International Journal on Advanced Science Engineering Information Technology

Vol.9 (2019) No. 2
ISSN: 2088-5334

On Tackling Real-Life Optimization Problems

Nadia Abd-Alsabour[#]

[#] *Cairo University, Cairo, Egypt*
E-mail: nadia.abdalsabour@cu.edu.eg

***Abstract*—Most real-world applications are concerned with minimizing or maximizing some quantity so as to enhance some result. This emphasizes the importance of optimization and subsequently the significance of the optimization methods that are able to tackle these real-life optimization problems. There are a number of practical reasons for which traditional optimization and exhaustive algorithms cannot deal with a variety of these real-life optimization applications although there are numerous optimization problems that can benefit from applying these traditional optimization algorithms to handle them. Therefore, their is a need for propsong new optimization algorithms (such as nature inspired optimization methods) and optimize the capabilities of the existing ones (such as hybridization and parallelization) as well. This paper investigates the most recent optimization directions for dealing with the real-life optimization problems with an application to one of the most common and important optimization problems in a variety of financial fields and other fields which is the portfolio optimization problem since it is considered one of the most crucial problems in the modern financial management and has a variety of applications such as asset management and building strategic asset allocation. The computational results were got utilizing benchmark data from the OR library with the use of modern optimization algorithms. In addition, the article highlights the differences and similarities among the utilized optimization methods. In addition, recent advancements to the utilized optimization methods are highlighted.**

***Keywords*—real-world problems; nature-inspired algorithms; differential evolution (DE); particle swarm optimization (PSO).**

I. INTRODUCTION

While there are numerous optimization problems that can benefit from applying traditional optimization algorithms to handle them, there is a number of practical reasons behind which these methods cannot deal with a variety of real-life applications. These methods are mostly local search ones that can not ensure getting the global optimum (except when handling convex and linear problems). This is because their outcomes are on the basis of the initial starting points [1].

Besides, exhaustive algorithms (looking through all of the conceivable solutions) are usually time-consuming and hence intractable. This is because it has been proven that they are not appropriate for tackling complex and large optimization problems such as real-life ones as they do not get optimal results in a reasonable time [2]-[3].

This has motivated the advancement of recent heuristic optimization algorithms (such as recent evolutionary algorithms such as differential evolution) and new features to the current ones (such as hybridization and parallelization).

Heuristic search refers to the possibility of making some smart decisions without considering the whole picture but on the basis of the minimum given information. The term heuristic is utilized for the methods that discover solutions among all conceivable ones without ensuring discovering the best one. Consequently they get roughly close results.

When utilizing a heuristic method for tackling an optimization problem, it is required to tell whether the ideal solution convergence will take place in the closest future (will the present solution get closer to the best one?) or will always be a gap to the best solution? [4].

Metaheuristics are not problem-specific and are approximate strategies that guide the search procedure to efficiently investigate the search space. They get within acceptable time satisfying solution rather than ensuring discovering the best one [5].

A good instance of modern metaheuristics is nature-inspired optimization methods that are a set of novel algorithms whose ideas are motivated from nature. This is because there has been a common belief that nature provides optimal results for a variety of complicated problems. Ideas and concepts existed in nature have been studied in order to propose algorithms that simulating these ideas and concepts and can handle successfully these difficult problems. Nature-inspired optimization methods have successfully handled a variety of real-life optimization problems. Therefore, they have attracted considerable attention from numerous researchers from a variety of domains. Examples are evolutionary and swarm intelligence methods [2], [6].

In the last decades, nature inspired metaheuristic optimization methods such as evolutionary methods and swarm intelligence methods have been extremely well known for tackling complex optimization problems such as complete real-life ones. This is because they have merits over the conventional ones such as they demand less domain-specific information (this feature is vital when tackling numerous optimization problems for which getting domain-specific information is considered intractable). Their ideas came from nature as many systems in nature perform tasks in an excellent manner. This is because they all have a common feature which is searching for the optimum. They have to accomplish objectives and satisfy constraints within which this optimum has to be discovered. Such optimum searching can be structured as an optimization problem (getting the optimum solution that is assessed via an objective function) [7].

Recently, various ways were advanced to these algorithms to increase their capabilities and the number of applications they can handle. In addition, recent evolutionary and nature-inspired methods have been proposed. This is researched in this article with the utilization of the portfolio (a portfolio is a suitable collection of various investments held by persons or organizations since decreasing the investment risk necessitates this diversification in the invested portfolios when making financial decisions) optimization problem since it is required in a variety of financial areas and a well-known and an essential problem in finance and economics.

