Application of Metaheuristics Algorithms and Signed Graphs to Portfolio Turnover Management

C. Perina, N. Buckley, and A. K. Nagar

Abstract—**Portfolio turnover has taken an important place in portfolio management because of its impact on the trading cost. There are few methods to assess the probability of a portfolio turnover, the most famous one being the representation with signed graphs and even fewer ways to extract from a range of assets the portfolio with the smallest probability of turnover. We use the signed graph method combined to several optimization algorithms to solve this problem. We demonstrate the efficiency of our methods with the data provided by an American asset management company and show how it is possible to extract from a range of assets the portfolio with the lowest turnover probability.**

Index Terms—**Graph theory, portfolio turnover management, metaheuristic algorithms.**

I. Introduction

Risk management is one of the main areas of portfolio management. However, when speaking about portfolio and risk-management, many forget a very important notion: Portfolio turnover. The turnover is a measure of how frequently assets are bought or sold. This is an important aspect that shouldn't be forgotten since portfolio turnover brings extra expenses because of the taxes and the trading cost. Some researches already showed that picking portfolio based on risk alone can result in unstable portfolio and therefore high expenses [1]. It is therefore important when building a portfolio to take into account not only some notions on risk management such as the variance or the presence of hedging assets but also the probability of portfolio turnover.

It has already been demonstrated that we could estimate the balance of a given portfolio, however, building portfolio and checking afterwards the probability of its turnover isn't an efficient way of building a portfolio. It is far more useful to use a method to directly extract from a range of assets a portfolio with very low chances of turnover.

To reach this result, we combine the Balance theory and 3 different optimization algorithms: Genetic Algorithm, Simulated Annealing and Ant-colony Algorithm to find the more balanced portfolio out of a large range of assets.

The advantage of this method lies in its implementation features: The size of the portfolio won't have any significant impact on the computation time. On top of that it doesn't use any forecasting method and has therefore a good accuracy.

This paper is organized as follows: In the next section we will present some related work, then in the 3rd section we will explain the signed graph theory, the Balance theory and how they are used to determine the balance of a portfolio. Afterwards, in the 4th section we will quickly present the different optimization methods used and present their application in this problem. And finally in the 5th section we will use the method on some data and present the results.

II. Related Work

Portfolio optimization is a recurrent topic of research, many researchers such as Connor, Goldberg, Korajczyk or Jorion have developed the risk analysis further (ref [2] & [3]). Some research papers have been written about portfolio turnover. Reference [4] showed that Portfolio turnover was something manageable: They carried out an Empirical study to compare the return and risk on stocks bought, on stocks sold and the risk of a market index to conclude that portfolio turnover isn't always bad for the investor, some of them enhance their portfolio performance with every transaction they make while others suffer losses.

The unexpectedly high frequency of portfolio turnover and its consequences was studied in [5], bringing into light the fact that portfolio turnover is a notion regularly neglected because of “volatile markets, signals from clients, and short-term incentives” and that it can be disastrous to the client. However, despite these results, few researches about the estimation of the likelihood of portfolio turnover have been published.

Some interesting researches were conducted on portfolio turnover, for example in 2005, Kahn and Shaffer noted in [6] that reducing portfolio turnover is an efficient way to reduce transaction cost implied by portfolios with many assets and, in [7], Qian, Sorensen and Hua conducted researches on portfolio management while taking into account portfolio turnover: They maximized the Information Ratio under portfolio turnover constraints and determined a relation between portfolio Turnover and autocorrelation. However, they worked with an estimated value of the turnover, calculated with changes in forecast. They didn’t try to avoid turnover but rather to find a good portfolio with a turnover not too imposing. Our purpose is to avoid turnover and will rely on an estimation of portfolio turnover probability.

Last year, an efficient method to estimate the probability of portfolio turnover, based on Frank Harary's work was presented in [1].

Frank Harary's work is the seminal of the analysis of the balance of a signed graph. His work already showed that it was possible to represent many things using the signed

Manuscript received December 24, 2016; revised February 15, 2017.

