A Fast Converging Evolutionary Algorithm for Constrained Multiobjective Portfolio Optimization

Yi Chen[1](✉), Hemant Kumar Singh[2], Aimin Zhou[1], and Tapabrata Ray[2]

[1] East China Normal University, Shanghai 200062, China
52164500006@stu.ecnu.edu.cn, amzhou@cs.ecnu.edu.cn
[2] The University of New South Wales, Canberra, ACT 2600, Australia
{h.singh,t.ray}@adfa.edu.au

Abstract. Portfolio optimization is a well-known problem in the domain of finance with reports dating as far back as 1952. It aims to find a trade-off between risk and expected return for the investors, who want to invest finite capital in a set of available assets. Furthermore, *constrained* portfolio optimization problems are of particular interest in real-world scenarios where practical aspects such as cardinality (among others) are considered. Both mathematical programming and meta-heuristic approaches have been employed for handling this problem. Evolutionary Algorithms (EAs) are often preferred for constrained portfolio optimization problems involving non-convex models. In this paper, we propose an EA with a tailored variable representation and initialization scheme to solve the problem. The proposed approach uses a short variable vector, regardless of the size of the assets available to choose from, making it more scalable. The solutions generated do not need to be repaired and satisfy some of the constraints implicitly rather than requiring a dedicated technique. Empirical experiments on 20 instances with the numbers of assets, ranging from 31 to 2235, indicate that the proposed components can significantly expedite the convergence of the algorithm towards the Pareto front.

Keywords: Representation for evolutionary algorithm · Multi-objective portfolio optimization · Constrained portfolio optimization

1 Introduction and Background

Portfolio optimization is a prominent application problem in the field of finance. A number of studies have been reported on this subject since 1952, as reviewed in [9]. The core task is to find the optimal allocation of limited capital among certain given assets so that two conflicting goals, minimizing risk and maximizing returns, can be achieved. Mean-Variance (MV) model is an original portfolio optimization model, which estimates the expected return by mean of the profit

© Springer Nature Switzerland AG 2021
H. Ishibuchi et al. (Eds.): EMO 2021, LNCS 12654, pp. 283–295, 2021.
https://doi.org/10.1007/978-3-030-72062-9_23

and measures the risk by variance of the profit [11]. This model is a quadratic problem since it assumes that the risk of an investment should be assessed by a covariance matrix based on the variance and the correlation among the assets. As a general quadratic problem, it is not difficult to solve using contemporary optimization techniques [14]. However, practical portfolio problems involve a range of constraints, such as cardinality, minimum lot size, pre-selected assets, etc. Incorporating these constraints result in mixed integer/combinatorial models. There are also some exact algorithms, which have been designed and utilized to tackle the constrained portfolio optimization problems [1]. Nonetheless, there is no polynomial-time algorithm for the case when a cardinality constraint is involved [13].

Therefore, meta-heuristic algorithms such as Evolutionary Algorithms (EAs) have gained attention to solve complex instances of portfolio optimization problems. Moreover, population-based metaheuristics are also inherently suited for solving multiobjective optimization problems, since the optimum of such problems consists of a trade-off solution set (Pareto optimal front). They have been used to solve different constrained portfolio optimization problems based on the mean-variance (MV) model, with different numbers of objectives and various different types of constraints. For single-objective problems, Chang et al. [2] used three different meta-heuristics, namely genetic algorithms, tabu search, and simulated annealing, to handle the problem with a fixed amount of invested assets. Gaspero et al. [8] proposed a hybrid algorithm with a local search and a quadratic programming solver. The numerical experiments on a portfolio optimization problem with three constraints (cardinality, quantity, and pre-assignment) showed that it performed better than some of the widely used softwares such as CPLEX and MOSEK. As for the constrained multi-objective problems, Pouya et al. [15] transformed the problem into a single-objective form with a fuzzy normalization. Subsequently, they used a metaheurstic method to solve the transformed problem and obtained improved results over Particle Swarm Optimization [15]. Deb et al. [6] solved the problem using a customized hybrid algorithm with specialized evolutionary operators, repair, local search and clustering. Lwin et al. [10] proposed a learning-guided multi-objective evolutionary algorithm (MODEwAwL), utilizing the information about selection of assets from previous generation to guide the search. It was based on a multiobjective Differential Evolution (DE) algorithm and the study compared results obtained with four well known Multi-Objective Evolutionary Algorithms (MOEAs). The simulation results on a MV model based problem, including four constraints such as cardinality, quantity, pre-assignment, and round lot constraints, indicated that the proposed algorithm improved the performance over the others. Yi et al. [3,4] introduced a tailored representation, Compressed Coding Scheme (CCS) for the same problem as that in [10]. This representation used one gene fragment to represent both the selection and allocation of one asset. The comparison between CCS and the direct representation illustrated that this tailored representation was superior for this problem with regard to the obtained efficient front since it could reduce the interaction between the selection and allocation elements of the problem.

