Modelling Cost into a Genetic Algorithm-based Portfolio Optimization System by Seeding and Objective Sharing

C. Aranha, H. Iba, *The University of Tokyo*

***Abstract*— Portfolio Optimization by GA is a problem that has recently received a lot of attention. However, most works in this area have so far ignored the effects of cost on Portfolio Optimization, and haven't directly addressed the problem of portfolio management (continuous optimization of a portfolio over time). In this work, we use the Euclidean Distance between the portfolio selection in two consecutive time periods as measure of cost, and the objective sharing method to balance the goals of maximizing returns and minimizing distance over time. We also improve the GA method by adding genetic material from previous runs into the new population (seeding). We experiment our method on historical monthly data from the NASDAQ and NIKKEI indexes, and obtain a better result than pure GA, defeating the index under non-bubble market conditions.**

I. Introduction

There are many applications of Evolutionary Computation techniques to the field of financial technical analysis. In particular, the Portfolio Optimization problem has seen an increase of EC-related research in recent years [3], [5], [4], [10].

Portfolio Optimization is an interesting problem from a GA point of view. The basic idea is that we want to choose a weighted group of assets from a possibly large number of available securities, in order to maximize the expected return given an acceptable risk rate. In other words, it is a resource allocation problem with competing objectives.

Among the current applications of Evolutionary Computation to the Portfolio problem we find: Multi-Objective Genetic Algorithms and Memetic Algorithms [8], Multi-State GA [10], Memetic Algorithms, Genetic Programming [9], [3], among others.

All the above techniques have been applied to optimize one isolated position. In other words, they optimize the weights of the different securities in a portfolio without taking into account the weights held previously, or planned future positions. However, there are costs related with selling and buying the needed stocks to change a portfolio structure, and not taking these costs into account may cause large differences between theoretical and practical results [1], [8]. In [3], for instance, there is up to a 150% reduction of returns when cost measures are added to the model.

In this work, we adapt a known GA-based Portfolio Optimization method to take into consideration the previously held position, aiming to minimize the transaction costs by minimizing the difference between the previous held portfolio and the portfolio for the current time period. We change the method in two ways to achieve this: establishing the Euclidean distance between the held portfolio and the goal portfolio as a second objective for the GA, and seeding the population with individuals from the optimization run from the previous position.

In the next section, we explain the Portfolio Optimization problem in more detail. In section 3, we describe the GA-based Portfolio Optimization method, and our modifications to it. In section 4 we make an experimental analysis of the modifications, and compare them with the vanilla method. We discuss the results on section 5.

II. Problem Description

The Portfolio Problem can be described as a resource allocation problem. Given a finite amount of resources, and a group of processes to which these resources can be allocated to, what is the the allocation which maximizes a given goal function? Here the resource is the available capital for investment, the processes are the different assets we can invest into, and the goal is the return of this investment.

This is the problem of generating an *Investment Portfolio*. The problem can be thought out in a strategic (high) or tactical (low) level. In a high level elaboration of the problem, the investor has to think about what are his goals for the invested capital - for instance, if he desires liquidity, a regular income (dividends), or rapid valorization - both for the short term and long term. Based on these goals the investor can plan on allocating his capital to different kinds of investments, like foreign currency, bonds or stock.

The low level elaboration of the problem is the balancing of a group of assets of the same kind, after the definition of the strategic goals. Suppose that, as part a investing strategy, an investor allocates c_i of capital to technology stock. He must then choose a subgroup among the many available technology assets that will maximize his return to a given risk rate. This portfolio must then be accompanied over time, and adapted to changes in the market, or in the goals of the investor.

This problem can be modelled by the *Modern Portfolio Theory*.

A. Modern Portfolio Theory

Markowitz proposed the Modern Portfolio Theory, which states that by choosing a combination of assets to invest in, a investor could get higher returns with the same amount of risk [6]. He also proposed the mean-variance method [7], which could be used to maximize the portfolio selection problem.

Intuitively, in the mean-variance approach, the expected return of the portfolio is the weighted mean of its composing

1-4244-1340-0/07/$25.00 ©2007 IEEE

assets' expected returns, and the expected risk of the portfolio is the variance of this return. Thus the risk is a measure of how reliable the expected return is. The formulas for the expected return and risk of a portfolio is given by equations (1) and (2) below, respectively:

$$E(r_p) = \sum_{i=1}^{n} E(r_i) w_i, \quad (1)$$

$$\sigma_p = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} w_i w_j, \quad (2)$$

where w_i is the weight of asset i in the portfolio ($0 \leq w_i \leq 1$ and $\sum w_i = 1$), $E(x)$ is the expected value of the random variable x, r_i is the return for asset i, r_p is the return for the portfolio, and σ_{ij} is the covariance between assets i and j.

