Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 1970 – 1981

doi:10.3906/elk-1807-18

Identifying preferred solutions in multiobjective combinatorial optimization problems

Banu LOKMAN[1*], **Murat KÖKSALAN**[2]

[1]Portsmouth Business School, University of Portsmouth, Portsmouth, UK

[2]Department of Industrial Engineering, Faculty of Engineering, Middle East Technical University, Ankara, Turkey

Received: 03.07.2018 • **Accepted/Published Online:** 26.03.2019 • **Final Version:** 15.05.2019

Abstract: We develop an evolutionary algorithm for multiobjective combinatorial optimization problems. The algorithm aims at converging the preferred solutions of a decision-maker. We test the performance of the algorithm on the multiobjective knapsack and multiobjective spanning tree problems. We generate the true nondominated solutions using an exact algorithm and compare the results with those of the evolutionary algorithm. We observe that the evolutionary algorithm works well in approximating the solutions in the preferred regions.

Key words: Evolutionary algorithm, preferred region, nondominated frontier, multiobjective combinatorial optimization

1. Introduction

Multiobjective combinatorial optimization (MOCO) problems have applications in a wide variety of areas such as investment and resource planning, logistics, facility location, scheduling, telecommunication and communications systems, and energy since they represent real-life situations in large organizations. The decision-makers usually have conflicting objectives and hence there does not exist a unique optimal solution, and the aim is to generate nondominated solutions for which an improvement in one of the objectives is not possible without sacrificing from other objectives.

Due to the rapid growth of nondominated solutions with the problem size, many heuristic approaches have been developed for MOCO problems [see 1]. Modern heuristic searches in general and evolutionary algorithms in particular have been widely used [see 2–4]. In [5], Ehrgott and Gandibleux reviewed some classical metaheuristics for MOCO problems and focused on the hybrid metaheuristics that combine exact and heuristic approaches. Many approaches try to approximately generate the whole nondominated frontier. As discussed in [6], incorporating the preferences of the decision-maker (DM) into evolutionary multiobjective optimization is important since it is not practical to generate all nondominated solutions and present them to the DM. The work in [7] presents a review of preference incorporation in multiobjective evolutionary algorithms and discusses preference models and implementation strategies. The authors point out that the application procedure plays a vital role for the preference models and scalability issue according to the criteria is the main concern.

The authors of [8] developed the first interactive evolutionary method (IEM) for multiobjective problems that tries to converge the preferred solutions by obtaining preference information from the DM progressively. The work in [9] also develops an interactive evolutionary algorithm. In the interaction process, the DM selects his or her best solution among a set of representative solutions and the algorithm guides the solution effort

*Correspondence: banu.lokman@port.ac.uk

to the neighborhood of this solution. The algorithm defines a territory around each solution in the archive population and a solution is only accepted if it does not violate this territory. The diversity of the population is maintained by this territory defining property. The work in [10] develops an evolutionary metaheuristic to approximate preference-nondominated points (EMAPS) for MOCO problems. EMAPS differs from guided-search evolutionary algorithms since the preference information is obtained by using qualitative statements. EMAPS uses this information in an individualized fitness function and evolves a population of solutions to focus on the preferred regions of the nondominated frontier.

Recently, there are a number of hybrid approaches that aim to generate a set of points representing the nondominated frontier. For example, [11] integrates an epsilon-constraint method into an evolutionary algorithm and generates points from different parts of the nondominated frontier by changing the thresholds on each objective, while [12] incorporates self-organizing maps into an evolutionary algorithm. The authors define a neighborhood relation by self-organizing maps and the evolutionary algorithm iteratively generates new points using two neighbors. Although these approaches are designed to generate representative points, they do not incorporate the preferences of the DM into the solution process.

Multiobjective knapsack and multiobjective spanning problems are combinatorial optimization problems that have many practical applications. Examples for multicriteria knapsack problems can be found in capital budgeting, investment planning, and resource planning applications. The authors of [13] develop a genetic local search algorithm for MOCO problems and test the performance of the algorithm on MOKPs. The algorithm aims to generate a set of approximately efficient solutions from all regions of the efficient frontier in a reasonable time. The work in [14] develops a favorable weight-based evolutionary algorithm (FWEA) to obtain well-distributed solutions close to the nondominated frontier, while [15] adapts the FWEA to approximate the nondominated frontier of the multidimensional multiobjective knapsack problem (MMOKP). The work in [16] also proposes a multiobjective Chebyshev-based genetic algorithm, MOTGA, for MMOKP. The MOTGA focuses on different parts of the nondominated frontier at each stage to obtain a good approximation of a set of nondominated solutions. Multiobjective spanning tree (MOST) problems also have many real-life applications including communication networks, electric power systems, and network designs. The work in [17] proposes a greedy randomized adaptive search procedure (GRASP) algorithm for the MOST problem while [18] develops a nongenerational genetic algorithm (GA) for MOST problems. In [19], Zhou and Gen developed a genetic algorithm for MOST. The algorithm aims at generating all nondominated solutions close to the ideal point or well-distributed solutions along the nondominated frontier according to the preferences of the DM. Reference [20] reviews some approaches that incorporate the preferences of the DM and argues that there is a need for more work to be done, especially for preference-based MOCO problems with more than two criteria.

