Knowledge-Based Systems 198 (2020) 105944

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A parallel variable neighborhood search algorithm with quadratic programming for cardinality constrained portfolio optimization

Mehmet Anil Akbay, Can B. Kalayci *, Olcay Polat

Pamukkale University, Faculty of Engineering, Department of Industrial Engineering, Kinikli, 20160, Denizli, Turkey

ARTICLE INFO

Article history:
Received 3 December 2019
Received in revised form 16 April 2020
Accepted 18 April 2020
Available online 23 April 2020

Keywords:
Metaheuristics
Variable neighborhood search
Asynchronous parallelization
Quadratic programming
Portfolio optimization

ABSTRACT

Over the years, portfolio optimization remains an important decision-making strategy for investment. The most familiar and widely used approach in the field of portfolio optimization is the mean–variance framework introduced by Markowitz. Following this pioneering work, many researchers have extended this model to make it more practical and adapt to real-life problems. In this study, one of these extensions, the cardinality constrained portfolio optimization problem, is considered. Cardinality constraints transform the quadratic optimization model into the mixed-integer quadratic programming problem, which is proved to be NP-Hard, making it harder to obtain an optimal solution within a reasonable time by using exact solution methodologies. Hence, the vast majority of the researchers have taken advantage of approximate algorithms to overcome arising computational difficulties. To develop an efficient solution approach for cardinality constrained portfolio optimization, in this study, a parallel variable neighborhood search algorithm combined with quadratic programming is proposed. While the variable neighborhood search algorithm decides the combination of assets to be held in the portfolio, quadratic programming quickly calculates the proportions of assets. The performance of the proposed algorithm is tested on five well-known datasets and compared with other solution approaches in the literature. Obtained results confirm that the proposed solution approach is very efficient especially on the portfolios with low risk and highly competitive with state-of-the-art algorithms.

1. Introduction

One of the key issues in investment is to make a higher profit at a reasonable level of risk. In this context, one of the most common strategies is the "portfolio selection" defined as the compiling of the best asset allocation among various alternatives by considering the tradeoff between two conflicting objectives; return maximization and risk minimization. The main idea behind the portfolio selection strategy is to spread the capital among multiple securities efficiently rather than investing in a single security. In the early 50 s, Markowitz [1] proposed a mean–variance framework as the pioneering work of the modern portfolio era for the portfolio selection problem. Although diversification of investment alternatives has been a long-standing practice, the mean–variance approach has become a revolutionary work that brings a quantitative strategy and modeling approach from the operations research perspective for selecting best asset combination assuming that the historical prices of the market can reflect the future prices [2].

Although the mean–variance framework of Markowitz remains a prominent strategy for the financial investment field explaining the trade-off between risk and return, because of the lack of real-life constraints the original model falls short to generate realistic solutions for real-life problems. For instance, holding too many assets in a specific portfolio may not be logical for real-life cases because of additional costs. Therefore, an additional constraint, namely cardinality constraint (CC) is introduced to the original model to restrict the number of assets [3]. It is also evident from the advancing literature that there is an increasing interest in enhancing the original model with real-life constraints, especially CC and boundary constraints (BC) [4]. Even though it is possible to solve the standard mean–variance model by using convex quadratic programming (QP), adding CC and BC transforms the structure of the original model from quadratic optimization into the mixed-integer quadratic programming problem, namely cardinality constrained portfolio optimization (CCPO) problem, which is proved to be NP-Hard [5]. Since exact solution techniques may be ineffective to solve this problem, there is an increasing interest in developing efficient heuristic algorithms to obtain near-optimal solutions.

In this study, an efficient two-phased solution methodology combining a metaheuristic (variable neighborhood search) and

* Corresponding author.
E-mail addresses: m.anil.akbay@gmail.com (M.A. Akbay), cbkalayci@pau.edu.tr (C.B. Kalayci), opolat@pau.edu.tr (O. Polat).

https://doi.org/10.1016/j.knosys.2020.105944
0950-7051/

exact (quadratic programming) approach is presented to solve the CCPO problem. While most of the researchers consider the stock selection process with the determination of the asset weights simultaneously, in the proposed solution approach, selection of the assets included in the portfolio and calculation of the optimal proportions are carried out separately, which contributes to a great algorithmic performance with simplicity and effectiveness. Moreover, compared to other solution methodologies such as population-based metaheuristic algorithms, VNS does not require long algorithm run times and relatively large memory resources. Therefore, these features enable it to be applied successfully on CCPO as well as other combinatorial and global optimization problems. Firstly, the variable neighborhood search (VNS) algorithm selects the assets to be included in the portfolio. Once the assets are decided, the mixed-integer quadratic programming problem is transformed back into the quadratic programming model without the need for any other procedure to satisfy CC. Afterward, exact optimal proportions for the previously selected assets are determined via QP.

To the best of our knowledge, this study presents the first implementation of VNS combined with an exact solution method in the portfolio optimization (PO) literature. In addition to the two-stage algorithm structure, the initial solution construction procedure, sequential neighborhood exchange, optimal pool size selection mechanisms, and asynchronous parallelization strategy can be counted as critical algorithmic components of the proposed solution approach which distinguish it from other algorithms in the literature and provide superiority. Finally, the proposed algorithm is tested on well-known benchmark datasets and compared with the other solution algorithms in the literature. Experimental results are presented to show the effectiveness of the developed algorithm along with the evaluation of the competitiveness against other methodologies presented in the literature.

The remainder of this paper is structured as follows: The literature review on the CCPO problem is presented in Section 2. The mathematical formulation of the problem is given and explained in Section 3. Detailed information about the proposed solution approach is presented in Section 4. The results of the experimental study are reported in Section 5 while Section 6 presents the conclusion with future research directions.

2. Literature review

Following Markowitz's inspiring work, the portfolio optimization (PO) problem attracted a lot of interest in the literature. Surveys focusing on this problem from different perspectives are summarized as follows: Kolm et al. [6] addressed portfolio optimization from a practical perspective considering some of the challenges encountered with real-life constraints while Metaxiotis and Liagkouras [7] reviewed the design and implementations of multi-objective evolutionary algorithms (MOEAs). Aouni et al. [8] took into consideration of exact attempts along with various risk measures. Then, Ertenlice and Kalayci [9] reviewed swarm intelligence algorithms adopted to solve in portfolio optimization. Recently, Kalayci et al. [4] presented a comprehensive review on deterministic mean–variance portfolio optimization (MVPO) problem analyzing various solution approaches, models, performance measures and real-life constraints. They mainly focused on exact and inexact solution approaches from the perspective of a computational analysis. Doering et al. [10] on the other side, examined how well the metaheuristic algorithms developed for NP-Hard class portfolio optimization problems generate high-quality solutions by considering the relationship between portfolio optimization and risk management.

While a vast majority of the researchers adapt approximate methods to obtain solutions for the CCPO problem due to its computational difficulty, there is a limited attempt to solve the problem by exact solution approaches with relaxation methods and QP mechanisms. The publications utilizing various solution approaches for the CCPO problem along with the broad classification is presented in Fig. 1.

Overall, solution approaches developed for the CCPO problem significantly increased in the last decade while population-based algorithms have the greatest number of implementations. Although, the performance of the algorithm and its superiority over others may vary according to the type of the problem applied, one of the most preferred approaches among population-based solution approaches is evolutionary algorithms due to advantages such as conceptual simplicity, a wide range of application areas and the possibility of hybridization with other algorithms. Chang et al. [3] introduced cardinality constraints to the MVPO problem for the first time and analyzed the effect on the efficient frontier as well as arising difficulty of calculating optimal solution by comparing the performance of genetic algorithms, tabu search, and simulated annealing algorithms. Based on the formulation of Chang et al. [3], Woodside-Oriakhi et al. [20] using the same heuristics, integrated subset optimization problem in which the desired return is introduced as inequality. Anagnostopoulos and Mamanis [27] applied various evolutionary algorithms to successfully deal with the computational challenges of the constrained portfolio optimization model. Especially on large scale instances, the single objective algorithms are outperformed by the multi-objective ones. Chen et al. [42] proposed an extended algorithm based on a multi-objective evolutionary structure combining local search and non-dominated sorting to solve CCPO. Soleimani et al. [25] introduced the market (sector) capitalization constraint in addition to the transaction lots and cardinality constraint for the first time in the literature and utilized a genetic algorithm to solve the problem. Chang et al. [22] proposed GA to solve the CCPO problem using different risk measures such as semi-variance, mean absolute deviation and variance with skewness. Besides, Pai and Michel [50] proposed an evolutionary optimization algorithm to solve the constrained PO problem and utilized a k-means clustering analysis to handle cardinality constraint. Because of the limited capabilities of existing techniques in solving large-scale combinatorial problems, Liagkouras [31] introduced a new MOEA incorporates a new three-dimensional encoding structure.

