea of utilizing of agent-based co-evolutionary approach for supporting decisions of financial investor through generating possible investment strategies is presented and experimentally verified.

1 Introduction

Almost simultaneously with the growing maturity of financial markets, international stocks, etc., researchers paid their attention to mathematical models, formulas and theorems supporting investment decisions. It is enough to mention in this place such models supporting building effective portfolio as Nobel prize winner Harry Markowitz' Modern Portfolio Theory (MPT) proposed in 1952 [8] [9], or its extension proposed in 1958 by James Tobin [15]. Although, mentioned theory lays the foundations of modern capital investments practically it is nowadays rather only historically-important method of assets pricing. Next, Capital Asset Pricing Model (CAPM) was proposed by J. Traynor [16], J. Lintner [7], J. Mossin and formalized by W. Sharpe [14]—and it was based of course on previous work of Markowitz and his MPT theory. On the basis of the critique of CAPM (e.g. so called Roll's Critique)—Arbitrage Pricing Theory (APT) was proposed by Stephen A. Ross in mid-1970s [13]. In 1990s so-called Post Modern Portfolio Theory (PMPT) was proposed. The notion of PMPT was used for the first time probably by B. M. Rom and K. W. Ferguson in 1993 [12].

The real and wide support of the investments-related decision making was possible along with the use and applying computational units and dedicated computational methods and algorithms. Because of the complexity and difficulty of market-related problems attempts of applying computational intelligence heuristics rather than accurate numerical methods was natural and justified. However, observing market situations and behaviors—entrepreneurs, small and medium enterprises (SMEs), and corporations—all of them all the time have to be more innovative, cheaper, more effective etc. than the others in order to maximize their profits. That is why, all the time some enterprises introduce some organizational, financial or technological innovations and the rest of the market-game participants have to respond to such changes. All the time we are eye

M. Bubak et al. (Eds.): ICCS 2008, Part III, LNCS 5103, pp. 664–673, 2008.

witnesses of a peculiar “arms race”, which can be compared to “arms races” caused by co-evolutionary interactions. The range of dependencies that can be seen on the market can be pretty wide—from co-operation, through competition until antagonism. It is obvious however, that all activities of each participant of the market game are conformed to one overriding goal—to survive and to maximize profits (analogies to Darwin’s “survival of the fittest”). From the interactions with another enterprises point of view it can be realized by: eliminating from the market as many (weak) rivals as possible and taking over their customers, products, delivery channels etc. (analogy to “predator-prey” interactions), by sucking out of another (“stronger”) enterprises’ customers, technologies, products etc. (analogy to “host-parasite” interactions), by supplementing partners’ portfolio with additional products, technologies, customers etc., and by co-operating with other actors of the market (analogies to living in “symbiosis”). It is seen clearly, that one of the most important activity of all market-game participants is co-existence with co-development—and from the computational intelligence point of view we would say—co-evolution (this subject is wider covered in [4]).

Because (generally speaking of course and under additional conditions) participants of the market game are autonomous entities (from the computational intelligence point of view we would say—“agents”), they are distributed, they act asynchronously, and they interact with another entities and with the environment to achieve their individual goal—maximizing profit—in the natural way, applying co-evolutionary multi agent systems seems to be the perfect approach for modeling such phenomenons and environments. In this paper we will present exemplary agent-based co-evolutionary approach for supporting investment decisions.

In the literature there can be found some attempts for applying evolutionary-based approaches for supporting investing-related decisions. S. K. Kassicieh, T. L. Paez and G. Vora applied the genetic algorithm to making investment decisions problem [6]. The tasks of the algorithm included selecting the company to invest in. Their algorithm operated on historical stock data. O. V. Pictet, M. M. Dacorogna, R. D. Dave, B. Chopard, R. Schirru and M. Tomassini ([10]) presented the genetic algorithm for automatic generation of trade models represented by financial indicators. Three algorithms were implemented: genetic algorithm, genetic algorithm with fitness sharing technique developed by X. Yin and N. Germay ([17]), and genetic algorithm with fitness sharing technique developed by the authors themselves in order to prevent the concentration of individuals around “peaks” of fitness function. Proposed algorithms selected parameters for indicators and combined them to create new, more complex ones. F. Allen and R. Karjalainen ([1]) used genetic algorithm for finding trading rules for S&P 500 index. The algorithm could select the structures and parameters for rules.