According to Modern Portfolio Theory, when balancing a portfolio we want to maximize the value of equation 1 and minimize equation 2. Besides the difficulties of acquiring good estimates for the expected return values, as the number of available assets grow, the calculation of the risk becomes more and more complex, and the search space become larger. Thus, automated methods to calculate optimal portfolios were developed.

B. Modelling Cost

While return and risk are used to create a theoretical model of the portfolio problem, in the real world we have a third face of the problem: transaction costs.

If we want to change the weights of a portfolio P_1 to that of a portfolio P_2, we will have to buy stock that belongs to P_2 but not P_1, and sell stock from P_1 which is not in P_2. Buying and selling an asset has an associated cost, so as the difference between P_1 and P_2 increases - in other words, the difference between the current position and the desired position - the cost of changing to the new position increases as well. If this cost becomes too high, it may overcome the value of the expected return.

It is difficult to correctly insert a measure of cost into the portfolio model. Stock dealers may have different policies regarding the costs of financial transactions. Often, an arbitrary value for cost is chosen, like in [8], [3]. Sometimes, a more abstract approach is taken, and the cost value is let unfixed [1].

In this work we use an indirect approach to cost. We assume that the transaction costs are based on the amount of an asset sold or bought (like the above works), but we do not set a fixed value to this cost. Instead, we define a third goal to the model - the distance between the current position and the desired position. In this definition, distance is the amount by which the weights in the two positions differ, which is given by the Euclidean distance of the weight vectors. This measure has not been used in GA optimized portfolio approaches.

By using this measure of cost, we allow ourselves more freedom to search for solutions that satisfy each goal to different degrees. Since automated financial agents usually act as assistants to human investors, instead of acting independently, this design flexibility is desirable.

However, the choice of the Euclidean Distance as a measure of cost has some drawbacks. The major one is that it introduces the assumption that large transactions of a single asset are less desirable than small transactions of many assets. This often does not hold in practice.

III. System Implementation

There are many works which implement GA techniques for Portfolio optimization with solid results. Yang [10] uses Multi-State GA with a single weight-array portfolio representation where all assets are introduced in the portfolio. While Lin [4] uses a "knapsack" system with a single index-array portfolio representation. Streichert [8] uses a mixed approach where a weight array and an index array represent one portfolio proposal.

In this work we implement and describe a Simple GA system that draws from the common points of the above works. We then use this Simple GA algorithm as an starting point for the changes necessary to add cost as a new objective function to the system.

This system is based on a model with the same assumptions as those of the works mentioned here. For example, trading volumes are not taken into account (they do not influence the availability or cost of trades, nor the price of assets), and the investor is supposed to be concerned only with the return and risk of the portfolio.

A. Portfolio Representation

In this work, we represent one portfolio by two arrays, index (I) and weights (w). Given a universe of n assets, I is a binary array which determines if an asset i ($0 \leq i < n$) belongs to the portfolio or not (I_i is true if it belongs, false if not). w is a real-valued array which stores the weight of asset i in the portfolio.

When using a single array for weights, simple crossover of weights cannot handle the elimination of assets or introduction of new assets into the portfolio easily. One possible solution would be to create more complex mutation rules that include the introduction/removal of assets. However, we found it to be simpler to implement and modify the system if addition/removal and balancing of weights were dealt with in different structures.

Thus, the goal of the binary index array is to restrict the size of the search in the real valued weight array. The GA process chooses which assets will be part of our portfolio by evolving the index array. The real values in the weight array will then be evolved to balance the final portfolio. By leaving the GA to manipulate both the index array and the weight array (instead of just the weight array), we avoid using arbitrary rules to choose which assets to be balanced.

With this representation, we can estimate the expected return and the risk of a portfolio for a given time t in the following manner. First we calculate the past return of the portfolio for a training time range N (eq. (3)). Then we

estimate the future return of time t by the average of the portfolio returns (r_p^i) from $t-N$ to $t-1$ (eq. (4)).

$$r_p = \sum_{i=1}^{n} I_i w_i r_i \tag{3}$$

$$E(r_p^t) = \frac{\sum_{i=t-N}^{t-1} r_p^i}{N} \tag{4}$$

The risk for the portfolio is calculated as the standard deviation of the portfolio returns over the training time range against the average return:

$$\sigma_p = \sqrt{\frac{\sum_{i=t-N}^{t-1}(r_p^i - E(r_p))^2}{N}} \tag{5}$$

B. Portfolio Optimization

In our GA System each individual's genome is given by the arrays I and W explained above. A random generated individual has a random assignment of $true$ or $false$ for each I, and random real weights for W on the range $[0.0, 1.0]$, which are normalized for calculating the return of that individual.

