A multi-objective open set orienteering problem

Article *in* Neural Computing and Applications · September 2020

DOI: 10.1007/s00521-020-04798-7

5 authors, including:

Joydeep Dutta

7 PUBLICATIONS **11** CITATIONS

SEE PROFILE

Partha sarathi Barma

NSHM Knowledge Campus

6 PUBLICATIONS **9** CITATIONS

SEE PROFILE

Anupam Mukherjee

National Institute of Technology, Durgapur

5 PUBLICATIONS **19** CITATIONS

SEE PROFILE

Samarjit Kar

National Institute of Technology, Durgapur

248 PUBLICATIONS **3,152** CITATIONS

SEE PROFILE

Neural Computing and Applications
https://doi.org/10.1007/s00521-020-04798-7

ORIGINAL ARTICLE

A multi-objective open set orienteering problem

Joydeep Dutta[1] · Partha Sarathi Barma[1] · Anupam Mukherjee[2] · Samarjit Kar[2] · Tanmay De[3]

Received: 27 August 2019 / Accepted: 17 February 2020

Abstract
We have constructed a multi-objective set orienteering problem to model real-world problems more fittingly than the existing models. It attaches (i) a predefined profit associated with each cluster of customers, and (ii) a preset maximum service time associated with each customer of all the clusters. When a customer from cluster visits, it allows the earning of a profit score. Our purpose is primarily to search for a route that (i) on the one hand allows us to service each cluster for maximizing customer satisfaction and (ii) on the other hand allows us to maximize our profits, too. The model assumes that the more time we spend on servicing, the more customer satisfaction it yields. It tries to cover as many clusters as possible within a specified time budget. In this paper, we also consider third-party logistics to allow the flexibility of ending our journey at any cluster of our choice. The proposed model is solved using the nondominated sorting genetic algorithm and the strength Pareto evolutionary algorithm. Here, we also generate the dataset to test the proposed model by using instances from the literature of the generalized traveling salesman problem. Finally, we present a comparative result analysis with the help of some statistical tools and discuss the results.

Keywords Set orienteering problem · Multi-objective optimization · NSGAII · SPEA2

1 Introduction

In today's world, successful organizations keep a constant focus on the enhancement of products and plan continuously to improve customer through suitable service. Transportation has a vital role in any industry because a proper routing plan can serve the customers more efficiently, and hence, it increases customer satisfaction, thereby increasing the goodwill of the organization. Efficient routing can reduce a lot in the cost of transportation that directly decreases product price and increases product acceptability in the market leading to an increase in consumption of the products in question. That is why researchers have given massive attention to the routing of products or services for the last few decades. As a result, several new concepts and models of routing plans have been invented. These models directly or indirectly map real-world problems. The traveling salesman problem (TSP) published in the 1800s is the oldest and the essential concept. Here, the challenge is to find a Hamiltonian cycle that starts from and ends in the same city and visits all the cities exactly once. This problem is described in detail in the book of Lawer et al. [1]. Soon after, a version of TSP, called prize-collecting TSP, was presented by Balas [2]. Here, a salesman gets prizes for the cities he visits and conversely pays penalties for the cities that he has omitted. Prize-collecting TSP is thus a maximization problem.

Fischetti et al. [3] published a paper on generalized TSP (GTSP) that partitioned all the cities into some clusters and tried to search for a minimum cost path that covers all the clusters by visiting one city from each cluster. Feillet et al. [4] presented a survey paper on TSP with profit. It considered only a single vehicle to earn profits. Dantzig and Ramser [5] introduced the concept of vehicle routing problem (VRP) on their paper on truck dispatching. VRP considers more than one vehicle to provide the services to a set of customers positioned in different locations. But the

✉ Samarjit Kar
samarjit.kar@maths.nitdgp.ac.in

[1] Department of Computer Science and Engineering, NSHM Knowledge Campus, Durgapur, West Bengal 713212, India

[2] Department of Mathematics, NIT Durgapur, Durgapur, West Bengal 713209, India

[3] Department of Computer Science and Engineering, NIT Durgapur, Durgapur, West Bengal 713212, India

Published online: 03 March 2020

concept of profit is not there in VRP. Generally, VRP is a minimization problem on the traveling distance or cost.

A well-known routing problem with profit is orienteering problem (OP). Tsiligirides [6] first introduced it. Here, each city is associated with a profit. The challenge in OP is to maximize the total profit by visiting the cities within a predefined time limit. OP involves the concept of both the knapsack problem and TSP. Since then, researchers have developed several models of OP like the team orienteering problem, the team orienteering problem with time windows, the time-dependent team orienteering problem, etc. Angelelli et al. [7] introduced the clustered orienteering problem (COP), which is the combination of OP and GTSP. Here, all the customers are grouped into some clusters, and each cluster is associated with a profit. The profit score of a cluster is collected only after visiting all the customers belonging to that cluster. The target of this COP is to find a cycle that maximizes the collected profit within a specified time limit.

Recently, Archetti et al. [8] introduced a variant of the OP in analogy with COP of Angelelli et al. [7]. They coined the term set orienteering problem (SOP). Like COP, SOP uses a set of clusters with profit, and each cluster has some customers. Unlike COP, in SOP, the profit of a cluster is collected after visiting at least one of the customers belonging to that cluster. SOP also tries to find the cycle that maximizes the collected profit within a specified time limit. Both the COP and SOP have a considerable number of applications, especially in mass distribution products. In the case of COP, the carrier will serve all the customers who belong to the same cluster, whereas in the case of SOP, it uses two levels of carriers. The first level of transport will serve only one customer of a cluster. Mostly based on the distance, the system chooses the customer on a cluster. Some other internal carriers serve the other customers of the cluster. We can apply SOP in all the applications of GTSP with a limited budget. Very recently, Pěnička et al. [9–11] presented that many OP variants like OP with Neighborhoods (OPN) (2016), the Dubins OP (DOP) (2017), etc., can also be mapped as SOP.

Till now, all the papers published on SOP considered only one objective, i.e., the maximization of total profit. If we look into the real-life applications of SOP, the choice of a particular customer from a set of customers of a cluster to supply the total orders of all the customers of the same cluster is not only based on only one objective; instead, we used to consider so many parameters. These are like distance, customer's goodwill, customer satisfaction, the capacity of the customer, customer's payment status, customer's social status, urgency, order amount, etc. So to fill this gap, we have introduced first the multi-objective SOP. In this paper, we consider two objectives to solve the SOP.

In addition to that, we use third-party logistic; that is, the vehicle will start from a fixed depot, but it will not return to the depot, and there is no fixed endpoint. So we have used the term open set orienteering problem (OSOP). Till now, very few papers have been published on the SOP. Archetti et al. [8] have first introduced the concept of SOP, and they considered the same start and end depot belonging to a cluster having only one point, i.e., the depot. Next, Pěnička et al. [9] have published the paper on SOP by considering a fixed starting cluster and a fixed end cluster, both of which may have multiple nodes. In this work, there will be a fixed starting cluster having only one node, i.e., the depot, but there is no such fixed end cluster. To make this model more realistic, we have given this flexibility to stop the tour at any cluster except the depot as we are considering third-party logistic. Therefore, we implement a multi-objective open set orienteering problem (MOOSOP).

To solve the proposed multi-objective model, we use the nondominated sorting genetic algorithm (NSGAII) and the strength Pareto evolutionary algorithm (SPEA2). The solution of the model will give a route to serve a set of customers belonging to different clusters with the consideration of the trade-off between the maximization of the total profit and the maximization of the average customer satisfaction score.

The main contributions of this article are summarized as follows.

a. The multi-objective version of the set orienteering problem has been developed. We use third-party logistics to make it open, as the proposed model has no restriction on the endpoint of the route. The definition of the SOP by Archetti et al. [8] is a single-objective model that considered a fixed same point for start and end. But as per the proposed model, both the beginning and endpoints are different. As compared to Pěnička et al. [9], the proposed model has given the flexibility to choose any cluster as the end cluster except the staring one.
b. NSGAII and SPEA2 are applied to solve the proposed MOOSOP model using the instances of the GTSP.
c. Statistical analysis is also performed to measure the performance of both the algorithms to solve the proposed model.

The paper is organized as follows. In the beginning, Sect. 1 introduces the work. The literature survey of Sect. 2 describes the previous studies on different types of orienteering problems and specially set orienteering problems. Section 3 highlights the reason to start this work. Section 4 presents the mathematical modeling of the proposed work. Section 5 describes the methodology used to solve the proposed model. Section 6 describes the numerical dataset used to assess the proposed study. The

computational results are presented in Sect. 7 that show the applicability of the used algorithms in the proposed model. Finally, Sect. 8 describes the main findings and the conclusion of the work.