Besides the popularity of the evolutionary-based algorithms, swarm-based solution approaches have also attracted considerable attention in the last decade. Cura [32] applied particle swarm optimization (PSO) approach for the first time on mean–variance CCPO problem and compared the performances of the proposed algorithm with a genetic algorithm (GA), tabu search (TS), and simulated annealing (SA) algorithms on well-known data sets. Following, Zhu et al. [51] performed a comparative return analysis between the proposed PSO algorithm and the other solution approaches such as GA and the Visual Basic Application Solver considering both restricted and unrestricted scenarios. Golmakani and Fazel [34] added three more constraints such as bounds on holdings, minimum transaction lots and sector capitalization to the model and proposed an improved PSO with various mutation strategies to overcome the complex structure of this constrained model. Deng et al. [33] proposed an improved PSO algorithm by utilizing reflection, minimum hold and mutation strategies for the CCPO problem presenting a better performance than the original PSO algorithm, especially on low-risk portfolios. Pouya et al. [52] adapted invasive weed optimization algorithm to solve multi-objective portfolio optimization transforming into a single-objective programming model using fuzzy normalization and uniform design method. After a thorough computational analysis, Kalayci et al. [38] determined that repairing infeasible solutions may have a negative impact on the convergence of the

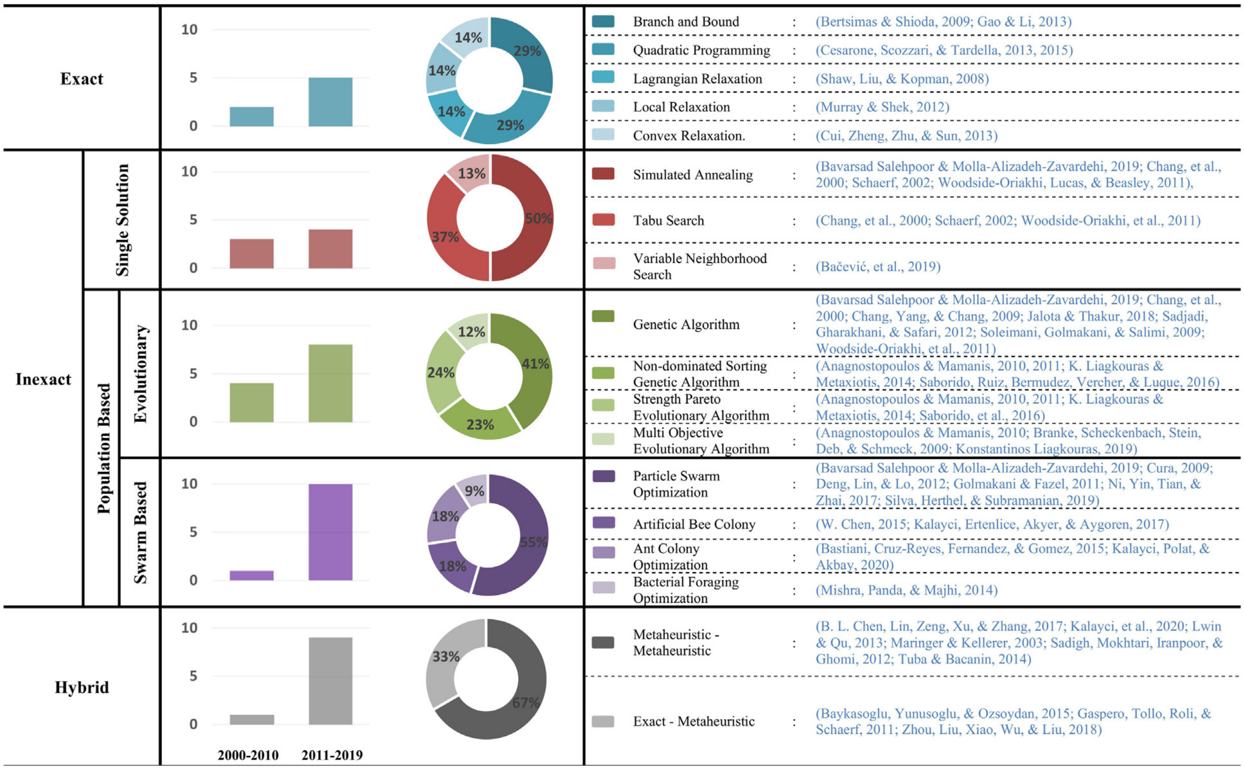

Fig. 1. Solution approaches proposed for the CCPO problem. (See Refs. [3,11–20,20–49]).

algorithm. Therefore, they enhanced their artificial bee colony (ABC) algorithm by using infeasibility toleration along with the feasibility enforcement strategies. Silva et al. [36] proposed a unified PSO algorithm to deal with different versions of the PO problem introducing an adaptive ranking procedure based on non-dominated sorting, crowding distance and cost–benefit.

It is evident from the number of studies in the PO literature [4], single solution-based solution approaches are less preferable than the population-based algorithms. Although not preferred as a stand-alone solution approach, they are combined with many other solution algorithms due to their powerful exploitation capabilities, simple structures and quick running times. Schaerf [19] improved the local search techniques proposed by Chang et al. [3] with a broader set of possible neighborhood structures to solve the CCPO problem. Bačević et al. [21] developed a VNS based heuristic for portfolio optimization problem together with additional non-convex constraints such as market capitalization and sector constraints and cardinality constraints along with a rebalancing strategy. In this strategy, they implemented a local search strategy based on a continuous VNS algorithm since problem variables are continuous.

In recent studies, various researchers preferred solution methodologies hybridizing multiple methods rather than using a single heuristic algorithm to overcome deficiencies and improve exploration and exploitation capabilities of the relevant techniques. Maringer and Kellerer [44] proposed a hybrid algorithm combining simulated annealing and an evolutionary strategy for the CCPO problem. While the proposed algorithm uses a population of crystals, elimination of the population members is based on evolutionary principles and acceptance criteria are based on the SA algorithm. Gaspero et al. [48] developed a hybrid solution approach based on local search and QP with cardinality, quantity, and pre-assignment constraints. While local search is used for selecting assets to be included in the portfolio, QP is used for determining asset weights for portfolio optimization. Baykasoglu et al. [47] adopted a greedy randomized adaptive search procedure (GRASP) together with quadratic programming where stock indices are selected by GRASP and proportion determination is optimized with QP. Recently, Kalayci et al. [40] developed a hybrid solution approach based on continuous ant colony optimization enhanced with critical components from artificial bee colony and genetic algorithms to specifically deal with the CCPO problem.

3. Portfolio optimization with Cardinality constraints

The mathematical model of the CCPO based on Chang et al. [3] is presented as follows:

Parameters

λ	The trade-off parameter of variance and return
ε_i and δ_i	The minimum and the maximum proportion of an asset i
N	Number of securities in the index
K	The exact number of securities to be held in the portfolio
σ_{ij}	The covariance between a security i and a security j
μ_i	The expected return of a security i

Variables

w_i	Proportion of securities
$z_i = \begin{cases} 1 \\ 0 \end{cases}$	If asset i is included in portfolio 1; otherwise 0.

$$\min \lambda \left[\sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \right] - (1-\lambda) \left[\sum_{i=1}^{N} w_i \mu_i \right] \tag{1}$$

Subject to :

$$\sum_{i=1}^{N} w_i = 1 \tag{2}$$

$$\sum_{i=1}^{N} z_i = K \tag{3}$$

$$\varepsilon_i z_i \leq w_i \leq \delta_i z_i \qquad i = 1, \ldots, N \tag{4}$$

$$z_i \in \{0, 1\} \qquad i = 1, \ldots, N \tag{5}$$

$$0 \leq w_i \leq 1 \qquad i = 1, \ldots, N \tag{6}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1 \qquad i = 1, \ldots, N \tag{7}$$

Eq. (1) involves two conflicting objectives, return maximization and risk minimization. As a weighting parameter, λ determines the trade-off between return and risk by linearly increasing its value from 0 to 1. When λ is equal to 0, then the model looks for the maximum return, while when λ is equal to 1, the model looks for the minimum risk. Eq. (2) guarantees that the security proportions add to one. Eq. (3), namely cardinality constraint, restricts the portfolio size and ensures that exactly K number of securities are held in the portfolio. Thus, the continuous structure of the efficient frontier turns into discontinuous by introducing cardinality constraint to the model [3]. This transformation increases the complexity and computational difficulty of the problem. Eq. (4) puts a restriction on the asset proportions to be the pre-determined minimum and maximum limit values. Eqs. (5)–(7) presents decision variables and variable domains.

4. Variable neighborhood search and parallelization strategies

VNS is a single solution and local search-based metaheuristic algorithm developed by Mladenovic and Hansen [53] to solve combinatorial optimization problems. Unlike most of the local search-based algorithms, VNS uses a set of neighborhood structures instead of a single one. The main idea behind using multiple neighborhood structure is the fact that a local minimum within a neighborhood may not be so for another one. Furthermore, VNS systematically changes the neighborhoods during the search. Thus, it ensures diversification in the search space and overcomes the problem of getting stuck into local optima frequently encountered in local search-based heuristics.

Let $G_k, (k = 1, \ldots, k_{max})$ and $G_k(x)$ signify the set of neighborhoods and the set of solutions within the kth neighborhood of the solution x, respectively. The main structure of the basic VNS is presented via the pseudo-code given in Fig. 2.

Compared to other heuristic algorithms, VNS provides substantial advantages over other algorithms due to its simple structure, integrability with different solution techniques and requiring very few parameters [54]. Moreover, as with population-based algorithms, it does not require a long algorithm run times and relatively large memory resources. These features enable it to be applied successfully on various combinatorial and global optimization problems such as location problems [55], clustering problems [56], graph problems [57], knapsack and packing problems [58], timetabling [59], various scheduling problems [60,61] and vehicle routing problems [62,63], assembly and disassembly line balancing [64–66] and portfolio optimization [21].

Although the basic VNS performs well in many different combinatorial optimization problem types, various extensions [67] have been developed to overcome problem-specific difficulties and improve the performance of the algorithm. Furthermore, with the development of hardware technology, it is possible to run various algorithm structures simultaneously using multiple processors. Thus, distributing some part of the algorithm among the available processors reduces computational time and enhances the exploration capability of the relevant solution methodologies. In the literature, several parallelization strategies are developed for the VNS algorithm. García-López et al. [68] compared three parallelization strategies: synchronous parallel VNS, i.e., parallelizing the local search phase; replicated parallel VNS, i.e., running several VNS simultaneously applying multi-start strategy; and replicated shaking VNS, i.e., shaking and local search are controlled by slave processors while the main algorithm is run via the master processor. Crainic et al. [69] developed cooperative neighborhood VNS, i.e., asynchronous master–slave relationship, considering various search control mechanisms, information exchange methods and search strategy rules.

5. The proposed methodology

The pseudo-code describing the main structure of the proposed methodology is presented in Fig. 3. The related hybrid solution approach consists of two phases: selecting the assets included in the portfolio and determining the asset weights, respectively. While the assets included in the portfolio are selected by the VNS algorithm, the asset proportions are determined by using QP.