We use a simple linear crossover for both arrays, where the offsprings receive the weight for each asset from one of the parents with equal probability. This applies for both the index vector and the weight vector.

For the mutation operator, each asset in the index vector of an individual has a parametric chance p_{mut} of being flipped (turned from true to false or false to true). Also, each asset in the weight vector has the same chance p_{mut} of being perturbed by +/- 10% (within a normal distribution).

The fitness measure for each individual is given by the Sharpe Ratio (eq. (6)). The Sharpe Ratio is a well known financial indicator that compensates the return value of an asset by its risk - often used to measure the performance of risky portfolios or assets, when compared to a known low-risk security.

$$Sr = \frac{E(r_p) - R}{\sigma_p} \tag{6}$$

Where R is a Risk Free Asset, or Benchmark asset: one asset with a low, but constant, return rate. US Treasury bonds are often used as a measure of R in this formula.

Our evolutionary strategy is as follows: We use elite strategy, where the best p_{elite} individuals from one generation are directly copied to the next generation. The remaining individuals in the new generation are then created by crossover with a certain probability. Selection is made by a deterministic tournament strategy, where k individuals from the old generation are picked at random, and the fittest among these k individuals is chosen to generate offspring.

C. Population Seeding

One of the techniques we introduce in this work to generate a portfolio strategy consistent over time is *seeding*. When creating the new population to optimize the portfolio selection for a time period i, we introduce some individuals from the result population for time period $i-1$. These new

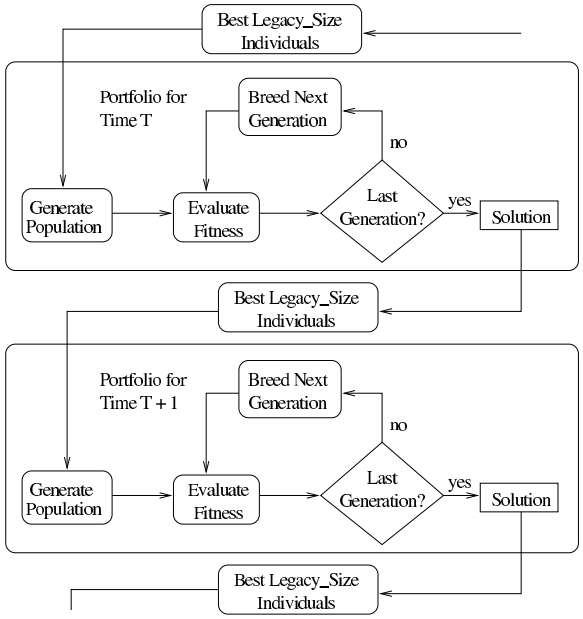

Fig. 1. Description of the Seeding method.

members are copied into the first generation of the new population, and from then on act as if they were normal, randomly generated individuals (see Figure 1).

In practice, this technique generates a bias in the evolutionary search towards the region of the search space that contained the solution in the previous time period. Unless the return values changed very drastically, the seeded individuals will have a fitness value slightly higher than most randomly-generated individuals, and will reproduce more, leading the population to focus its search on the area of the previous winner.

We added a *legacy_size* parameter to our GA system. Its value indicates the number of individuals from the previous population that are copied to the new population. However as will be shown later, we found that the number of seeded individuals is much less important than the existence/absence of seeders.

D. Objective Sharing

The second technique we introduce to the GA system in order to improve the Portfolio optimization over time is *Objective Sharing*. This is a modified version of the lexicographic ordering [2].

We introduce a second objective to the system: the Euclidean distance between the individual (current solution), and the previous best solution. The Euclidean distance between two solutions here is calculated in the usual way:

$$d(a,b) = \sqrt{\sum_i (w_i^a I_i^a - w_i^b I_i^b)^2} \tag{7}$$

Where w_i is the weight of the i-th index in the portfolio, and I_i is the 0 or 1 value which indicates the presence or absence of that particular index in the portfolio.

Once we have this new objective measurement, we have to change the evolutionary process to take it into account. At each generation, one of two fitness measures is used to evaluate the population, with a probability given by the parameter $p_{os} > 1$. The main fitness measure, Sharpe Ratio, is chosen with probability $P_{sr} = 1 - (1/p_{os})$, and the secondary fitness measure, Euclidean Distance, is chosen with probability $P_d = 1/p_{os}$.

Once one of the two objectives is chosen, selection and breeding happens the same way as described before, and another objective is then chosen for the next generation. In this way, the population is directed towards both objectives. The value of p_{os} can be adjusted to determine the relative priority of both the objectives.

IV. Experiments

In all the following experiments, we used similar values for the evolution parameters, which were found to give acceptable convergence rates. The evolution parameters used are listed in Table I. The difference between Generations and Population size is due to the larger data size of the NIKKEI dataset. The difference between training lengths is reasoned below.