2 Literature review

The root of all the routing plans starts from the TSP problem. But TSP does not consider any profit score associated with the cities. The routing problems that involve profits are more appropriate to fit the real-world issues. That is why the researchers are paying massive attention in this section of the routing problems. In this context, the concept of orienteering problem (OP), as described by Chao et al. [12], is the perfect one. OP helps to model several issues of the real-world scenario. They are like "selective traveling salesman problem" [13], "maximum collection problem" [14], "bank robber problem" [15], etc. Vansteenwegen et al. [16] published a survey paper on the orienteering problem. Gunawan et al. [17] published a survey on the recent variants of OP. Mukhina et al. [18] proposed a new version of OP with functional profits. Here, the profit of any node depends on several factors like geographical position and the other attributes of the node. They used open-source framework to solve it and use the ant colony optimization method and a greedy method. Hanafi et al. [19] published a paper on OP by introducing an extension of team orienteering problem. They incorporated the concept of priority on the tasks of the customers. Here, the vehicles used are heterogeneous in nature. They enhanced the kernel search framework and solve the model using the branch-and-cut method. Yu et al. [20] introduced a selective discrete particle swarm optimization method to solve the model of team orienteering problem with time window. They used a partial score associated with each city based on a set of attributes. Authors used a discrete version of PSO to solve the model.

Schilde et al. [21] worked to develop a heuristic method to solve the multi-objective OP. They applied the model in the domain of tourism and focused on the different types of interests of different kinds of tourists using the many profits associated with each point of interest. Using Pareto ant colony optimization algorithm and variable neighborhood search, they have solved the model and tested on several benchmark instances as well as on some practical examples from Austrian regions and the cities of Vienna. Chen et al. [22] extended the orienteering problem with time windows by associating more than one profit values with each point or customers. They used ant colony optimization to solve the model based on 76 benchmark instances. Mei et al. [23] introduced a multi-objective time-dependent orienteering problem in 2016. They considered two critical factors, namely time-dependent travel time and multiple preferences together. They proposed two multi-objective methods, namely multi-objective memetic algorithm and a multi-objective ant colony system, to solve the above model. Yu et al. [24] studied a multi-objective OP. They considered separate benefits for various categories for each point of interest. Thus, multi-objective decision making arises. It was solved using simulated annealing. Wang et al. [25] considered both the uncertainty and the multi-objective in the OP. They used the uncertainty theory. It was solved using a discrete multi-objective bat algorithm and multi-objective local search. They applied the model in reconnaissance mission planning. In the recent years, Hu et al. [26] published a paper on the multi-objective team orienteering problem with time windows. They included multiple profits per city. They solved the model using a multi-objective method based on decomposition and constraint programming. Some well-known benchmark instances used this model. Hapsari et al. [27] published a paper, especially for the tourism industry. They developed a model on multi-objective orienteering problem that focus on minimizing the total travelled time and maximizing the total score. To solve the model, they used some real-world data of east Java and some data from the literature. They applied an adjustment iterated local search method to solve it and compare with other state-of-the-art algorithms.

Angelelli et al. [7] first coined the term clustered orienteering problem (COD), which is a generalization of OP. Like GTSP, COD also has a different set of clusters that are a group of customers. Each such cluster is associated with a profit, and a tour plan can earn this profit if it covers all the cities of a cluster. They solved the model using both the exact and heuristic approaches. After that, Yahiaoui et al. [28] published a paper on COP based on the order first cluster second approach. In the same year, Alvarez-Miranda et al. [29] gave a new concept of generalized clustered orienteering problem. The popular game Pokemon GO motivates them to do this work. This work includes the ideas of several routing problems like TSP, GTSP, OP, etc. They have designed branch-and-bound (B&B) and branch-and-cut (B&C) algorithms to solve it.

As already, we have discussed in Introduction section that very recently, in the last half of 2017, a new variant of OP has come into the literature called set orienteering problem (SOP). Archetti et al. [8] published the paper. Here, the customers are associated with a cluster, and each cluster is also associated with a profit. A tour plan can gain the profit of a cluster if it covers at last one customer of that cluster. Later, Pěnička et al. [9] presented a paper on the SOP and showed that many variants of OP could be modeled as SOP. They also solved the model using a variable neighborhood search. Till now, only these two

papers have been published, and many researchers are working on it to best utilize the concept of this model in the real-life scenario.

3 The motivation of the work

The real-life supply chain system of several products initiates the main interest in doing this work. We observed that several well-known companies select a particular customer among a set of customers from a local area as their authorized dealer. The companies used to supply the bulk orders of all the customers from a local area to the selected dealers. In the next step, the internal distribution of the city is completed either by the dealer or by the other customers of that area. There are several routing models in the literature. The model of the set orienteering problem (SOP), as described by Archetti et al. [8], exactly matches the requirements of this case mentioned above. That is why we have selected SOP as the base model for our work. As a recently developed model, the literature on SOP has very few papers. All of these papers have implemented their model as a single-objective case. But in reality, it is different, as choosing a customer as an authorized dealer and making a tour plan always involve several issues. So we have designed our work as a multi-objective SOP. At the same time, none of the published papers considered the third-party logistics that are mostly in use nowadays. Therefore, we implement the multi-objective open SOP where the vehicles used for supply are not bound to return to the depot.

4 Problem definition and mathematical model of MOOSOP

In this paper, the aim is to solve the multi-objective open set orienteering problem (MOOSOP), which extends the basic model of SOP by considering two objectives and the flexibility on ending cluster being anyone except the starting one. Like SOP, the MOOSOP can also be defined with the help of a directed graph $DG = (V, A)$, where $V = \{v_0, v_1, \ldots, v_m\}$ is the set of vertices and $A = \{a_{ij}\}$ is the set of arcs such that between every pair of vertices v_i and v_j, there will be an arc a_{ij} associated with a cost of c_{ij}. All the vertices are grouped into a set of clusters $S = \{s_0, s_1 \ldots, s_n\}$ such that $s_i \cap s_j = \emptyset$ for $i \neq j, 0 \leq i, j \geq n$. Every vertex $v_{i=0,1,\ldots,m}$ must belong to only one cluster in S. Each cluster $s_{i=0,1,\ldots,n}$ has a profit $p_{i=0,1,\ldots,n}$ that can be earned by visiting at least one of the vertices from that cluster. In this model, we consider a fixed cluster s_0 as a starting cluster that holds only one vertex, i.e., the depot from where the tour begins. Therefore, $|s_0| = 1, |s_{i=1,2..,n}| \geq 1$. It is also assumed that the ending cluster s_f can be any vertex except the starting cluster. Therefore, $f = 1, 2, \ldots, n$ as $s_f \neq s_0$.

But in comparison with the previous works, there is no such fixed ending cluster. As the tour may end at any cluster except the starting one, we assume that the profit corresponding to the starting cluster is zero, i.e., $p_0 = 0$. Here, we have considered two objectives. One of the goals is to search a route that maximizes the collected profit by visiting the clusters as maximum as possible within the predefined budget $T_{\max}$. Based on the triangular inequality, we can easily say that an optimal tour should not include more than one vertex for each visited cluster. The other objective used in this model is to maximize customer satisfaction. Here, we assume a maximum service time t_i associated with each cluster s_i. The customers of a cluster are expecting that the service provider will stay this much of time to provide the service to get the full customer satisfaction. Let t_i' be the actual time the service provider spent on the cluster s_i.

The proposed model may be formed as an integer linear programming problem. Here, we have assumed some variables given below.

y_i is a binary variable which is equal to 1 if at least one vertex is visited from the cluster s_i, else it is zero. x_{ij} is a binary variable which is equal to 1 if the tour plan includes a_{ij}, else it is zero.

Now the proposed formulation of the MOOSOP is given below:

$$\text{Maximize } Z_1 = \sum_{s_i \in S} p_i y_i \tag{1}$$

$$\text{Maximize } Z_2 = \left(\sum_{i=1}^{n} \left\{ 1 - \left(\frac{t_i - t_i'}{t_i} \right) \right\} y_i \right) / \sum_{i=1}^{n} y_i \tag{2}$$

subject to

$$\sum_{a_{ij} \in A} c_{ij} x_{ij} \leq T_{\max} \tag{3}$$

$$\sum_{v_i \in V \setminus \{s_q\}} x_{ij} = \sum_{v_i \in V \setminus \{s_q\}} x_{ji} \quad \forall s_q \in S \setminus \{s_0, s_f\},\ f \neq 0,\ \forall v_j \in s_q \tag{4}$$

$$\sum_{v_i \in V \setminus \{s_q\}} \sum_{v_j \in s_q} x_{ij} = y_q \ \forall s_q \in S \setminus \{s_0\} \tag{5}$$

$$\sum_{v_i \in V \setminus \{s_q\}} \sum_{v_j \in s_q} x_{ji} = y_q \quad \forall s_q \in S \setminus \{s_f\},\ f \neq 0 \tag{6}$$

$$\sum_{v_i \in U} \sum_{v_j \in U} x_{ji} \leq \sum_{s_q \in U \setminus \{s_t\}} y_q \quad \forall U \in S \setminus \{s_0, s_f\},\ f \neq 0,\ s_t \in U \tag{7}$$

$$y_0 = 1,\ y_f = 1 \tag{8}$$

$$y_q \in \{0,1\},\ s_q \in S,\ x_{ij} \in \{0,1\},\ a_{ij} \in A \tag{9}$$

The objective function (1) represents the maximization of collected profit. The objective function (2) represents the maximization of customer satisfaction. Constraint (3) denotes the budget constraint. The clusters except the starting and the ending cluster must have an equal number of arcs entering and exiting. Equation (4) implements this fact. Constraint (5) ensures that at least one arc must be entered in every visited cluster except the starting cluster. Similarly, constraint (6) provides that at least one arc must be leaving from every visited cluster except the ending cluster. Constraint (7) represents subtour elimination. Constraint (8) ensures that the starting and ending clusters must be visited. The domains of some variables are implemented using constraint (9).