In this paper, we develop an evolutionary algorithm that aims to converge the region that is of interest to the DM. We test the performance of the algorithm on the MMOKP and MOST problems. We compare the obtained solutions with true nondominated solutions generated with an exact algorithm that we developed. We give some definitions in Section 2 and develop the evolutionary algorithm in Section 3. We present our computational experiments in Section 4 and then draw conclusions in Section 5.

2. Definitions

A general multiobjective problem can be formulated as:

(P):

$$\begin{aligned} \text{“Max”} \quad & z(\mathbf{x}) = (z_1(\mathbf{x}), z_2(\mathbf{x}), \ldots, z_q(\mathbf{x})) \\ & \text{subject to} \\ & \mathbf{x} \in \mathbf{X}, \end{aligned}$$

where $\mathbf{x}$ denotes a decision vector in the solution space, $\mathbf{X}$, and q is the number of the objectives. Since the maximization of a vector is not a well-defined mathematical operation, the quotation marks are used. Let $\mathbf{Z}$ be the feasible set in the criterion space and $\mathbf{z} = (z_1, \ldots, z_q)$ denote a point in the criterion space with a criterion value of z_i for the ith criterion. Point $\mathbf{z^2} \in \mathbf{Z}$ is said to dominate $\mathbf{z^1} \in \mathbf{Z}$ if $\mathbf{z^2} \geq \mathbf{z^1}$ and $\mathbf{z^2} \neq \mathbf{z^1}$. If there does not exist such a $\mathbf{z^2}$, then point $\mathbf{z^1}$ is said to be nondominated and the corresponding decision vector, $\mathbf{x^1}$, is said to be efficient. The set of all nondominated solutions defines the nondominated (efficient) frontier. $\mathbf{z}^{IP} = (z_1^{IP}, \ldots, z_q^{IP})$ and $\mathbf{z}^{NP} = (z_1^{NP}, \ldots, z_q^{NP})$ denote the ideal and nadir point, respectively. The ideal point corresponds to $z_i^{IP} = \text{Max}_{\mathbf{x} \in \mathbf{X}}\, z_i(\mathbf{x})$ for a maximization problem. If $\mathbf{E}$ denotes the set of efficient solutions, then $z_i^{NP} = \text{Max}_{\mathbf{x} \in \mathbf{E}}\, z_i(\mathbf{x})$ for a maximization problem.

3. The evolutionary algorithm

We design a preference-based evolutionary algorithm, PBEA, to generate approximately nondominated solutions in a region defined by an upper (lower) bound for each criterion where criteria are minimization (maximization) type. Assuming q criteria, we define the vector of these bounds $\mathbf{b} = (b_1, \ldots, b_q)$ as the reference point. We maintain a regular population, P_t, and an elite population, E_t, in each generation t. The latter contains only the solutions in the desired region defined by the bounds.

3.1. Fitness assignment

The work in [21] defines a hypersurface to approximate the locations of the nondominated solutions. It first generates several nondominated (or approximately nondominated) solutions and then fits a hypersurface that passes through these solutions. It scales the objective function values, z_k , $k = 1, \ldots, q$, and denotes the scaled objective values as z'_k , $k = 1, \ldots, q$, by using $z'_k = \dfrac{z_k - z_k^{IP}}{z_k^{NP} - z_k^{IP}}$ such that points $(0, \ldots, 0)$ and $(1, \ldots, 1)$ correspond to the ideal and nadir points, respectively. After scaling the objective function values between 0 and 1, the hypersurface is defined as $\sum_{k=1}^{q}(1 - z'_k)^p = 1$ for some $p > 0$. The authors demonstrate on a variety of MOCO problems that this hypersurface represents the efficient solutions well. We utilize the same idea for the purpose of assigning fitness values to solutions. We treat $1 - \sum_{k=1}^{q}(1 - z'_k)^p$ as the contour on which the solution lies, where smaller values are desirable as they correspond to contours close to the contour of the ideal point.

3.2. Initialization

Initially, we generate $(q + 1)$ ‘seed’ solutions that are ‘close’ to the nondominated frontier. This can be done using problem-specific techniques or heuristic procedures. Then we generate random solutions until the preset maximum population size is achieved. We insert each solution into E_0 or P_0 by checking whether it is in the preferred region or not.

3.3. Evolution

The algorithm creates the offspring from two parents where the first and second parents are probabilistically selected from E_t and $P_t \cup E_t$, respectively. If an offspring is not dominated and is in the preferred region, it is inserted in E_{t+1}. Then the members that are dominated by the new member are removed from E_{t+1}. If the maximum elite population size is achieved, the solution with the lowest fitness score in E_{t+1} is removed from the elite population. The offspring that is not in the preferred region is inserted in P_{t+1}. P_{t+1} is maintained by eliminating the solution having the minimum fitness value if all its members are nondominated relative to each other. If there are dominated members, then the elimination is done among them, again based on fitness. Evolution continues for a predetermined number of generations.