The algorithm starts with the construction of an initial solution and asset selection pool. Afterward, the main part of the algorithm searches for the optimal portfolio by iteratively applying shaking, local search, and quadratic programming procedures. The constructed loop structure continues to run until the maximum number of iterations is reached. If the obtained solution after shaking procedure and subsequently applying QP is improved, shaking is continued with the improved one without applying local search. Otherwise, the algorithm continues to the local search phase with the solution generated via the shaking procedure. If the solution is improved in either shaking or local search phases supported with QP, the algorithm is continued with the initial neighborhood structure. Otherwise, i.e., if both solutions obtained from shaking and local search are failed to ensure improvement, neighborhood is increased, and the search continues with the next shaking procedure.

If improvement is achieved (Fig. 3, Step 22), the variable neighborhood search algorithm is set to work back from the first neighborhood (Fig. 3, Step 24), i.e. is restarted. Therefore, it can be considered as a recursive algorithm that brings a higher computational complexity. However, the maximum number of evaluations (Fig. 3, Step 9 and Step 28) limits the algorithm execution to a certain level of time. On the other hand, if an improvement is not achieved, the algorithm can be considered as iterative, i.e., the time complexity of the proposed algorithm is directly proportional to the number of assets held in the portfolio (K) and pool size used in the local search phase ($poolsize_L$). In the local search step, utilizing the best improvement strategy requires $O(K \times (poolsize_L - K))$ of time complexity which results in calling of quadratic programming function $(K \times (poolsize_L - K))$ times. Hence, the worst-case time complexity of the algorithm is proportional to the capability of the MATLAB Quadprog function. The outer loop (Fig. 3, Step 12) brings a time complexity of $(K^2 \times (poolsize_L - K))$ for the entire algorithm. Moreover, the restriction of the search pool and running the algorithm simultaneously on the multiple processors significantly reduce the time complexity of the variable neighborhood search algorithm.

As the main components of the proposed approach; initial solution construction, neighborhood selection as well as shaking and local search procedures are presented in detail in the following sections.

5.1. Asset selection

The asset selection phase of the proposed solution approach consists of three phases: constructing an initial solution and the search pool (i), shaking (ii) and local search (iii).

Algorithm: Variable Neighborhood Search
Begin
Determine the set of neighborhood structure G_k, ($k = 1, \dots, k_{max}$)
Construct an initial solution
Determine a stopping condition
Repeat the following steps until the stopping condition is met:

1. Set $k \leftarrow 1$
2. Repeat the following steps until $k = k_{max}$
   - **a.** Shaking: Generate a solution x' from $k'th$ neighborhood of solution x (x' : $G_k(x)$)
   - **b.** Local Search: Apply local search method to the solution x' and denote with the best-found local solution as x''.
   - **c.** If the local optimum found is better than the incumbent, set $x = x''$ and continue with the initial neighborhood.
     Otherwise, set $k \leftarrow k + 1$

End

Fig. 2. Variable neighborhood search algorithm [54].

Algorithm: Variable neighborhood search
W: the incoming solution to the procedure
W_S: generated solution after shaking phase with QP
W_L: generated solution after local search phase with QP
W_{GB}: global best so far solution
Begin
Select the neighborhood structure G_k, $k = 1, \dots, k_{max}$
Construct an initial solution W and the search pool, set $It \leftarrow 0$.
Repeat the following steps until the maximum number of evaluations ($MaxIt$) is reached
    Set $k \leftarrow 1$
    Repeat the following steps until $k = k_{max}$
        Shaking: Generate W_S from the k^{th} neighborhood of W (W_S: $G_k(W)$)
        Quadratic programming: Determine asset weights of W_S via QP
        $It \leftarrow It + 1$
        if W is not improved by W_S **then**
            Local Search: Apply local search to W_S and obtain W_L (W_L: $G_k(W_S)$)
            Quadratic programming: Determine asset weights of W_L via QP
            $It \leftarrow It + 1$
        end if
        if an improvement is achieved either in local search OR shaking **then**
            $W \leftarrow W_S$ OR $W \leftarrow W_L$
            $k \leftarrow 1$
        else
            $k \leftarrow k + 1$
        end if
        if $It > MaxIt$
            Break
        end if
    end
    if W_{GB} is improved **then**
        $W_{GB} \leftarrow W$
    end if
End
End

Fig. 3. Pseudo-code of the proposed solution approach.

5.1.1. Constructing initial solution and search pool

The initial solution can be generated randomly [3,33,38], or through various solution construction strategies [32,45,47]. However, using methodologies that enable reaching (near) optimal solutions as quickly as possible matters since the algorithm performance and convergence to the efficient frontier may be directly affected. Therefore, in this study, rather than starting with a random initial solution, the method proposed by Cura [32] has been adapted to both construct an initial solution and to construct the search pool to be used in the asset selection stage of the proposed solution approach.

According to this methodology, each asset is sorted based on c values, which indicate the proportion between average return and average risk according to the trade-off parameter (λ). In some cases, the datasets also include assets with negative returns and risk levels. Since, negative values may cause some calculation errors, Eqs. (10) and (11) are used to avoid miscalculations. Then, c values for each asset are calculated by using Eq. (12), and finally, all assets are sorted based on calculated c values. After sorting operation, assets with the highest c value represent the most preferable assets, whereas assets with the lower c value are classified as relatively less preferable.

$$\theta_i = 1 + (1 - \lambda)\mu_i \qquad i = 1, \dots, N \tag{8}$$

$$\rho_i = 1 + \lambda \frac{\sum_{j=1}^{N} \sigma_{ij}}{N} \qquad i = 1, \dots, N \tag{9}$$

$$\Omega = -1 \times \min(0, \theta_1, \dots, \theta_N) \tag{10}$$

$$\psi = -1 \times \min(0, \rho_1, \dots, \rho_N) \tag{11}$$

$$c_i = \frac{\theta_i + \Omega}{\rho_i + \psi} \qquad i = 1, \ldots, N \tag{12}$$

Let $a(i)$ represent each asset in the index. An example initial solution and search pool representation belong to FTSE 100 dataset are presented in Fig. 4. While the first K assets which indicate the assets to be kept in the portfolio are accepted as an initial solution after determining assets weights via QP, the rest of the assets constitute the selection pool which is going to be used in the shaking and local search steps later on.

In some cases, narrowing the search pool to a certain extent may improve search performance and thus enable the optimal solution to be found in less time. Therefore, instead of including all the assets in the search pool, it should be restricted. Example pool sizes for 5 levels determined based on the total number of assets holding in the portfolio ($K = 10$) and the total number of assets in the index (N) are also shown in Fig. 4. Details about narrowing the search pool and finding the optimal pool size are given in the further sections.

After the initial solution and the search-pool are constructed, shaking and local search steps are systematically applied.

5.1.2. Shaking

One of the most critical parts of the VNS algorithm is the mechanism of changing neighborhoods systematically which has a crucial importance for the search performance. Therefore, designing an effective search procedure considering both exploration and exploitation capability of the algorithm becomes one of the most important goals of this solution approach. Following the aforementioned goals; two main components of the shaking procedure: determining of neighborhood structures and neighborhood exchange mechanisms are of great importance. On the one hand, the chosen neighborhood structure and change strategy should effectively diversify searching to avoid a stuck into local optima, on the other hand, it should ensure to reach global optimum as quickly as possible.

In the developed algorithm, each neighborhood represents how many of the assets will be removed from the current solution and how many new assets will be inserted. This information is transferred to the shaking procedure via the variable $k = (1, \ldots, k_{max})$ as shown in Figs. 5 and 6 and the number of assets is removed and inserted based on this transferred information in the shaking step. Initially, the shaking phase starts with removing/inserting only one asset by selecting randomly, and the number of assets to be removed/inserted is increased. K denotes the desired number of assets included in the portfolio. The maximum number of assets can be removed and inserted is $K (k_{max} = K)$. In each iteration, the solution W_S is generated randomly from the incoming solution W by exchanging k assets. In other words, the solution W_S is within the neighborhood $G_k(W)$.

Besides neighborhood structures to be used, neighborhood exchange mechanisms have great importance on diversification. Over a pre-determined set of neighborhoods, various strategies can be implemented such as random, stochastic or sequential selection. In this study, sequential neighborhood selection is used. According to this strategy, changing starts with the initial neighborhood and continues by increasing the neighborhood in each iteration. At any stage of shaking along with the current neighborhood (G_k), if no improvement is made, shaking continues with the next neighborhood (G_{k+1}); otherwise, it continues with the first neighborhood (G_1) for the next iteration.

Basic representation of the sequential neighborhood exchange mechanisms and shaking structure are presented in Fig. 5. and in the pseudo-codes showed in Figs. 3 and 6.

5.1.3. Local search

In the local search step of the proposed solution approach, the best improvement strategy allowing the algorithm to run until all possible movements are done is applied. Although it is known that the best improvement is much more time consuming compared to the first improvement strategy, it is reported that better sometimes faster results are obtained with the best improvement when a good initial solution generated by a constructive heuristic or an initial solution construction strategy rather than the random initial solution [70].

In the local search phase, each asset in the current portfolio is removed, and an asset chosen from the search pool is inserted. The pseudocode of the local search procedure used in the proposed algorithm is presented in Fig. 7.

5.1.4. Parallelization

In this study, a simple parallelization strategy is applied aiming to shorten the computational time. Based on the three-dimensional taxonomy proposed by Crainic et al. [71], applied strategy is categorized as $pC/C/MPSS$. While the first dimension of the taxonomy represents the search control cardinality, and the second and third dimensions describe communication control and search strategy, respectively. As seen in Fig. 8, search control is maintained by running the algorithm independently and asynchronously on different processors for each λ value rather than using a master–slave structure. Communication and information transfer between the processors are not realized. Furthermore, while the same search strategy is adopted in each VNS algorithm distributed to the processors because the objective function with different λ values yields different results, different initial solutions are used for each algorithm in threads. Consequently, it becomes possible to collect optimal results of different λ values simultaneously from each thread of the processor.

It is always possible to develop different parallelization structures to improve performance considering the local search phase as one of the most time-consuming components of the sequential VNS algorithm or to improve exploration capability by simultaneously starting multiple searches at different points of the search space. See Crainic and Hail [72] for the parallel applications for the various metaheuristics.