It deals with distributing financial resources to a selected set of assets such that minimizing the total risk associated with this set as well as ensuring a particular level of returns at the same time. This is because investigation in a portfolio of items is favoured by organizations and persons over the investigation in a single asset because this allows dampening the risk via diversifying the investments without influencing the expected returns [5], [8] - [9].

In this article, we utilize differential evolution (as an example of recent evolutionary algorithms), and particle swarm optimization (as an example of recent nature-inspired algorithms) for tackling the portfolio optimization problem as an example of the most common optimization problems in numerous real-life fields.

Section A addresses the portfolio optimization problem. The fundamentals of the utilized optimization algorithms are addressed in Section B. Section C describes the data utilized in the experiments, and the experiments implemented. Section 3 addresses the obtained results and discusses the experiments' results. The last section finishes up this article and features the possible work for future.

II. MATERIALS AND METHOD

A. The portfolio optimization problem

It is considered one of the most crucial problems in the modern financial management. It is one of the most studied problems in the finance field and the financial mathematics. It has a variety of applications such as asset management and building strategic asset allocation.

The pioneering work to this optimization problem was the work of Markowitz who set the establishments of modern portfolio theory where he worked on the mean-variance model. He considered the profits of the assets as random variables and estimates the risk as the standard deviation for the given dataset. This theory recommends the way investors should enhance their portfolio of risky assets in order to benefit from the diversification impacts since the risk relies upon the pair wise correlation between the risky items and it is not additive. Since his work, numerous investigations have been performed on computational methods for tackling this problem. His work has been widely extended due to the persistent work of numerous scientists.

In practice, this problem may nictitates long time to be tackled. This requires the utilization of recent search methods other than exhaustive ones. Besides, local search algorithms will not be able to provide good solutions as opposite to the differential evolution and particle swarm optimization methods since they are global optimization methods.

There are several forms of this optimization problem. We use in this article the well known and common framework which is the mean-variance framework to which the theory of portfolio optimization is generally associated.

This framework depends on the variance-covariance matrix of the returns. Therefore, we utilized in this paper the covariance (a common statistical calculation) of the returns for all possible combinations of items as a measure of risk. It quantifies whether 2 stocks move in inverse or same direction if it equivalents to a negative or positive value respectively. This can help predict how stocks might move together or not in the future which in turn can help make important decisions such as deciding on the complimenting items in order to decrease the potential risk and increment the potential return. Thus, the risk measure utilized in this article is much better than the standard deviation and the variance that are used as risk measures in some models of the utilized optimization problem. Utilizing them is not accurate since they measure the stability (variability) for one variable only which is not appropriate (because the investor is interested in a set of items as he/she needs to know which will move together). Even if the stability for one variable is required, then it is more reliable to measure the stability (variability) for one variable to use the relative variability which is known statistically as the coefficient of variation (100 * the standard deviation / mean; the smaller the more stable) that can be used also to compare the stability of various variables even if they are with different measurement units because it is independent of them.

This framework aims at constructing the portfolio having the minimum risk with a given minimal return which implies that a diversified portfolio is favoured over a non-diversified one. This can be treated as a multi-criteria optimization problem to be tackled utilizing metaheuristics (are usually very proficient in discovering close-optimal solutions and based on which approximate methods can be proposed) and hybrid methods. This is because these methods are considered the state of the art nowadays in dealing with a variety of real-world constrained optimization problems. In addition, recent advancements in tools and methodologies for proposing (like development frameworks and languages) metaheuristics make them more appealing also as engineering techniques for general problem solving in the industry [3], [5], [8], [10]-[12].

The main disadvantage of this framework is getting enough data for estimating the returns and the risk. Besides, the estimation of covariance (for the risk definition) and return is so sensitive to measurement errors [8]. These are handled in this work by getting benchmark data.

Besides, it needs to be more realistic which can be addressed by including constraints to be able to deal more with practical applications. By this inclusion of necessary realistic restriction, this problem turns to be much harder and impractical computationally which enforces researchers to utilize recent optimization methods to handle it such as nature inspired ones. Notwithstanding, there are a few utilizations of nature-inspired techniques in literature for tackling this problem such as the work of Bacanin and Tuba [5] and Kresta and Slova [13].