5 Methodology

Most of the real-world problems are multi-objective in nature. The issue of delivery of products is a widespread scenario, and it is also a multi-objective problem. The MOOSOP model is very much useful to map the transportation of different products. So the model presented in this study is a multi-objective SOP, and to solve it, the multi-objective evolutionary algorithms are very much suitable. Nowadays, so many multi-objective algorithms already exist. Some state-of-the-art multi-objective algorithms are trendy such as NSGAII, SPEA2, MOALO, MOPSO, AbYSS, etc. Here, we have used NSGAII and SPEA2 to solve the proposed model.

5.1 NSGAII

NSGAII is a nondominated sorting-based multi-objective evolutionary algorithm (MOEA), which was first proposed by Deb et al. [30]. Unlike the other MOEAs consisting of external archives, the NSGAII deals with a population updating process from the union of populations before and after exploration–exploitation. Compared to earlier MOEAs, the NSGAII executes a nondominated solution set much faster, and the diversity preservation is also maintained more easily. Many researchers employed the NSGAII in their problem-specific way to evaluate discrete as well as continuous multi-objective problems so far, and this algorithm seems to produce effective results in those cases. Hence, we select the NSGAII as a popular state-of-the-art MOEA to solve the proposed model. Initially, it creates a random parent population, and then the selection, crossover and mutation are performed similarly as in GA. In the next step, both the parent and child (after crossover and mutation) populations are combined to get a new population with double size (2 m) of the population (size m). Then, the new double-size population (size 2 m) is sorted using the crowding distance calculation. The next step incorporates the frontification process of the solutions based on nondominance checking. The best m solutions are forwarded to the next generation, while the remaining m individuals are discarded. As the termination condition is reached, the first front is obtained as a near-optimal solution. The necessary steps of the algorithm are given below (Table 1).

Table 1 NSGAII procedure

NSGAII pseudo code
Procedure of NSGAII
Initialize: Randomly generate an initial population of the possible solution, P_t of size m; Set crossover and mutation probability
Evaluate the fitness of the population
Make a non-dominated sorting of the population
Apply selection operation and store the solution in Selected
Apply crossover and mutation and store the solution in Children
While(stopping criteria not satisfied)
Evaluate the fitness of the Children ;
Merge population and children;
Make a non-dominated sorting and find the fronts;
Evaluate crowding distance. Based on the crowding distance update population.
Perform selection and store in Selected,
Apply crossover and mutation on selected and generate Children.
end while
end NSGAII

Table 2 SPEA2 procedure

SPEA2 pseudo code
Procedure of SPEA2 *Initialize : Generate population of possible solution, P_t ;* *empty external set (archive), Z_t with size N;* *$Z_t = 0$;* *generation number, $t = 0$;* *for $t = 1$ to g do* *Fitness assignment on each individual in P_t and Z_t ;* *Copy all non-dominated individual to Z_{t+1} ;* *If the capacity of Z_{t+1} exceeded N then use the truncation operator to remove elements from Z_{t+1}.* *If Z_{t+1} does not exceed the capacity, then the dominated individual will be sorted and filled to Z_{t+1}.* *Perform binary tournament selection to fill the mating pool,* *Apply crossover and mutation to the mating pool.* *end for* *end SPEA2*

5.1.1 Complexity analysis of one iteration of NSGAII

Let us assume that the M objectives are used and the number of chromosomes taken in the population is N. The worst-case complexity [30] of the important operations is given below.

1. Fast nondominated sorting: $O(MN^2)$
2. Crowding distance assignment: $O(M(2N)\log(2N))$
3. Sorting based on crowded-comparison operator:$O(2N\log(2N))$

So the complexity of NSGAII algorithm is

$$O(MN^2) + O(M(2N)\log(2N)) + O(2N\log(2N)) \approx O(MN^2).$$

5.2 SPEA2

Several Pareto-based multi-objective evolutionary algorithms are found in the literature. Each of the algorithms has some advantages and disadvantages when solving real-life problems depending on the nature of the problems. Here, we use SPEA2 to get a set of alternative solutions for the proposed multi-objective SOP. Zitzler et al. [31] developed SPEA2, which performs better than several other algorithms of its peer in many problems. The necessary steps of the algorithm are given in Table 2.

5.2.1 Time complexity of SPEA2

Let N and N' be the population size and archive size, respectively, and $K = N + N'$. The computational complexity [31] of each of the operators is as follows:

Construction of distance list for each individual: $O(K^2)$
Fitness assignment procedure: $O(K^2 \log K)$
Removal of an individual: $O(K^2)$

Thus, the overall complexity SPEA2 for an iteration is:

$$O(K^2) + O(K^2 \log K) + O(K^2) \approx O(K^2 \log K)$$

5.3 Encoding

This paper uses two well-known multi-objective evolutionary algorithms, viz. NSGAII and SPEA2, to solve the proposed model. Any solution to this model will contain the cluster sequence, the time to service the clusters and the cities within each cluster to visit within the predefined time limit to maximize the score and the customer satisfaction. Therefore, to solve the proposed model using NSGAII and SPEA2, we encode one single solution with the help of three strings of length N given in Fig. 1. N is the number of clusters in the problem. Here, the first string or the first row contains the cluster sequence that is the order of clusters to visit. The second string contains the time to spend in each cluster, and the third string includes the cities to be visited corresponding to each cluster. For example, the route first covers cluster 9, and the time to be spent on cluster 9 is 10. Among all the cities of cluster 9, the selected city to visit is city 52.

5.4 Initialization

For both the algorithms, we assume m as the population size and N as the number of clusters. We take m number of chromosomes, and each chromosome is of size $3 \times N$. The first row of each chromosome is a random permutation of

9	2	6	5	4	3	7	10	8	1
10	16	6	32	56	60	24	16	12	8
52	13	37	17	29	46	22	25	16	6

Fig. 1 An example of a solution

clusters between 1 to N. The second row is filled up by a random number between 1 and the maximum service time of the corresponding cluster. The third row is filled up by a randomly selected city from the set of cities under the same cluster.

5.5 Fitness evaluation

As the proposed model has two objectives, viz. maximization of profit score and maximization of customer satisfaction, we use two fitness functions directly for the two objective functions as presented by Eqs. 1 and 2. In both the methods SPEA2 and NSGAII, we use the same fitness functions. One of the fitness functions calculates the total profit scores by adding the profits of the clusters as per the sequence of the clusters of the chromosomes until it crosses the budget constraint. Another fitness function calculates the average customer satisfaction by dividing the total customer satisfaction of the visited clusters by the total number of clusters visited.

5.6 Selection

In SPEA2, we use the well-known tournament selection with a suitable selection pressure, which represents the number of solutions participating in a tournament. In this method, we randomly choose some solutions from the population and sort them based on the crowding distance metric. The solution with the rank of one is sent to the mating pool. In our implementation of SPEA2, we consider selection pressure as four. Choosing a selection pressure is a matter to perform several pilot tests as a low selection pressure leads to a slow convergence rate, whereas a high selection pressure leads to premature convergence. In NSGAII, we use conventional Roulette wheel selection. Here, the probability of selection of a solution is also calculated using the crowding distance.

5.7 Crossover

Crossover operator is responsible for exploring the search space. Both the NSGAII and SPEA2 use a well-known partially mapped crossover (PMX) for the cluster sequence, i.e., the first string of chromosomes. The third row of chromosome contains a list of cities, each of which belongs to different clusters. During the crossover, the cities of the third row of chromosome change simultaneously according to the changes in the cluster sequence. The second row of the chromosome is a list of service times spent on associated clusters. We use the following crossover technique to find the service time of clusters in the child:

$$\begin{aligned}&\text{Servic\_Time}(\text{Child } 1, i)\\&\quad = 1 + (\text{Servic\_Time}(\text{Parent } 1, i)\\&\qquad + \text{Servic\_Time}(\text{Parent } 2, i))\\&\qquad \bmod(\text{Max\_Servic\_Time}(\text{Child } 1, i))\\&\qquad \text{for } i = 1, 2, \ldots N\end{aligned}$$

Goldberg and Lingle [32] proposed the PMX crossover. It first selects two random crossover points. Then, the substrings between these two crossover points of the parents are swapped. Then, the repeating elements within one permutation are exchanged with the allele corresponding to those mapped by the other parent. This operation is explained in detail using Fig. 2 assuming the maximum service time of the clusters {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} as {10, 24, 70, 65, 40, 30, 12, 25, 30, 30}, respectively. In the first step, two random crossover points are selected and are marked using blue arrows over the parent. Then, the substring (5, 4, 3) of parent 1 is swapped with the substring (6, 9, 5) of parent 2. Then, the new permutations generated for offspring 1 and offspring 2 are (9, 2, 7, 6, 9, 5, 6, 10, 8, 1) and (2, 8, 3, 5, 4, 3, 10, 7, 4, 1), respectively. Now to remove the duplicate values from each permutation, we need to consider the mapping between the substrings (6, 9, 5) and (5, 4, 3). As per the below figure, it is clear that 6 is mapped with 5, 5 is mapped with 3 and 9 is mapped with 4. Using this mapping scheme, the duplicate values 6 and 9 of the first permutation are replaced with 3 and 4, respectively. Similarly, the duplicate values 4 and 3 of the second permutation are replaced with 9 and 6, respectively. Finally, the crossover for the first row becomes complete. The second row is generated using the previously mentioned formula of service time. For example, the service time for cluster 9 of offspring 1 is 23. It is derived as per the below calculation:

$$\begin{aligned}&1 + (\text{Servic\_Time}(\text{Parent } 1, 0) + \text{Servic\_Time}(\text{Parent } 2, 0))\\&\quad \bmod (\text{Max\_Servic\_Time}(\text{Child } 1, 0))\\&= 1 + (\text{Servic\_Time}(9) + \text{Servic\_Time}(2))\\&\quad \bmod (\text{Max\_Servic\_Time}(9))\\&= 1 + (10 + 12) \bmod (30)\\&= 23\end{aligned}$$

The third row is generated from the parent according to the cluster–city relationship. The first value of the third row in offspring 1 is 52 as city corresponding to cluster 9 is 52 in the parent 1.