3.4. Parent selection

After assigning the fitness score to each member of E_t, we rank the solutions according to their fitness scores. Then the first parent is selected probabilistically with respect to its rank as suggested by [22]. Similarly, the second parent is selected from solutions in $P_t \cup E_t$ probabilistically based on their fitness scores. If $|E_t| = 0$, we calculate the total deviation of each solution $\mathbf{z} = (z_1, \ldots, z_q)$ that is nondominated relative to the current population from the reference point $\mathbf{b} = (b_1, \ldots, b_q)$ considering only the criteria in which the former is worse. That is, denoting by B_k the index set of criteria for which $z_k > b_k$ $(z_k < b_k)$ for minimization (maximization) type criteria, we calculate the deviation as $d(\mathbf{z}, \mathbf{b}) = \sum_{k \in \mathbf{B}_k} \frac{z_k - b_k}{z_k^{NP} - z_k^{IP}}$. Then we rank these solutions in the ascending order of $d(\mathbf{z}, \mathbf{b})$ and assign a selection probability based on their ranks.

3.5. The algorithm

We next present the steps of the PBEA.

4. Computational experiments

We apply the algorithm to the MMOKP and the MOST problem for two, three, and four objectives. We initialize the parameters as follows:

- N_E (Maximum elite population size) = 50
- N_P (Maximum population size) = 100
- G (Total number of generations to be completed) = 20,000 for MOKP and 40,000 for MOST problem (corresponds to 40,000 offspring in each case)
- p_c (Crossover probability) = 1 and p_m (mutation probability) = 0.9

4.1. Multidimensional multiobjective knapsack problem

We consider a multidimensional multiobjective knapsack problem having q knapsacks, m items, and q objectives:

Algorithm 1 PBEA

Step0: *Initialization*
Initialize the parameters N_E, N_P, G, p_c, p_m, t where:
N_E: Maximum elite population size
N_P: Population size
G: Total number of generations to be completed
p_c: Crossover probability
p_m: Mutation probability
t: Generation counter

Step1: *Generation of the initial population*
Initialize $t = 0$. Generate the initial populations, E_0 and P_0.

Step2: *Selecting the first parent*
Step2.1: If $|E_t| > 0$, then assign a selection probability to each member of E_t based on its fitness rank and select the first parent from E_t. Go to Step 3. Otherwise, go to Step 2.2.
Step2.2: Assign a selection probability to each currently nondominated solution $\mathbf{z} = (z_1, \ldots, z_q)$ in P_t based on its rank measured by its distance from the reference point, $d(\mathbf{z}, \mathbf{b}) = \sum_{k \in \mathbf{B}_k} \frac{z_k - b_k}{z_k^{NP} - z_k^{IP}}$, and select the first parent.

Step3: *Selecting the second parent*
Assign a selection probability to each member of $P_t \cup E_t$ based on its fitness rank and select the second parent from $P_t \cup E_t$.

Step4: *Crossover operation*
Employ crossover operator to the parents to generate offspring.

Step5: *Mutation*
Mutate each offspring with probability p_m and use the repair operator (if needed) to obtain feasible offspring. Assign fitness scores to the offspring and let $P_{t+1} = P_t$ and $E_{t+1} = E_t$.

Step6: *Insertion*
For each offspring, repeat the following steps and then go to Step 7. If the offspring is in the preferred region, go to Step 6.1. Otherwise, go to Step 6.2.
Step6.1: Insert the offspring in E_{t+1} and remove all solutions in E_{t+1} dominated by the offspring. If there does not exist any dominated solution and $|E_{t+1}| > N_E$, then remove the solution with the worst fitness score.
Step6.2: Insert the offspring in P_{t+1} and remove a solution based on fitness. If there are dominated solutions, consider only those solutions for elimination.

Step7: *Termination*
Set $t = t + 1$. If $t = G$, stop. Otherwise, go to Step 2.

$(MMOKP)$:

$$
\begin{aligned}
\text{“Max”} \quad & z(\mathbf{x}) = (z_1(\mathbf{x}), z_2(\mathbf{x}), \ldots, z_q(\mathbf{x})) \\
& \text{subject to} \\
& \sum_{j=1}^{m} w_{ij} x_j \leq C_i \qquad\qquad i = 1, \ldots, q \\
& z_i(\mathbf{x}) = \sum_{j=1}^{m} p_{ij} x_j \qquad\qquad i = 1, \ldots, q \\
& \mathbf{x} \in \mathbf{X},
\end{aligned}
$$