In the methods available to solve constrained multiobjective optimization problems, including those discussed above, the representation of a solution vector is typically done using a binary vector that indicates whether a particular asset is selected or not. Consequently, the length of the solution vector increases significantly as the number of assets increases; resulting in a corresponding (exponential) increase in the solution search space. Therefore, in this paper, we propose and show initial results on a new representation scheme that remains of a fixed size (equal to the cardinality of the portfolio) regardless of the number of assets available to choose from. Moreover, we also propose a customized initialization which guarantees one of the global extremities of the true Pareto front (PF). An evolutionary algorithm constructed with the proposed components shows a commendable speed-up in convergence, delivering a good PF approximation in relatively small number (≈20%) of evaluations compared to the peer algorithms. The mathematical model for the problem is detailed next in Sect. 2, followed by the proposed method in Sect. 3. Numerical experiments and conclusions are presented in Sects. 4 and 5, respectively.

2 Mathematical Model

The constrained portfolio optimization problem studied here has two conflicting objectives (minimum risk, maximum return) and is subject to four constraints: cardinality, quantity, pre-assignment, and round lot constraints [10]. The following notations are used in the formulation:

N	the number of available assets
K	the number of assets in a portfolio, i.e., the cardinality
L	the number of assets in the pre-assignment set
w_i	the proportion of capital invested in the i^{th} asset
ρ_{ij}	the correlation coefficient of the returns of i^{th} and j^{th} assets
σ_i	the standard deviation of i^{th} asset
σ_{ij}	the covariance of i^{th} and j^{th} assets
μ_i	the expected return of the i^{th} asset
υ_i	the minimum trading lot of the i^{th} asset
ϵ_i	the lower limit on the investment of the i^{th} asset
δ_i	the upper limit on the investment of the i^{th} asset
y_i	the multiple of the minimum trading lot in the i^{th} asset

$$\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$$

$$s_i = \begin{cases} 1, & \textbf{if} \text{ the } i^{th} \ (i = 1, \ldots, N) \text{ asset is chosen} \\ 0, & \text{otherwise} \end{cases}$$

$$z_i = \begin{cases} 1, & \textbf{if} \text{ the } i^{th} \text{ asset is in the pre-assigned set} \\ 0, & \text{otherwise} \end{cases}$$

With above notations, the problem can be formulated as:

$$\text{Minimize} \quad f_1 = \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{1}$$

$$\text{Maximize} \quad f_2 = \sum_{i=1}^{N} w_i \mu_i, \tag{2}$$

$$\text{subject to} \quad \sum_{i=1}^{N} w_i = 1, \qquad 0 \leq w_i \leq 1, \tag{3}$$

$$\sum_{i=1}^{N} s_i = K, \tag{4}$$

$$\epsilon_i s_i \leq w_i \leq \delta_i s_i, \qquad i = 1, ..., N, \tag{5}$$

$$s_i \geq z_i, \qquad i = 1, ..., N, \tag{6}$$

$$w_i = y_i v_i, \qquad i = 1, ..., N, \quad y_i \in \mathbb{Z}_+, \tag{7}$$

$$s_i, z_i \in \{0, 1\}, \qquad i = 1, ..., N, \tag{8}$$

The Eqs. (1) and (2) represent the two conflicting objectives, i.e., risk minimization and return maximization. Equation (3) ensures that all available capital should be invested. Equation (4) is the *cardinality constraint* enforcing that exactly K assets are selected. Equation (5) is the *floor and ceiling constraint*, it defines that investment in the i^{th} asset should lie between ϵ_i to δ_i. Further, Eq. (6) designates the *pre-assignment constraint* that certain asset(s) *must be* selected ($z_i = 1$) in the portfolio, eg, based on investors' preferences. Equation (7) defines the *round lot constraint*, specifying a minimum number of lots to be traded for a selected asset. Finally, Eq. (8), *discrete constraint*, limits both s_i and z_i to be binary. In this study, these constraints are set in consistency with the past study [3] as follows:

- Cardinality $K = 10$, floor $\epsilon_i = 0.01$, ceiling $\delta_i = 1.0$, pre-assignment $z_{30} = 1$ and round lot $v_i = 0.008$.