5.8 Mutation

The mutation is an evolutionary operator that exploits the search space. Applying mutation with appropriate mutation probability leads to faster convergence.

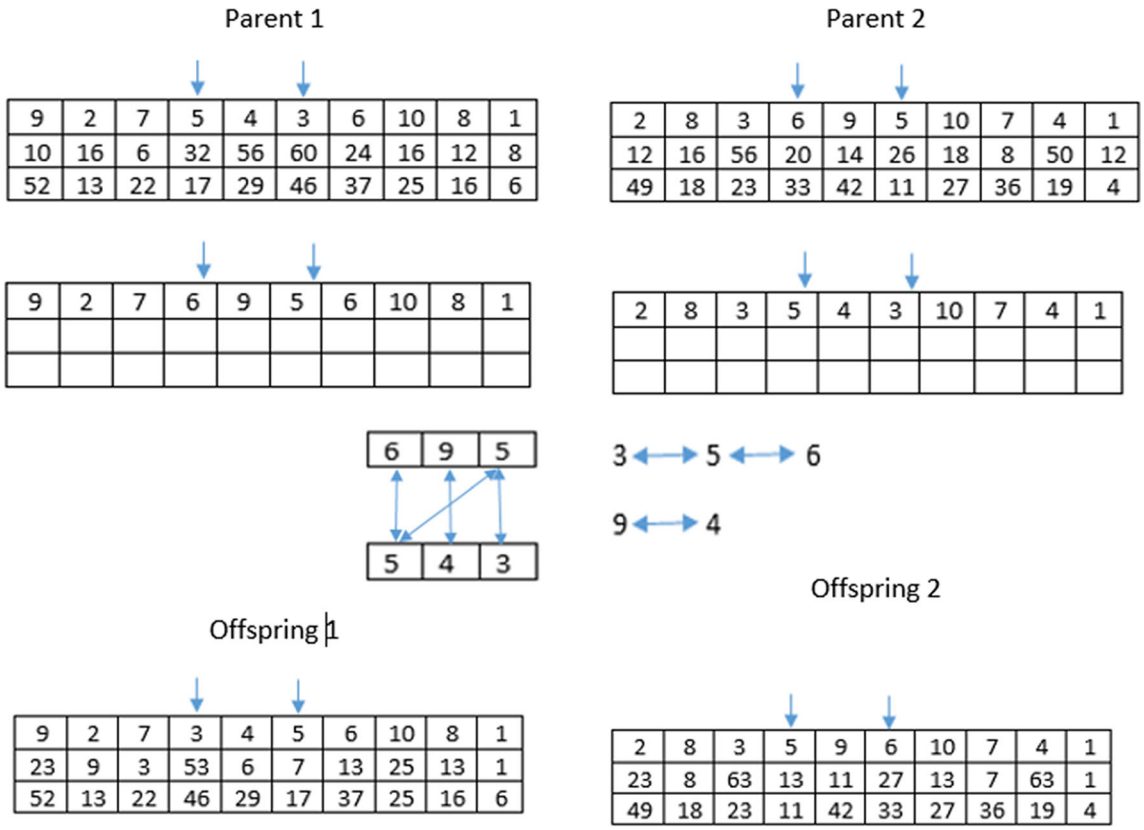


Fig. 2 PMX crossover

In this paper, we use an inversion mutation on the chromosomes in SPEA2. It selects two random mutation points. Then, the substrings between these two selected points are just inverted for all the rows of the parent chromosome. Figure 3 shows the inversion mutation. Here, two random mutation points are marked using blue arrows, which are pointing to 7 and 3. The substrings (7, 5, 4, 3), (6, 32, 56, 60) and (22, 17, 29, 46) are inverted in the newly generated mutated chromosomes.

In NSGAII, we use swap mutation. It selects two random mutation points. Then, the cluster numbers of the first row and the service times of the second row of the corresponding selected points are just swapped. In the case of the city sequence, we choose a different city from the same cluster corresponding to those selected points. Figure 4 explains the above method with a problem of size 10. Here, two random mutation points are marked using blue arrows, which are pointing to 7 and 6. Numbers 7 and 6 of the first row and numbers 6 and 24 of the second row are swapped. However, in the newly mutated chromosome, the cities corresponding to clusters 6 and 7 are randomly selected

Fig. 3 Inversion mutation (used in SPEA2)

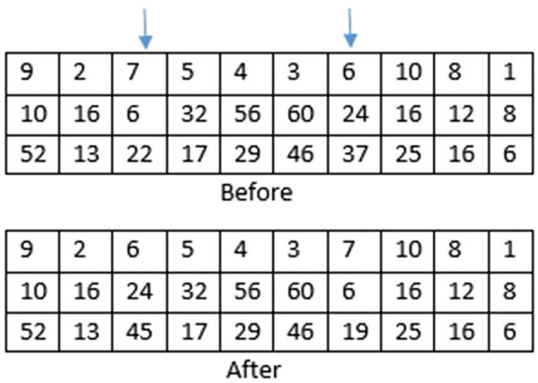


Fig. 4 Swap mutation (used in NSGAII)

except 37 and 22, respectively. Here, the selected cities are 45 of cluster 6 and 19 of cluster 7.

6 Performance measures

The performance of a multi-objective evolutionary algorithm can be measured using several metrics. Some of the most used performance metrics are hypervolume (HV), spread, generalized distance (GD) and inverted generalized distance (IGD). In this paper, we use GD, IGD and spread to compare the algorithms.

6.1 GD

Veldhuizen and Lamont [33] proposed this metric. The objective of the metric is to determine the distance between the nondominated solution and the nearest element from the Pareto front. The formula for GD is given below:

$$\mathrm{GD} = \sqrt{\frac{\sqrt{\sum_{i=1}^{K} d_i^2}}{K}}$$

where K is the number of solutions in the approximate front and d_i represents the distance between the solution and the nearest number in Pareto front. GD metric measures the convergence of a multi-objective evolutionary algorithm by measuring the gap between the approximate front and Pareto front (PF).

6.2 IGD

Veldhuizen and Lamont [33] also proposed this metric. This metric measures the diversity as well as the convergence of a multi-objective evolutionary algorithm. The formula of IGD is presented as follows.

$$IGD = \frac{\sum_{i=1}^{|P|} d(\varrho_i, N_D)}{|P|} \forall\, \varrho_i \in PF$$

where $d(\varrho_i, N_D)$ represents the distance between the elements of the Pareto front from the nearest neighborhood element from the approximate front. There, the smaller the IGD, the better the approximate solution.

6.3 Spread (Δ)

Spread measures the diversity of a nondominated solution set. This measure is proposed by Deb et al. [30] for the biobjective problem. It finds the distance between the nearest solutions present in an approximate front. Zhou et al. [34] extended this measure for multi-objective optimization problems where it finds the distance between a nondominated solution and its nearest neighbor. In best case, value of spread vanishes. The formula for the spread is given below:

$$\Delta = \frac{\sum_{i=1}^{m} d(e_i - N_D) + \sum_{x \in \mathrm{PF}} \left| d(X, N_D) - \bar{d} \right|}{\sum_{i=1}^{m} d(e_i - N_D) + |\mathrm{PF}| \times \bar{d}} \quad \forall e_i \in \mathrm{PF}$$

where $d(X, N_D) = \min_{Y \in N_D, Y \neq X} F(X) - F(Y)^2$ N_D is the set of nondominated solutions, and m is the number of objectives. $e_i (i = 1, 2, \ldots m)$ are extreme solutions in pareto front:

$$\bar{d} = \frac{1}{|\mathrm{PF}|} \sum_{X \in \mathrm{PF}} d(X, N_D).$$

7 Numerical illustration

7.1 Instance settings

The MOOSOP is a new model, and that is why there is no such benchmark dataset to evaluate the performance of our work. The concept of our proposed model is derived from the standard set orienteering problem (SOP) proposed by Archetti et al. [8]. Even SOP has no benchmark dataset in the literature. They have generated the dataset to test their model by using the instances from generalized traveling salesman problem (GTSP). Here, all the customers are grouped into several clusters, and the model tries to find the shortest cycle covering all the clusters by visiting at least one customer from each cluster. To test our proposed MOOSOP model, we also use the instances from the GTSP. We use only those instances for which we can get the Euclidean distance between two customers. We use 12 instances, which are mentioned in the first and the fourth columns of Table 4. One of them is 11berlin52. Here, the first number, i.e., 11, represents the number of clusters, and the last number, i.e., 52, represents the number of nodes or customers. We apply the instances in our proposed model as follows. As there is no concept of the depot in the GTSP, we select the first node as the depot, and we remove the first node from the cluster where it is included in GTSP. For example, in the case of 11berlin52, the first node is removed from its original cluster, which initially has 21 customers. So after removal, it becomes the depot, and that cluster will have now 20 customers. Then, we reduce the customers' number by one. That is, customer 2 will become customer 1, customer 3 will become customer 2 and similarly, customer 52 will become customer 51.