where p_{ij} denotes the profit of placing item j in the knapsack and w_{ij} denotes the capacity usage of item j in knapsack i. The binary variable x_j takes a value of 1 if item j is placed in the knapsacks and 0 otherwise. We set the capacity of the knapsacks to half of the total capacity requirement of all items, i.e. $C_i = \frac{\sum_{j=1}^{m} w_{ij}}{2}$. We generate random instances of MMOKP for which p_{ij} and w_{ij} are generated as uniformly distributed integers between 10 and 100. We use binary chromosome representation for MMOKP, where a gene takes the value of 1 if the corresponding item is placed in the knapsacks and takes the value of zero if not. We employ a randomly generated binary chromosome mask in the crossover operation. If the element of the mask is 1, the first offspring inherits the jth gene from the first parent and the second offspring inherits the jth gene from the second parent. Similarly, if the jth element of the mask is 0, then the first offspring and second offspring inherit the jth gene from the second and first parents, respectively. The mutation operator flips the current value of the selected gene. To generate seed solutions, we modify the single-objective, single-knapsack greedy algorithm by sorting the items in the descending order of $\frac{\sum_{i=1}^{q} \lambda_i^r p_{ij}}{\sum_{i=1}^{q} w_{ij}}$, where $\boldsymbol{\lambda}^r = (\lambda_1^r, \ldots, \lambda_q^r)$ denotes the rth weight vector given to each criterion. We define $q+1$ different weight vectors. With the first q vectors ($r = 1, \ldots, q$), we try to obtain solutions that represent the best values of objectives by setting $\lambda_r^r = 1 - (q-1)\epsilon$, $\lambda_i^r = \epsilon$ for $i \neq r$, where ϵ is an arbitrarily small positive constant. With the $(q+1)$st vector, we try to obtain a central solution by setting $\lambda_i^{q+1} = 1/q$ for $i = 1, \ldots, q$. Since it is possible to exceed the knapsack capacities or we may not fully utilize the knapsack capacities as a result of the crossover or mutation operators, we need a repair operator. Instead of adding items randomly when the knapsack capacities are not fully utilized, we search for the items iteratively and add the item that decreases the weighted Chebyshev distance from the ideal point most without exceeding knapsack capacities. Similarly, when the knapsack capacities are exceeded, we search for the items to remove in such a way that the weighted Chebyshev distance from the ideal point increases least. The weight vector, $\mathbf{w} = (w_1, \ldots, w_q)$, used in the Chebyshev distance is calculated using the direction from the ideal point towards the reference point as follows [see 23, p. 441]:

$$w_k = \begin{cases} \frac{1}{b_k - z_k^{IP}} \left[\sum_{j=1}^{q} \frac{1}{b_j - z_j^{IP}} \right]^{-1} & b_j \neq z_j^{IP} \quad \forall j \\ 1 & b_k = z_k^{IP} \\ 0 & b_k \neq z_k^{IP}, \quad \exists j : b_j = z_j^{IP}. \end{cases} \tag{1}$$

4.2. Multiobjective spanning tree problem

A spanning tree, T, defined on a graph, $G = (S, E)$, is a collection of $|S-1|$ edges connecting $|S|$ nodes without forming any cycles. The MOST problem can be formulated as follows:
$(MOST)$:

$$\begin{aligned} \text{“Min”} \quad & z(\mathbf{x}) = (z_1(T), z_2(T), \ldots, z_q(T)) \\ & \text{subject to} \\ & z_i(T) = \sum_{e \in E(T)} c_i(e) \qquad\qquad i = 1, \ldots, q \\ & T \in \tau(G), \end{aligned}$$

where $\tau(G)$ denotes the set of all spanning trees, $E(T)$ denotes the set of edges in T, and $(c_1(e), \ldots, c_q(e)$ is

the cost vector associated with edge e. We generate $c_k(e)$ values as uniformly distributed integers between 10 and 100. To generate seed solutions corresponding to each weight vector, λ^r, we use a greedy algorithm where the edges are sorted in ascending order of $\sum_{i=1}^{q} \lambda_i^r c_i(e)$ and those not creating a cycle with previously selected ones are selected one at a time until a spanning tree is formed. We use the same weight vectors as in the MMOKP. We keep the previous node information for each node where the direction is determined according to the last node in the tree. In the crossover operation, we take the union of edges of both parents and then try to select $|S-1|$ of them for which the weighted Chebyshev distance from the ideal point is minimum. We apply a modified version of the greedy algorithm, where the edges are sorted in the ascending order of $\max_{i=1,\ldots,q} w_i c_i(e)$ where the weight vector is also determined by using the direction from the ideal point towards the reference point (see Eq. (1)). The mutation operator removes a randomly chosen edge from the spanning tree and adds another edge that does not create a cycle. Different from the MMOKP, we obtain only one offspring from the two parents.

4.3. Results

In order to define the region and evaluate the results, we first generate all nondominated solutions using the algorithm developed in [24]. This algorithm efficiently generates the nondominated solutions for any multiobjective integer program. It iteratively introduces bounds to $q-1$ of the objectives and solves integer programs sequentially, eventually generating all nondominated solutions. By using these solutions on hand, we calculate upper (lower) bounds for each criterion for each minimization (maximization) problem. We apply the evolutionary algorithm on the MMOKP and the MOST problems with different lower or upper bound settings. In order to test the performance of the algorithm on different regions of each problem, we apply the algorithm to five replications of the same problem. In each replication, we first randomly generate a weight vector, $\mathbf{w} = (w_1, \ldots, w_q)$. Then, utilizing the true nondominated solutions that have already been generated, we find the first 50 nondominated solutions, $\mathbf{z}^n = (z_{n1}, \ldots, z_{nq})$ $n = 1, \ldots, 50$, that minimize the weighted Chebyshev distance from the ideal point, $WD(\mathbf{z}^n, \mathbf{z}^{IP}) = \max_{k=1,\ldots,q} w_k \frac{|z_{nk} - z_k^{IP}|}{|z_k^{NP} - z_k^{IP}|}$. That is, we find the 50 nondominated solutions in direction $\mathbf{d} = (\frac{1}{w_1}, \ldots, \frac{1}{w_q})$ from the ideal point. Then we set lower (upper) bounds to each criterion that cover all these 50 nondominated solutions for a given maximization (minimization) problem as demonstrated in Figure 1. We keep the number of nondominated solutions at the same value (50) in the region of interest in order to obtain uniformity among the problems solved.