The goal in solving the above problem is to seek a set of portfolios that lie close to/on the Pareto front, with a good diversity to cater to the investors with different risk-return appetite.

3 Proposed Method

The proposed approach follows a canonical evolutionary framework as shown in Algorithm 1. The algorithm uses customized representation and operators (including initialization, offspring generation) to suit the constrained multi-objective portfolio optimization problem considered. The intent behind the customization is to improve the convergence rate, and they assume no more prior knowledge about the problem than already given in the existing works (e.g. [3]). The key components of the algorithm are discussed in the following subsections.

Algorithm 1. Proposed method for multiobjective portfolio optimization

```
Input: Problem data (See Sect. 2), Population size ($N_p$), Total evaluations ($FE_{max}$)
Output: PF approximation
Set #function evaluation $FE = 0$, archive of evaluated solutions $\mathcal{A} = \emptyset$, generation count $t = 0$
Initialize the population $P^0$ (of size $N_p$) with the tailored method in Sect. 3.2, and evaluate all solutions. Update $\mathcal{A}$, $FE$
while ($FE < FE_{max}$) do
    Generate offspring solutions $C^t$ from current population $P^t$ in three different ways with probabilities, $pr_1$, $pr_2$ and $pr_3$ respectively.
    Evaluate offspring solutions $C^t$ Update $\mathcal{A}$, $FE$
    Rank $T^t = P^t \cup C^t$
    Select top $N_p$ individuals in $T^t$ as the next generation population $P^{t+1}$ Update $pr1$, $pr2$, $pr3$

    Set $t = t + 1$
end while
Output the non-dominated solutions $\mathcal{N} \in \mathcal{A}$ as the PF approximation
```

3.1 Solution Representation

From Sect. 2, one can observe that the solution vector under the standard notations includes $2N$ decision variables. The first N variables $s_i, i = 1, \ldots, N$, are binary, indicating whether i^{th} is selected or not; whereas the next N variables $w_i, i = 1, \ldots, N$, are discrete, indicating the proportion of the capital invested in i^{th} asset. Given that the given number of assets (N) may often comprise a large set, this representation gives rise to a large number of variables.

In this work, we propose and investigate a compact representation, which consists of merely $2K$ variables. The first K variables $a_i, i = 1, \ldots, K, 1 \leq a_i \leq N$, are *categorical*, indicating the labels (ids) of the chosen assets. The next K variables $q_i, i = 1, \ldots, K$, are discrete (positive integers), representing the *lot size* acquired of these K assets.

Note that the above representation helps in implicitly satisfying some of constraints and avoiding repair operators. Since the round lot $v_i = 0.008$, the total number of trading lots can be calculated as $p = \sum_i^N q_i = 1/0.008 = 125$. Now, since, floor $\epsilon_i = 0.01$ and ceiling $\delta_i = 1.0$, therefore, $2 \leq q_i \leq 125$. Thus, having q_i as integers in the given range automatically satisfy the round lot and floor-ceiling constraints. Moreover, the problem converts to a form akin to distributing the available capital (given number of lots ($p = 125$)) lots to a selected number (K) of assets. Such types of problems commonly occur in probability theory, and there is an opportunity to leverage some of the existing techniques to solve them (as detailed in further subsections).

3.2 Initialization

The initialization of the solutions occurs in two parts. First half is a customized initialization to obtain solutions close to the solution with the best return. Of these the one with the maximum return is *guaranteed globally* for the problem, while the others are created by perturbing it. The second half is generated randomly.