We then generate the profit p_i, the service time t_i required of each cluster and budget $T_{\max}$.

Profit p_i: The profit p_i of cluster s_i is generated by calculating the total number of customers of that cluster, i.e., $|s_i|$.

Service time t_i : The service time t_i of a cluster s_i is calculated by adding the service time required for all the customers of the cluster s_i. The service time required for a customer is calculated using the formula "$1 + (7141\ j + 73)$ mod(100)" to get the pseudorandom number. Angelelli et al. [7] for the clustered OP and Archetti et al. [8] for SOP also used this rule.

Budget $T_{\max}$: The cost of the best-known solution value of the GTSP taken from Fischetti et al. [3] is considered as the budget $T_{\max}$.

We use the same number of clusters in all the instances of GTSP and the same set of customers per clusters except the first one that is already discussed.

7.2 Parameter settings

While using any multi-objective method, parameter tuning has a very influential role. Here, for both the methods, we have used the same values for the characteristic parameters for each instance of the dataset. The essential parameters are population size, the maximum number of generations, crossover probability, mutation probability and archive size. The optimal parameter settings after several trials and errors are given in Table 3. For SPEA2, tournament selection with selection pressure 4 is used, and in the case of NSGAII, Roulette wheel selection mechanism is used. The crossover and mutation mechanisms used in NSGAII and SPEA2 are described in Sects. 5.7 and 5.8. The archive size used in SPEA2 is the same as the population size.

8 Result analysis and discussion

The proposed MOOSOP model is solved using two multi-objective evolutionary algorithms, which are implemented in *C* language on a system of Intel Core i5 CPU T6500 at 2.44 GHz, under Ubuntu 18.4.

8.1 Results

In Results section, we present the outcome of the different tests conducted to measure the performance of the proposed MOOSOP model. Here, each of the 12 instances is solved using NSGAII and SPEA2 for the proposed MOOSOP model. The first front generated using both the algorithms for an independent run on different instances is presented in Table 4. The detailed result of one of the solutions of the first front of an independent run on the instance 11berlin52 using NSGAII is presented in Table 5. The first front obtained on an independent run on two different instances 20rat99 and 22pr107 for the proposed MOOSOP model using two different algorithms NSGAII and SPEA2 is depicted in Fig. 5.

Table 3 Parameter settings of SPEA2 and NSGAII

Parameter	Values
Population size	100
Maximum number of generations	500
Crossover probability	0.9
Mutation probability	0.1
Archive size	100

As we construct a new variant of SOP, no Pareto front is available in the literature. So first, we generate an approximate front close to the Pareto front for both the algorithms. To generate the approximate front for each instance, we make 50 independent runs of both the algorithms and store the first front in an external archive. Then, the nondominated solutions from the archives are treated as the approximate front. For the instance of 11berlin52, the approximate front and the first front of an independent run using NSGAII and SPEA2 are presented in Fig. 6.

To measure the performance of NSGAII and SPEA2, we evaluate three performance measures of convergence and diversity, namely GD, IGD and spread. To perform the statistical analysis, we calculate two of central tendency measures, namely mean and median, and two of variability measures, namely standard deviation and interquartile range, for all the performance metrics. Mean, median, standard deviation and IQR of GD, IGD and spread after 50 runs of NSGAII and SPEA2 for each instance are presented in Table 6. To compare the algorithms, we plot the results. The values in bold are the better outcome when compared for both the NSGAII and SPEA2.

The bar charts of mean, median, standard deviation and IQR of GD, IGD and spread for the proposed model using SPEA2 and NSGAII are given in Figs. 7, 8, 9. The box plots of IGD, GD and spread for the proposed model using NSGAII and SPEA2 are presented in Fig. 10.

8.2 Discussion

Compared to the basic model of SOP, our model here accounts for a number of aspects more in selecting a customer from a cluster, and this makes the model more realistic. Finding only the profit as the primary means of customers' selection does not always generate better customer satisfaction, so we should consider most of the vital issues. We solve the proposed model by using two popular evolutionary algorithms NSGAII and SPEA2. Table 4 presents the first front generated using both the algorithms for an independent run each. From Table 4, we observe the following: Firstly, in most cases, the number of solutions in the first front generated by NSGAII is better than SPEA2,

Table 4 Results of the proposed MOOSOP model using NSGAII and SPEA2

Instance	NSGAII		SPEA2		Instance	NSGAII		SPEA2	
	Score	Satisfaction	Score	Satisfaction		Score	Satisfaction	Score	Satisfaction
11berlin52	25	0.911131	10	0.795053	20KroA100	23	0.806908	30	0.840378
	26	0.859422	42	0.615637		31	0.764991	32	0.780118
	30	0.797503	29	0.778599		44	0.754361	36	0.711796
	33	0.73364				45	0.42299	40	0.665993
	36	0.715077						46	0.62048
	42	0.638843						48	0.528135
16eil76	3	0.993056	12	0.996377	20rat99	5	0.981949	11	0.996497
	12	0.98913	17	0.407891		6	0.976031	14	0.562747
	17	0.407891	21	0.304404		8	0.751323	18	0.4215
	21	0.304404				12	0.537127	23	0.400378
						15	0.475363	25	0.278769
						18	0.380451	29	0.16805
11eil51	17	0.145776	15	0.136553	20rd100	17	0.855372	43	0.354695
	9	0.499847	12	0.264865		29	0.853439	21	0.885277
	14	0.279016	8	0.397541		37	0.634615	22	0.76819
	11	0.300447	7	0.997925		38	0.624906	40	0.516866
	10	0.328767				45	0.315767	36	0.728092
	7	0.997925							
14st70	7	0.997416	3	0.993421	21eil101	6	0.9725	23	0.418166
	15	0.57796	27	0.189239		9	0.915709	16	0.998915
	16	0.481541	6	0.987654		11	0.509931	19	0.496918
	19	0.439358	8	0.973684		13	0.334176	18	0.503886
	20	0.38256	25	0.308839		15	0.259507		
	22	0.321028				18	0.240728		
	25	0.308839							
	26	0.256603							
	29	0.151							
	34	0.129146							
16pr76	46	0.770578	30	0.812443	21lin105	18	0.96209	49	0.621864
	58	0.625219	36	0.709713		41	0.86469	47	0.843526
	55	0.762072	42	0.709346		42	0.782046		
	53	0.537719	44	0.640836		46	0.608355		
	27	0.771851	51	0.637589					
	50	0.627731	53	0.63628					
	33	0.692688	56	0.620514					
24pr124	10	0.838549	16	0.796787	22pr107	11	0.940607	15	0.86529
	18	0.799657	22	0.742784		12	0.779539	34	0.583521
	31	0.753000	27	0.738662		15	0.724448	32	0.747036
	38	0.714322	31	0.713963		16	0.651724	28	0.775902
	43	0.696375	34	0.689739		24	0.624993		
	52	0.623741				30	0.619732		
						37	0.583984		

Table 5 Detailed results of the proposed MOOSOP model on instance 11berlin52 using NSGAII

Sl. No.	Score	Satisfaction	Solution	
1	42	0.638843	Cluster sequence	8 1 4 3 7 10 9 6 2 11 5
			City sequence	51 13 6 19 3 45 20 47 2 26 33
			Service time	87.0 41.0 14.0 218.0 166.0 55.0 43.0 0.0 179.0 33.0 1.0
2	25	0.911131	Cluster sequence	8 1 4 3 7 10 9 6 2 11 5
			City sequence	51 52 4 8 31 43 20 47 2 26 33
			Service time	72.0 60.0 1116.0 210.0 177.0 4.0 114.0 0.0
3	30	0.797503	Cluster sequence	8 1 4 3 7 10 9 6 2 11 5
			City sequence	51 13 6 19 3 45 20 47 2 26 33
			Service time	69.0 40.0 1130.0 191.0 118.0 3.0 157.0 0.0 102.0 92.0 12.0
4	36	0.715077	Cluster sequence	8 1 4 3 7 10 9 6 2 11 5
			City sequence	51 13 6 19 31 45 20 47 2 26 33
			Service time	87.0 41.0 14.0 218.0 166.0 55.0 43.0 0.0 179.0 33.0 1.0
5	33	0.733640	Cluster sequence	4 5 7 10 9 11 2 3 1 8 6
			City sequence	33 31 45 20 28 2 8 14 11 47
			Service time	698.0 26.0 166.0 28.0 207.0 76.0 155.0 70.0 33.0 28.0 0.0
5	26	0.859422	Cluster sequence	10 4 7 1 8 11 9 3 6 2 5
			City sequence	43 48 17 52 11 28 50 19 47 7 33
			Service time	55.0 972.0 173.0 39.0 66.0 121.0 115.0 94.0 0.0 86.0 12.0