Creating a system which uses co-evolutionary and agent-based approach to investment strategy generation is interesting because such systems probably could better explore solution space and slow down a convergence process. However, to our best knowledge there have been no attempts of implementation of such systems so far. In the next sections the realization of agent-based co-evolutionary system for generating investment strategies is presented and its properties are experimentally verified.

The paper is organized as follows: in the Section 2 the general idea and architecture of evolutionary system for generating investment strategies is presented. The architecture of the system has been already described in detail in [3], so in this paper we will only focus on the algorithms used in the system. In the Section 3 some selected results obtained with the use of the proposed approach are presented. As compared to the results presented in [3], now we deeper analyze the proposed agent-based co-evolutionary algorithm for generating investment strategies. Finally this work is briefly summarized and concluded in the Section 4.

2 Agent-Based Co-Evolutionary System for Generating Investment Strategies

As it was said, the architecture of the evolutionary system for generating investment strategies has been described in [3], so in this section we will mainly focus on the algorithms used as the computational components. The algorithms used in the system include: evolutionary algorithm (*EA*), co-evolutionary algorithm (*CCEA*), and agent-based co-evolutionary algorithm (*CoEMAS*).

The system has component architecture ([3]) what implies that we can easily replace the given element of the system with another one. The most important parts of the system are computational component, historical data component, visualization component, and strategies evaluation component.

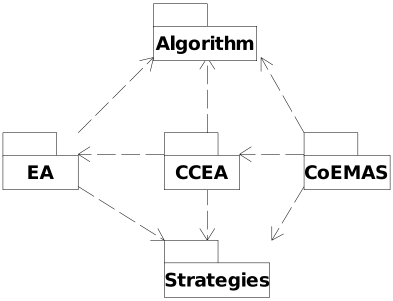

Fig. 1. Dependencies between the evolutionary algorithms packages

The classes implementing the algorithms used as computational components was organized into packages shown in the Fig. 1. Among the most important packages the following should be mentioned:

1. *Algorithm*—it is the most general package containing classes, which are the basis for other classes of strategies generation algorithms;
2. *Strategies*—all classes connected with the representation of strategies are placed within this package;
3. *EA*—it contains the implementation of the evolutionary algorithm;
4. *CCEA*—it contains the implementation of the co-evolutionary algorithm;
5. *CoEMAS*—which contains the implementation of the agent-based co-evolutionary algorithm.

In all three algorithms the strategy is a pair of formulas. First formula indicates when one should enter the market and the second indicates when one should exit the market. Each formula, which is a part of a strategy, can be represented as a tree, which leafs and nodes are functions performing some operations. Such tree has a root and any quantity of child nodes. The function placed in the root always returns logical value. The Fig. 2 shows the tree of exemplary formula, which can be symbolically written in the following way: $MACD > Mov(MACD, 6, EXPONENTIAL)$.

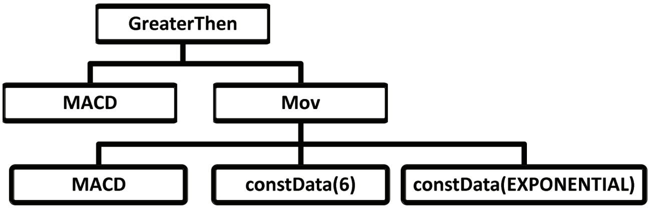

Fig. 2. The tree of exemplary formula

Functions which formulas are built of (there are altogether 93 such functions in the system) are divided into four categories: functions returning stock data (e.g. Close (returns close prices)), mathematical functions (e.g. Sin (sine function)), tool functions (e.g. Cross (finds a cross point of two functions)), and indicator functions (e.g. AD, which calculates the Accumulation/Distribution indicator).