Initializing the Best Return Solution(s): In order to maximize the overall return, one would need to buy the maximum quantity of the asset with highest return, while satisfying the prescribed constraints discussed in Sect. 2. We do so in these steps:

- Sort the assets in the decreasing order of their expected return
- Identify the top K assets in the sorted list, and assign the minimum trading lot ($q_i = 2$) to each. This list is referred to as S_K
- If the pre-assigned asset (z_{30}) already exists in the above list, move to next step, otherwise replace the last asset in S_K by the pre-assigned asset.
- Thereafter, assign $q_i = p - 2 \times (K - 1)$ lots to the first asset in S_K (the one with the largest return) and assign $q_i = 2$ lots to the remaining $K - 1$ assets. This assignment satisfies all constraints, sums up to p lots traded, and assigns maximum possible lots to the highest return asset. Thus, it constitutes the solution with maximum overall return, denoted by x_{maxR}
- For generating the rest of the $N_p/2 - 1$ solutions of the first half, the quantity of the first asset in S_k is decreased by 1, while that of the second asset is increased by 1 to construct each subsequent solution.

Initializing the Remaining Solutions: The remaining $N_p/2$ solutions are initialized randomly. For generating each solution, $K - 1$ assets are selected randomly, and the pre-assigned asset z_{30} is appended to the list to complete the selection. As for their lot sizes, the total p round lots are divided among K assets using the *stars and bars* (SaB) approach in probability theory [7]. This particular instance is equivalent to dividing n indistinguishable objects into m partitions. SaB approach solves this problem using a graphical aid by representing n objects as indistinguishable symbols ('stars') in a single row, and placing $m - 1$ separators ('bars') between them to specify the number assigned to each bin. For example, for dividing 5 objects in 3 bins, 2 bars can be placed between the them in different positions. Assuming each bin needs to have at least 1 object, the bars cannot be placed at either of the ends of the row. Thus, a total of $\binom{n-1}{m-1}$ possibilities exist for such an assignment. Correspondingly, for the given portfolio problem, there are $\binom{p-2K-1}{K-1}$ distributions possible where each chosen asset has the minimum round lot $q_i = 2$ allocated. One of such combinations is chosen randomly, that satisfies the minimum round lot constraint.

3.3 Creating Offspring Solutions

The method proposed to create of the offspring solutions attempts to generate offspring solutions targeting different parts of the PF approximation, i.e., some toward low risk solutions, some towards high return solutions, and some across whole PF. The probability of generating the solution in either of these parts is controlled via a set of self-adapting parameters $\{pr_1, pr_2, pr_3\}$, such that $\sum pr_i = 1$. The values are initialized to be equal at the beginning of the run, i.e., $\{1/3, 1/3, 1/3\}$.

The offspring solutions are generated through pairwise recombination among the current population. For two parent solutions px_1 and px_2, the recombination proceeds as follows:

- Let us denote the assets in px_1 and px_2 as apx_1 and apx_2, respectively. The common assets between the two parents are identified as $ap_c = apx_1 \cap apx_2$ Correspondingly, the assets that are not common between the two parents can be identified as $ap_{u1} = apx_1 \backslash ap_c$ and $ap_{u2} = apx_2 \backslash ap_c$.
- The assets of the offspring solutions are then assigned as follows. The common assets (ap_c) are kept in both offspring solutions. The remaining $K - |ap_c|$ assets are chosen randomly from $ap_{u1} \cup ap_{u2}$, while ensuring no asset is assigned to both offspring solutions.
- The lot size for each asset for the offspring solutions are assigned as follows. The lot sizes of the common assets are directly inherited from the parent solutions. For the remaining (not common) assets, first of all, the lot sizes vector $q_i, i = 1 \ldots |K - |ap_c||$, are generated using the same approach as used in initialization (i.e., SaB approach). Then,
  - With probability pr_1, the lot sizes are assigned in decreasing order to the assets with highest to lowest return. This is to encourage generation of solutions towards the PF extremity with high return.
  - With probability pr_3, the lot sizes are assigned in decreasing order by picking the assets with lowest expected risk based on the risk covariance matrix σ. This is to encourage generation of solutions towards the extremity of the PF with lowest risk.
  - With probability pr_2, the lot sizes are assigned randomly, to encourage the solutions across the PF with no bias.

Adjustment of Probabilities $\{pr_1, pr_2, pr_3\}$: After each generation, the probabilities $\{pr_1, pr_2, pr_3\}$ are adjusted by the algorithm to allow focus on unexplored regions as per the above conditions. The adjustment is done as follows:

- The curve length C_l of the current non-dominated set (in the objective space) is calculated assuming it to be piece-wise linear among the non-dominated points.
- The number of solutions in the left (min. risk), middle, and right (max return) are identified as n_1, n_2, n_3 are counted as those within length $l \leq C_l/3$, $C_l/3 \leq l \leq 2C_l/3$ and $2C_l/3 \leq l \leq C_l$ from the one extremity (say min. risk) along the curve. These numbers are then normalized as $\{N_1, N_2, N_3\} = exp(\{n_1, n_2, n_3\}/(n_1 + n_2 + n_3))$. In the final step, the probabilities are calculated as $\{pr_1, pr_2, pr_3\} = \frac{\{1/N_1, 1/N_2, 1/N_3\}}{1/N_1 + 1/N_2 + 1/N_3}$.