Our experiments indicate that the algorithm converges the desired regions very well in all the problems we solved. The solutions we find either coincide with the true nondominated solutions or are very close to them in the desired regions. Figure 2 demonstrates how well the solutions are represented by the set of solutions in the elite population on several instances we solved. While Figure 2a and Figure 2b illustrate the results for bicriteria problems, the solutions for the three-criteria problems are shown in Figures 2c and 2d.

For each nondominated solution in the preferred region, $\mathbf{z}^n = (z_{n1}, \ldots, z_{nq})$, $n = 1, \ldots, N$, we find a representative solution, $\mathbf{rz}^n = (rz_{n1}, \ldots, rz_{nq})$, in E_G such that the representative solution is at minimum distance from the corresponding nondominated solution. We use the Chebyshev metric to find the distance between the nondominated point and the representative point, i.e. we use $D(\mathbf{z}^n, \mathbf{rz}^n) = \max_{k=1,\ldots,q} \frac{|z_{nk} - rz_{nk}|}{|z_k^{NP} - z_k^{IP}|}$. In order to evaluate whether the nondominated solutions in the preferred region are represented well or not by the solutions in E_G, we calculate the average Chebyshev distance by using $AvgDist = \frac{\sum_{n=1}^{N} D(\mathbf{z}^n, \mathbf{rz}^n)}{N}$,

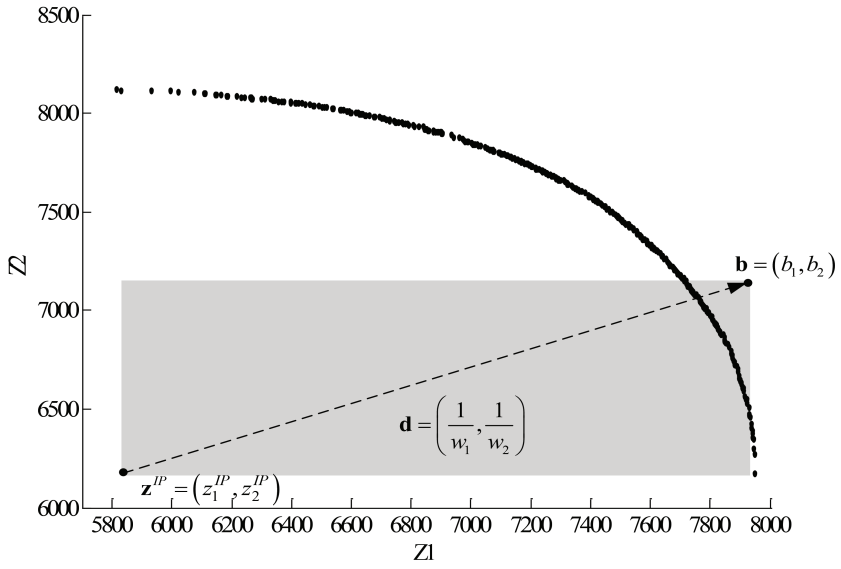
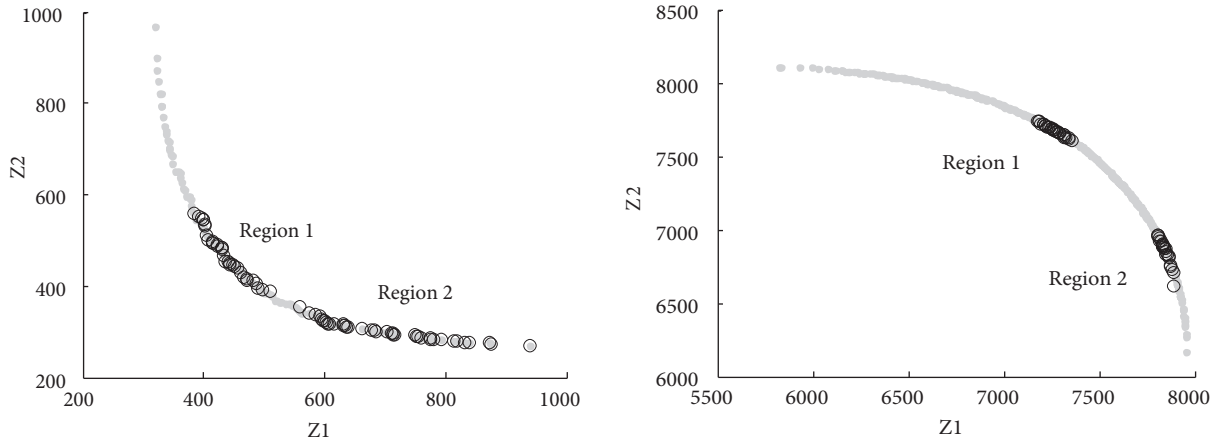
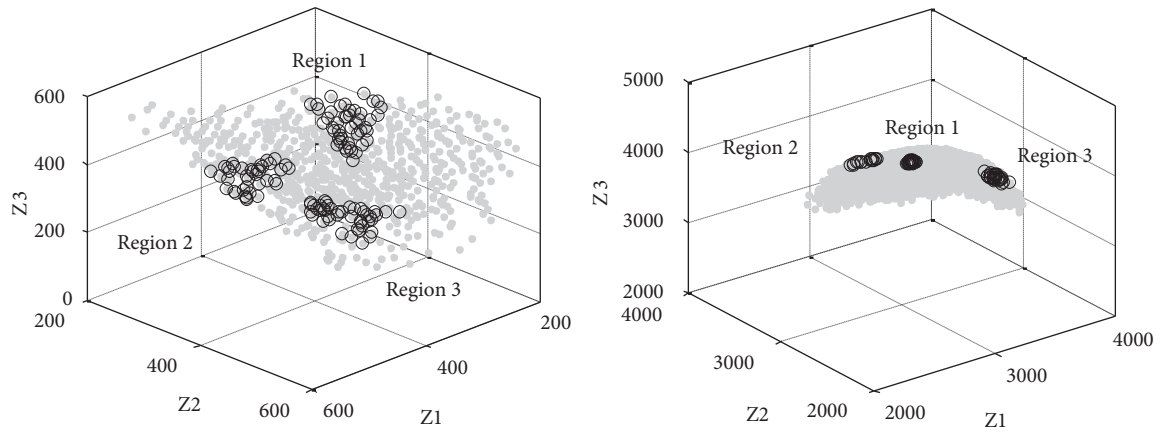