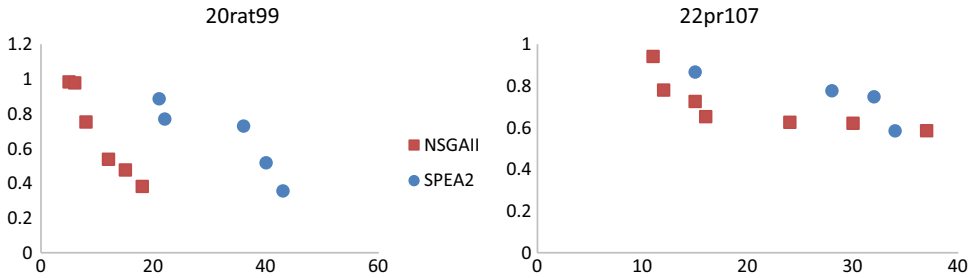


Fig. 5 First front obtained for the instances 20rat99 and 22pr107 using NSGAII and SPEA2

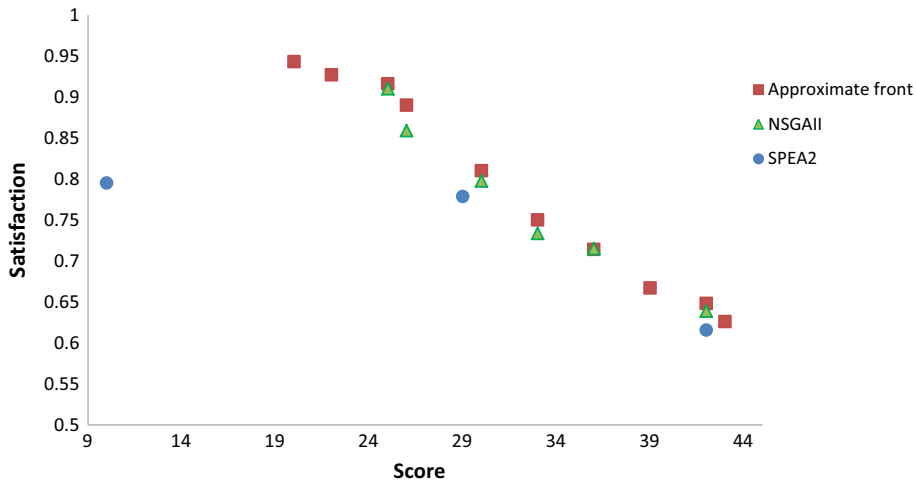


Fig. 6 Graphical representation of the approximate front and the first front of an independent run using NSGAII and SPEA2 for the instance 11berlin52

Table 6 Performance matrices for NSGAII and SPEA2

Instance	NSGAII						SPEA2					
	IGD		GD		Spread		IGD		GD		Spread	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
11berlin52	0.042144	0.007514	**0.034391**	0.004493	0.055172	**0.015543**	**0.039561**	**0.005829**	0.0371	**0.003947**	**0.038457**	0.022756
16eil76	0.041938	0.005309	**0.036008**	**0.002954**	0.046643	0.020259	**0.040906**	**0.005266**	0.036874	0.002981	**0**	**0**
11eil51	0.045767	**0.007971**	**0.035983**	**0.003407**	**0.050241**	**0.030441**	**0.042245**	0.009865	0.048811	0.007458	0.059453	0.039919
14st70	0.040291	0.006389	0.041416	0.004052	**0**	**0**	**0.038916**	**0.006362**	**0.039529**	**0.003996**	0.049073	0.028695
16pr76	0.044183	0.007733	0.036781	**0.003869**	**0.038189**	0.026826	**0.044113**	**0.004219**	**0.035778**	0.005609	0.041732	**0.022778**
24pr124	**0.039208**	**0.00588**	**0.036471**	0.003964	0.055406	**0.018309**	0.045016	0.007031	0.04004	**0.003015**	**0.032742**	0.023439
20kroa100	0.042282	0.005288	**0.033415**	**0.003465**	**0**	**0**	**0.037995**	**0.003958**	0.03913	0.003626	0.025367	0.008402
20rat99	0.043704	**0.007145**	0.036407	0.004151	**0.038737**	**0.021228**	**0.03986**	0.009348	**0.035441**	**0.003556**	0.051481	0.029122
20rd100	0.045481	0.008488	**0.034389**	**0.004575**	**0.008908**	**0.001873**	**0.044915**	**0.005005**	0.040756	0.006327	0.059586	0.033997
21eil101	**0.038963**	**0.007845**	0.035141	**0.003633**	0.049126	0.017938	0.044226	0.009012	**0.033217**	0.0044	**0.009052**	**0.001968**
21lin105	**0.038064**	**0.004761**	0.040407	**0.003957**	**0.050727**	0.007747	0.046116	0.0069	**0.040223**	0.005646	0.058357	**0.038723**
22pr107	**0.038276**	**0.005323**	**0.037045**	**0.004219**	**0**	**0**	0.044155	0.007744	0.03722	0.005538	0.023348	0.008035
Instance	NSGAII						SPEA2					
	IGD		GD		Spread		IGD		GD		Spread	
	Median	IQR	Median	IQR	Median	IQR	Median	IQR	Median	IQR	Median	IQR
11berlin52	0.041746	0.013467	**0.036481**	0.007956	0.057132	**0.033185**	**0.040063**	**0.0126**	0.03737	**0.006288**	**0.034205**	0.046229
16eil76	0.042268	**0.009402**	**0.035532**	0.006137	0.045993	0.03932	**0.040596**	0.010185	0.037307	**0.004586**	**0**	**0**
11eil51	0.046286	**0.012871**	**0.036773**	**0.005318**	**0.057528**	**0.063934**	**0.043527**	0.014499	0.045177	0.011955	0.059016	0.088033
14st70	**0.039091**	0.012621	0.041403	0.008962	**0**	**0**	0.039532	**0.011061**	**0.041059**	**0.007848**	0.045479	0.053887
16pr76	0.044322	0.01379	**0.036167**	**0.008579**	**0.038877**	0.060384	**0.043776**	**0.007388**	0.037187	0.008865	0.051991	**0.048315**
24pr124	**0.038761**	0.011523	**0.037136**	0.005609	0.059599	**0.032896**	0.043593	**0.009145**	0.039849	**0.004169**	**0.040018**	0.052732
20kroa100	0.041943	0.006731	**0.033252**	0.006061	**0**	**0**	**0.037465**	**0.005674**	0.039628	**0.005351**	0.028267	0.009715
20rat99	0.042132	**0.015176**	0.036072	0.007362	**0.046493**	**0.041356**	**0.036062**	0.019512	**0.035427**	**0.006761**	0.059163	0.065868
20rd100	0.046493	**0.010217**	**0.03416**	**0.008442**	**0.008388**	**0.00223**	**0.045748**	0.010813	0.040245	0.008852	0.056382	0.065123
21eil101	**0.037148**	0.011592	0.035204	**0.006886**	0.051918	0.037439	0.042406	**0.010963**	**0.034435**	0.008925	**0.009045**	**0.002083**
21lin105	**0.037705**	**0.007851**	**0.041004**	**0.007708**	**0.051538**	**0.010308**	0.046076	0.011353	0.042546	0.011806	0.075591	0.087389
22pr107	**0.036864**	**0.01032**	0.036429	**0.005489**	**0**	**0**	0.046705	0.015183	**0.035444**	0.009968	0.023442	0.010753