The first three sets of experiments study the effects of some parameters in the proposed system. The last set compares the different techniques with Simple GA for portfolio optimization, and economic indexes.

A. Datasets

We choose 2 datasets from our experiments to analyze in this work. Both are groups of assets belonging to well known indexes. Both datasets are very large, leading to the difficult problem of optimizing hundreds of real valued variables at once.

Our goal was to see whether we could use the proposed system to build portfolios that consistently beat the indexes over a long period of time. The stock value for these assets were obtained from public sources, and the return information was calculated from the monthly adjusted closing prices by using the logarithmic returns' formula (eq. (8) below, where p^k is the price of the asset at time k, and r^k is the return at time k).

$$r^i = ln\left(\frac{p^i}{p^{i-1}}\right) \tag{8}$$

The first dataset is a subset of the NASDAQ 100 index. We've chosen 89 assets, out of the 100 that compose the index, which have been on the market for more than 72 months (the 6 year period from Nov/2000 to Oct/2006).

The NASDAQ index is composed by a wide variety of technology industries, with very disperse levels of maturity and economic behavior. With a high number of assets, it becomes difficult to apply traditional balancing techniques. These traditional techniques usually require selecting a subset of the assets that have some agreed upon property - for example the highest expected return - and then run the balancing technique upon this subset. Our method, however, showed itself robust for this dataset without this selection step.

TABLE I
Evolutionary Parameters.

Name	Value NASDAQ	Value NIKKEI
Generations	60	100
Population Size	200	400
Training Length	11	6
Initial Fill Ratio	0.2	0.2
Tournament K	10	10
Elite Size	10	10
Crossover Rate	0.8	0.8
Mutation Rate	0.05	0.05

The second dataset is a subset of the NIKKEI 225 index. We've chosen 205 assets, out of the 225 that compose the index, which have been on the market for the period of 106 months (from Jan/1998 to Dec/2006).

The NIKKEI index is composed of companies in many different fields of the Japanese stock market. There is a wider variety of assets in this group than in the NASDAQ dataset - ranging from technological industry to fishing companies. However, the NIKKEI index suffered from a bubble burst between 1999 and early 2005, so the value of its component assets suffer heavy drops followed by sudden rises. This makes for a challenging environment for technical trading.

For both datasets, we use the last 53 months of the period as the evaluation data. The length of the training data was decided empirically, according to the experiments described in the next subsection.

B. Training Length

The Training Length parameter indicates how many months prior to the current date will be taken into consideration when calculating the expected return. We tested a range from 3 to 30 months in both the NASDAQ and NIKKEI datasets, running the Simple GA algorithm to decide the best training length for these experiments.

The results can be seen at figures 2, for the NASDAQ dataset, and 3 for the NIKKEI dataset. The values in the figures are the average total profit out of 30 runs of the algorithm (with different random seeds).

In the NASDAQ dataset, we have two peaks, at the points of 11 and 18 months. In the NIKKEI dataset, we have a peak at 6 months, and then the results quickly lower. We have observed that the training length value is a problem-dependent parameter. The values used for the reported experiment are showed on table I.

The proper value for the training parameter needs to be chosen carefully. There are local optima and the result's quality drops sharply for badly chosen parameters. Based on our experiments, the method we found best for choosing the Training Range parameter is to first find the best training

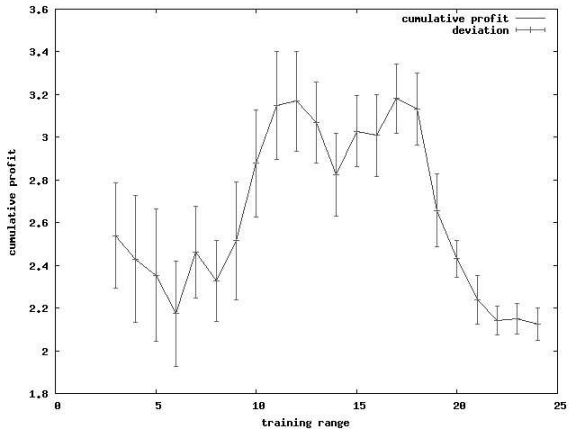

Fig. 2. Total return as a function of training length for NASDAQ data. Best profit and Sharpe Ratio is also at 11 months.