The cardinality constraints (for practical causes) force a limit on the total quantity of items (items' no. lies between two limits; this is utilized in this article as this is more practical than the other option indicated in Equation 2) or to have a specified number of items to be incorporated into the portfolio. This is illustrated in Equation 1 and 2 respectively.

$$min \leq \sum_{i=1}^{N} s_i \leq max \quad (1)$$

where:

N is the quantity of the given items. It equals to 31 in the benchmark data utilized in this article,

min is the minimum no. of invested items,

max is the maximum no. of invested items, and

$s_i \in \{0, 1\}$, $i = 1, \ldots\ldots, N$ and it means whether item *i* is invested or not i.e., if it = 1, then item *i* is invested and if it equals 0, then item *i* is not invested.

$$\sum_{i=1}^{N} s_i = num \quad (2)$$

where *num* is the desired quantity of items to be invested.

Involving the cardinality constraints (which is the most difficult component of tackling this optimization problem) is to reduce the transaction costs and the tax or to simplify administrating this portfolio. This is because limiting the quantity of various items in the portfolio reduces the transaction cost that can be so large. In addition, observing the organization's results is simpler and costs less with a smaller number of items in the portfolio [3], [13].

Moral-Escudero et al. [14] proved that when having this constraint which is interested in the real-life applications, this problem will be NP-hard. Thus, numerous methods can't deal with it such as mathematical programming algorithms. In spite of the fact that they were viewed as overwhelmed for managing this problem, we can not utilize them while having this constraint [3], [13]. Another example is the work of Bienstock [15] and Bertsimas and Shioda [16] in which the authors utilized exact methods to tackle it but they did not get the optimal results in a reasonable time.

Therefore, as it is difficult to be tackled optimally, numerous scientists have applied different optimization methods to tackle it and there are not so much work in literature related to this sort of portfolio optimization problem [3].

B. tackling the real-life problems

Almost all real-world problems can be modelled as optimization problems. Consequently, optimization became one of the most essential and applicable research areas in various domains such as computer science, operations research, and mathematics. Recently, nature-inspired techniques have been proposed and tackled effectively a variety of hard and complex optimization problems in various fields. Nature-inspired metaheuristics are metaheuristics that mimic rules and principles from nature [5].

This section addresses the recent directions for tackling real-life optimization problems with an application to the portfolio optimization problem.

B.1 Evolutionary algorithms

In nature, the most complex species are consequences of ongoing evolution over time that began with basic ones which will be developed into newer ones over time where one is more qualified to the changing environments in correlations with the prior ones. The results are the most adjusted species that we find today. Evolutionary methods mimic these concepts and hence they keep showing signs of improvement over time as a result of continuous adaptation and adjustments. They are population based heuristic search methods.

In these techniques, at first, the first generation is created by having all generated randomly individuals (or user-supplied). Then different operators are utilized to create a new one. Since the goodness of the solution varies from a generation to another, a fitness function is used to quantify an individual (in which each bit corresponds to an asset in the utilized optimization problem). The point is to discover the best one that accomplishes the best result.

In other words, these methods evolve a set of solutions aiming to enhance it iteratively by blending solutions and exchanging bits at random. In every generation, such randomly-changed members substitute worse ones after comparing them with the original population and found that they are better [8], [17] – [18].

A recent research direction of these methods is to hybridize them with different methods in order to exploit these 2 strategies. The other one is to utilize them in parallel. Lastly, recent evolutionary algorithms' variants have been proposed such as the differential evolution that is utilized in this article as the first two research directions were investigated in our previous work.

1) Hybrid algorithms

Recently, hybridizing optimization methods together to tackle a given optimization problem turns out to be so noteworthy and gives ideal outcomes over utilizing a metaheuristic individually. A well known type of hybridizing algorithms together is that the utilized method constructs the initial solution that will be refined through the use of a local search method. The reason is with heuristic optimization methods failure in local optimum (the same solution is got again and again) can happen and investigating the candidate solutions will end. This has motivated the use of local search algorithms integrated with the host optimization method to avoid the local minima via growing the neighbourhood of the present solution. Consequently,

improving the investigation ability of the host procedure [12], [19].

Notwithstanding, there are circumstances in which this integration becomes pointless. This is with the optimization problems whose utilized local search efficient polynomial neighbourhood will not have solutions or will get a small number of them and discovering a feasible solution is extremely difficult [20] – [21]. An example is so strongly constraint problems.

On the other side, local search individually encounters discovering good beginning solutions [22].