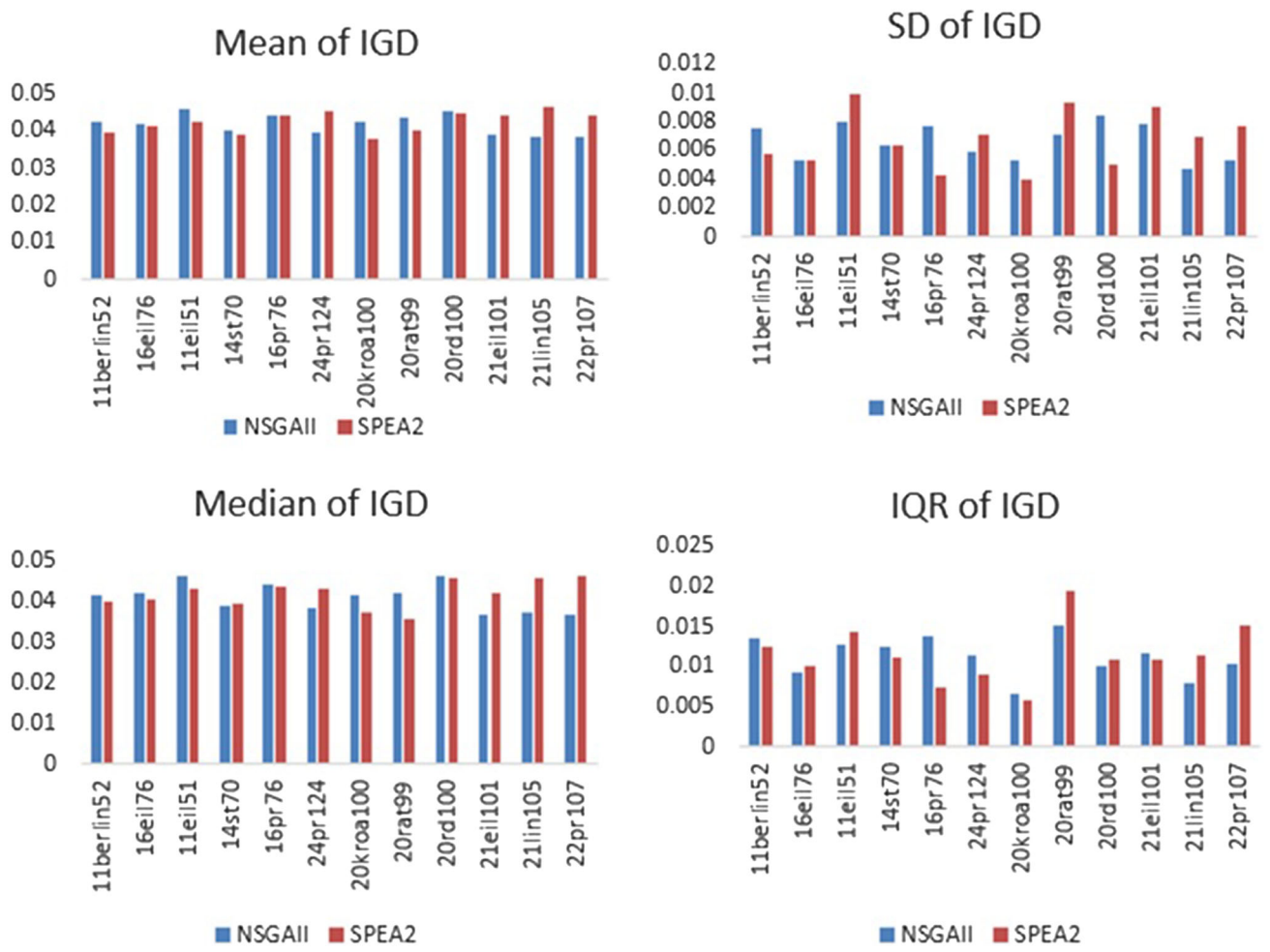


Fig. 7 Mean, SD, median and IQR of IGD for both the MOEAs

and secondly, in general, the first front generated by both the algorithms intersects and in few cases, they dominate each other. Therefore, in this scenario, deciding the best algorithm between these two is difficult. Hence, we use some performance metrics to compare the algorithms. The central tendency measures of the performance matrices, viz. GD, IGD, and spread, are presented in Table 6. The bar charts are used to compare these results and are depicted in Figs. 7, 8, 9.

In some cases, the mean and standard deviation of the IGD are better for SPEA2. When we consider GD, in most of the instances NSGAII performs better than SPEA2, but when we consider spread, SPEA2 is better than NSGAII. The box plots of GD, IGD and spread are presented in Fig. 10. From the box plot, it is clear that NSGAII performs better than SPEA2 concerning IGD and GD, but SPEA2 is better than NSGAII for spread.

9 Conclusion

The multi-objective open set orienteering problem (MOOSOP) deals with more than one objective. In this paper, we consider a practical variant of SOP in which the goals are the maximization of the profit score and the maximization of customer satisfaction. The model considers third-party logistics and is solved by NSGAII and SPEA2 algorithms. From result analysis, we can conclude that although many of them can be used equally well here, the success rate in reaching the goal depends on the tuning of the parameters and the method of implementation. In our work, we use three performance metrics, namely GD, IGD and spread, to analyze the performance of the algorithms. Finally, we can conclude that NSGAII performs better than SPEA2 in this work by considering the best of these three measures, as NSGAII performs better than SPEA2 with respect to IGD and GD, but SPEA2 performs better than NSGA2 with respect to spread.

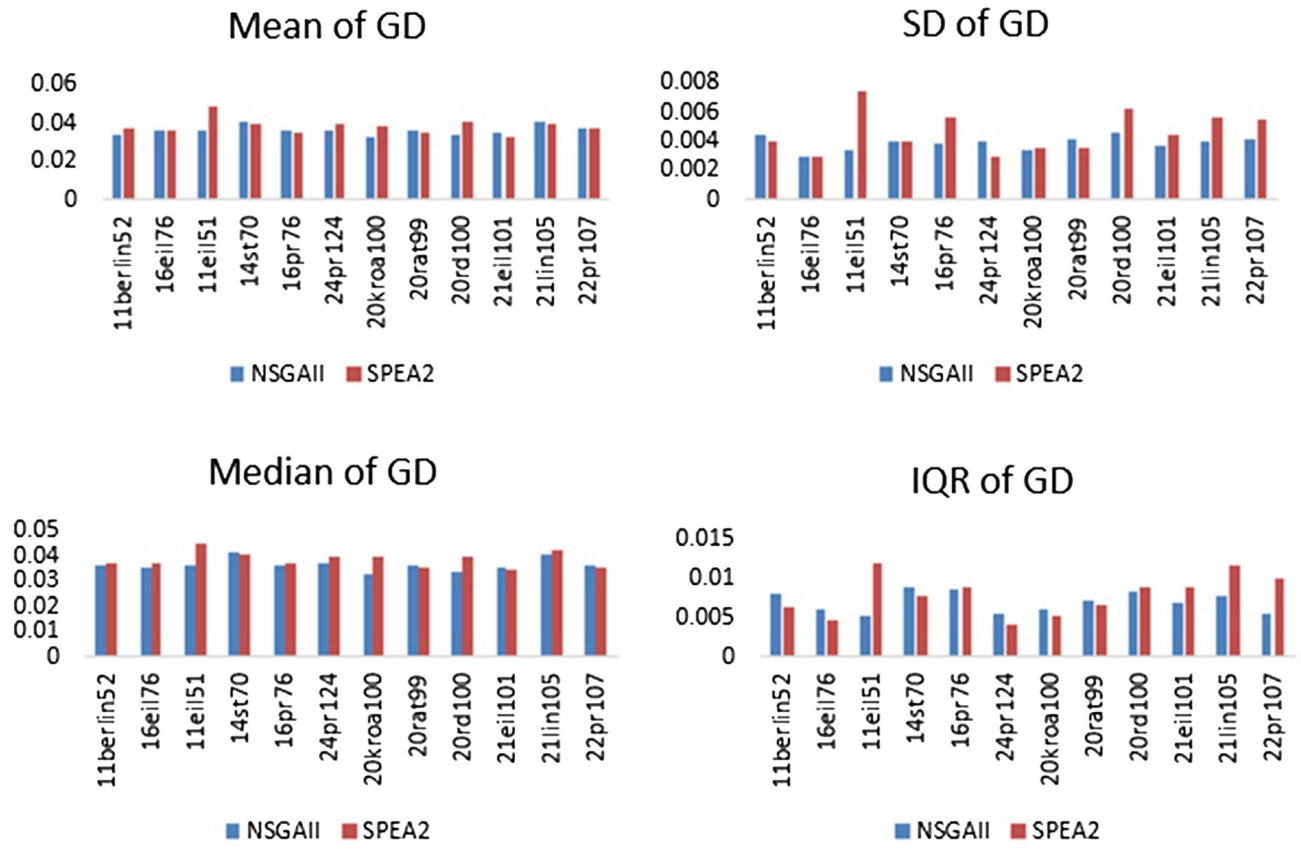
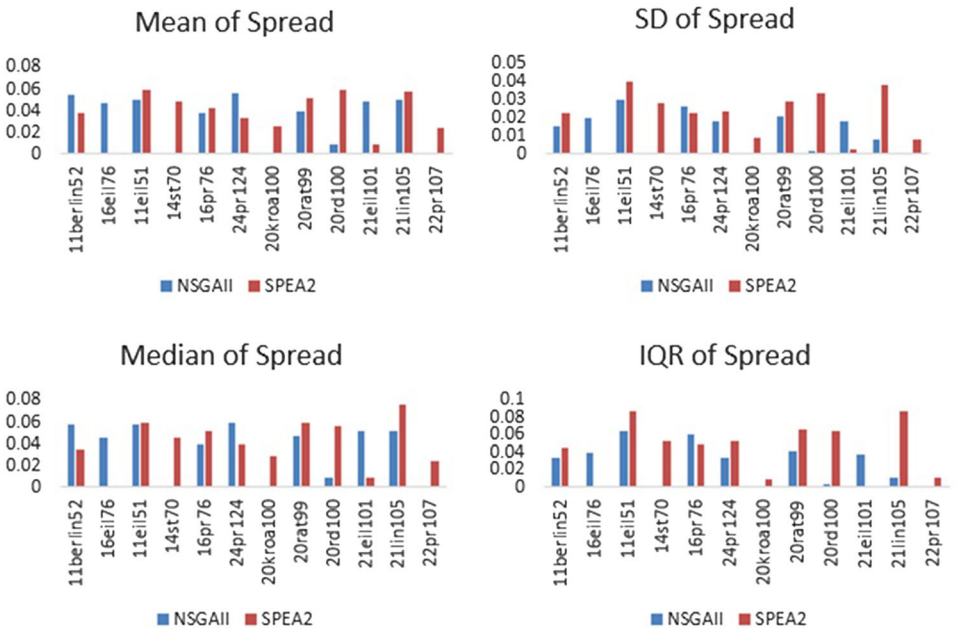


Fig. 8 Mean, SD, median and IQR of GD for both the MOEAs

Fig. 9 Mean, SD, median and IQR of spread for both the MOEAs



Future research may focus on the use of uncertainties for different parameters such as travel time, service time, operating costs, etc. In this paper, we also seek attention to future research in the domain of the orienteering problem in a sustainable environment. Nowadays, many researchers are enormously working on environmental issues.