C. Perina is with the Department of Mathematics and Computer Science, Faculty of Science, Liverpool Hope University, Hope Park, Liverpool, L16 9JD, United Kingdom C. Perina is also with Department of Mathematics, Institut National des Sciences Appliquées de Rouen, France (e-mail: cedrik.perina@insa-rouen.fr)

N Buckley and A. K. Nagar are with the Department of Mathematics and Computer Science, Faculty of Science, Liverpool Hope University, Hope Park, Liverpool, L16 9JD, United Kingdom.

graph theory in [8]. In this case, every vertex represents an asset and the edges represent the correlation coefficient between the assets.

Frank Harary also invented the Balance Theory in [9], showed the way to use it to "investigate complex structures of inter-related entities", and even used them to analyze international relations between countries. It is here relevant to represent a portfolio with this method; it enables to determine the stability of the portfolio by studying the balance of the associated graph since the Balance Theory states that an unbalanced graph will undergo changes which would lead in this case to portfolio turnover.

In [8] this representation was used to work on risk management. They showed that the stability of a portfolio can be improved just by adding or switching some carefully chosen assets.

Reference [1] used the signed graph representation and the balance theory to present a method to estimate the percentage of balance of a portfolio. They took every 3 vertex size graph that can possibly be extracted from a portfolio of asset and determined how many of these sub graphs were balanced. This method is efficient and presents numerous computational advantages; however it only enables to estimate the balance of a given portfolio. Our method goes further; we use their work on the estimation of the likelihood of portfolio turnover and mix it with some metaheuristics algorithms such as the Genetic Algorithm to improve the numerical value representing the quality of the portfolio (determined thanks to the probability of turnover of the portfolio). With this method, we found a way to extract from a large number of assets an almost perfectly balanced portfolio and to enable the improvement of the portfolio turnover management while taking into account some notions of risk-management.

III. Preliminaries

A. Signed Graphs and Portfolio

A signed graph consists of a finite nonempty set of vertices and a set of edges labeled as positive or negative. Harary proved that a signed graph is balanced if every cycle in this graph has an even number of negative edge, and is unbalanced if there exist at least one cycle with an odd number of negative edge. This implies that if a 3 vertex size sub graph is unbalanced, any graph containing this sub graph will also be unbalanced. Therefore the analysis of the balance of a graph can be reduced to the analysis of every 3 vertex size sub graph within the graph. This is a huge computational advantage since no matter how big the portfolio is, the program will never have to deal with any graph of more than 3 vertexes.

To represent a portfolio with the signed graph theory, we use the asset as the vertices and the correlation coefficient between the assets as edges. We choose two thresholds, if the correlation coefficient is above the positive one, then the edge is positive. If the correlation coefficient is lower than the negative threshold, the edge is negative and finally if the correlation coefficient is between the two thresholds then we consider that there is no edge between the vertices, we can give this edge the value of 0.

A 3 vertex size graph with at least one edge equal to 0 is considered to be balanced since it isn't a cycle. Hence, it is very easy to determine if a 3 sized graph is balanced: When we multiply the value of the 3 edges of a 3 vertex size graph, if the result is negative the graph is unbalanced, otherwise, it is balanced. *N*, The total number of 3 vertex size graph within a portfolio of k assets is given by the binomial coefficient:

$$N=\binom{K}{3}$$

Then by studying the *N* different 3 vertex size graph feasible with the portfolio, we find the number *nb* of balanced graph. The percentage of balance of the portfolio that will give us an estimation of the likelihood of a turnover is *(nb×100)/N*. With this method demonstrated in [1] it is possible to find the percentage of balance of a portfolio of any size. However the problem now is to extract the portfolio with the Best percentage of stability from a large range of assets

B. Metaheurstic Algorithms

To find this best percentage, metaheuristic Algorithms are a perfect match since these optimization algorithms are often used to find an optimum amongst a large number of solutions. We work with 3 of the most famous metaheuristic algorithms: The Genetic Algorithm, the Simulated Annealing and the ant-colony algorithm. Here is a brief presentation.

