Expert Systems With Applications 176 (2021) 114779

Contents lists available at ScienceDirect

Expert Systems With Applications

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

NSGA-II with objective-specific variation operators for multiobjective vehicle routing problem with time windows

Gaurav Srivastava [a], Alok Singh [a], Rammohan Mallipeddi [b,*]

[a] School of Computer and Information Sciences, University of Hyderabad, Hyderabad 500 046, India
[b] Deaprtment of Artificial Intelligence, School of Electronics Engineering, Kyungpook National University, Daegu 41566, Republic of Korea

ARTICLE INFO

Keywords:
Evolutionary algorithms
Multiobjective optimization
Nondominated sorting genetic algorithm II
Vehicle routing problem with time windows

ABSTRACT

Vehicle routing problem with time windows (VRPTW) is a pivotal problem in logistics domain as it possesses multiobjective characteristics in real-world applications. Literature contains a general multiobjective VRPTW (*MOVRPTW*) with five objectives along with *MOVRPTW* benchmark instances that are derived from real-world data. In this paper, we have proposed a nondominated sorting genetic algorithm II (NSGA-II) based approach with objective-specific variation operators to address the *MOVRPTW*. In the proposed NSGA-II approach, the crossover and mutation operators are designed by exploiting the problem characteristics as well as the attributes of each objective. The performance of the proposed approach is evaluated on the standard benchmark instances of the problem and compared with the state-of-the-art approach available in literature. The computational results demonstrate the superiority of our approach over the state-of-the-art approach for the *MOVRPTW*.

1. Introduction

Over the last several decades, vehicle routing problem (VRP) and its variants have attained massive popularity owing to their ability to model a wide range of real-world applications pertaining to various domains. Their applications include transportation planning, supply chain management in logistics networks, production management and so on (Hoff, Andersson, Christiansen, Hasle, & Løkketangen, 2010; Jozefowiez, Semet, & Talbi, 2008). The objective of VRP is to design an optimal set of delivery routes for a fleet of vehicles in order to serve a given set of customers of a company's supply chain. It represents the essence of assignment and routing of vehicles with minimum cost in a transportation supply management. Hence, it is a crucial problem in logistics management and also one of the most widely studied problems in the domain of combinatorial optimization. VRP is an $\mathcal{NP}$-hard problem as it generalizes traveling salesman problem which is $\mathcal{NP}$-hard (Garey & Johnson, 1979; Lenstra & Kan, 1981). Since its inception in 1959 (Dantzig & Ramser, 1959), many variants of VRP have been proposed in the literature such as Ip, Wang, and Cho (2013) and Chen, Chou, Hsueh, and Ho (2011), due to the incorporation of various constraints arising in real-world applications. The reader is referred to Eksioglu, Vural, and Reisman (2009) for a comprehensive survey on VRP and its variants. Some recent works on VRP can be found in Gmira, Gendreau, Lodi, and Potvin (2021), Molina, Salmeron, and Eguia (2020), Osaba, Yang, and Del Ser (2020), Fachini and Armentano (2020).

This paper is concerned with an important variant of VRP known as VRP with time windows (VRPTW) in the literature. VRPTW has the characteristics of VRP along with time window constraints on the customers, and hence, it is also an $\mathcal{NP}$-hard problem. VRPTW is more relevant than VRP in real-world applications like those presented in Prescott-Gagnon, Desaulniers, Drexl, and Rousseau (2010), Buhrkal, Larsen, and Ropke (2012) and Benjamin and Beasley (2010). The problem seeks the assignment and routing of a fleet of vehicles departed from a central depot, in order to serve a given set of customers. Each vehicle has a fixed capacity, and, each customer needs to be served once by exactly one vehicle. Each customer has a time window during which only the delivery to that customer can be made. The objective of VRPTW is to find such assignments and routing of vehicles under the given constraints so that the cost is minimized. In VRPTW, the cost can be number of vehicles/routes, total distance traveled by all vehicles, total travel time, etc. The cost can also be in terms of customer satisfaction, so in some applications, cost is defined as service penalties when a customer is served after their time window or receives an incomplete delivery (Jozefowiez et al., 2008). In case of multiple and diverse costs, all costs should be optimized simultaneously.