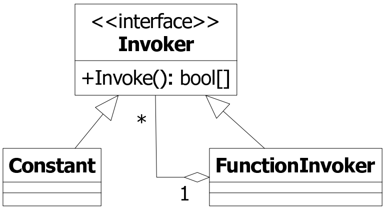

Fig. 3. Class diagram for formula representation

The Fig. 3 shows how formula tree is represented. There are three main classifiers. The interface *Invoker* allows to treat all tree nodes and leaves in the same manner. Class *FunctionInvoker* represents a node of the tree if it wraps function with arguments or represents leaf if it wraps function without arguments. Class *Constant* wraps constant value and it can be also treated as a function which returns always the same value.

2.1 Implemented Algorithms

In the *evolutionary algorithm* the genotype of individual contains two formulas. One is treated as exiting the market formula and the second as entering the market formula.

Fitness estimation is performed with the use of historical stock session data. As a result of formula execution two arrays with logical values are created. One of them concerns entering the market actions and the second one exiting the market actions. The algorithm which computes the profit processes these arrays and determines when the *buy* and when the *sell* actions occur. If the system is outside of the market and in the *buy* array is the *true* value then entering the market occurs. If the system is inside the market and in the *sell* array is the *true* value then exiting the market occurs. In the other cases no operation is performed.

Equation (1) shows the manner of the profit computation:

$$PR = \sum_{i=1}^{t} P_{en} - C_{en} + P_{ex} - C_{ex} \tag{1}$$

where PR is the profit earned as a result of the strategy use, t is the number of transactions, P_{en} and P_{ex} are respectively the prices of entering and exiting the market, C_{en} and C_{ex} are respectively commissions paid at entering and exiting the market.

During the computing of the fitness, besides the profit, also another objectives are taken into consideration: formula complexity—to complicated formula could cause computations slow down, and transaction length which depends on user's preferences. For the sake of simplicity weighted multi-objective optimization was used.

Equation (2) shows how average transaction length is computed:

$$TL = \frac{\sum_{i=1}^{t} BN_i}{t} \tag{2}$$

where TL is the average length of the transaction, and BN_i is the number of the bars (daily quotations) the transaction spans.

Equation (3) is used to calculate the fitness:

$$FV = W_{pr} \cdot PR - W_{fc} \cdot FC_b - W_{fc} \cdot FC_s + W_{tl} \cdot TL \tag{3}$$

where FV is the fitness value, W_{pr} is the weight of the profit, W_{fc} is the weight of the formula complexity, FC_b is the complexity of the buy formula, FC_s is the complexity of the sell formula, W_{tl} is the weight of the transaction length.

In the evolutionary algorithm the tournament selection mechanism [2] was used. Three kinds of recombination were used in the evolutionary algorithm:

- *return value* recombination—it requires two functions in the formulas with the same return type. The functions with their arguments are swapped between individuals.
- *argument* recombination—it requires two functions in the formulas with the same arguments. The arguments of the functions are swapped between individuals.
- *function* recombination—it requires two functions in the formulas with the same arguments and with the same return type. The functions are swapped between individuals and the arguments stay at its original places.

Two kinds of mutations were used:

- *function arguments* mutation—it requires constant value argument. Such constant is changed to another from the allowed range.
- *function* mutation—it replaces the function with another one. There are three variants of this mutation: the function is replaced by another one with the same parameter and the same return value, the function is replaced by another one with the same return type, but parameters are created randomly, the function is replaced by another one with the same return type, but parameters are preserved if it is possible—the parameters which cannot be taken from the function being replaced are created randomly.