Figure 1. Defining a region of interest for a minimization problem.



a. MOST Problem with 2 objectives, 20 nodes, 139 nondominated solutions b. MMOKP with 2 objectives, 200 items, 414 nondominated solutions



c. MOST Problem with 3 objectives, 10 nodes, 704 nondominated solutions d. MMOKP with 3 objectives, 100 items, 8288 nondominated solutions

Figure 2. Demonstration of identified solutions in different focused regions.

where the number of nondominated solutions in the preferred region is denoted as N. We also calculate the hypervolume metric, HV, as a performance measure [see 25, 26]. The nadir point is used as a reference point in the calculation of HV. We report these performance measures as well as the solution times for MMOKP and MOST problems in Tables 1 and 2, respectively. Considering each of the 150 problem instances of MMOKP summarized in Table 1, $AvgDist$ takes an overall average value of 0.00266 with a standard deviation of 0.00278. The average $AvgDist$ value and standard deviation are 0.00906 and 0.01138, respectively, for the 75 instances of the MOST problem summarized in Table 2. These results show that the solutions in the elite population

Table 1. MMOKP results obtained for different settings*.

No. of objectives	No. of items	Pr.	No. of nond. solns	$AvgDist$		$\frac{HV^N(ElitePop.)}{HV^N(Nond.Frt.)}$		Sol. time (s)	
				Avg.	Std. dev.	Avg.	Std. dev.	Avg.	Std. dev.
2	100	1	125	0.01009	0.00252	0.96485	0.01095	23.30	2.12
		2	169	0.00735	0.00606	0.97306	0.01610	25.91	5.88
		3	152	0.00242	0.00083	0.97853	0.00480	19.96	1.26
		4	154	0.00691	0.00321	0.96154	0.02613	25.16	11.54
		5	125	0.00789	0.00305	0.96695	0.01474	20.85	1.14
	200	1	448	0.00256	0.00074	0.98055	0.00354	189.90	41.41
		2	457	0.00256	0.00072	0.98524	0.00450	85.42	19.65
		3	414	0.00310	0.00145	0.98123	0.01160	134.93	37.58
		4	439	0.00216	0.00101	0.98174	0.00669	110.86	35.44
		5	407	0.00303	0.00161	0.98208	0.00828	131.09	78.01
3	25	1	182	0.00077	0.00065	0.99496	0.00830	9.62	2.15
		2	168	0.00140	0.00078	0.98145	0.02031	8.89	0.59
		3	76	0.00148	0.00179	0.99881	0.00184	8.57	0.54
		4	163	0.00033	0.00032	0.99969	0.00032	8.46	0.60
		5	470	0.00004	0.00006	0.99998	0.00005	8.64	0.95
	50	1	784	0.00227	0.00100	0.98881	0.01207	41.71	10.25
		2	912	0.00151	0.00045	0.99086	0.00472	42.38	2.89
		3	519	0.00190	0.00027	0.99158	0.00943	46.57	17.63
		4	280	0.00190	0.00128	0.98995	0.00680	29.00	2.68
		5	356	0.00207	0.00079	0.99130	0.00806	34.72	8.21
	100	1	2790	0.00148	0.00071	0.99427	0.00500	73.45	11.92
		2	8288	0.00331	0.00208	0.97380	0.01934	136.74	60.16
		3	10701	0.00246	0.00126	0.96701	0.02580	144.43	101.88
		4	5652	0.00242	0.00093	0.97694	0.02556	155.43	53.90
		5	6500	0.00232	0.00076	0.95975	0.02346	121.56	49.49
4	25	1	211	0.00094	0.00113	0.99800	0.00412	8.60	0.83
		2	401	0.00153	0.00115	0.97843	0.03171	7.47	0.44
		3	489	0.00139	0.00111	0.98906	0.01963	11.10	1.62
		4	396	0.00100	0.00094	0.99729	0.00454	12.19	2.26
		5	629	0.00125	0.00054	0.99683	0.00278	17.44	5.88