* Corresponding author.
E-mail addresses: gauravsrignp@gmail.com (G. Srivastava), alokcs@uohyd.ernet.in (A. Singh), mallipeddi.ram@gmail.com (R. Mallipeddi).

https://doi.org/10.1016/j.eswa.2021.114779
Received 20 March 2020; Received in revised form 6 December 2020; Accepted 20 February 2021
Available online 13 March 2021

Owing to the practical importance of VRPTW, intensive research efforts have been made to develop exact, heuristic, and metaheuristic approaches (Bräysy & Gendreau, 2005). Being an $\mathcal{NP}$-hard problem, the suitability of the exact approaches (Baldacci, Mingozzi, & Roberti, 2012) is limited to small problem instances only, and hence, metaheuristic approaches are usually employed to solve large VRPTW instances in a reasonable amount of computation time. As a result, many metaheuristic approaches (Oliveira & Vasconcelos, 2010; Ursani, Essam, Cornforth, & Stocker, 2011; Gong et al., 2012; Bent & Van Hentenryck, 2004; Bräysy, Hasle, & Dullaert, 2004; Lim & Zhang, 2007; Repoussis, Tarantilis, & Ioannou, 2009; Garcia-Najera & Bullinaria, 2011; Tan, Chew, & Lee, 2006; Ombuki, Ross, & Hanshar, 2006; Ghoseiri & Ghannadpour, 2010; Hsu & Chiang, 2012) have been proposed in the literature for VRPTW. Several previous studies considered VRPTW as a single objective problem. For example, Oliveira and Vasconcelos (2010) and Ursani et al. (2011) presented approaches based on simulated annealing and genetic algorithm respectively to minimize the total travel distance as the sole objective. Literature also contains several research studies in which VRPTW is addressed as a multiobjective problem. Often, two objectives, viz. minimization of total travel distance and minimization of number of routes are considered. In most of such problem variants, multiple objectives are transformed into a single-objective by using scalar techniques like the one presented in Gong et al. (2012).

The multiobjective VRPTW, referred to as *MOVRPTW* in the literature, may have objectives so that the optimization of one objective may lead to deterioration in the values of other objectives. Further, relative importance of different objectives may vary from one domain to another or from one scenario to another. Therefore, instead of finding a single solution for VRPTW, it is desirable to present a set of solutions having trade-offs among different objectives, as the preference of decision maker may not be known a priori (Garcia-Najera & Bullinaria, 2011). Thus, VRPTW is primarily a multiobjective optimization problem (MOP). However, there exist only a few works in the literature which address multiobjective VRPTW. The main characteristics of multiobjective approaches is to handle all the objectives with the same preference, and hence, these approaches should generate a set of Pareto optimal solutions. Garcia-Najera and Bullinaria (2011), Tan et al. (2006), Ombuki et al. (2006), Ghoseiri and Ghannadpour (2010) and Hsu and Chiang (2012) report approaches which considered VRPTW as a bi-objective problem. However, these approaches were tested on the widely used Solomon's problem instances (Solomon, 1987). The Solomon's problem instances (Solomon, 1987) were originally proposed for single-objective VRPTW, i.e., minimizing the total travel distance as the sole objective. Castro-Gutierrez, Landa-Silva, and Pérez (2011) and Castro-Gutierrez (2012) analyzed the conflicting nature of various objectives in Solomon's instances (Solomon, 1987), and concluded that the Solomon's instances do not represent the ideal benchmark scenarios for multiobjective VRPTW due to the following reasons (Castro-Gutierrez et al., 2011):

1. Dependence relationships between objectives are weak in Solomon's instances.
2. The data in Solomon's instances are based on Euclidean distance for both travel distance and travel time, which does not represent a realistic scenario with respect to real-world applications.

Castro-Gutierrez et al. (2011) defined a general *MOVRPTW* with five objectives used commonly in various multiobjective VRPTW variants in the literature. These five objectives are as follows:

1. Minimizing the number of vehicles,
2. Minimizing the total travel distance by all the vehicles,
3. Minimizing the makespan, i.e., longest travel time among all routes (from/to depot),
4. Minimizing the total waiting time due to early arrivals
5. Minimizing the total delay time due to late arrivals