2) Parallel algorithms

Parallel computing arose as a type of high-performance computing because of the large decrease in the cost and wealth of computational resources (hardware and software) since the last decade of the past century. In this scheme, a set of calculations are executed concurrently on the basis of a huge problem can be separated to littler problems which will then be tackled simultaneously [23].

Recently, several parallel methodologies have been introduced to these methods to avoid the premature convergence that they may encounter. They perform parallel computing via assigning unused processors with the sub-populations. They separate a huge population into smaller ones and execute concurrent irregular searches among them [24].

The coarse-grained methods have numerous populations interconnected in a particular topology performing sparse migrations of individuals between its islands [25]. It has been demonstrated that they have favoured performance over the traditional ones [23] and utilizing them frequently prompts favoured performance and quicker methods as well. This is on the grounds that they can converge rapidly since the number of individuals on any islands is smaller than the number of individuals that the ordinary genetic methods use. Besides, each island investigates different parts of the whole search space and along these lines improving the exploratory stance of these methods [26]-[27].

3) Differential evolution (a recent evolutionary algorithm)

It was proposed in the 1990s [28]. From that point forward, it has risen as a standout amongst the most adaptable and competitive of the evolutionary methods and has tackled successfully a variety of real-life problems. It is considered one of the most efficient evolutionary methods for handling continuous problems. It is a type of evolutionary algorithms that inspires aspects from biology. These are its operations (selection, mutation, and crossover), and the utilization of successive iterations that are generations in nature [3], [23], [29] – [31].

It is a population-based metaheuristic approach. Beginning with uniformly random solutions from the search space, each generation of it works through the identical steps utilized by the standard evolutionary methods. It evolves a set of solutions though a number of generations. In a given iteration, new solutions are generated and assessed. Better ones substitute worse ones. At the end, the best one is outcome [18]. However, these operations are different from those of evolutionary algorithms with the same names. Besides, it is different from them in creating new vectors (solutions are known as vectors in DE) which is achieved via combining several ones with the candidate one.

Mutation here is the process of generating a trial solution (for a given target one; as a trial solution is constructed for every target one) by generating a mutant solution (by mutating 3 randomly chosen solutions excluding the target solution) and then the crossover of it and the target solution. A large value for the crossover probability prompts favouring the mutant solution and a small value produces a trial solution more similar to the target one.

Selection aims at keeping trial or target solution. The basic scheme is to keep the one having better fitness that will *immediately* substitute the other one and becomes eligible to be chosen to constitute the next mutant solution (each solution takes turns as a target or candidate solution). This is considered a crucial distinction from other evolutionary methods as any enhancements can impact other solutions without awaiting the entire population to complete the update.

Its initial iteration is comprised of initialization, mutation, crossover, and selection. Only the last 3 stages are repeated to the ensuing generations that proceed until satisfying the stopping condition [23], [30]. The algorithm is illustrated in Fig. 1.

In contrast to some other evolutionary algorithms, it is very simple and requires a few control parameters (the crossover probability, the scale factor, and the population size). Nonetheless, it shows remarkable performance when tackling various objective functions in terms of computational speed, and the obtained accuracy.

DE has a few varieties depending on the utilized types of recombination operators and solutions, and the quantity of members for which the algorithm computes the mutation values [3].

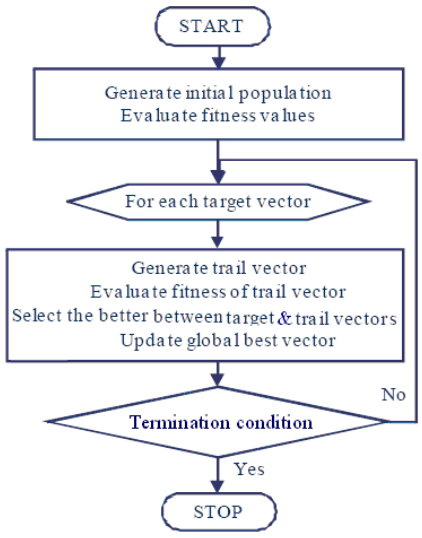

Fig. 1 The DE steps, adapted from [31]

Various recent advancements to the differential evolution have been proposed in the last 8 years [24]. For instance, differential evolution (like other evolutionary) has been parallelized to refine its speed and accuracy on various applications. Another example is the cooperative co-evolutionary differential evolution with adaptive subcomponents that applies in parallel several

decompositions during short learning stages [32]. These researches with differential evolution have prompted expanding its applications in various domains [23].