5.2. Determination of asset proportions

After selection of the assets to be included in the optimal portfolio, cardinality constraint is eliminated, and the model was transformed into a simple quadratic programming. After the elimination of the cardinality constraint, the complexity of the problem significantly decreased. Furthermore, the search space was considerably narrowed because the number of assets evaluated reduced from N to K. After this phase, the weights of the assets can be calculated using a simple heuristic algorithm. However, since the problem is simplified to a quadratic optimization model, it can be easily solved with exact solution techniques, rather than adopting a heuristic algorithm to obtain near-optimal solutions. Therefore, QP is utilized to get exactly optimal results in the second phase [47]. In this study, MATLAB quadratic programming solver (MATLAB R2019a, Optimization Toolbox, Version 8.3) is used to calculate asset proportions since VNS is also developed in the same platform, thus no integration necessity required.

6. Computational experiments

6.1. Implementation

The proposed algorithm has been programmed in MATLAB R2019a and the statistical tests are performed with MINITAB 18.

i	1	…	10	…	28	…	32	…	40	…	55	…	89
$a(i)$	18	…	9	…	30	…	48	…	80	…	49	…	45
$c(i)$	1,0039	…	1,0022	…	1,0015	…	1,0014	…	1,0012	…	1,0009	…	0,9993
Initial solution			K										
Search pools based on the pool size	pool size $K + N/5$												
	pool size $K + N/4$												
	pool size $K + N/3$												
	pool size $K + N/2$												
	pool size N												

Fig. 4. An example of search-pool and assets in the initial portfolio for the FTSE 100 dataset.

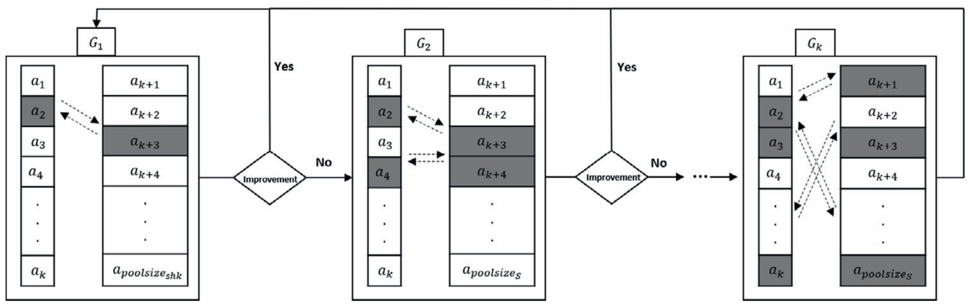

Fig. 5. Representation of sequential neighborhood change.

Procedure: Shaking
Input: $W, poolsize_S$
Output: W
W: the incoming solution to the procedure
$poolsize_S$: The size of the asset selection pool for the shaking phase
P: the set of assets in the portfolio
A: the set of assets in the restricted asset selection pool limited with $poolsize_S$
k: number of assets to remove/add ($k \le K$)
Begin
$REMOVE$ ←choose k assets randomly from P /*assets to be removed*/
ADD ←choose k assets randomly from $A \setminus P$ /*assets to be added*/
replace $REMOVE$ in W with ADD
end

Fig. 6. Pseudo-code of shaking procedure.

Procedure: Local search
Input: W_S, K
Output: W_L
W_S: the incoming solution to procedure
W_L: outcoming solution from the procedure
W_B: best-found solution in the local search step
K: the desired number of assets to be held in the portfolio
$poolsize_L$: pool size used for the local search phase
P: the set of assets in the portfolio
A: the set of assets in the restricted asset selection pool limited with $poolsize_L$ parameter
Begin
$W_L \leftarrow W_S$ and $W_B \leftarrow W_S$
$i = 1$
while $i \le K$
  for $j = K$ to $poolsize_L$
    W_L ←Replace i^{th} asset of W_S with j^{th} asset of $A \setminus P$
    Determine the weights of the assets in W_L via QP
    if W_B is improved by W_L **then**
      $W_B \leftarrow W_L$
    end if
  end for
  $i = i + 1$
end while
End

Fig. 7. Pseudo-code of local search procedure.

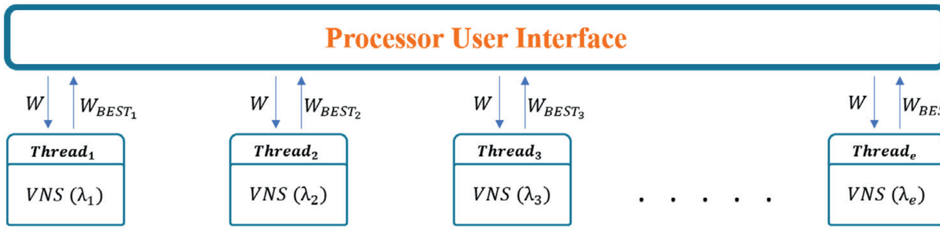

Fig. 8. Parallelization strategy for the variable neighborhood search algorithm.

The computational experiments were run on a workstation with an Intel Xeon E5-2650 2.0 GHz processor and 32 GB RAM.

6.2. Test problems

The proposed algorithm is tested on five publicly available benchmark datasets, which are widely utilized in the literature [73] to analyze performance and make a comparison with other solution approaches proposed by Chang et al. [3], Cura [32], Deng et al. [33], Lwin and Qu [43], Baykasoglu et al. [47], and Kalayci et al. [38]. These datasets contain weekly prices of Hang Seng, DAX 100, FTSE 100, S&P 100, and Nikkei 225 indices between March 1992 and September 1997. The number of stocks (N) for each dataset is 31, 85, 89, 98, and 225, respectively.

6.3. Parameter settings

It is needed to determine appropriate parameter values for the proposed solution approach to ensure the best algorithmic performance. In this context, parameter tests were performed on each dataset in line with the following assumptions: Number of assets included in the portfolio ($K = 10$), $\varepsilon_i = 0.01$ ($i = 1, \ldots, N$) and a total number of evaluations = ($N \times 1000$). The number of replications = 5. Levels tested for the parameters of the proposed methodology is presented in Table 1.

One of the most critical parameters that may affect search performance and speed of the algorithm is the size of the pool from which assets included in the portfolio are chosen. Although choosing a large search pool is good because of the increased diversity in the search, the algorithm may lose extra time to reach optimal solutions. On the other hand, narrowing the search pool may increase speed as well as exploitation capability of the algorithm, while posing the problem of getting caught in local optimum points. Therefore, it is needed to search for an optimal pool size for the developed algorithm. However, due to the different asset selection mechanisms applied in the shaking and local search, the optimal pool size to be used in these stages may be different. Therefore, the pool size parameter is set separately for both the shaking and local search steps. For each parameter, 5 levels are determined based on the number of assets (N) in the relevant dataset. As K denotes the number of assets in the portfolio, which is fixed to 10, an example pool size representation for the FTSE 100 dataset is presented in Table 2.

Based on the levels determined, the optimal pool size was searched by analyzing the interaction between shaking and local search steps. Test results belong to the FTSE dataset are presented in Figs. 9 and 10.

Fig. 9. represents the interaction plot and Fig. 10 demonstrates the main effect plot for pool size applied in the shaking and local search steps based on the mean of the optimal solution. According to the test results, selecting a narrower pool in local search yielded better results and algorithmic performance. Although the pool size selected during the shaking phase did not significantly affect the average results as much as in the local search phase, selecting larger pool size appears to worsen the performance slightly when the pool size applied for shaking is adjusted to cover relatively less assets indicated by the first 3 level tested (see Fig. 9).

6.4. Performance measures

To compare the performance of the proposed solution approach and the other algorithms in the literature, mean percentage error ($MEAPE$), median percentage error ($MEDPE$) minimum percentage error ($MINPE$) and maximum percentage error ($MAXPE$) performance measures introduced by Chang et al. [3] and mean Euclidean distance ($MEUCD$), mean return error (MRE) and variance of return error (VRE) performance measures introduced by Cura [32] have been used.

Let (x_i, y_i) represents each discrete point on the standard efficient frontier and (x^*, y^*) represents each optimal solution presenting variance and return values respectively produced by the proposed algorithm.

Firstly, x^{**} and y^{**} values representing linearly interpolated horizontal and vertical projections of each value on CCEF respectively are calculated as presented in the following formulations:

$$x^{**} = x_k + (x_j - x_k)\left[\frac{(y^* - y_k)}{(y_j - y_k)}\right] \tag{13}$$

$$y^{**} = y_k + (y_j - y_k)\left[\frac{(x^* - x_k)}{(x_j - x_k)}\right] \tag{14}$$

where x'_k and x'_j represent the variance values where $y_k = \max\,[y_i|y_i \le y^*]$, $y_j = \min\,[y_i|y_i \ge y^*]$, y'_k andy'_j represent the return values where $x_k = \max\,[x_i|x_i \le x^*]$ and $x_j = \min\,[x_i|x_i \ge x^*]$, respectively. Afterward, percentage deviation errors are calculated in both directions:

$$\varphi_j = 100\left|\frac{y^* - y^{**}}{y^{**}}\right| \tag{15}$$

$$\omega_j = 100\left|\frac{x^* - x^{**}}{x^{**}}\right| \tag{16}$$

While Eq. (18). defines the percentage deviation on return, Eq. (20). indicates the percentage deviation of variance.

$$MEAPE = \frac{\sum_j^E \min(\varphi_i, \psi_i)}{E} \tag{17}$$

$$\text{Where} \quad \psi_i = 100\left|\frac{\sqrt{X^{**}} - \sqrt{X^*}}{X^{**}}\right| \tag{18}$$

$$MEDPE = \min\left(median\left\{\varphi_1, \varphi_2, \ldots, \varphi_E\right\}, median\left\{\psi_1, \psi_2, \ldots, \psi_E\right\}\right) \tag{19}$$

$$MINPE = \min\left(\min\left\{\varphi_1, \varphi_2, \ldots, \varphi_E\right\}, \min\left\{\varphi_1, \varphi_2, \ldots, \varphi_E\right\}\right) \tag{20}$$

$$MAXPE = \max\left(\max\left\{\varphi_1, \varphi_2, \ldots, \varphi_E\right\}, \max\left\{\varphi_1, \varphi_2, \ldots, \varphi_E\right\}\right) \tag{21}$$

Table 1
Levels tested for the parameters of the proposed methodology.