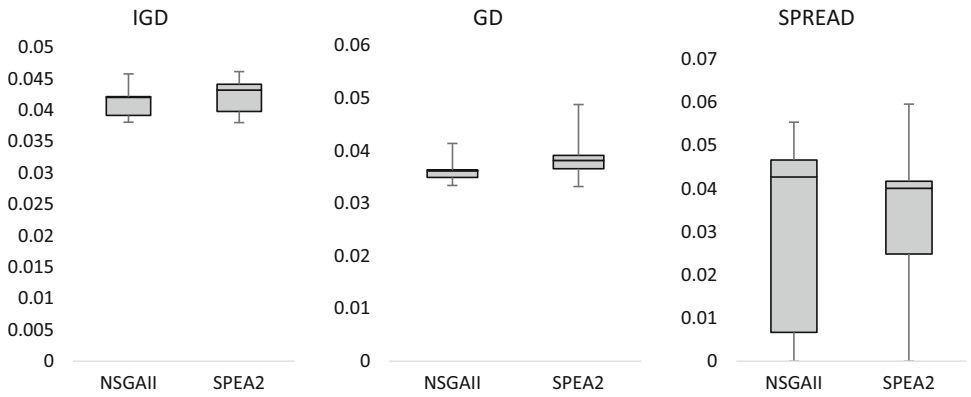


Fig. 10 Box plot of IGD, GD and spread for both the MOEAs

Therefore, a green MOOSOP may be an up-and-coming area of research for further works based on this paper.

Compliance with ethical standards

Conflict of interest All authors of this research paper declare that there is no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Informed consent was obtained from all participants included in the study.

References

1. Lawler E, Lenstra J, Rinnooy K, Shmoys D (1985) The traveling salesman problem: a guided tour of combinatorial optimization. Wiley, NY
2. Balas E (1989) The prize collecting traveling salesman problem. Networks 19:621–636
3. Fischetti M, Gonzlez JJS, Toth P (1997) A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. Oper Res 45(3):378–394
4. Feillet D, Dejax P, Gendreau M (2005) Traveling salesman problems with profits. Transpor Sci 39:188–205
5. Dantzig G, Ramser J (1959) The truck dispatching problem'. Manag Sci 6:80. https://doi.org/10.1287/mnsc.6.1.80
6. Tsiligirides T (1984) Heuristic methods applied to orienteering. J Oper Res Soc 35:797–809
7. Angelelli E, Archetti C, Vindigni M (2014) The clustered orienteering problem. Eur J Oper Res 238:404–414
8. Archetti C, Carrabs F, Cerulli R (2017) The set orienteering problem. Eur J Oper Res 267(1):264–272
9. Pěnička R, Faigl J, Saska M (2019) Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants. Eur J Oper Res. https://doi.org/10.1016/j.ejor.2019.01.047
10. Faigl J, Pěnička R, Best G (2016) Self-organizing map-based solution for the orienteering problem with neighborhoods. In: Proceedings of the IEEE international conference on systems, man, and cybernetics, pp 1315–1321
11. Pěnička R, Faigl J, Váňa P, Saska M (2017) Dubins orienteering problem. IEEE Robot Autom Lett 2(2):1210–1217
12. Chao I, Golden B, Wasil E (1996) Theory and methodology—a fast and effective heuristic for the orienteering problem. Eur J Oper Res 88:475–489
13. Laporte G, Martello S (1990) The selective traveling salesman problem. Discrete Appl Math 26:193–207
14. Kataoka S, Morito S (1988) An algorithm for the single constraint maximum collection problem. J Oper Res Soc Jpn 31(4):515–530
15. Arkin E, Mitchell J, Narasimhan G (1998) Resource-constrained geometric network optimization. In: Proceedings 14th ACM symposium on computational geometry, June, pp 307–316
16. Vansteenwegen P, Souffriau W, Van Oudheusden D (2011) The orienteering problem: a survey. Eur J Oper Res 209:1–10. https://doi.org/10.1016/j.ejor.2010.03.045
17. Gunawan A, Lau H, Vansteenwegen P (2016) Orienteering problem: a survey of recent variants, solution approaches, and applications. Eur J Oper Res. https://doi.org/10.1016/j.ejor.2016.04.059
18. Mukhina K, Visheratin A, Nasonov D (2019) Orienteering problem with functional profits for multi-source dynamic path construction. PLoS ONE 14(4):e0213777. https://doi.org/10.1371/journal.pone.0213777
19. Hanafi S, Mansini R, Zanotti R (2019) The multi-visit team orienteering problem with precedence constraints. In: European journal of operational research (**In Press**)
20. Yu V, Redi A, Jewpanya P, Gunawan A (2019) Selective discrete particle swarm optimization for the team orienteering problem with time windows and partial scores. Comput Ind Eng. https://doi.org/10.1016/j.cie.2019.106084
21. Schilde M, Doerner KF, Hartl RF, Kiechle G (2009) Metaheuristics for the bi-objective orienteering problem. Swarm Intelligence. 3(3):179–201. https://doi.org/10.1007/s11721-009-0029-5
22. Chen YH, Sun WJ, Chiang TC (2015) Multiobjective orienteering problem with time windows: An ant colony optimization algorithm. In: 2015 conference on technologies and applications of artificial intelligence, pp 128–135 (TAAI) https://doi.org/10.13140/rg.2.1.2461.3849
23. Mei Y, Salim F, Li X (2016) Efficient meta-heuristics for the multi-objective time-dependent orienteering problem. Eur J Oper Res. https://doi.org/10.1016/j.ejor.2016.03.053
24. Yu V, Jewpanya P, Yang ZY, Redi P, Agus Y, Idrakarna P (2017) Solving the multi-objective orienteering problem with time windows using simulated annealing. In: Proceedings of the international conference on innovation and management 2017, Tokyo, Japan
25. Wang J, Guo J, Zheng M, Wang Z, Li Z (2018) Uncertain multiobjective orienteering problem and its application to UAV reconnaissance mission planning. J Intell Fuzzy Syst 34(4):2287–2299. https://doi.org/10.3233/JIFS-171331

26. Hu W, Fathi M, Pardalos P (2018) A multi-objective evolutionary algorithm based on decomposition and constraint programming for the multi-objective team orienteering problem with time windows. Appl Soft Comput 73:383–393
27. Hapsari I, Surjandari I, Komarudin KJ, Int Ind Eng (2019) Solving multi-objective team orienteering problem with time windows using adjustment iterated local search. J Ind Eng Int 15(4):679–693. https://doi.org/10.1007/s40092-019-0315-9
28. Yahiaoui AE, Moukrim A, Serairi M (2017) Hybrid Heuristic for the clustered orienteering problem. In: Bektaş T, Coniglio S, Martinez-Sykora A, Voß S. (eds) Computational logistics. ICCL 2017. Lecture Notes in Computer Science, Springer, Cham, vol 10572, pp 19–33. https://doi.org/10.1007/978-3-319-68496-3_2
29. Álvarez-Miranda E, Luipersbeck M, Sinnl M (2017) Gotta (efficiently) catch them all: pokémon GO meets orienteering problems. Eur J Oper Res. https://doi.org/10.1016/j.ejor.2017.08.012
30. Deb K, Agrawal S, Pratap A, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197
31. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength pareto evolutionary Algorithm. TIK-Report, p 103
32. Goldberg D, Lingle R (1985) Alleles, Loci and the traveling salesman problem. In: Proceedings of the 1st international conference on genetic algorithms and their applications, Los Angeles, USA, pp 154–159. https://doi.org/10.1155/2017/7430125
33. Veldhuizen DA, Lamont GB (1998) Multiobjective evolutionary algorithm research: a history and analysis. Technical Report TR-98-03, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright Paterson, AFB, OH
34. Zhou A, Jin Y, Zhang Q, Sendho B, Tsang E (2006) Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion. In: 2006 IEEE congress on evolutionary computation (Sheraton Vancouver Wall Center Vancouver, BC, Canada, pp 3234–3241