The rationale behind the above adjustments is to simply map the sections of the non-dominated set with fewer solutions to high probability of exploration and vice-versa. Other mathematical forms that achieve similar mapping can also be used.

3.4 Ranking and Reduction

For ranking the population, the widely used non-dominated sorting and crowding distance are used [5]. The former targets convergence towards the PF and the latter helps promote diversity among the non-dominated solutions during the search process.

4 Empirical Study

In this section, we discuss the experimental set-up and the results of the benchmarking.

4.1 Experiment Configuration

Two sets of test problems are considered here. One is established recently based on the data from Yahoo Finance website (D1−15) with the number of available assets as high as 2235. The other one is a classic but small scale (D16−20) benchmark suite[1]. Their details are given in Table 1.

Table 1. Twenty benchmark instances considered in this study

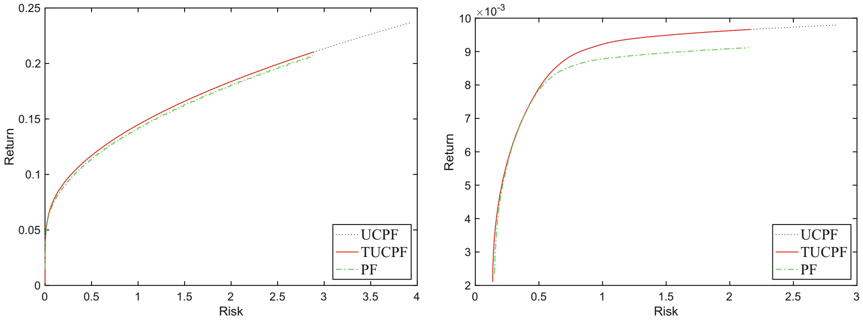
Instance	Origin	Name	#Assets	Instance	Origin	Name	#Assets
$D1$	Korea	KOSPI Composite	562	$D11$	USA	NASDQ Industrial	808
$D2$	USA	AMEX Composite	1893	$D12$	USA	NASDQ Telecom	139
$D3$	USA	NASDAQ	2235	$D13$	USA	NYSE US100	94
$D4$	Australia	All ordinaries	264	$D14$	USA	NYSE World	170
$D5$	Italy	MIBTEL	167	$D15$	USA	S&P 500	469
$D6$	UK	FTSE ACT250	128	$D16$	Hong Kong	Hang Seng	31
$D7$	USA	NASDAQ Bank	380	$D17$	Germany	DAX100	85
$D8$	USA	NASDAQ Biotech	130	$D18$	UK	FTSE 100	89
$D9$	USA	NASDQ Computer	417	$D19$	US	S&P 500	98
$D10$	USA	NASDQ Financial00	91	$D20$	Japan	Nikkei	225

For benchmarking with peer algorithms, the popularly used multiobjective performance metric, Hypervolume (HV) [16], is used in a normalized objective space. HV measures the volume of dominated space by a non-dominated set Q relative to a reference point r. The reference point needs to be carefully chosen for computing HV, to ensure that the contributions of the extreme points are counted but do not overwhelmingly contribute more than those of the other points in the set. Thus it is recommended to set r slightly dominated by the nadir point of the true PF (where known). In this work, we have set $r = (1.2, 1.2)$ for this reason in the normalized space. Note that while calculating HV, the second objective (maximize return) is also converted to minimization by negating the values. The second key aspect is to select the normalization bounds.

[1] Datasets can be accessed at https://github.com/CYLOL2019/Portfolio-Instances.