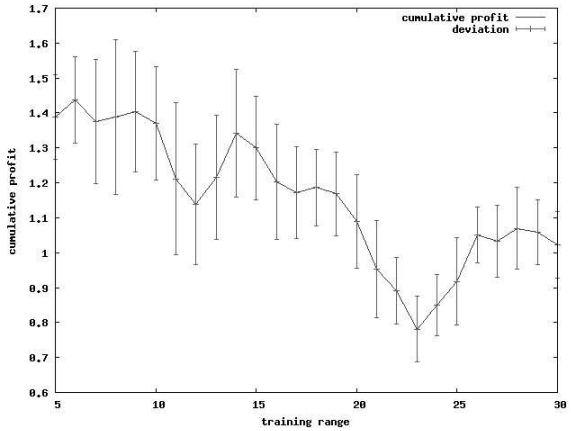

Fig. 3. Total return as a function of training length for NIKKEI data. Best profit and Sharpe Ratio is at 6 months.

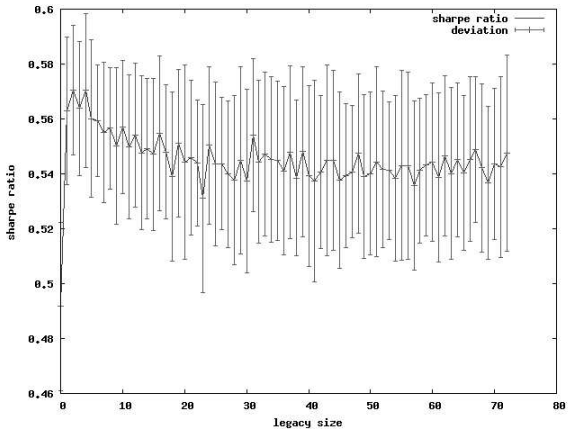

Fig. 4. Sharpe Ratio time of the GA optimized portfolio as a function of the legacy parameter for the NASDAQ dataset. The lowest value happens when legacy is zero, then we see a small drop for legacy values above 5.

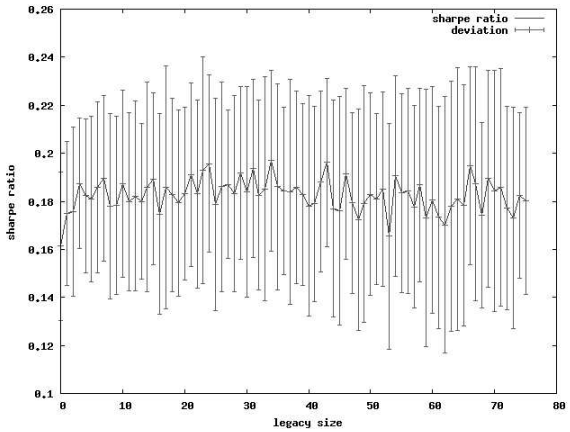

Fig. 5. Sharpe Ratio time of the GA optimized portfolio as a function of the legacy parameter for the NIKKEI dataset. The lowest value is also when legacy is zero, but we see a smaller drop than when compared with the NASDAQ dataset.

length of a subset of the known training data, and then using this value to manage the portfolio for a time period.

C. Seeding Analysis

The Population Seeding technique introduces a new parameter to the Genetic Algorithm, the *Legacy_Size*, which determines how many individuals from the previous genetic search should be included in the new population.

As the number of assets available to a portfolio increases, the search space for the optimal portfolio also increases exponentially, making the problem harder. The Population Seeding technique is a heuristic which says that "the last $legacy\_size$ solutions are probably good enough to be used again", providing a head-start for the new population. The new population will begin its search where the last population was found to be successful.

To check this hypothesis, we tested a Legacy Size from 0 to 30% of the population size for both NIKKEI and NASDAQ datasets, and the results can be seen in figures 4 and 5.

These results show us that our initial hypothesis isn't entirely correct. For both the NIKKEI data and the NASDAQ data, there is a significant difference between Legacy Size $=$ 0 and Legacy Size $>$ 0. The average cumulative profit and Sharpe ratio is much lower when Seeding is not used. However, when seeding is used, the effect of the legacy parameter value over the results quickly stabilizes.

These results suggest that Seeding has the following effect in the algorithm: the seeded individual will have a fitness value higher than that of the randomly generated individuals, and then soon dominate the population, directing it towards its region in search space. Since the seeded individual quickly dominates all the randomly generated ones, additional individuals will not have as much of an effect as the first one.

TABLE II
SHARPE RATIO AND DISTANCE AS A FUNCTION OF P_{os} PARAMETER.

P_{os}	Sharpe Ratio	Distance
None	0.55	0.909
2	0.397	0.294
3	0.450	0.309
4	0.497	0.374
5	0.521	0.427
6	0.535	0.661
7	0.542	0.792

D. Objective Sharing Analysis

The Objective Sharing technique introduces the parameter p_{os} to the Genetic Algorithm. Its value determines the rate at which one of the two goals will be chosen for every generation.

We expect that a lower p_{os} value will result in a portfolio strategy with smaller distances and returns, and that as we raise this parameter, we can find different degrees of compromise between the two measures until an acceptable solution is found.