In the *co-evolutionary algorithm* the co-operative approach was used ([11]). In this model of co-operative co-evolution the problem is divided into sub-problems which are solved by sub-populations (species). Each sub-population is processed by the evolutionary algorithm. Individuals from the different populations interact only during the fitness estimation stage. There are two kinds of species in the CCEA: entering the market and exiting the market. Each individual has one formula encoded in its genotype. This formula is treated as formula of entering the market or as formula of exiting the market—depending on to which sub-population (species) the given individual belongs. Individual of the given species was evaluated in the group of individuals. Such group consists of evaluated individual and the chosen representative individual coming from the other sub-population. At the initialization stage of the CCEA algorithm the representative individual is chosen randomly. After the initialization stage the representative individual is the best individual from the last generation. Genetic operators and selection mechanism used in the CCEA are the same as in case of the evolutionary algorithm.

Co-evolutionary multi-agent algorithm algorithm is the agent-based version of the *CCEA* algorithm. Now, each co-evolutionary algorithm is an agent which independently performs computations. The populations of each co-evolutionary algorithm also consist of agents-individuals. The populations exist within the environment which has the graph-like structure with computational nodes ("islands") and paths between them. In each node of the environment there exist resources. Such resources circulate between the agents and the environment. During the evolution process agents give back certain amount of the resources to the environment. Resources are also given to the agents by the environment in such a way that "better" (according to the fitness function) agents are given more resources. Agents with the resource amount less than the value of the parameter specified by the user die. Such mechanism plays the role of selection—thanks to this the agents neither live too long and keep the resources unnecessarily nor they cause the stagnation of the evolutionary processes.

The fitness estimation and the genetic operators are generally the same as in the case of previously described algorithms. Quite different is the reproduction mechanism, which is now based on the resources. During the reproduction process the parents are chosen on the basis of the amount of resources they possess. After the recombination the parents give the children certain amount of their resources. During the mutation phase the amounts of the resources possessed by the individuals do not change. The agents-individuals can migrate between the nodes of the environment. During the migration of the individual its resource amount is decreased by the certain constant—the resources are given back to the environment.

3 Experimental Results

In [3] the implemented algorithms were analyzed taking into consideration their efficiency, convergence properties and generalization properties. In this paper we deeper analyze properties of the agent-based co-evolutionary algorithm, for example the influence of some parameters' values on the best individual's fitness, and the ability of rules generalization. We also additionally verify the properties mentioned above.

During the experiments all the algorithms worked through 500 generations. They generated strategies for 10 stocks and historical stocks data coming from the period of 5 years were taken into consideration. The session stock data came from the WIG index ([5]) and the period from 2001-09-29 to 2006-09-29.

In the case of analyzing additional aspects of CoEMAS, the system worked through 300 generations, and generated strategies for 3 stocks. Data came from the period of 5 years, whereas while analyzing the ability of the generalization the algorithm worked through 500 generations and the data came from a randomly selected period of at least 5 years. Strategies were generated for 4, 7, 10, 13, and 16 securities chosen randomly.

All the experiments were carried out with the population size equal to 40. In the CoEMAS that population size was achieved using environment with the resource level equal to 980 (CoEMAS is the algorithm with variable population size, which depends on the total amount of the resources within the system). Presented experiments were carried out on the PC with one AMD Sempron 2600+ processor.

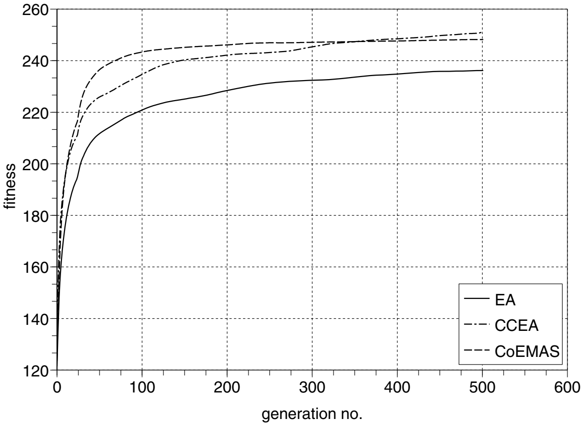

Fig. 4. Fitness of the best individuals in the population (average values from 20 experiments)