The Genetic Algorithm first evaluates the quality of every individual thanks to a fitness function. This function returns a numerical value for each individual and will therefore enable us to spot the best individuals. Then we select the individual that we will use for the reproduction and the creation of the next generation of individuals. Two parents are chosen among the selected individuals, and with a combination of their genes new individuals are created, this is the cross-over. The children obtained are considered as a part of the «next generation ». We carry out as many cross-over as necessary to have as many individuals in the new generation that we had in the previous one. Finally there can be some mutation; some genes of some individuals can be randomly modified.

The purpose of the mutation is to avoid getting stuck at some local peak. Once the new generation is «completed », the iteration is over and the next one begins. We repeat this process as many times as wanted or just until the value returned by the fitness function for one of the individual reach a previously chosen value.

The Simulated Annealing observes the following steps:

1) A temperature is initialized and a random solution has to be generated.
2) A function is used to determine the worth of this solution (like the fitness function of the Genetic Algorithm). Let's call its value *result1*.
3) A neighboring solution is created and its worth is determined thanks to the function. Let's call its value *result2*.
4) A variable Delta is introduced. *Delta=result2-result1*

Since we are looking for a maximum: If $Delta \geq 0$ then

result2 ≥ result1
The neighboring solution is then better, we keep it and it becomes our new «first solution». (*result1:=result2*)

If *Delta<0* then *result2<result1*. The probability to accept the neighboring solution as the first solution becomes *exp (-Delta/Temperature*). That is the reason why the initial temperature should be high to let the program explore the entire space of solutions in at first before cooling down.

1) Then we repeat the step 3 and 4 as many times as chosen or until we find a suitable solution.
2) We decrease the temperature according to a chosen function and run again step 3-4-5 until we reach a previously chosen threshold.

The ant-colony Algorithm has different variants and can therefore take various forms but is often represented as follows:

The ants navigate from a nest to look for some food. They all move on their own and let some pheromone on their path. The more pheromone there is on a path, the more attractive this path become. At first the navigation to the food is random but logically, during the same amount of time, more ants would have passed through the shortest path to the food than through the other paths. As a consequence, there will be more pheromone on the shortest path therefore even more ants should follow that path at the next iteration. After every iteration, the quantity of pheromone on the shortest past increases. Of course the pheromones aren’t everlasting, after some times, it disappears.

IV. Proposed Method

Now that we have a way to estimate the probability of a portfolio turnover thanks to the signed graph theory and 3 different metaheuristics algorithms really useful to find an optimum, it's time to use all of these notions to find the best portfolio (in term of balance) among a large range of asset.

A. Metaheuristic Solution to Portfolio Management

The first method uses the Genetic Algorithm:

We start by randomly generating as many portfolios of k different assets as possible. This number is given by the result of the Euclidean division of *n*, the number of assets by *k*. Every portfolio of k assets would represent an individual and the assets would be the genes. For the fitness function we use the function that gives the percentage of balance of each portfolio of a table of portfolio and added some modifications to take into consideration risk management:

Algorithm 1 Percentage of balance
```
$nbStable := 0;$
For $j = 1$ to $p - 2$
  For $z = j + 1$to $p - 1$
    For $l = z + 1$to $p$
      If $((Correlation[j, z] * Correlation[j, l] * Correlation[z, l]) \geq 0)$
        then $nbStable := nbStable + 1;$
      End If
    End For
  End For
End For
$nbgraphes = factorial[p]/(factorial[p - 3] * (factorial[3]))$
$percentage = nbStable * 100/nbgraphes$
```

As you can see on the Algorithm 1, this function works with the previously explained signed graph method: *Correlation* is a matrix which contains the value of the edge between the assets (1 , 0 or -1 depending on their correlation coefficient). Just as explained previously, they are multiplied and if the result is positive or equal to 0 then the graph is balanced. Using this method the balance of every 3 vertex size graph is analyzed and the percentage of balance is calculated. We added the calculation of the variance and made it carry weight in the fitness function.