B.2 Particle swarm optimization

The social behaviour of swarms such as birds, fish, ants, and bees, has been utilized as an inspiring source for proposing artificial intelligent systems known as swarm intelligence systems [5].

An instance of these systems is the PSO that is a successful branch of nature-inspired search methods particularly when handling continuous optimization problems. It is so appealing due to its simple conceptual structure. It was introduced in 1995 by Eberhart and Kennedy simulating the social conduct of the bird flocking, fish schooling, and particles [8], [33] that encouraged conceptual visualization of the search procedure.

It is a population-based metaheuristic. The possible solutions are represented by a population of particles (a swarm of particles is called the population of solutions) which move within a multi-dimensional search space with given speeds (a solution is considered a position in the search space). Its speed, position, and a record of its previous performance define every member that encodes an intersection of all search dimensions. The related location and velocity (the direction into which a solution moves; it is the change in a given solution in a given generation) of every member will be created randomly. The velocity and position are the key features of any member and updating them are main operations. This update necessitates just a basic mathematical operation which is considered an important computational advantage. During the search procedure, the members tend to move toward better parts of the search space. When achieving a new location, the best location of the population and every member are refreshed. At that point, the speed of each member is adjusted. This procedure is repeated until fulfilling the given termination condition.

In other words, in every iteration, the velocity of the individual is adjusted stochastically based on the historical best location of the neighborhood and the particle itself (by adjusting the velocity of a member, a constraint on how it is changed can be forced). This means that, in order to find the ideal solution, every individual changes its searching direction based on the best experience of the other individuals (gbest- known as the social portion) and its own best previous experience (pbest- known as the cognition portion). This is practiced via the fitness function. The algorithm is illustrated in Fig. 2.

The pioneers having the best performers impact particles. In every flight cycle, every individual (as for its present position) is assessed via the objective function so as to quantify the quality of the member as well as define the leader in the sub-groups and the whole one [31], [34]-[35].

From the previous demonstration, it is obvious that the PSO has similarities with the evolutionary algorithms. It has the initialization phase in which the creation of the initial swarm of particles takes places (initialization with a random location and speed). In addition, it has the solution representation phase. This encourages authors to classify it as an evolutionary method [31].

A recent research direction with the PSO is introducing new variants of it (such as fully informed particle swarm [36]) to be suitable for various optimization problems. Besides, incorporating with it other algorithms in order to tackle various optimization problems has been increased recently. As an illustration, Kumar [37] integrated a PSO and a genetic method in order to get well classification rules. Also, Nazir et al. [38] integrated a genetic method with a particle optimization algorithm in order to tackle the feature selection problem [39].

However, the utilization of PSO in general for tackling the portfolio optimization problem is still limited [8] and more particularly the utilization of the previous research directions has not tackled the portfolio optimization problem yet.

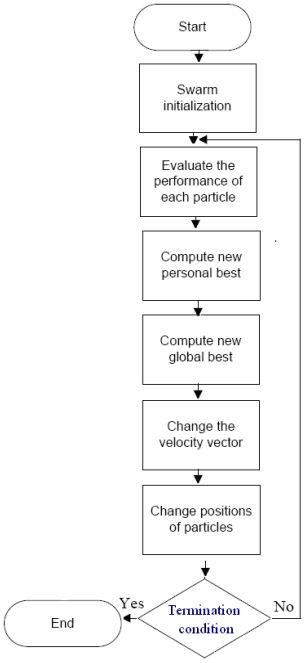

Fig. 2 The PSO steps

C. Computational experiments.

This article examines the utilization of recent optimization methods for tackling real-life optimization problems with a case study of the portfolio optimization problem. The experiments performed are as follows.

C.1 Data

Two experiments were implemented and tested utilizing benchmark data that is taken from the OR library [40]. Table 1 depicts the details of the utilized benchmark data for 31 items. In this table, the first column contains the mean return for each item, and the second column contains the standard deviation of the return for each item.

The correlation between all pairs of the 31 items is provided in Table 2 (supplementary file) [41]. It characterizes the strength of the relationship between 2 stocks as it tells the degree to which these 2 assets move together or not. It is utilized in this article for computing the variance-covariance matrix of the returns of the items that holds the covariance between all possible combinations of

items. The covariance between items *m* and *n* is computed according to Equation 3.

$$\text{Covariance}(m, n) = \rho(m, n) * \sigma_m * \sigma_n \quad (3)$$

where:

$\rho(m, n)$ is the correlation between items *m* and *n*,

σ_m is the standard deviation of the return of item *m*, and

σ_n is the standard deviation of the return of item *n*.