For the datasets from OR-Library (D16–D20), the true PFs can be obtained with a quadratic approach using CPLEX and a mathematical multiobjective method [12]. Therefore, true ideal and nadir points of the true PF are used as normalization bounds for these cases. However, the true PFs are unavailable for some problems in NGINX dataset (D1–D15) due to large number of assets. For these cases, the best known unconstrained Pareto fronts (UCPFs) are considered with the following modification. They are truncated by the best return solution (known to be feasible and globally optimal from Sect. 3). This truncated version of the UCPF is referred to as TUCPF. The PFs, UCPFs and TUCPFs for selected problems are depicted in Fig. 1. It can be seen that PFs and TUCPFs are significantly different from the UCPFs for both instances. Many solutions of UCPFs are infeasible. For instance, on D6, the return for UCPF ranges from 0 to 4, however, the solutions with return higher than 3 are all unavailable. On the other hand, although PFs are very close to TUCPFs on D6, the differences are distinct on D17. These results demonstrate the necessity of utilizing the PFs (where available) or TUCPFs (where PFs are not available) instead of the UCPFs contrary to some past studies that have used UCPFs. Since true PFs are unknown for some problems, we utilize the bounds of TUCPFs for normalizing the obtained PF approximations in those cases.

The proposed method is compared with two recent peer algorithms, namely MODEwAwL [10] and CCS [3]. The number of objective function evaluations are limited to 5000, and each algorithm is run 31 times with a population size of 100 for gathering the statistics of performance (HV).



(a) Three PFs on D6

(b) Three PFs on D17

Fig. 1. The three different PFs.

4.2 Results and Comparisons

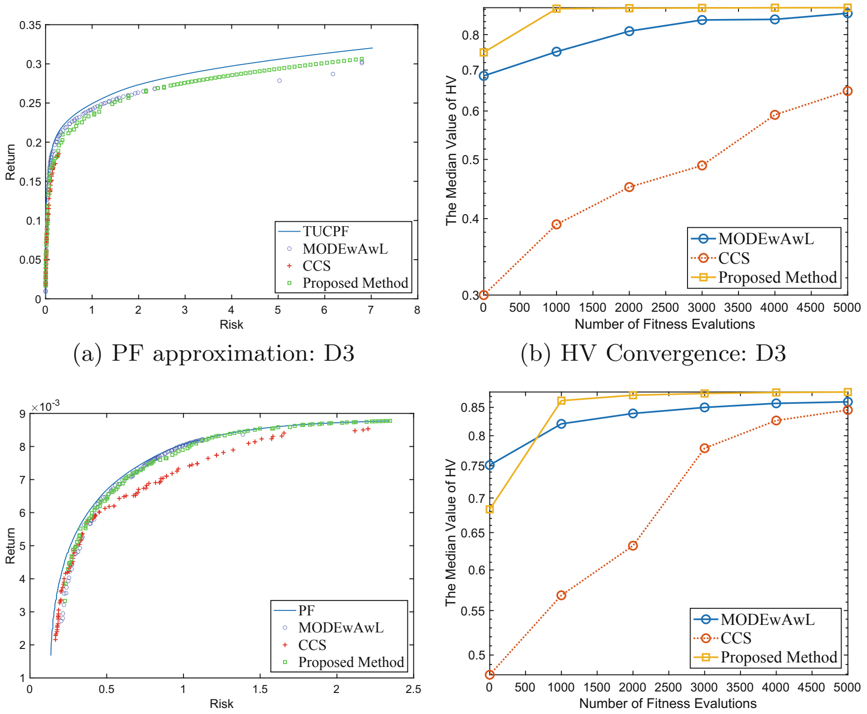
The statistics of HV obtained using the three algorithms are shown in Table 2. The rank and mean rank of each algorithm on all instances is listed in parenthesis [∗] and last row, respectively; where lower value indicates better performance. The best result of each instance is highlighted in gray. The symbol “+/−/=”

indicates better, worse, and equal performance relationship between paired algorithms in terms of Wilcoxon rank-sum test (5% significant level). Lastly, if a PF is unavailable (for HV normalization bounds calculation) after running the mathematical approach for 7 days, the datasets are shown with a star(*). In such cases, TUCPFs are used for HV normalization bounds during benchmarking. We note that the problems are unsolvable with a 7 days time limit for the mathematical approaches when the available number of assets exceed 225.

Table 2. HV obtained using the three algorithms. Note: for brevity, we denote MODEwAwL, CCS and proposed method with 'A', 'B' 'C' respectively here.