*Different lower and upper bounds are used for the same problem (different regions).

Table 2. MOST results obtained for different settings*.

No. of objectives	No. of nodes	Pr.	No. of nond. solns	*AvgDist*		$\frac{HV^N(ElitePop.)}{HV^N(Nond.Frt.)}$		Sol. time (s)	
				Avg.	Std. dev.	Avg.	Std. dev.	Avg.	Std. dev.
2	15	1	76	0.01488	0.00487	0.91349	0.02665	5.36	0.63
		2	68	0.01271	0.00991	0.94117	0.04143	5.01	0.19
		3	94	0.00650	0.00073	0.96227	0.00800	5.36	0.28
		4	86	0.00469	0.00086	0.96975	0.00753	5.54	0.62
		5	85	0.02385	0.01241	0.89244	0.04414	4.89	0.53
	20	1	143	0.00677	0.00596	0.94888	0.04576	13.27	1.02
		2	133	0.01043	0.00703	0.94415	0.03572	8.86	4.02
		3	245	0.01028	0.00521	0.92959	0.04245	10.96	4.61
		4	139	0.02435	0.03034	0.87059	0.14535	6.36	0.71
		5	162	0.00987	0.01071	0.94460	0.05296	7.44	0.72
3	10	1	655	0.00252	0.00165	0.97787	0.03128	13.10	1.23
		2	486	0.00121	0.00125	0.99013	0.01924	12.46	0.54
		3	704	0.00161	0.00091	0.98950	0.01295	13.13	0.58
		4	549	0.00248	0.00240	0.98182	0.02019	8.10	2.70
		5	733	0.00382	0.00235	0.94817	0.06171	6.89	0.46

*Different lower and upper bounds are used for the same problem (different regions).

represent the nondominated solutions in the preferred region well for both MMOKP and MOST problems. In addition, the hypervolume ratio, $\frac{HV^N(ElitePop.)}{HV^N(Nond.Frt.)}$, has an average value of 0.98382 and standard deviation of 0.01742 for the MMOKP. For the MOST problem, this ratio takes an average value of 0.94696 with standard deviation of 0.05715. Note that the best ratio that could be obtained would be 1.00000, which corresponds to identifying all true nondominated solutions in the region exactly. Based on these results, we can conclude that the hypervolume metric of the elite solutions is very close to the hypervolume metric for the nondominated solutions in the desired region. We observe that solution times and both performance measures are consistently small for all problems, indicating that the algorithm works well in convergence and diversity.

5. Conclusions

We developed an evolutionary algorithm to represent the nondominated solutions in a predetermined region. We test the performance of the algorithm on MMOKP and MOST problems, focusing on various parts of the solution space. The results show that the algorithm approximates the desired regions of the nondominated frontier well in these problems. Different than existing approaches, our algorithm concentrates only on the regions the DM is interested in and generates a set of points to represent the preferred nondominated points. In contrast to the existing literature, we also integrate an approximation procedure into the evolutionary algorithm and evaluate the fitness of each solution based on the approximation method. We evaluate the performance of our algorithm comparing our output with that of the exact algorithm in [24] that is designed to generate all nondominated points for any MOCO problem.

Modifying the algorithm and experimenting for other MOCO problems is a direction for future research. Using the main idea of focusing on desired regions, we can develop an interactive algorithm. The algorithm

would utilize the responses of the decision-maker to update the bounds throughout the algorithm and converge the preferred regions.

References

[1] Ehrgott M, Gandibleux X. Approximative solution methods for multi-objective combinatorial optimization. Top 2004; 12 (1): 1-63. doi: 10.1007/BF02578918

[2] Deb K. Multiobjective Optimization Using Evolutionary Algorithms. New York, NY, USA: Wiley, 2001.

[3] Hansen MP. Tabu search for multiobjective optimization: MOTS. In: 13th International Conference on Multiple Criteria Decision Making; Cape Town, South Africa; 1997. pp. 574-586.

[4] Ulungu EL, Teghem J, Fortemps PH, Tuyttens D. MOSA method: A tool for solving multiobjective combinatorial optimization problems. Journal of Multi-Criteria Decision Analysis 1999; 8: 221-236.

[5] Ehrgott M, Gandibleux X. Hybrid metaheuristics for multi-objective combinatorial optimization. In: Blum C, Aguilera MJB, Roli A, Sampels M (editors). Studies in Computational Intelligence, Vol. 114. Berlin, Germany: Springer, 2008, pp. 221-259.