Both, Table 1 and Fig. 4 show that CCEA has the best efficiency—the fitness value is the highest in this case. In the case of CCEA high convergence does not cause the fitness worsening. The EA algorithm reveals the worst efficiency and also the highest convergence. The CoEMAS algorithm obtained slightly worse fitness then CCEA, but has

Table 1. The comparison of efficiency, work time and convergence of the algorithms

Algorithm	The best fitness in last generation	Convergence in last generation	Algorithm work time (m:s)
EA	236.171	59.481	1:57.27
CCEA	251.052	58.792	12:9.341
CoEMAS	248.222	44.151	14:49.048

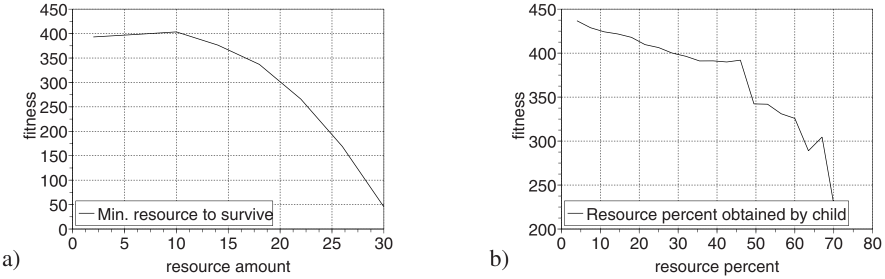

Fig. 5. The minimal amount of resource needed for individuals to survive in CoEMAS (each point is the average of 10 experiments) (a), and the percentage of resources transferred from parental agents to their children during recombination (b)

the smallest convergence. The Fig. 4 shows that the best fitness of the EA during the entire evolution process is much worse than the best fitness of the remaining two algorithms. The best fitness of the CCEA until 350 generation is worse than the best fitness of the CoEMAS, but later (after 350 generation) CCEA obtained a little better results.

Providing the high quality results is costly. The cost is manifested by the computational time. The work time of both CCEA and CoEMAS is considerably longer (six and seven times respectively) than this of EA.

Search for optimal values of the parameters of all algorithms was performed by using back to back experiments. For a given value of the parameter each algorithm was launched a few times and then the average results was computed. Then the value of the particular parameter was modified and efficiency was estimated again. The conclusions coming from such experiments are as follows: for most of the parameters no regularity was found. But there are some exceptions. The most interesting plots/diagrams are presented in the Fig. 5.

Table 2. Results of examining the CoEMAS capability of generalization

Number of stocks in group (n)	Profit (%) per year	Profit (%) per year for random stocks
4	83.71	5.62
7	132.71	23.64
10	48.79	27.11
13	35.33	26.20
16	100.83	25.93

The Fig. 5a shows the influence of the minimal amount of resource needed for individuals to survive on CoEMAS efficiency. At the beginning, with the growth of the value of this parameter the efficiency of the CoEMAS slightly increases. But when the resource amount exceeds 10 units the same efficiency systematically decreases.

The Fig. 5b presents the percentage of the resources transferred from parental agents to their children during the recombination in CoEMAS. It indicates that transferring a large amount of the resource causes a decrease in the efficiency of the algorithm. Whereas transferring a small amount of the resource causes increase in the algorithm efficiency.

As it was mentioned, CoEMAS was also analyzed from the ability for generalization point of view. In the Table 2 the results of experiments concerning generalization are presented. The results show that the number of stocks in a group does not influence the results of strategy generation for the given stocks. Moreover, small amount of stocks in a group (less than 7) causes obtaining poor results for random stocks along with the small capability for generalization. When the number of stocks taken for strategy generating is greater or equal to 7, then the algorithm gives good results in spite of the random securities.

4 Summary and Conclusions

In the course of this paper three evolutionary based approaches for supporting investors with their decisions related to entering (purchasing stocks) and exiting (selling stocks) the market were presented and experimentally verified.