Data\ Alg.		A	B	C	Data\ Alg.		A	B	C
$D1^*$	Median	7.64e-01[3]	7.84e-01[2]	8.07e-01[1]	$D11^*$	Median	8.57e-01[2]	7.77e-01[3]	8.83e-01[1]
	Std	4.04e-02	9.49e-02	1.44e-04		Std	2.68e-02	1.08e-01	1.60e-03
$D2^*$	Median	7.80e-01[2]	7.70e-01[3]	7.80e-01[1]	$D12$	Median	8.38e-01[2]	8.11e-01[3]	8.56e-01[1]
	Std	1.10e-02	1.25e-01	1.90e-04		Std	3.71e-02	1.14e-01	3.01e-04
$D3^*$	Median	8.67e-01[2]	6.47e-01[3]	8.85e-01[1]	$D13$	Median	8.45e-01[2]	8.11e-01[3]	8.45e-01[1]
	Std	2.74e-02	1.03e-01	3.68e-04		Std	1.48e-02	4.62e-02	4.00e-03
$D4^*$	Median	8.68e-01[2]	7.40e-01[3]	8.96e-01[1]	$D14$	Median	8.42e-01[1]	7.54e-01[3]	8.39e-01[2]
	Std	3.72e-02	1.12e-01	2.63e-04		Std	3.42e-02	1.22e-01	2.26e-03
$D5$	Median	8.91e-01[2]	8.39e-01[3]	8.92e-01[1]	$D15^*$	Median	9.10e-01[2]	8.04e-01[3]	9.15e-01[1]
	Std	6.56e-03	3.35e-02	5.68e-03		Std	9.08e-03	9.43e-02	3.88e-03
$D6$	Median	8.07e-01[1]	8.01e-01[3]	8.06e-01[2]	$D16$	Median	8.13e-01[3]	8.32e-01[1]	8.24e-01[2]
	Std	1.32e-02	7.01e-03	5.20e-04		Std	3.52e-02	2.34e-03	1.11e-03
$D7^*$	Median	8.65e-01[1]	8.33e-01[3]	8.63e-01[2]	$D17$	Median	8.97e-01[2]	8.84e-01[3]	9.01e-01[1]
	Std	1.85e-02	1.12e-01	2.53e-04		Std	3.11e-02	1.86e-02	4.32e-03
$D8$	Median	8.40e-01[1]	8.28e-01[3]	8.29e-01[2]	$D18$	Median	8.33e-01[1]	8.19e-01[2]	8.19e-01[3]
	Std	3.03e-02	6.94e-03	3.11e-03		Std	1.70e-02	3.26e-02	4.22e-03
$D9^*$	Median	8.44e-01[1]	6.99e-01[3]	8.39e-01[2]	$D19$	Median	8.60e-01[2]	8.45e-01[3]	8.78e-01[1]
	Std	2.97e-02	1.20e-01	1.67e-03		Std	1.12e-02	2.72e-02	2.62e-03
$D10$	Median	8.99e-01[2]	8.77e-01[3]	9.07e-01[1]	$D20$	Median	8.82e-01[1]	7.84e-01[3]	8.47e-01[2]
	Std	6.03e-03	1.92e-02	1.58e-03		Std	8.29e-03	2.29e-02	1.60e-02
MeanRank		1.8(A)	2.8(B)	1.4(C)	+/-/=		9/4/7(C&A)	16/1/3(C&B)	-

The results show that the proposed algorithm performs first or second in all problems with an exception of D18. Furthermore, it achieves the best mean rank (1.4) among the three approaches. With regards to the Wilcoxon rank-sum tests, the proposed algorithm outperforms the others on most problems. For example, it shows better performance over MODEwAwL and CCS for large number of instances (9 and 16 times respectively), while it is worse on small number of instances (4 and 1 respectively). Furthermore, the differences are also clear from Fig. 2 in terms of the PF approximations obtained corresponding to the median HV. Due to space limitations, we only show the results for two instances, D3 and D19; but others show a similar trend. The figures show the improved PF approximation obtained by the proposed algorithm. It is able to find solutions covering most regions of the PF, especially the high return end. For example, in Fig. 2(a), the PF approximations of MODEwAwL and CCS only focus on the low risk area, whereas the proposed algorithm achieves much



(a) PF approximation: D3

(b) HV Convergence: D3

(c) PF approximation: D19

(d) HV Convergence: D19

Fig. 2. PF approximations and the corresponding HV convergence curves.

better coverage. Furthermore, the HV convergence plots indicate the significant speedup using the proposed method. It converges to the near-final PF within 1000 fitness evaluations, while the other two algorithms require 5000 fitness evaluations to get to equivalent stage.