Table II illustrates the results achieved when applying the algorithm with different values for P_{os} on the NASDAQ dataset. When P_{os} is 2, we have the largest loss and the smallest distance. As we raise the value of P_{os}, we get smaller gains of Sharpe Ratio for larger losses in distance. The results reflect our expectations, as we can see the trade-off between Sharpe Ratio and average Distance, as we increase P_{os}. Results for the NIKKEI dataset were similar.

From these particular results, we can see that changing the value of 1 to 2 will result in a rather great increase of returns for a small increase in the difference. After that, increasing the value of the parameter will increase the difference between the positions for smaller increases in the returns. This kind of information may allow the analyst to tune the algorithm according to his particular cost structure, by balancing the two goals.

E. Comparison with Simple GA

To evaluate whether the proposed methods can achieve their objectives, we compare them with a simple GA algorithm, and with the historical values of the indexes. The index value represents the average trend of the component assets. It is an indicator of whether the suggested portfolio strategy managed to "beat" the market, by picking the assets with above-average performance for the period. The simple GA is used to compare the proposed techniques with previous works.

We choose 5 combinations of the two techniques proposed in this paper, and ran them on the NASDAQ and NIKKEI dataset 30 times each with different RNG seeds, averaging the results. *Seeded GA* is the Simple GA algorithm with only the addition of the seeding technique. We use set the parameter $legacy\_size$ to 1. *SOx GA* is the Simple GA with the Objective Sharing technique, with parameter p_{os} equal to x. The values of 2 and 5 were the smallest and highest values that showed a significant difference in results from other systems. *Seeded SOx GA* includes both modifications to the system at the same time.

We noticed that there was no significant change in processing time for any of the combinations during the experiment. As can be seen from the heuristics descriptions, neither of the two are computationally intensive.

The results of the experiments are summed up on Table III. INDEX is the result for the financial index corresponding to that dataset. SO2 and SO5 refer to the Objective Sharing modification with $p_{os} = 2$ and 5, respectively. Seeded refers to the Population Seeding modification.

Cumulative return, and Sharpe ratio are measures of the "efficiency" of the portfolio strategy. They should be compared with the values given for the index. Cumulative return is how much the portfolio's value increased with relation to its initial value. Sharpe Ratio indicates the risk - for similar cumulative returns, a higher Sharpe Ratio indicates a smaller amount of risk.

Distance is the average distance between portfolio positions through the strategy. It is a linear measure of the cost associated with that strategy. Higher value means that the portfolio position changes more during the period, and thus there will be a higher associated cost. The final cost will depend on the volume of trade, and policies of the trading company.

Comparing the GA methods, we notice that their performances relative with one another were consistent in both datasets, while their performance with relation to the index were quite different in each dataset. We'll first examine the relative performance, and then address the index performance.

Seeded GA dominated Simple GA in both datasets, with higher Sharpe Ratio and Cumulative Returns, and lower average distance. This shows that while we introduced the seeding technique to reduce distance, it has also managed to bias the search towards a better region of the search space.

The Objective Sharing results were as expected. SO2 had a smaller Sharpe Ratio when compared with Simple GA, in exchange for a much smaller Average Distance. Interestingly, the Cumulative Return in the NASDAQ dataset remained near the same, which means that sometimes the SO algorithm may trade only risk for distance, instead of return.

When we combined the two methods, mixed results were obtained. In the NASDAQ dataset, Seeded SOx dominated the SOx methods, while the opposite happened in the NIKKEI dataset. We supposed that the difference can be explained because of the bubble behavior of the NIKKEI assets. With sudden changes on the direction of the market, information from previous successful portfolios might become damaging if not modelled properly in a bubble environment. However, more testing is needed to fully understand this phenomenon.

Figures 6, 7, 8 and 9 give us a better image of metrics in table III. We compare the methods with best overall results with simple GA and the index values.

TABLE III

COMPARISON OF THE DIFFERENT METHODS. 30 RUNS ARE PERFORMED, AND THE AVERAGE RESULTS OF THE BEST INDIVIDUAL FOR EACH METHOD IS REPORTED. THE VALUE BETWEEN PARENTHESIS IS THE STANDARD DEVIATION OF THIS AVERAGE.