The significance of these objectives can vary from domain to domain. For example, with reference to food delivery and healthcare industries, delay time is of utmost importance. Goods transportation industry may consider total distance travelled as a crucial objective to minimize than other objectives as consumption of fuel is directly proportional to distance travelled, and minimizing the total distance travelled by all the vehicles is important from the perspective of monetary considerations. For small scale industries, minimization of number of vehicle may be of highest priority in comparison to other objectives. Castro-Gutierrez et al. (2011) also presented a set of *MOVRPTW* instances utilizing the data from a distribution company in Tenerife, Spain. These *MOVRPTW* instances are more realistic as well as truly multiobjective in nature as far as *MOVRPTW* is concerned. In Castro-Gutierrez et al. (2011), a non-dominated sorting genetic algorithm II (NSGA-II) (Deb, Pratap, Agarwal, & Meyarivan, 2002) based approach is presented which is also the first approach using the new *MOVRPTW* instances. These instances are publicly available for download (Castro-Gutierrez, Landa-Silva, & Pérez, 2012).

The *NSGA-II* approach presented in Castro-Gutierrez et al. (2011) uses generic crossover and mutation operators to generate offspring. Generic operators do not exploit the structure of the problem and the characteristics of the objectives, and therefore may fail to yield high quality solutions. A new local search-based multiobjective algorithm (*LSMOVRPTW*) is presented in Zhou and Wang (2015), which is the second approach for *MOVRPTW* in the literature. In *LSMOVRPTW* approach, for each objective, one local search procedure is devised by considering the problem structure of *MOVRPTW* with corresponding objective. First, a solution is randomly selected from the archive and then all local search procedures are applied iteratively. The newly generated solutions are updated in the archive by using $\in$-dominance concept (Laumanns, Thiele, Deb, & Zitzler, 2002; Deb, Mohan, & Mishra, 2005). Computational results and their analysis presented in Zhou and Wang (2015) show that *LSMOVRPTW* performed better than *NSGA-II* approach presented in Castro-Gutierrez et al. (2011). *LSMOVRPTW* is currently the state-of-the-art approach for *MOVRPTW*.

MOVRPTW is a relatively understudied problem as the only approaches mentioned in the previous paragraph are available in the literature. The *LSMOVRPTW* uses local search approach to generate new solutions. Usually, local search approaches are considered as exploitative in nature and lacks the exploration characteristics. NSGA-II algorithm is a Pareto-based multiobjective optimization technique that is among most successful techniques in the literature for addressing multiobjective problems. Despite these facts, *LSMOVRPTW* is able to beat *NSGA-II* approach of Castro-Gutierrez et al. (2011) quite comfortably due to the generic nature of variation operators (crossover and mutation) used in the latter approach. These generic variation operators do not make use of characteristics specific to *MOVRPTW* and its various objectives, and, hence lack appropriate exploitation characteristics. As a result, this genetic algorithm yields inferior solutions in comparison to *LSMOVRPTW*. This stressed the need for the variation operators appropriately utilizing the characteristics of *MOVRPTW* and its various objectives. This has motivated us to develop such variation operators and a new approach based on NSGA-II utilizing such variation operators. As our crossover and mutation operators are designed as per the characteristics of *MOVRPTW* and the characteristics of each objective, we have a dedicated crossover operator and a dedicated mutation operator for each objective. Hence, our approach uses five different crossover operators and five different mutation operators. Another key feature of our approach is that for the last 25% generations, crossover is not used and only mutation is used. This is due to the fact that our mutation operators have better exploitation ability than our crossover operators. Each mutation operator at these final iterations acts as a local search to some extent for its respective objective and improves the solution quality further in the population with respect to its objective. In addition to these two key features, our new NSGA-II approach differs from the NSGA-II approach of Castro-Gutierrez et al. (2011) in the manner in

which initial population is generated and parents are selected for crossover and mutation. NSGA-II approach of Castro-Gutierrez et al. (2011) uses a constructive method for initial population generation that aims at satisfying first the customers farthest from the depot. On the other hand, we have used a method similar to the method used by Zhou and Wang (2015) to generate an initial solution. NSGA-II approach of Castro-Gutierrez et al. (2011) uses deterministic binary tournament selection, whereas our method has used probabilistic binary tournament selection. Computational results on benchmark instances from the literature demonstrate that our new approach is capable of producing superior solutions in comparison to *LSMOVRPTW*.