The return is measured by its mean. The total risk that ought to be limited while ensuring a given level of returns is measured by the covariance of returns.

TABLE I
THE UTILIZED BENCHMARK DATA

The mean of the return	The standard deviation
.001309	.043208
.004177	.040258
.001487	.041342
.004515	.044896
.010865	.069105
.001759	.053671
.002594	.046613
.004950	.045492
.007115	.053634
.003186	.046923
.002093	.047043
.005202	.042955
.004489	.039754
.003642	.050224
.003960	.036183
.000141	.038844
.000282	.043980
.000392	.050467
.005294	.058710
.004801	.050000
.002699	.044716
.001879	.038128
.004656	.045625
.003842	.050449
.002690	.053335
.004793	.045450
.003286	.047931
.002338	.036047
.005817	.035848
.001993	.036762
.002380	.039827

C.2 Method

The following systems were developed in the experiments:

1. Differential evolution, and
2. Particle swarm optimization.

These systems were implemented with the utilization of the R language [42]. In the experiments, we set enough iterations for the utilized algorithms in order to overcome any bad setting for their parameters.

In order to make sure that these algorithms work properly, we run them a number of times and get the average of these runs since it is not sufficient with stochastic methods to depend only on one result [18]. In each of the experiments, we recorded the elapsed time to tell the amount of time needed by each of these algorithms.

III. RESULTS AND DISCUSSION

We in this article addressed the recent research direction in handling real-life optimization problems via tackling the portfolio optimization problem utilizing a modern evolutionary algorithm (which is the DE) and a modern nature inspired algorithm (which is the PSO). This was through utilizing benchmark data from the OR library.

The results of the utilized methods are depicted in the following table. The name of the utilized optimization method is in the first column. The objective function's value of the best solution of each utilized method is in the second column. The third column contains the elapsed time (in seconds) consumed by each utilized method.

TABLE III
THE OBTAINED RESULTS

Utilized method	Objective function's value	Elapsed time
PSO	0.0004985774	2.39
DE	0.0004985472	2.41

The attraction of the utilized methods is that they are population-based methods, do not require particular problem-specific information, and can deal effectively with continuous data as they have pulled in large consideration particularly for tackling continuous problems. In addition, both are conceptually so simple, and utilize just simple mathematical operators.

In contrast to DE, in PSO, every member adjusts its move depending on the experience of its neighbors' move and its move experience as well [43]. In contrast to PSO and evolutionary methods, differential evolution creates new solutions through joining a few ones with the candidate one.

Solution representation is utilized in both of the utilized methods. Moreover, both two methods in contrast to evolutionary algorithms do not need ranking the values of the objective functions of the obtained solutions in any stage. This viewed as a crucial computational merit particularly when having large no. of potential solutions.

Parallel algorithms, hybrid algorithms, and hybrid parallel algorithms are recent and successful tools for tackling real-world optimization problems but they were investigated previously in our previous work [44]-[46]. Therefore, they are not utilized in this article. It should be noticed that there are other variants of the utilized optimization algorithms (rather than the utilized ones) that may have different results than the obtained ones. For instance, JADE version of the differential evolution [47] may get high quality solutions because of its diversification capabilities. These capabilities are considered to be incorporated into the utilized version in this article in the future work.

The previous results should not be utilized for comparing between these methods utilized in this article and other results from literature. This is because different setting for the algorithms' parameters, diverse algorithms implementations, and distinctive platforms may lead to unfair comparisons.

IV. CONCLUSIONS

This article utilized recent optimization techniques for handling real-life optimization problems such as the cardinality constrained portfolio optimization problem.

Besides, it provided different research directions related to each of the utilized algorithms. In addition, the differences among the used optimization methods are demonstrated. The point was to deliver a variety of systems to tackle one of the most common real-life optimization problems in the financial fields to direct the researchers to valuable directions. As a research direction, more methods motivated from nature have to be utilized. Besides, the used methods' performance can be optimized in a variety of ways that should be investigated. Another point to be considered for the future work in this area is the utilization of other types of the used optimization problem and other real-life optimization problems as well.

REFERENCES

[1] X. Yang, "Nature-Inspired Optimization Algorithms: A Tutorial", IDEAL, 2018.