On the other end of the spectrum, one relative shortcoming of the proposed algorithm also seems apparent, which is in obtaining the solutions on the low-risk end. For instance, in Fig. 2(c), the PF approximation of CCS can achieve much better low-risk solutions (below 2.5e−4). Moreover, we also observed in preliminary experiments that when large number of evaluations are made available (say 40,000–100,000), the relative advantage of the proposed algorithm is not as significant as its performance for lower evaluation budgets.

To summarize, initial experiments establish that the proposed algorithm is competitive to the state-of-the-art forms and shows significant promise in achieving fast convergence. However, a more in-depth investigation, extensive experimentation and parametric studies need to be further conducted to understand the impact of each component individually, and to improve their performance.

5 Conclusion and Future Work

This paper proposes a new compact representation and initialization for multi-objective portfolio optimization problem. The proposed approach can implicitly satisfy some of the constraints, avoiding the need for specialized repair operators, and can guarantee to obtain one of the extremities (highest return) of the PF. Numerical experiments are conducted on a range of problems with assets up to 2235, and results compared with two other established algorithms. The proposed algorithm shows favorable results for most problems, especially demonstrating remarkable fast convergence rate when compared to the peer algorithms. While the performance is promising, there is further scope for improving performance in terms of obtaining low-risk solutions and further improving its performance for higher computational budgets, which will be studied in the future work.

Acknowledgement. The authors would like to thank China Scholarship Council and UNSW Research Practicum program for supporting the work.

References

1. Bertsimas, D., Shioda, R.: Algorithm for cardinality-constrained quadratic optimization. Comput. Optim. Appl. **43**(1), 1–22 (2009)
2. Chang, T.J., Meade, N., Beasley, J.E., Sharaiha, Y.M.: Heuristics for cardinality constrained portfolio optimisation. Comput. Oper. Res. **27**(13), 1271–1302 (2000)
3. Chen, Y., Zhou, A., Das, S.: A compressed coding scheme for evolutionary algorithms in mixed-integer programming: a case study on multi-objective constrained portfolio optimization (2019)
4. Chen, Y., Zhou, A., Dou, L.: An evolutionary algorithm with a new operator and an adaptive strategy for large-scale portfolio problems. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 247–248. ACM (2018)
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
6. Deb, K., Steuer, R.E., Tewari, R., Tewari, R.: Bi-objective portfolio optimization using a customized hybrid NSGA-II procedure. In: Takahashi, R.H.C., Deb, K., Wanner, E.F., Greco, S. (eds.) EMO 2011. LNCS, vol. 6576, pp. 358–373. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19893-9_25
7. Feller, W.: An introduction to probability theory and its applications (1957)
8. Gaspero, L.D., Tollo, G.D., Roli, A., Schaerf, A.: Hybrid metaheuristics for constrained portfolio selection problems. Quant. Finan. **11**(10), 1473–1487 (2011)
9. Kolm, P.N., Tütüncü, R., Fabozzi, F.J.: 60 years of portfolio optimization: practical challenges and current trends. Eur. J. Oper. Res. **234**(2), 356–371 (2014)
10. Lwin, K., Qu, R., Kendall, G.: A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization. Appl. Soft Comput. **24**, 757–772 (2014)
11. Markowitz, H.: Portfolio selection. J. Finan. **7**(1), 77–91 (1952)
12. Mavrotas, G., Florios, K.: An improved version of the augmented ε-constraint method (augmecon2) for finding the exact pareto set in multi-objective integer programming problems. Appl. Math. Comput. **219**(18), 9652–9669 (2013)

13. Newman, A.M., Weiss, M.: A survey of linear and mixed-integer optimization tutorials. INFORMS Trans. Educ. **14**(1), 26–38 (2013)
14. Noceda, J., Wright, S.: Numerical optimization. Springer Series in Operations Research and Financial Engineering. ORFE. Springer, Cham (2006). https://doi.org/10.1007/978-0-387-40065-5
15. Pouya, A.R., Solimanpur, M., Rezaee, M.J.: Solving multi-objective portfolio optimization problem using invasive weed optimization. Swarm Evol. Comput. **28**, 42–57 (2016)
16. While, L., Hingston, P., Barone, L., Huband, S.: A faster algorithm for calculating hypervolume. IEEE Trans. Evol. Comput. **10**(1), 29–38 (2006)