Dataset	NASDAQ			NIKKEI		
Method	Sharpe Ratio	Distance	Cum. Return	Sharpe Ratio	Distance	Cum. Return
Index	0.239	n/a	1.721	0.216	n/a	1.562
Simple GA	0.486 (0.0006)	1.477 (0.004)	2.82 (0.02)	0.168 (0.001)	8.51 (0.5)	1.439 (0.02)
Seeded GA	0.55 (0.0009)	0.909 (0.012)	3.14 (0.01)	0.181 (0.001)	8.21 (0.49)	1.49 (0.01)
SO2 GA	0.425 (0.0011)	0.429 (0.002)	2.85 (0.023)	0.154 (0.0017)	1.32 (0.06)	1.376 (0.002)
Seeded SO2 GA	0.397 (0.002)	0.294 (0.003)	2.59 (0.01)	0.125 (0.002)	1.19 (0.07)	1.28 (0.015)
SO5 GA	0.492 (0.0007)	0.818 (0.0009)	2.89 (0.001)	0.174 (0.0009)	2.55 (0.09)	1.446 (0.006)
Seeded SO5 GA	0.521 (0.0008)	0.42 (0.0039)	3.12 (0.001)	0.156 (0.0015)	2.48 (0.11)	1.38 (0.01)

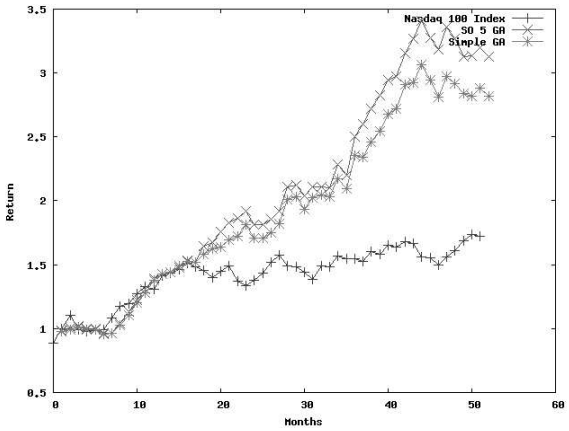

Fig. 6. Cumulative returns for the NASDAQ dataset. We compare the Simple GA and Seeded SO5 GA with the index value. The GA methods use the index stability to pick the most profitable assets, and slowly raise the cumulative return.

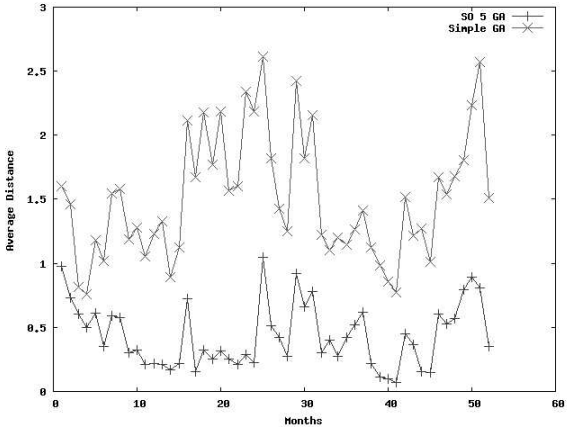

Fig. 7. Distance values for the NASDAQ dataset. The graph shows the distance between the portfolio held in the current month and the previous portfolio. Simple GA shows much more spikes on the changes than SO5 Seeded GA.

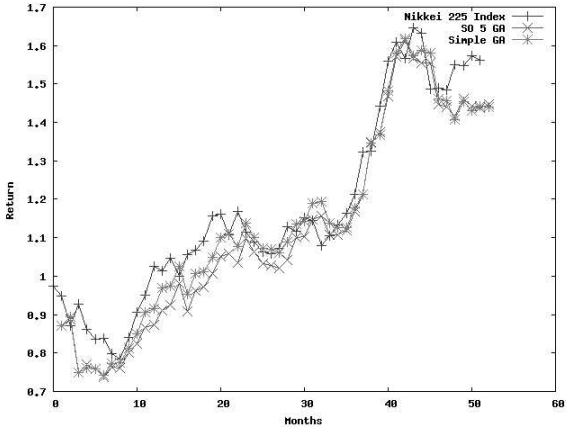

Fig. 8. Cumulative returns for the NIKKEI dataset. We compare the Simple GA and SO5 GA with the index value. In the sharp drop in month 2 both methods make a wrong decision, and have to play catch up with the index. Around the 45th month, another sudden drop makes the portfolio cumulative return to sharply drop

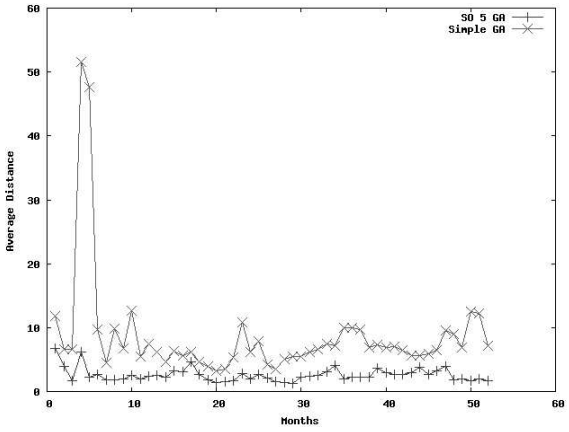

Fig. 9. Distance values for the NIKKEI dataset. Simple GA show extreme changes during the first drop, while SO5 manages to recover using a much simpler change in the portfolio.