As it was mentioned in the Section 1, using co-evolutionary based approaches seems to be a natural way of modeling market-related interactions, relations and behaviors. In this paper such an approach was used for generating robust investment strategies.

As it can be observed in the Section 3 preliminary experiments and obtained results are very promising since the co-evolutionary algorithms (both "classical" and agent-based) allow for obtaining much more valuable results than compared evolutionary algorithm. During the experiments it was possible to distinguish such parameters of the proposed CoEMAS approach that influence the system and obtained results most significantly and in most regular way.

Authors are obviously still researching and experimenting with the proposed algorithms and the future works will include wide range of additional experiments (also with data coming not only from Warsaw Stock Exchange). On the basis of obtained results further modification and tuning up of proposed approaches will be performed. Also comparisons of obtained results with strategies generated not only on the basis of another (evolutionary based) heuristics but also with the use of classical models, systems and algorithms which are being used by investors for supporting their financial decisions are included in the future research plans. Additionally, as it was mentioned in the Section 1, since the goal of this paper was to experimentally verify the proposed agent-based co-evolutionary approach—it is natural that more interesting were comparisons of efficiency, and the influence of some parameters' values on the obtained results than analyzing particular strategies obtained during experiments. It explains the way of presenting obtained results in the Section 3 but in the future work not only quantitative but also qualitative comparisons of generated strategies will be performed.

References

1. Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51(2), 245–271 (1999)
2. Bäck, T., Fogel, D., Michalewicz, Z.: Handbook of Evolutionary Computation. IOP Publishing and Oxford University Press (1997)
3. Dreżewski, R., Sepielak, J.: Evolutionary system for generating investment strategies. In: Giacobini, M. (ed.) EvoWorkshops 2008. LNCS, vol. 4974, pp. 83–92. Springer, Heidelberg (2008)
4. Dreżewski, R., Siwik, L.: Co-evolutionary multi-agent system for portfolio optimization. In: Brabazon, A., O'Neill, M. (eds.) Natural Computation in Computational Finance, pp. 273–303. Springer, Heidelberg (in print, 2008)
5. Historical stock data, `http://www.parkiet.com/dane/dane_atxt.jsp`
6. Kassicieh, S.K., Paez, T.L., Vora, G.: Investment decisions using genetic algorithms. In: Proceedings of the 30th Hawaii International Conference on System Sciences, vol. 5. IEEE Computer Society, Los Alamitos (1997)
7. Lintner, J.: The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. Review of Economics and Statistics 47, 13–37 (1965)
8. Markowitz, H.: Portfolio selection. Journal of Finance 7(1), 77–91 (1952)
9. Markowitz, H.: The early history of portfolio theory: 1600-1960. Financial Analysts Journal 55(4), 5–16 (1999)
10. Pictet, O.V., Dacorogna, M.M., Dave, R.D., Chopard, B., Schirru, R., Tomassini, M.: Genetic algorithms with collective sharing for robust optimization in financial applications. Technical Report OVP.1995-02-06, Olsen & Associates Research Institure for Applied Economics (1995)
11. Potter, M.A., De Jong, K.A.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evolutionary Computation 8(1), 1–29 (2000)
12. Rom, B., Ferguson, K.: Post-modern portfolio theory comes of age. The Journal of Investing (Winter, 1993)
13. Ross, S.: The arbitrage theory of capital asset pricing. Journal of Economic Theory 13(3) (1976)
14. Sharpe, W.F.: Capital asset prices: A theory of market equilibrium under conditions of risk. Journal of Finance 19(3), 425–442 (1964)
15. Tobin, J.: Liquidity preference as behavior towards risk. The Review of Economic Studies 25, 65–86 (1958)
16. Treynor, J.: Towards a theory of market value of risky assets (unpublished manuscript, 1961)
17. Yin, X.: A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In: Forrest, S. (ed.) Proceedings of the Fifth International Conference on Genetic Algorithms. Morgan Kaufman, San Francisco (1993)