Parameter	Denotes to	Levels tested				
Δ	Threshold for improvement	Fixed to: $1e-16$				
MaxIt	Maximum number of evaluations	Fixed to: $Nx1000$				
$poolsize_S$	Pool size for shaking	$K+N/5$	$K+N/4$	$K+N/3$	$K+N/2$	N
$poolsize_L$	Pool size for local search	$K+N/5$	$K+N/4$	$K+N/3$	$K+N/2$	N

Table 2
Example pool size representation for the FTSE 100 dataset.

Parameter	Denotes to	Levels tested				
		$K+N/5$	$K+N/4$	$K+N/3$	$K+N/2$	N
$poolsize_S$	Pool size for shaking	28	32	40	55	89
$poolsize_L$	Pool size for local search	28	32	40	55	89

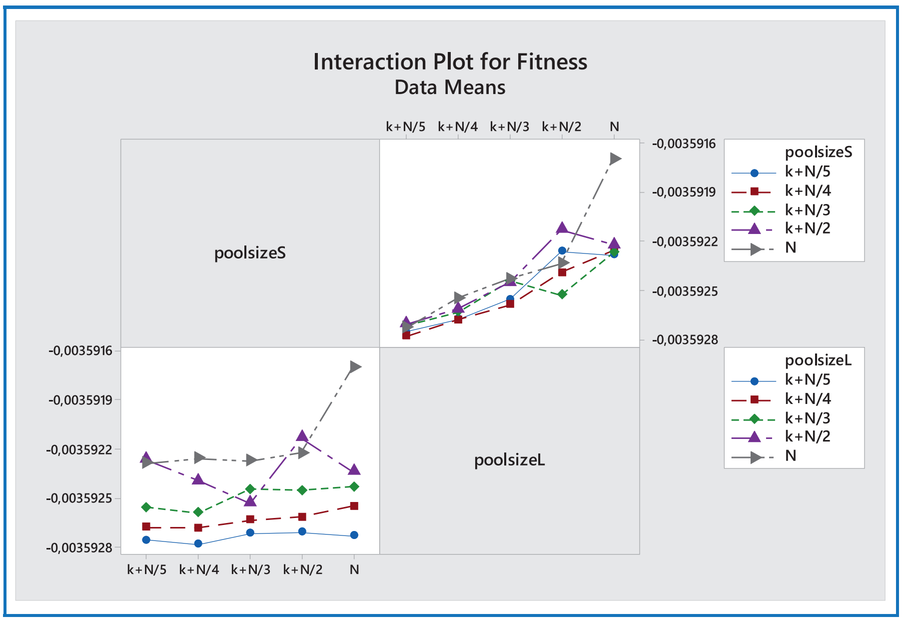

Fig. 9. Interaction plot of the pool size for the shaking and local search phases.

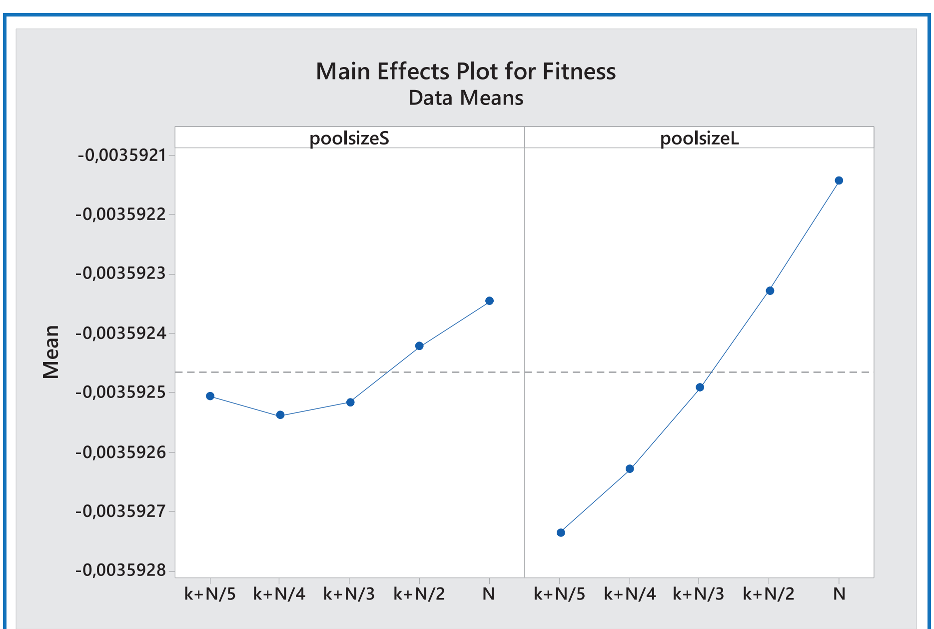

Fig. 10. Main effect plot of the pool size for the shaking and local search phases.

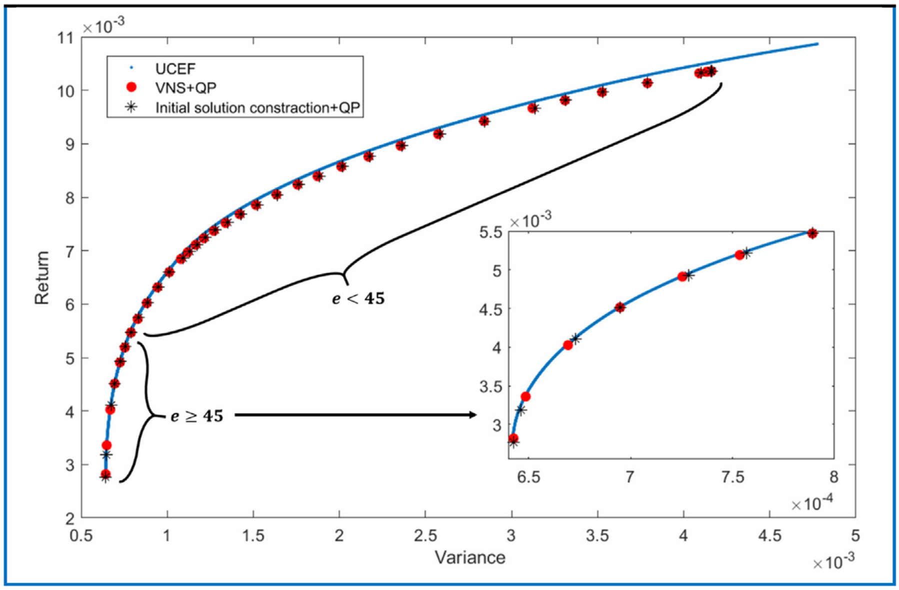

Fig. 11. Efficient frontiers for the Hang Seng dataset.

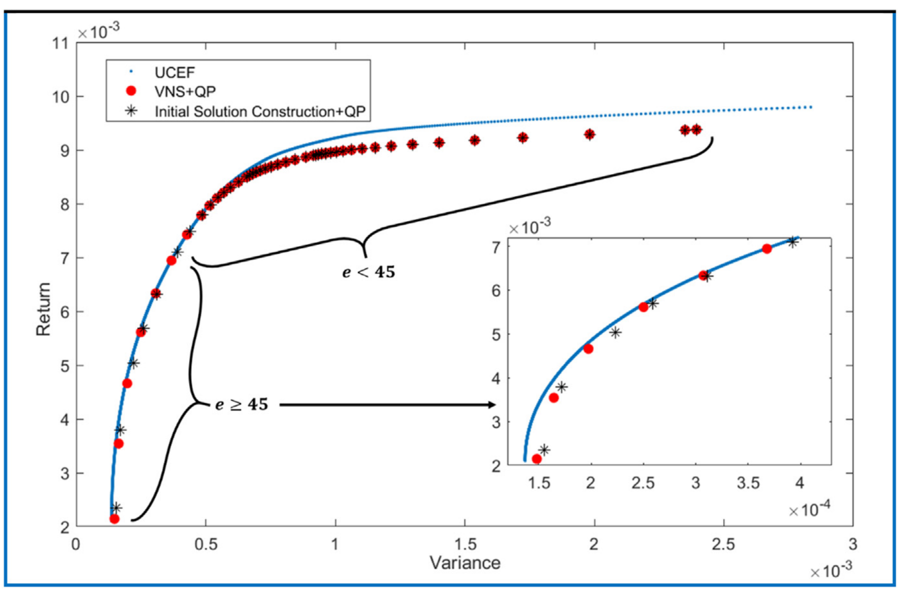

Fig. 12. Efficient frontiers for the DAX 100 dataset.

$$MEUCD = \frac{\sum_{j=1}^{E} \sqrt{(x^{***} - x^{*}) + (y^{***} - y^{*})}}{E} \tag{22}$$

$$VRE = \frac{\sum_{j=1}^{E} \frac{100\left|(x^{***} - x^{*})\right|}{x^{*}}}{E} \tag{23}$$

$$MRE = \frac{\sum_{j=1}^{E} \frac{100\left|(y^{***} - y^{*})\right|}{y^{*}}}{E} \tag{24}$$

6.5. Computational results

In this section, the computational results of the proposed algorithm are presented comparing with the other solution approaches in the literature by testing on 5 well-known benchmark instances. All tests were done by accepting $K = 10$, $\varepsilon_i = 0.01$ $(i = 1, \ldots, N)$ and $\delta_i = 1$ $(i = 1, \ldots, N)$ for the problem formulation.

Let $e = (1, \ldots, 50)$ denotes each solution point generated by different approaches corresponding to each λ value. The first experimental test is done to analyze the performance of the proposed algorithm for both the low values of $\lambda(e < 45)$ and high values of $\lambda(e \geq 45)$ because the linearity/nonlinearity of the objective function is changed according to the λ value. For instance, while $\lambda = 0$, objective function turns into almost linear, increasing of λ value result in increasing non-linearity of the objective function. Thus, the convergence of the generated solutions by different algorithms may be affected based on λ parameter.