The remaining part of the paper is structured as follows: Section 2 presents the problem formulation of the *MOVRPTW*. In Section 3, we present an overview of existing approaches for *MOVRPTW* and the multiobjective optimization. The proposed NSGA-II approach for *MOVRPTW* is presented in Section 4. Section 5 presents the computational results. Finally, Section 6 presents several important conclusions and the directions in which future research based on the work reported in the paper can be carried out.

2. Problem description

This section introduces the notational conventions used and provides a formal definition of *MOVRPTW*. This paper uses the same formulation and notational conventions as presented in Zhou and Wang (2015). VRPTW contains a set of vertices $v = \{0, \ldots, N\}$, where vertex 0 corresponds to depot and remaining vertices represent customers. The fleet of vehicles depart from the depot in order to serve the customers, and each vehicle has a capacity C. Each customer i is associated with a demand of goods $g_i > 0$ and a time window $[b_i, e_i]$. The time window constraint represents that the customer i must be served in-between and including the time b_i and e_i. In other words, e_i is the latest service time such that the vehicle should arrive at customer i's location. If vehicle arrives at customer i's location before the earliest service time b_i, then vehicle must wait until b_i in order to serve the customer. This situation leads to waiting of the vehicle, and, waiting time will be the difference in time between the earliest service time b_i and the arrival time. Each customer i has a service time s_i, which is the time taken in delivery of goods after arrival of vehicle at customer's location. The time window constraint for depot is $[0,e_0]$, i.e., vehicles start from depot at time 0 and must return to depot on or before time e_0. It is considered that the demand of goods for depot is $g_0 = 0$. Here, d_{ij} and t_{ij} represent the travel distance and travel time between vertices i and j respectively. The aim of the *MOVRPTW* is to find the set of M routes $R = \{r_1, \ldots, r_M\}$ with the minimum cost, such that each customer is served exactly once by some vehicle and each vehicle should start at time 0 & return to the depot on or before time e_0. Let r_j represents the j^{th} route such that, $r_j = \langle c(1,j), \ldots, c(N_j,j)\rangle$ is the sequence of customers served in j^{th} route which is having N_j customers and $c(i,j)$ specifies some customer visited at i^{th} position in this route. Furthermore, the depot is represented as $c(0,j) = c(N_j + 1,j) = 0$.

The total travel distance of the j^{th} route is defined as

$$Dist_j = \sum_{i=0}^{N_j} d_{c(i,j)c(i+1,j)}. \tag{1}$$

Here, $i = 0 = N_j + 1$ represents the depot and $d_{c(i,j)c(i+1,j)}$ provides the distance between customer i and customer $i + 1$.

To find the total travel time, we need to define arrival time and leaving time of a vehicle at a customer location. Let $a_{c(i,j)}$ be the arrival time and $l_{c(i,j)}$ be the leaving time of vehicle j at the i^{th} customer $c(i,j)$ in this route. It is pertinent to mention that vehicle j represents the j^{th} route.

The arrival time is defined as

$$a_{c(i,j)} = l_{c(i-1,j)} + t_{c(i-1,j)c(i,j)}. \tag{2}$$

The leaving time of vehicle j from depot is considered as 0, i.e., $l_{c(0,j)} = 0$ and $t_{c(i-1,j)c(i,j)}$ represents the travel time between customer $i-1$ and i.

If a vehicle arrives early at a customer location then the vehicle needs to wait until the earliest service time of this customer. This scenario incurs a waiting time for the vehicle. The waiting time of vehicle j at i^{th} customer $c(i,j)$ can be computed as

$$w_{c(i,j)} = \begin{cases} 0 & \text{if } a_{c(i,j)} \geqslant b_{c(i,j)} \\ b_{c(i,j)} - a_{c(i,j)}, & \text{otherwise.} \end{cases} \tag{3}$$

and the leaving time of vehicle j from i^{th} customer $c(i,j)$ is

$$l_{c(i,j)} = a_{c(i,j)} + w_{c(i,j)} + s_{c(i,j)}. \tag{4}$$

Hence, the total travel time of the route r_j can be computed as follows

$$T_j = \sum_{i=0}^{N_j} \left(t_{c(i,j)c(i+1,j)} + w_{c(i+1,j)} + s_{c(i+1,j)} \right). \tag{5}$$

The $s_{c(i+1,j)}$ presents the service time of $(i+1)^{th}$ customer in the j^{th} route. Here, waiting time and service time of depot is zero, i.e., $w_{c(N_j+1,j)} = 0$ and $s_{c(N_j+1,j)} = 0$. This is due to the fact there is no service requirement at the depot.