If all assets are positively correlated and something goes wrong, then all the value of all assets will go down and the loss will be huge. That is the reason why it is wise to have some asset negatively correlated to serve as a hedge: it's the hedging asset: For every portfolio we check whether they had a hedging asset: Since the priority is to create balanced portfolio we don't to give too much credit to the presence of a hedging asset. We just grant a bonus of 5 point for every portfolio with a hedging asset.

In the end, the fitness function returns for each portfolio: the percentage of balance minus the variance and if the portfolio has a hedging asset, 5 points are added to the result. The balance still has the biggest influence on the result but these modifications can make the difference between two equally stable portfolios and introduce some notions of risk management.

For the selection part, we divide the new generation in 4 parts: A quarter is composed of the most stable portfolio previously found thanks to the fitness function. Another quarter is made of random portfolios, to improve the genetic diversity and avoid finding a solution that would only be a local maximum and the other half of the group is composed of the «children» of the first half, obtained thanks to cross-over: We take 2 of the previously selected portfolio, we split them in half and regroup them in a different way to create 2 new pairs: that is the most simple crossover method. Please find below a representative drawing (see Fig. 1).

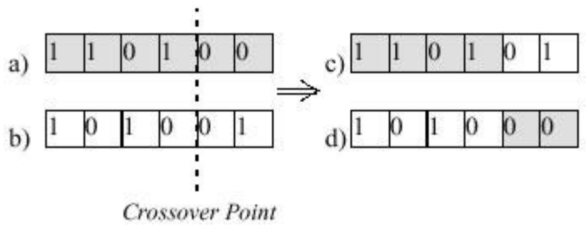

Fig. 1. Crossover method.

To avoid the repetition of one asset within a single portfolio during this crossover, the program check if there is any repeated asset and if there is one, he is replaced. We chose to replace it with a random asset to improve the diversity: we believe that there isn’t such thing as a perfect portfolio, the choice of a portfolio depends on the objectives, the financial means and the boldness of the investor. A less stable but more promising (concerning the incomes) portfolio may be considered as more interesting by some people. On top of that, there can also be more than one portfolio with a perfect balance.

For all of these reasons, we want to create a genetic algorithm that won't converge to one excellent portfolio but rather give different really good portfolio and afterward, the choice of the most fitted portfolio would be up to the investor. It is however strongly advised to avoid making too much mutation because it could lead to a random program so if the initial range of assets is divided into *p* portfolios we create *p/8* mutations at every generation. For the Simulated Annealing, we create a portfolio of *k* randomly chosen but different assets. Then, we find its worth thanks to a new function that is the same as the fitness function except that it works on a single portfolio.

As you can see in the algorithm 2 below, to create the neighboring solution, we change one asset of the portfolio. A random asset here represented by *temp(s)*, is chosen among the *n* assets, and then we confirm that this asset isn't already present in the portfolio, otherwise we choose another one.

Algorithm 2 Creation of the Neighboring Solution

```
$temp := randperm[n]$
$s := 1$
$Newasset = temp[s]$
$k := 1$
While $(k > 0)$
  $k = 0$
      For $j = 1$ to $nb$
         If $(portfolio[j] = temp[s])$ then
            $k := k + 1$
         End If
      End For

      If $k = 0$ then
         $temp3 := randperm[nb]$
         $portfolio[temp3[1]] := temp[s]$
      Else
         $s := s + 1$
      End If
End While
```

Concerning the probability of accepting the neighboring solution, we just create a random number between 0 and 1. If the number is lower than the probability of acceptance, the neighboring solution is accepted. Otherwise it is rejected. Concerning the temperature we make it start at 20 degree and then be multiplied by *ALPHA=0.5* at every iteration.

Concerning the method with the ant colony, just like the method with the Genetic Algorithm, we divide the range of asset into several portfolios. The different portfolios represent different paths and the assets represent some part of these paths. First, we create some random portfolio and use our usual function to obtain a worth for each one of them. Like previously, this worth depends on the percentage of balance, the variance and the presence of a hedging asset within the portfolio.