Figs. 11–15 shows the efficient frontiers obtained by utilizing the solution procedure consisting of initial solution construction

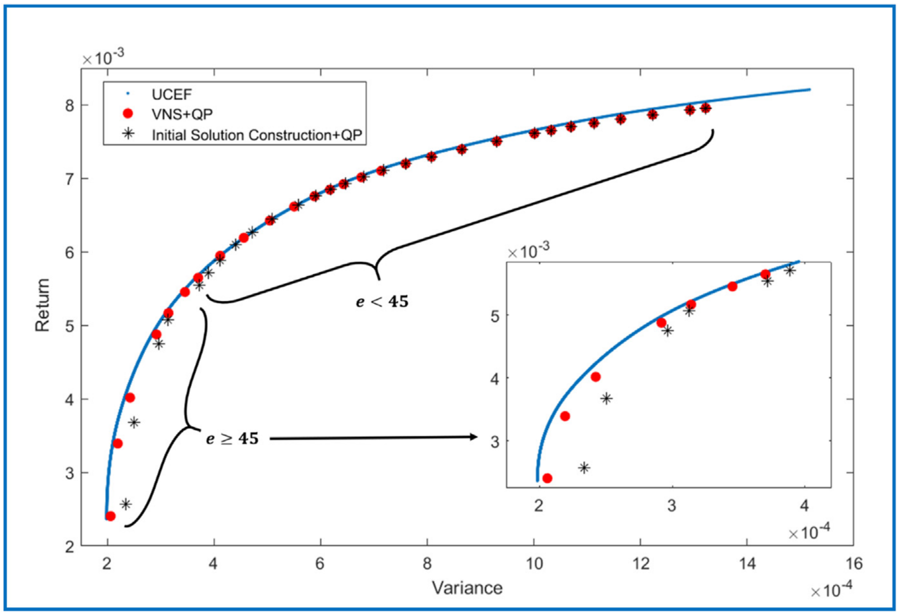

Fig. 13. Efficient frontiers for the FTSE 100 dataset.

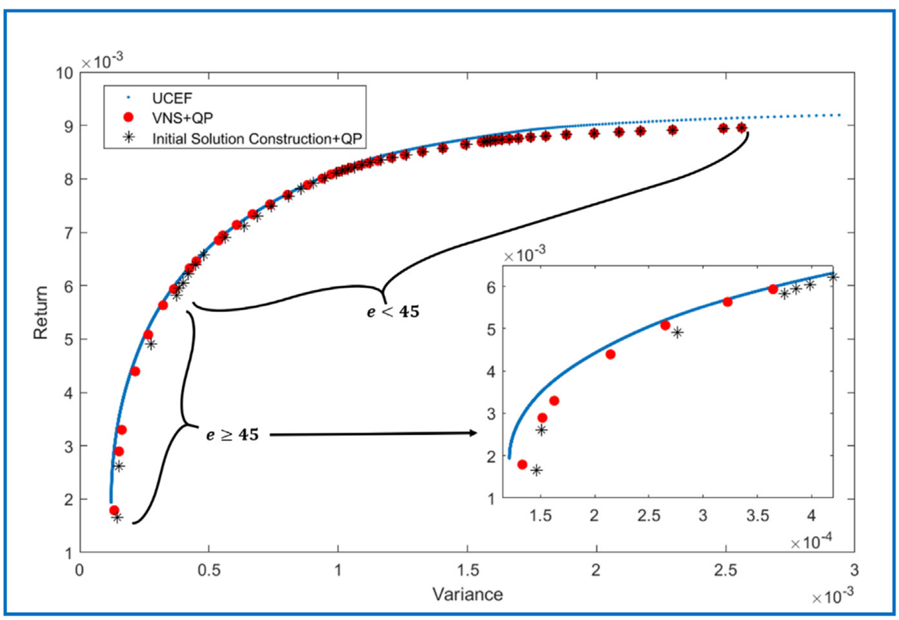

Fig. 14. Efficient frontiers for the S&P 100 dataset.

procedure combining with quadratic programming (INIQP) and the heuristic efficient frontier together with the unconstrained efficient frontier in concern with the CCPO problem for the XU030, Hang Seng, DAX 100, FTSE 100, S&P 100 datasets. In general, it is seen that VNS combined with QP (VNSQP) performs better convergence to the unconstrained efficient frontier (UCEF) in comparison with the INIQP. However, while the distance between the obtained efficient frontiers are closer for the lower values of λ ($e < 45$), the difference becomes much more significant for the higher values of λ ($e \geq 45$) especially for the larger datasets.

Computational results are obtained by the proposed solution method and presented in Tables 3–5 with tuned parameters in comparison with the other algorithms in the literature based on the performance indices outlined in Section 4. Due to the fact that performance measures, as well as the number of λ values, used varies in the compared studies, results are grouped into three. To make an accurate comparison, the maximum number of evaluations is set to 1000xN for the proposed methodology.

Firstly, the performance comparison is carried out for the proposed solution approach and PSO algorithm proposed by Deng et al. [33] and INIQP considering the low values of $\lambda(e < 45)$ and high values of $\lambda(e \geq 45)$. Table 3 presents the mean percentage errors of the results obtained by the VNSQP algorithm, INIQP and PSO proposed by Deng et al. [33]. According to the results, it is significantly seen that the proposed solution approach shows a superior performance comparing others especially for the higher

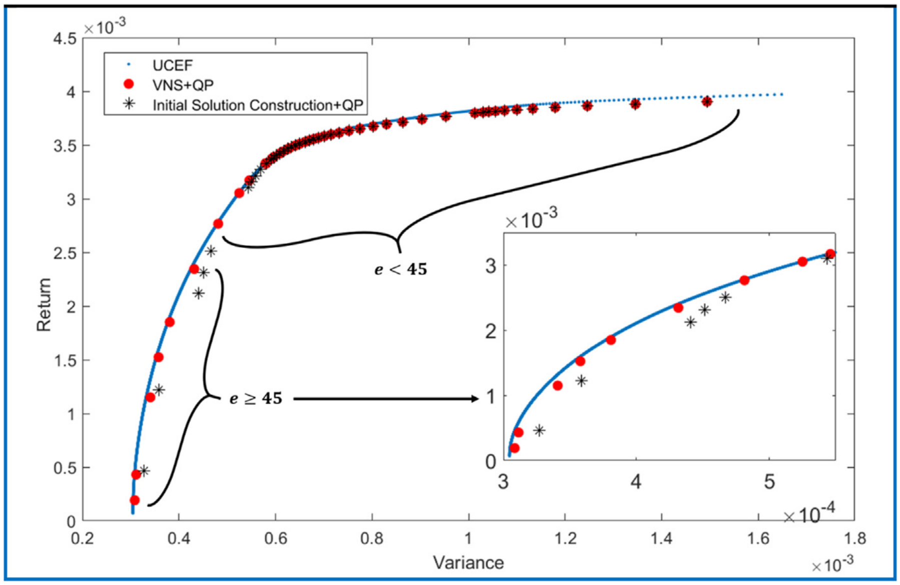

Fig. 15. Efficient frontiers for the NIKKEI dataset.

values of $\lambda(e \geq 45)$ while giving comparable results for the low values of $\lambda(e < 45)$. When DAX 100 and S&P 100 datasets are considered, even INIQP generates better solutions for the much more risk sensitive portfolios.

Results of the second performance comparison which involves mean, median, minimum and maximum percentage error scores of the algorithms proposed by Chang et al. [3], Deng et al. [33] and Lwin and Qu [43], INIQP and VNSQP is presented in Table 4. To make a fair performance comparison, each measure is calculated considering 50 different values of λ parameter. According to the obtained results, it is significantly seen that the proposed solution approach outperformed others for almost all performance measures.

Finally, Table 5 compares the test results of the proposed methodology with Cura [32], Baykasoglu et al. [47], Kalayci et al. [38] and INIQP considering performance measures such as MEUCD, VRE and MRE proposed by Cura [32]. Comparing to the best results announced by relevant researchers, the proposed solution approach performed better in terms of convergence to the unconstrained efficient frontier based on VRE especially for Hang Seng, DAX 100, FTSE 100 and S&P 100 datasets while showing comparable performance on the NIKKEI datasets.

These results confirm that the proposed methodology, variable neighborhood search algorithm with quadratic programming, is highly efficient in solving the CCPO. Moreover, the methodology combining initial solution construction strategy with quadratic programming is also quite competitive results when compared to others. It even outperformed some of the competitors for some datasets. This shows that the initial solution construction strategy used in this study provides a considerable advantage for the proposed solution approach.

7. Conclusion and discussion

In this study, a two-stage solution methodology was developed to solve the portfolio optimization problem with cardinality constraints hybridizing variable neighborhood search algorithm enhanced with asynchronous parallelization strategy and quadratic programming. In the first stage, the assets included in the portfolio were selected by using the VNS algorithm. A sorting strategy according to the risk coverage rate based on the λ parameter was used to determine the initial solution and search pool for the variable neighborhood search algorithm. After determining the assets to be included in the optimal portfolio, the problem turns into a simple quadratic programming model that seeks what ratios available capital should be distributed on the pre-selected assets. Since the exact solution of the obtained model can be found by using exact solution method, asset weights were determined by utilizing quadratic programming. The proposed methodology was compared with other solution methods in the literature testing on five well-known datasets. Obtained results revealed that the proposed methodology is highly efficient.

Some of the future research directions and discussions are summarized below:

- Since it is always possible to improve on the best-found solutions, developing much more efficient solution approaches and algorithm components which will answer problem-specific challenges will be always valuable for future research directions.
- Testing the algorithm on datasets belonging to different markets is important to see how well it responds to market-specific variability.
- Although the mean–variance framework is one of the most widely used approaches for modeling portfolio optimization problem, developing new structures considering different risk measures such as; value at risk, conditional value at risk, mean absolute deviation and developing successful solution methodologies for enhanced models with additional constraints are still a challenge to be handled.
- The testing of the proposed methodology on the multiple investment horizons is also worthy of investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 3
The results compared with Deng et al. [33] based on mean percentage error based on 50 different values of λ parameter.