[6] Deb K, Köksalan M. Guest editorial: Special issue on preference-based multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation 2010; 14 (5): 669-670. doi: 10.1109/TEVC.2010.2070371

[7] Rachmawati L, Srinivasan D. Preference incorporation in multi-objective evolutionary algorithms: a survey. In: IEEE 2006 Congress on Evolutionary Computation; Vancouver, BC, Canada; 2006. pp. 3385-3391.

[8] Phelps S, Köksalan M. An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. Management Science 2003; 49 (12): 1726-1738. doi: 10.1287/mnsc.49.12.1726.25117

[9] Köksalan M, Karahan İ. An interactive territory defining evolutionary algorithm: iTDEA. IEEE Transactions on Evolutionary Computation 2010; 14 (5): 702-722. doi: 10.1109/TEVC.2010.2070070

[10] Köksalan M, Phelps SP. An evolutionary metaheuristic for approximating preference-nondominated solutions. INFORMS Journal on Computing 2007; 19 (2): 291-301. doi: 10.1287/ijoc.1050.0170

[11] Chen J, Li J, Xin B. DMOEA-εC: decomposition-based multiobjective evolutionary algorithm with the ε-constraint framework. IEEE Transactions on Evolutionary Computation 2017; 21 (5): 714-730. doi: 10.1109/TEVC.2017.2671462

[12] Zhang H, Zhou A, Song S, Zhang Q, Gao XZ et al. A self-organizing multiobjective evolutionary algorithm. IEEE Transactions on Evolutionary Computation 2016; 20 (5): 792-806. doi: 10.1109/TEVC.2016.2521868

[13] Jaszkiewicz A. Genetic local search for multi-objective combinatorial optimization. European Journal of Operational Research 2002; 137: 50-71. doi: 10.1016/S0377-2217(01)00104-7

[14] Soylu B, Köksalan M. A favorable weight based evolutionary algorithm for multiple criteria problems. IEEE Transactions on Evolutionary Computation 2010; 14 (2): 191-205. doi: 10.1109/TEVC.2009.2027357

[15] Soylu B, Köksalan M. An evolutionary algorithm for the multi-objective multiple knapsack problem. In: Shi Y, Wang S, Peng Y, Li J, Zeng Y (editors). MCDM 2009: Cutting-Edge Research Topics on Multiple Criteria Decision Making. Communications in Computer and Information Science, Vol. 35. Berlin, Germany: Springer, 2009, pp. 1-8.

[16] Alves MJ, Almeida M. MOTGA: A multiobjective Tchebycheff based genetic algorithm for the multidimensional knapsack problem. Computers & Operations Research 2007; 34: 3458-3470. doi: 10.1016/j.cor.2006.02.008

[17] Arroyo JEC, Vieira PS, Vianna DS. A GRASP algorithm for the multi-criteria minimum spanning tree problem. Annals of Operations Research 2008; 159 (1): 125-133. doi: 10.1007/s10479-007-0263-4

[18] Chen G, Chen S, Guo W, Chen W. The multicriteria minimum spanning tree problem based genetic algorithm. Information Sciences 2007; 177 (22): 5050-5063. doi: 10.1016/j.ins.2007.06.005

[19] Zhou G, Gen M. Genetic algorithm approach on multi-criteria minimum spanning tree problem. European Journal of Operational Research 1999; 114: 141-152. doi: 10.1016/S0377-2217(98)00016-2

[20] Köksalan M. Multiobjective combinatorial optimization: some approaches. Journal of Multi-Criteria Decision Analysis 2009; 15: 69-78. doi: 10.1002/mcda.425

[21] Köksalan M, Lokman B. Approximating the nondominated frontiers of multi-objective combinatorial optimization problems. Naval Research Logistics 2009; 56: 191–198. doi: 10.1002/nav.20336

[22] Reeves CR. Genetic algorithms. In: Reeves CR (editor). Modern Heuristic Techniques for Combinatorial Problems. New York, NY, USA: Halsted Press, 1993. pp. 151-188.

[23] Steuer RE. Multiple Criteria Optimization: Theory, Computation, and Application. New York, NY, USA: Wiley, 1986.

[24] Lokman B, Köksalan M. Finding all nondominated points of multi-objective integer programs. Journal of Global Optimization 2013; 57 (2): 347-365. doi: 10.1007/s10898-012-9955-7

[25] Zitzler E, Brockhoff D, Thiele L. The hypervolume indicator revisited: on the design of Pareto-compliant indicators via weighted integration. In: Obayashi S, Deb K, Poloni C, Hiroyasu T, Murata T (editors). Conference on Evolutionary Multi-Criterion Optimization; Volume 4403 of LNCS. Berlin, Germany: Springer, 2007. pp. 862-876.

[26] Zitzler E, Thiele L, Laumanns M, Fonseca CM, Grunert da Fonseca V. Performance assessment of multiobjective optimizers: an analysis and review. IEEE Transactions on Evolutionary Computation 2003; 7 (2): 117–132. doi: 10.1109/TEVC.2003.810758