Then we calculate the pheromone for every asset: As represented in the algorithm 3, the pheromone of an asset is calculated thanks to the pheromone obtained at the previous generation (starting from the 2nd iteration) and the sum of the results of the function on every portfolio that includes the asset.

Once we found the pheromone, we use it to set the probability of every asset to be chosen when we create a new set of random portfolio: The method used in algorithm 4 is as follow: We filled a table with the assets as follows: If the number of pheromone of the asset 1 reached 100, then 100 squares of the table will be filled with the number 1.

When we create a new portfolio we just randomly pick a square of the table (while avoiding any repetition of asset within one portfolio). After each iteration, we decrease the pheromone of all assets to symbolize the disappearance of the pheromone.

In the end, the program return several portfolio with their percentage of stability, their variance and a variable that shows whether the portfolio possess any hedging asset.

B. Results

We now demonstrate the efficiency of these methods by applying the data provided by an American asset management company: This data comport 90 assets, and the variation of their performance over from the start of the year, 1 week, 1 month, 3 months and 3 years.

We set the parameter of the methods as follows:

The negative threshold is -0.01 and the positive one is 0.01

We want to create portfolio made of 5 assets. This implies that the Genetic Algorithm (and the ant-colony) returns 90/5=18 portfolios of 5 assets. However, as the number of generation (or travels) goes up, the convergence increases. Therefore among the 18 portfolio solutions, some will be identical. Like we previously explained, we believe that diversity is a better option to let the investor choose the portfolio that suits them best. Hence, we settle for 5 generations to get satisfying and diversified portfolio-solution. Since luck plays a role in the results of the algorithm, we launch every method 3 times and keep the mean value of the percentage of balance of all the portfolio-solution obtained after 3 attempts as our solution.

The simulated annealing only returns one portfolio therefore there is no need to be concerned about the convergence. However to fairly compare this method with the other, we choose a shutoff parameter to limit the number of iterations (see Table I).

TABLE I: PARAMETER OF THE SIMULATED ANNEALING

Initial temperature	20 °
Alpha	0.5
Number of iterations	10
Shutoff parameter	1 °

We also fix the number of iterations of the ant-colony algorithm at 5 to be fair to the Genetic Algorithm. After running the test with the three algorithms, we obtain the following results (see Table II).

TABLE II: RESULTS OF THE OPTIMIZATION ALGORITHMS

Results	Genetic Algorithm	Simulated Annealing	Ant colony Algorithm
Balance	99.44%	98.07%	100%
Mean Percentage of portfolio with at least one hedging asset	75.92	66.67	0
Duration of an iteration	0.88sec	0.03 sec	23.98 sec

The Genetic Algorithm almost always returns perfectly balanced portfolio and most of them include a hedging asset. However, the convergence is noticeable, out of the 18 portfolio-solution returned by the algorithm, at least 5 of them are identical. It reduces the number of choice for the investor.

The Simulated Annealing is the quickest algorithm;

however it only returns one portfolio.

The ant-colony is by far the slowest but it is most likely due to the method used to handle the probability of the selection of every gene. This method is the most efficient if we only focus on the balance. However because of the numerous factors which carry weight on the pheromone, the presence of a hedging asset has become negligible. That's the reason why not even one of the 54 portfolio-solutions created had a hedging asset: In this case, the risk management almost don't weight in the constitution of the portfolio-solutions. This method also offers a larger diversity than the genetic algorithm.

V. Conclusion

In this paper, we propose several optimizations methods combined to the signed graph theory to extract from a range of assets some portfolio with the lowest chances of portfolio turnover. We illustrate it by applying our methods to some data from an American asset management company about 90 assets.

Our results show that we can extract perfectly balanced portfolio from a range of asset. However, it also demonstrates that focusing on the balance can cause some carelessness in the risk management like in the case with the ant-colony algorithm.