Index	N	e	(Deng, et al., 2012)	INIQP	VNS + QP
Hang Seng	31	e < 45	**1.2398**	1.3119	1,2507
		e > 45	**0.0357**	0.0809	0,0516
DAX100	85	e < 45	**2.4189**	2.4475	2.4220
		e > 45	3.4420	2.9226	**1.8707**
FTSE100	89	e < 45	**0.7808**	0.8115	0.7854
		e > 45	3.1310	4.284	**1.5426**
S&P100	98	e < 45	**1.0702**	1.2582	1.0744
		e > 45	6.2273	4.2827	**3.5534**
Nikkei	225	e < 45	**0.6405**	0.7132	0.6696
		e > 45	1.0281	3.0434	**0.7451**
Average		e < 45	**1.2300**	1.3085	1.2404
		e > 45	2.7728	2.9227	**1.5527**

Table 4
The results based on 50 different values of λ parameter.

Index	N	Error Measure	GA (Chang, et al., 2000)	TS (Chang, et al., 2000)	SA (Chang, et al., 2000)	PSO (Deng, et al., 2012)	PBILDE (Lwin & Qu, 2013)	INIQP	VNS+QP
Hang Seng	31	*MEAPE*	1.0974	1.1217	**1.0957**	1.0953	1.1431	1.1640	1.0964
		MEDPE	1.2181	1.2181	1.2181	–	1.2390	1.3090	**1.2155**
		MINPE	–	–	–	–	–	**0.0000**	**0.0000**
		MAXPE	–	–	–	–	–	1.5540	**1.5538**
DAX 100	85	*MEAPE*	2.5424	3.3049	2.9297	2.5417	2.4251	2.5050	**2.3125**
		MEDPE	2.5466	2.6380	2.5661	–	2.5866	2.6120	**2.5630**
		MINPE	–	–	–	–	–	0.3619	**0.0059**
		MAXPE	–	–	–	–	–	6.3090	**4.0275**
FTSE 100	89	*MEAPE*	1.1076	1.6080	1.4623	1.0628	0.9706	1.2280	**0.8453**
		MEDPE	1.0841	1.0841	1.0841	–	1.0840	1.0840	**1.0840**
		MINPE	–	–	–	–	–	0.0108	**0.0045**
		MAXPE	–	–	–	–	–	8.4190	**2.0669**
S&P 100	98	*MEAPE*	1.9328	3.3092	3.0696	1.6890	1.6386	1.6210	**1.2649**
		MEDPE	1.2244	1.2882	1.1823	–	1.1692	1.2150	**1.1323**
		MINPE	–	–	–	–	–	0.0000	**0.0000**
		MAXPE	–	–	–	–	–	9.0600	**5.4551**
Nikkei	225	*MEAPE*	0.7961	0.8975	0.6066	0.6870	0.5972	0.9929	**0.5904**
		MEDPE	0.6133	0.6093	0.6732	–	0.5896	0.6514	**0.5857**
		MINPE	–	–	–	–	–	0.2836	**0.0000**
		MAXPE	–	–	–	–	–	4.8720	**1.1606**
Average		*MEAPE*	1,4953	2,0483	1,8328	1,4152	1,3549	1,5022	**1,2219**
		MEDPE	1,2146	1,2457	1,2101	–	1,2158	1,2440	**1,1990**
		MINPE	–	–	–	–	–	0,2048	**0,1192**
		MAXPE	–	–	–	–	–	5,1251	**2,6207**

Table 5
The results based on 51 different values of λ parameter.

Index	N	Error Measure	PSO (Cura, 2009)	GRASP–QP (Baykasoglu, et al., 2015)	ABC (Kalayci, et al., 2017)	INIQP	VNS+QP
Hang Seng	31	*MEUCD*	0.0049	**0.0001**	**0.0001**	**0.0001**	**0.0001**
		VRE	2.2421	1.6400	1.6432	1.7545	**1.6397**
		MRE	0.7427	0.6060	**0.6047**	0.6341	0.6058
DAX 100	85	*MEUCD*	0.0090	**0.0001**	**0.0001**	**0.0001**	**0.0001**
		VRE	6.8588	6.7593	6.7925	7.1446	**6.7583**
		MRE	1.5885	1.2769	**1.2761**	1.2798	1,2767
FTSE 100	89	*MEUCD*	0.0022	**0.0000**	**0.0000**	**0.0000**	**0.0000**
		VRE	3.0596	2.4350	2.4397	3.1653	**2.4349**
		MRE	0.3640	**0.3245**	0.3255	0.3282	0.3252
S&P 100	98	*MEUCD*	0.0052	**0.0001**	**0.0001**	**0.0001**	**0.0001**
		VRE	3.9136	2.5211	2.5260	3.6934	**2.5105**
		MRE	1.4040	0.9063	**0.8885**	1.0484	0.9072
Nikkei	225	*MEUCD*	0.0019	**0.0000**	**0.0000**	**0.0000**	**0.0000**
		VRE	2.4274	**0.8359**	0.8396	1.6031	0.8561
		MRE	0.7997	0.4184	**0.4127**	0.4259	0,4217
Average		*MEUCD*	0.0046	**0.0001**	**0.0001**	**0.0001**	**0.0001**
		VRE	3.7003	**2.8383**	2.8482	3.4722	2.8399
		MRE	0.9798	0.7064	**0.7015**	0.7433	0.7073

CRediT authorship contribution statement

Mehmet Anil Akbay: Conceptualization, Methodology, Visualization, Writing - review & editing. **Can B. Kalayci:** Conceptualization, Methodology, Visualization, Writing - review & editing. **Olcay Polat:** Conceptualization, Methodology, Visualization, Writing - review & editing.

Acknowledgment

This research is funded by the Scientific and Technological Research Council of Turkey (TUBITAK) with the grant number 214M224.

References

[1] H. Markowitz, Portfolio selection, J. Finance 7 (1952) 77–91.
[2] H.M. Markowitz, The early history of portfolio theory: 1600–1960, Financ. Anal. J. 55 (1999) 5–16.
[3] T.J. Chang, N. Meade, J.E. Beasley, Y.M. Sharaiha, Heuristics for cardinality constrained portfolio optimisation, Comput. Oper. Res. 27 (2000) 1271–1302.
[4] C.B. Kalayci, O. Ertenlice, M.A. Akbay, A comprehensive review of deterministic models and applications for mean–variance portfolio optimization, Expert Syst. Appl. 125 (2019) 345–368.
[5] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher, J.D. Bohlinger (Eds.), Complexity of Computer Computations: Proceedings of a Symposium on the Complexity of Computer Computations, March 20–22, 1972, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department. Publishing, 1972, pp. 85–103.
[6] P.N. Kolm, R. Tutuncu, F.J. Fabozzi, 60 Years of portfolio optimization: Practical challenges and current trends, European J. Oper. Res. 234 (2014) 356–371.
[7] K. Metaxiotis, K. Liagkouras, Multiobjective evolutionary algorithms for portfolio management: A comprehensive literature review, Expert Syst. Appl. 39 (2012) 11685–11698.
[8] B. Aouni, M. Doumpos, B. Pérez-Gladish, R.E. Steuer, On the increasing importance of multiple criteria decision aid methods for portfolio selection, J. Oper. Res. Soc. (2018) 1–18.
[9] O. Ertenlice, C.B. Kalayci, A survey of swarm intelligence for portfolio optimization: Algorithms and applications, Swarm Evol. Comput. (2018).
[10] J. Doering, R. Kizys, A.A. Juan, À. Fitó, O. Polat, Metaheuristics for rich portfolio optimisation and risk management: Current state and future trends, Oper. Res. Perspect. 6 (2019) 100121.
[11] D. Bertsimas, R. Shioda, Algorithm for cardinality-constrained quadratic optimization, Comput. Optim. Appl. 43 (2009) 1–22.
[12] J. Gao, D. Li, Optimal Cardinality constrained portfolio selection, Oper. Res. 61 (2013) 745–761.
[13] F. Cesarone, A. Scozzari, F. Tardella, A new method for mean–variance portfolio optimization with cardinality constraints, Ann. Oper. Res. 205 (2013) 213–234.
[14] F. Cesarone, A. Scozzari, F. Tardella, Linear vs. quadratic portfolio selection models with hard real-world constraints, Comput. Manag. Sci. 12 (2015) 345–370.
[15] D.X. Shaw, S.C. Liu, L. Kopman, Lagrangian relaxation procedure for cardinality-constrained portfolio optimization, Optim. Methods Softw. 23 (2008) 411–420.
[16] W. Murray, H. Shek, A local relaxation method for the cardinality constrained portfolio optimization problem, Comput. Optim. Appl. 53 (2012) 681–709.
[17] X.T. Cui, X.J. Zheng, S.S. Zhu, X.L. Sun, Convex relaxations and MIQCQP reformulations for a class of cardinality-constrained portfolio selection problems, J. Global Optim. 56 (2013) 1409–1423.
[18] I. Bavarsad Salehpoor, S. Molla-Alizadeh-Zavardehi, A constrained portfolio selection model at considering risk-adjusted measure by using hybrid meta-heuristic algorithms, Appl. Soft Comput. 75 (2019) 233–253.
[19] A. Schaerf, Local search techniques for constrained portfolio selection problems, Comput. Econ. 20 (2002) 177–190.
[20] M. Woodside-Oriakhi, C. Lucas, J.E. Beasley, Heuristic algorithms for the cardinality constrained efficient frontier, European J. Oper. Res. 213 (2011) 538–550.
[21] A. Bačević, N. Vilimonović, I. Dabić, J. Petrović, D. Damnjanović, D. Džamić, Variable neighborhood search heuristic for nonconvex portfolio optimization, Eng. Econ. 64 (2019) 254–274.
[22] T.J. Chang, S.C. Yang, K.J. Chang, Portfolio optimization problems in different risk measures using genetic algorithm, Expert Syst. Appl. 36 (2009) 10529–10537.
[23] H. Jalota, M. Thakur, Genetic algorithm designed for solving portfolio optimization problems subjected to cardinality constraint, Int. J. Syst. Assur. Eng. Manag. 9 (2018) 294–305.
[24] S.J. Sadjadi, M. Gharakhani, E. Safari, Robust optimization framework for cardinality constrained portfolio problem, Appl. Soft Comput. 12 (2012) 91–99.