[2] I. C. Obagbuwa, "Swarm Intelligence Algorithms and Applications to Real-world Optimization Problems: A Survey", *International Journal of Simulation, Systems, Science & Technology*, vol. 19, no. 2, pp.1-8. 8, 2018.

[3] K. Lwin, R. Qu, "A hybrid algorithm for constrained portfolio selection problems", *Applied intelligence*, Kluwer, 2013.

[4] W. J. Gutjahr, "A generalized convergence result for the graph-based ant system metaheuristic", *Probability in the Engineering and Informational Sciences*, vol. 17, no. 4, pp. 545-569, 2003.

[5] N. Bacanin, and M Tuba, "Firefly Algorithm for Cardinality Constrained Mean-Variance Portfolio Optimization Problem with Entropy Diversity Constraint", The Scientific World Journal, vol. 2014, Article ID 721521, 16 pages, 2014.

[6] W. Fang, X. Li, M. Zhang, and M. Hu, "Nature-Inspired Algorithms for Real-World Optimization Problems", *Journal of Applied Mathematics*, vol. 2015, Article ID 359203, 2 pages, 2015.

[7] N. Siddique and H. Adeli, "Nature Inspired Computing: An Overview and Some Future Directions", *Cognitive Computing*, vol. 7, pp. 706–714, 2015.

[8] G. Tollo, and A. Roli, "Metaheuristics for the Portfolio Selection Problem", *International Journal of Operations Research*, vol. 5, no. 1, pp. 13-35, 2008.

[9] J.N. Kapiamba, E.L.B. Ulungu, and P.K. Mubenga, "Simulated annealing vs. genetic algorithm to portfolio selection", *International Journal of Scientific and Innovative Mathematical Research*, vol. 3, pp.18-30, 2015.

[10] A. John, A. I. Logubayom, and J. Ackora-Prah, "Portfolio Optimization Using Matrix Approach: A Case of Some Stocks on the Ghana Stock Exchange", *International Journal of Accounting, Finance and Risk Management*, vol. 2, no. 1, 2017.

[11] Calculating covariance for stocks, available at: https://www.investopedia.com/.../11/calculating-covariance.asp. Last visited on 1-11-2018.

[12] C. Blum, and A. Roli, "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison", *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308, 2003.

[13] A. Kresta and, K. Slova, "Solving cardinality constrained portfolio optimization problem by binary particle swarm optimization algorithm", *Department of Mathematical Methods in Economics, Faculty of Economics, VŠB-Technical University of Ostrava, Sokolská třída*, vol.33, no. 701, 2011.

[14] R. Moral-Escudero, R. Ruiz-Torrubiano, and A. Su´arez, "Selection of optimal investment portfolios with cardinality constraints," in Proceedings of the 2006 Congress on Evolutionary Computation (CEC2006), 2006, pp. 2382–2388.

[15] D. Bienstock, "Computational study of a family of mixed-integer quadratic programming problems," *Mathematical programming*, vol. 74, pp. 121–140, 1996.

[16] D. Bertsimas and R. Shioda, "Algorithm for cardinality-constrained quadratic optimization," *Computational Optimization and Applications*, vol. 43, no. 1, pp. 1–22, May 2009.

[17] A. Shukla, R. Tiwari, and R.: Kala, "Real Life Applications of Soft Computing", CRC Press, 2010.

[18] M. Gilli, D. Maringer, and E. Schumann, "Numerical Methods and Optimization in Finance", Elsevier, 2011.

[19] S. Fidanova, "Ant colony optimization and multiple knapsack problem", In: Rennard JP (Ed.), Handbook of research on nature inspired computing for economics and management. Idea Group, pp. 498-509, 2007.

[20] V. Maniezzo, and M. Roffilli, "Very strongly constrained problems: an ant colony optimization approach", *Cybernetics and Systems: An International Journal*, vol. 39, no. 4, pp. 395-424. 2008.

[21] V. Maniezzo, and M. Milandri, "An Ant-Based Framework for Very Strongly Constrained Problems", in Dorigo, M. et al. (Eds.): Ants 2002, LNCS, vol. 2463, pp. 222-227, Springer, 2002.

[22] M., Dorigo, and T. Stutzle, "Ant Colony Optimization", MIT Press, Cambridge, 2004.

[23] S. Das, SS. Mullick, P.N. Suganthan, "Recent advances in differential evolution– An updated survey", *Swarm and Evolutionary Computation*, vol.2, no. 7, pp. 1–30, 2016.