The NASDAQ index was stable during the studied period, showing a very light "bullish" (raising) behavior. In this environment, the GA was able to pick the best performing assets and build a portfolio with good returns. The NIKKEI index, on the other hand, was observed during the later phase of its bubble stage, with steep rises and falls alternating. We observe that the system has difficulty dealing with these extreme movements, but still manages to "catch up" with the index.

V. Discussion

We have tested our methods against historical data, and confirmed that they can be used to reduce the distance between positions held over time without reducing the expected return too much.

By manipulating the parameters of the Objective Sharing method, we could get strategies with different emphasis on minimizing distance or maximizing returns. Therefore, we believe that modeling Cost as the distance between portfolios is a promising direction for further improvement of GA-based technical analysis methods. Objective Sharing was successful in greatly reducing the distance (cost) of solutions without reducing its return much.

The Seeding technique, while designed to reduce the distance between solutions, was observed to increase the performance of the population. We believe that this happens because the Seeding method adds information to the population which was not available before - the best solution found for the previous time step.

VI. Conclusions and Future Work

In this work, we have proposed extensions for GA methods of portfolio optimization, in order to add a measurement of transaction costs into the model. We model the cost as a distance minimization goal for the portfolio over time, and suggest two mechanisms to achieve this cost reduction: Seeding and Objective Sharing.

For future work, we intend to do a more through sensitivity analysis on the Objective Sharing, comparing it to Pareto Front-based Multi Objective GA techniques. The main question we want to research is how problem-dependent or problem-independent the parameter P_{os} is. We can then use this information to improve the system so that the user has more control over the evolutionary process of the portfolio strategy.

Also, in this work we tested our method against stable and bubble markets, and observed different behaviors in both. We intend to execute further experiments with historical data of other market environments (bullish market, bearish market, etc), in order to study how our system reacts to them.

An better abstract function for measuring distance needs to be found, in order to relax the assumptions around trading cost introduced by the use of the Euclidean Distance.

Besides considering it as an objective metric, there is another way to look at our Distance proposal for cost modelling. The Distance between two portfolios can also be thought of as a measure of genetic diversity. That's because the genotype and phenotype of an individual, after the normalization step, are the same. When we reward close distance from previous successful individuals during the evolutionary step, it can be thought of as if we were curbing genetic diversity. So a different way to proceed from this work is to study whether methods that guarantee or inhibit genetic diversity could be used along with the distance model for cost.

References

[1] S.-P. Chen, C. Li, S.-H. Li, and X.-W. Wu. Portfolio optimization model with transaction costs. *Acta Mathematicae Applicatae Sinica*, 18(2):231–248, June 2002.

[2] C. A. Coello Coello. A Short Tutorial on Evolutionary Multiobjective Optimization. In E. Zitzler, K. Deb, L. Thiele, C. A. C. Coello, and D. Corne, editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 21–40. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001.

[3] C. Fyfe, J. P. Marney, and H. Tarbet. Risk adjusted returns from technical trading: a genetic programming approach. *Applied Financial Economics*, 15:1073–1077, 2005.

[4] C.-M. Lin. An effective decision-based genetic algorithm approach to multiobjective portfolio optimizaton problem. *Applied Mathematical Sciences*, 1(5):201–210, 2007.

[5] D. Lin, X. Li, and M. Li. A genetic algorithm for solving portfolio optimization problems with transaction costs and minimum transacton lots. *LNCS*, (3612):808–811, 2005.

[6] H. Markowitz. Portfolio selection. *Journal of Finance*, 7:77–91, 1952.

[7] H. Markowitz. *Mean-Variance analysis in Portfolio Choice and Capital Market*. Basil Blackwell, New York, 1987.

[8] F. Streichert, H. Ulmer, and A. Zell. Evolutionary algorithms and the cardinality constrained portfolio optimization problem. In D. Ahr, R. Fahrion, M. Oswald, and G. Reinelt, editors, *Operations Research Proceedings*. Springer, September 2003.

[9] N. Svangard, P. Nordin, and S. Lloyd. Using genetic programming with negative parsimony pressure on exons for portfolio optimization. In R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, editors, *Proceedings of the 2003 Congress on Evolutionary Computation CEC2003*, pages 1014–1017, Canberra, 8-12 Dec. 2003. IEEE Press.

[10] X. Yang. Improving portfolio efficiency: A genetic algorithm approach. *Computational Economics*, 28(1):1–14, 2006.