[25] H. Soleimani, H.R. Golmakani, M.H. Salimi, Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm, Expert Syst. Appl. 36 (2009) 5058–5063.
[26] K.P. Anagnostopoulos, G. Mamanis, A portfolio optimization model with three objectives and discrete variables, Comput. Oper. Res. 37 (2010) 1285–1297.
[27] K.P. Anagnostopoulos, G. Mamanis, The mean–variance cardinality constrained portfolio optimization problem: An experimental evaluation of five multiobjective evolutionary algorithms, Expert Syst. Appl. 38 (2011) 14208–14217.
[28] K. Liagkouras, K. Metaxiotis, A new Probe Guided Mutation operator and its application for solving the cardinality constrained portfolio optimization problem, Expert Syst. Appl. 41 (2014) 6274–6290.
[29] R. Saborido, A.B. Ruiz, J.D. Bermudez, E. Vercher, M. Luque, Evolutionary multi-objective optimization algorithms for fuzzy portfolio selection, Appl. Soft Comput. 39 (2016) 48–63.
[30] J. Branke, B. Scheckenbach, M. Stein, K. Deb, H. Schmeck, Portfolio optimization with an envelope-based multi-objective evolutionary algorithm, European J. Oper. Res. 199 (2009) 684–693.
[31] K. Liagkouras, A new three-dimensional encoding multiobjective evolutionary algorithm with application to the portfolio optimization problem, Knowl.-Based Syst. 163 (2019) 186–203.
[32] T. Cura, Particle swarm optimization approach to portfolio optimization, Nonlinear Anal. RWA 10 (2009) 2396–2406.
[33] G.F. Deng, W.T. Lin, C.C. Lo, Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization, Expert Syst. Appl. 39 (2012) 4558–4566.
[34] H.R. Golmakani, M. Fazel, Constrained portfolio selection using particle swarm optimization, Expert Syst. Appl. 38 (2011) 8327–8335.
[35] Q.J. Ni, X.S. Yin, K.W. Tian, Y.Q. Zhai, Particle swarm optimization with dynamic random population topology strategies for a generalized portfolio selection problem, Nat. Comput. 16 (2017) 31–44.
[36] Y.L.T.V. Silva, A.B. Herthel, A. Subramanian, A multi-objective evolutionary algorithm for a class of mean–variance portfolio selection problems, Expert Syst. Appl. 133 (2019) 225–241.
[37] W. Chen, Artificial bee colony algorithm for constrained possibilistic portfolio optimization problem, Physica A 429 (2015) 125–139.
[38] C.B. Kalayci, O. Ertenlice, H. Akyer, H. Aygoren, An artificial bee colony algorithm with feasibility enforcement and infeasibility toleration procedures for cardinality constrained portfolio optimization, Expert Syst. Appl. 85 (2017) 61–75.
[39] S.S. Bastiani, L. Cruz-Reyes, E. Fernandez, C. Gomez, Portfolio optimization from a set of preference ordered projects using an ant colony based multi-objective approach, Int. J. Comput. Intell. Syst. 8 (2015) 41–53.
[40] C.B. Kalayci, O. Polat, M.A. Akbay, An efficient hybrid metaheuristic algorithm for cardinality constrained portfolio optimization, Swarm Evol. Comput. 54 (2020) 100662.
[41] S.K. Mishra, G. Panda, R. Majhi, Constrained portfolio asset selection using multiobjective bacteria foraging optimization, Oper. Res. Int. J. 14 (2014) 113–145.
[42] B.L. Chen, Y.B. Lin, W.H. Zeng, H. Xu, D.F. Zhang, The mean–variance cardinality constrained portfolio optimization problem using a local search-based multi-objective evolutionary algorithm, Appl. Intell. 47 (2017) 505–525.
[43] K. Lwin, R. Qu, A hybrid algorithm for constrained portfolio selection problems, Appl. Intell. 39 (2013) 251–266.
[44] D. Maringer, H. Kellerer, Optimization of cardinality constrained portfolios with a hybrid local search algorithm, OR Spectrum 25 (2003) 481–495.
[45] A.N. Sadigh, H. Mokhtari, M. Iranpoor, S.M.T.F. Ghomi, Cardinality constrained portfolio optimization using a hybrid approach based on particle swarm optimization and hopfield neural network, Adv. Sci. Lett. 17 (2012) 11–20.
[46] M. Tuba, N. Bacanin, Artificial bee colony algorithm hybridized with firefly algorithm for Cardinality constrained mean-variance portfolio selection problem, Appl. Math. Inf. Sci. 8 (2014) 2831–2844.
[47] A. Baykasoglu, M.G. Yunusoglu, F.B. Ozsoydan, A GRASP based solution approach to solve cardinality constrained portfolio optimization problems, Comput. Ind. Eng. 90 (2015) 339–351.
[48] L.D. Gaspero, G.D. Tollo, A. Roli, A. Schaerf, Hybrid metaheuristics for constrained portfolio selection problems, Quant. Finance 11 (2011) 1473–1487.
[49] Z. Zhou, X. Liu, H. Xiao, S. Wu, Y. Liu, A DEA-based MOEA/D algorithm for portfolio optimization, Cluster Comput. (2018).
[50] G.A.V. Pai, T. Michel, Evolutionary optimization of constrained k-means clustered assets for diversification in small portfolios, IEEE Trans. Evol. Comput. 13 (2009) 1030–1053.
[51] H.H. Zhu, Y. Wang, K.S. Wang, Y. Chen, Particle Swarm Optimization (PSO) for the constrained portfolio optimization problem, Expert Syst. Appl. 38 (2011) 10161–10169.
[52] A.R. Pouya, M. Solimanpur, M.J. Rezaee, Solving multi-objective portfolio optimization problem using invasive weed optimization, Swarm Evol. Comput. 28 (2016) 42–57.
[53] N. Mladenovic, P. Hansen, Variable neighborhood search, Comput. Oper. Res. 24 (1997) 1097–1100.
[54] P. Hansen, N. Mladenovic, Variable neighborhood search: Principles and applications, European J. Oper. Res. 130 (2001) 449–467.
[55] P. Hansen, N. Mladenović, D. Perez-Britos, Variable neighborhood decomposition search, J. Heuristics 7 (2001) 335–350.
[56] P. Hansen, N. Mladenović, J-Means: a new local search heuristic for minimum sum of squares clustering, Pattern Recognit. 34 (2001) 405–413.
[57] S.A. Canuto, M.G.C. Resende, C.C. Ribeiro, Local search with perturbations for the prize-collecting steiner tree problem in graphs, Networks 38 (2001) 50–58.
[58] J. Puchinger, G.R. Raidl, U. Pferschy, The Core Concept for the Multidimensional Knapsack Problem. Publishing, Berlin, Heidelberg, 2006, pp. 195–208.
[59] P. Côté, T. Wong, R. Sabourin, A Hybrid Multi-objective Evolutionary Algorithm for the Uncapacitated Exam Proximity Problem. Publishing, Berlin, Heidelberg, 2005, pp. 294–312.
[60] P. Hansen, N. Mladenović, An introduction to variable neighborhood search, in: S. Voß, S. Martello, I.H. Osman & C. Roucairol (Eds.), Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization. Publishing, Boston, MA, 1999, pp. 433–458.
[61] P. Hansen, N. Mladenović, D. Urošević, Variable neighborhood search and local branching, Comput. Oper. Res. 33 (2006) 3034–3045.
[62] C.B. Kalayci, C. Kaya, An ant colony system empowered variable neighborhood search algorithm for the vehicle routing problem with simultaneous pickup and delivery, Expert Syst. Appl. 66 (2016) 163–175.
[63] O. Polat, C.B. Kalayci, O. Kulak, H.O. Gunther, A perturbation based variable neighborhood search heuristic for solving the Vehicle Routing Problem with Simultaneous Pickup and Delivery with Time Limit, European J. Oper. Res. 242 (2015) 369–382.
[64] C.B. Kalayci, O. Polat, S.M. Gupta, A variable neighbourhood search algorithm for disassembly lines, J. Manufact. Technol. Manag. 26 (2015) 182–194.
[65] C.B. Kalayci, O. Polat, S.M. Gupta, A hybrid genetic algorithm for sequence-dependent disassembly line balancing problem, Ann. Oper. Res. 242 (2016) 321–354.
[66] O. Polat, C.B. Kalayci, Ö. Mutlu, S.M. Gupta, A two-phase variable neighbourhood search algorithm for assembly line worker assignment and balancing problem type-II: an industrial case study, Int. J. Prod. Res. 54 (2015) 722–741.
[67] P. Hansen, N. Mladenović, J.A. Moreno Pérez, Variable neighbourhood search: methods and applications, Ann. Oper. Res. 175 (2009) 367–407.
[68] F. García-López, B. Melián-Batista, J.A. Moreno-Pérez, J.M. Moreno-Vega, The parallel variable neighborhood search for the p-Median problem, J. Heuristics 8 (2002) 375–388.
[69] T.G. Crainic, M. Gendreau, P. Hansen, N. Mladenović, Cooperative parallel variable neighborhood search for the p-Median, J. Heuristics 10 (2004) 293–314.
[70] P. Hansen, N. Mladenović, First vs. best improvement: An empirical study, Discrete Appl. Math. 154 (2006) 802–817.
[71] T.G. Crainic, M. Toulouse, M. Gendreau, Toward a taxonomy of parallel tabu search heuristics, INFORMS J. Comput. 9 (1997) 61–72.
[72] T.G. Crainic, N. Hail, Parallel metaheuristics applications in: E. Alba (Ed.), Parallel Metaheuristics: A New Class of Algorithms. Publishing, Hoboken, New Jersey, 2005.
[73] J.E. Beasley, Or-library - Distributing test problems by electronic mail, J. Oper. Res. Soc. 41 (1990) 1069–1072.