[24] C.C. Li, H. Lin, and J.C. Liu, "Parallel genetic algorithms on the graphics processing units using island model and simulated annealing" *Advances in Mechanical Engineering*, vol. 9, no. 7, 2017, 1687814017707413.

[25] J. Madera, E. Alba, and A. Ochoa, "A Parallel Island Model for Estimation of Distribution Algorithms", In: Jose A. Lozano, P. Larranaga, I. Inza, E. Bengoetxea (Eds.): Towards a New Evolutionary Computation. Springer-Verlag Berlin Heidelberg. 2006.

[26] E. Cantú-Paz, "A survey of parallel genetic algorithms", *Calculateurs paralleles, reseaux et systems repartis*, vol. 10, no.2, pp.141-171. 1998.

[27] G. Luque, and E. Alba, Parallel Genetic Algorithms: Theory and Real World Applications. Springer. 2011.

[28] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[29] S. Das, P.N. Suganthan, "Differential evolution: a survey of the state-of-the-art", *IEEE Trans.Evol.Comput*.vol.15, no. 1, pp. 4–31, 2011.

[30] F. Neri, and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis", *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 61-106, 2010.

[31] V. Kachitvichyanukul, "Comparison of Three Evolutionary Algorithms: GA, PSO, and DE", *Industrial Engineering & Management Systems*, vol. 11, no. 3, pp.215-223, September 2012.

[32] G. A. Trunfio, "A Cooperative Co-evolutionary Differential Evolution Algorithm with Adaptive Subcomponents", ICCS International Conference on Computational Science, 2015.

[33] J. Kennedy, and R. Eberhart, Particle Swarm Optimization, *proceedings of IEEE International Conference on Neural Networks*, pp. 1942–1948, 1995.

[34] C. P. Lim, and L. C. Jain, "Advances in Swarm Intelligence", In: C.P. Lim et al. (Eds.): *Innovations in Swarm Intelligence, SCI*, vol. 248, pp. 1–7, Springer, 2009.

[35] J. Barrera, and C. A. Coello Coello, "A Review of Particle Swarm Optimization Methods Used for Multimodal Optimization", In: C.P. Lim et al. (Eds.): *Innovations in Swarm Intelligence, SCI*, vol. 248, pp. 1–7, Springer, 2009.

[36] R. Mendes, J. Kennedy, and J. Neves, "The Fully Informed Particle Swarm: Simpler, may be Better", *IEEE Transactions on Evolutionary Computation*, vol. 8, pp. 204–210, 2004.

[37] R. Wahono, and N. Suryana, "Combining PSO based Feature Selection and Bagging Technique for Software Defect Prediction", *International Journal of Software Engineering and Its Applications*, vol.7, no.5, pp. 153-166, 2013.

[38] B. Lui, "Web Data Mining", Springer, 2010.

[39] N. Abd-Alsabour, "A review on evolutionary feature selection", In *Modelling Symposium (EMS), 2014 European*, pp. 20-26. IEEE, 2014.

[40] Beasley, J E. "Operations Research (OR)-Library." Index of /~Mastjjb/Jeb/Orlib/Files, 2004, http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files

[41] Nadia Abd-Alsabour. (2019). Table2 The correlation between all pairs of the 31 items. http://doi.org/10.5281/zenodo.2649226

[42] R: A Language and Environment for Statistical Computing [http://www.R-project.org]. R Foundation for Statistical Computing, Vienna, Austria.

[43] S. Das, A. Abraham, and A. Konar, "Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives", Studies in Computational Intelligence, vol.116, pp.1–38, Springer-Verlag Berlin Heidelberg, 2008.

[44] N. Abd-Alsabour, "Investigating the influence of adding local search to search algorithms", 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2017.
[45] N. Abd-Alsabour, "Parallel evolutionary algorithms and high dimensional optimization problems", *Journal of Computers*, vol. 13, no. 11, pp.1265-1271, November 2018.
[46] N. Abd-Alsabour, "Local search for parallel optimization algorithms for high dimensional optimization problems", *MATEC Web of Conferences*, vol. 210, p. 04052. EDP Sciences, 2018.
[47] J. Zhang, and A. C. Sanderson, "JADE: Adaptive Differential Evolution with Optional External Archive", IEEE Transactions on Evolutionary Computation, vol.13, no. 5, pp. 945-958, 2009.