These methods advantages lies in the fact that they are easy to implement and allow the user to work with a large range of assets without any huge increase of the computation time. It also avoids any forecasting errors. However, this method shows its limits: As soon as we tried to introduce more notions of portfolio management (like the hedging asset), it became hard to find a portfolio both balanced and secured as showed in the Ant-colony algorithm. Portfolio management isn't all about turnover, it is important to take into account more notions such as the risk or the output. Yet, the more features we want to optimize, the harder it becomes to find a great fitness function that would find the best compromise between all of the notions.

To go further in the study of portfolio management, it would be interesting to look for the perfect equilibrium between the optimization of the portfolio balance and the risk management. It would also be interesting to look for an optimization method that would possess the efficiency of the ant-colony algorithm and the speed of the Simulated Annealing while taking into account the risk management like the Genetic Algorithm.

References

[1] B. Vasanthi, S. Arumugam, A. Nagar, and S. Mitra, "Application of signed graphs to portfolio analysis," in *Proc. 2nd Gloal Conference on Business and Social Science*, 2015.

[2] G. Connor, L. R. Goldberg, and R. A. Korajczyk, *Portfolio Risk Analysis*, 2010.

[3] P. Jorion, *Portfolio Optimization in Practice, Financial Analyst Journal*,1992.

[4] W. S. Baumann, E. R. Miller, and E. T. Veit, "Managing portfolio turnover: An empirical study," *Quaterly Journal of Business and Economics*, 2005.

[5] D. Guyatt and J. Lukomnik, "Does portfolio turnover exceed expectations?" *Rotman International Journal of Pension Management*, 2010.

[6] R. N. Kahn and J. S. Shaffer, “The surprisingly small impact of asset growth on expected alpha,” *The Journal of PortfolioManagement*, vol. 32, no. 1, pp. 49-60.

[7] E. Qian, E. Sorensen, and R. Hua, "Information horizon, portfolio turnover and optimal alpha models," *The Journal of Portfolio Management*, 2007.

[8] F. Harary, M. Lim, and D. C. Wunsch, "Signed graphs for portfolio analysis in risk management," *IMA Journal of Management Mathematics*, pp. 201-210, 2002.

[9] F. Harary, "On the notion of balance of a signed graph," *Michigan Math*, pp. 143-146, 1953.

C. Perina is an engineering student at the Institut National des Sciences Appliquées de Rouen (INSA). He was born in 1995 in Tremblay-en-France, C. Perina entered the INSA after getting his scientific high-school degree with honors in 2013 and is currently in fourth year, in the Department of Mathematics where he showed a keen interest in optimization methods.

He carried out a 3 months internship at Liverpool Hope University in 2016 to study optimization methods applied to finance.

Mr. Perina’s paper on metaheuristics algorithms and signed graphs applied to Portfolio turnover management is his first publication.

N. Buckley was born in Liverpool, England, and holds a Ph.D. in computer science from Liverpool Hope University

He is currently a temporary lecturer in the Department of Mathematics and Computer Science, teaching various programming languages and algorithmic design, as well as supervising final year student dissertations

Dr. Buckley has several publications in the specialized area of cryptography known as secret sharing, using various techniques such as optimization algorithms to improve the quality of the reconstructed secrets.

Atulya K. Nagar is the foundation chair and a professor of mathematical sciences, at Liverpool Hope University where he is the dean of the Faculty of Science and has been the head of the Department of Mathematics and Computer Science. A mathematician by training, Professor Nagar possesses multi-disciplinary expertise. He is an internationally recognised scholar working at the cutting edge of theoretical computer science, natural computing, applied mathematical analysis, operations research, and systems engineering and his work is underpinned by strong Complexity-theoretic foundations. He received a prestigious Commonwealth Fellowship for pursuing his doctorate (dphil) in applied non-linear mathematics, which he earned from the University of York in 1996. He holds BSc (Hons.), MSc, and MPhil (with distinction) from the MDS University of Ajmer, India. Prof Nagar has edited volumes on Intelligent Systems, and Applied Mathematics; he is the Editor-in-Chief of the International Journal of Artificial Intelligence and Soft Computing (IJAISC) and serves on editorial boards for a number of prestigious journals. He has an extensive background and experience of working in Universities in the UK and India.