Using above notations, the total waiting time of this route is

$$W_j = \sum_{i=1}^{N_j} w_{c(i,j)}. \tag{6}$$

There are two versions of VRP in the context of latest service time. The first version with *hard time windows* (Garcia-Najera & Bullinaria, 2011; Tan et al., 2006) does not allow the vehicle to arrive at a customer location after latest service time of that customer. This version of VRP is not very realistic as the travel time can vary a lot depending on external factors like road and traffic conditions which can be chaotic and unpredictable (Taş, Dellaert, Van Woensel, & De Kok, 2013; Hashimoto, Ibaraki, Imahori, & Yagiura, 2006). The second version with *soft time windows* provides a relaxation to latest service time and allows a small time window violation, since it can not be considered as a critical breach of service requirements in most practical applications. Literature contains so many practical applications of VRP with soft time windows, such as in Fu, Eglese, and Li (2008), Chiang and Russell (2004), Taillard, Badeau, Gendreau, Guertin, and Potvin (1997) and Müller (2010). It is observed that relaxing the requirements of hard time windows can yield lower cost solutions requiring fewer vehicles, shorter travel distance and less travel time. Furthermore, VRP with soft time windows can be considered as a generalization of VRP with hard time windows (Fu et al., 2008; Chiang & Russell, 2004). This paper addresses VRP with soft time windows, i.e., a vehicle is allowed to arrive late within a certain time after the latest service time, also known as maximum allowed time in the literature. The late arrival causes a delay time for the vehicle. Let *md* denotes the maximum allowed time that a vehicle is permitted to arrive after the end of time window.

The delay time of vehicle j at i^{th} customer of the route is

$$delay_{c(i,j)} = \begin{cases} 0 & \text{if } a_{c(i,j)} \leqslant e_{c(i,j)} \\ a_{c(i,j)} - e_{c(i,j)}, & \text{otherwise.} \end{cases} \tag{7}$$

Hence, the total delay time of this route is

$$Delay_j = \sum_{i=1}^{N_j} delay_{c(i,j)}. \tag{8}$$

It is pertinent to mention that, the early arrival of vehicle at a customer's location results in waiting time and late arrival of vehicle incurs the delay time, in the route covered by this vehicle.

The five objectives for *MOVRPTW*, each of which needs to be minimized, can be stated as follows:

f_1 (number of vehicles):

$$\text{Minimize } f_1 = |R| = M; \tag{9}$$

f_2 (total travel distance):

$$\text{Minimize } f_2 = \sum_{j=1}^{M} Dist_j; \tag{10}$$

f_3 (makespan, i.e., longest travel time among all routes (from/to depot)):

$$\text{Minimize } f_3 = \max\{T_j | j = 1, \ldots, M\}; \tag{11}$$

f_4 (total waiting time due to early arrivals):

$$\text{Minimize } f_4 = \sum_{j=1}^{M} W_j; \tag{12}$$

f_5 (total delay time due to late arrivals):

$$\text{Minimize } f_5 = \sum_{j=1}^{M} Delay_j; \tag{13}$$

The objectives f_1, f_2 and f_3 can be visualized as transportation costs. Objective f_3 is also concerned about balancing the drivers' workloads. On the other hand, f_4 and f_5 can be considered as service costs representing the customer satisfaction (Taş et al., 2013).

The constraints associated with *MOVRPTW* are stated below:

1. *Demand constraint:* The total demand of each route r_j must be less than or equal to the capacity of the vehicle, i.e.,

$$\sum_{i=1}^{N_j} g_{c(i,j)} \leqslant C \quad \forall j \in \{1, 2, \ldots, M\} \tag{14}$$

2. *Travel time constraint:* The delay time for all vehicles at any customer location must be less than or equal to the maximum allowed delay time, i.e.,

$$delay_{c(i,j)} \leqslant md \quad \forall i \in \{1, 2, \ldots, N\} \quad \forall j \in \{1, 2, \ldots, M\} \tag{15}$$

3. *Return time constraint:* All vehicles must return to depot before the closing time, i.e.,

$$a_{c(N_j+1,j)} \leqslant e_{c(N_j+1,j)} \quad \forall j \in \{1, 2, \ldots, M\} \tag{16}$$

Thus, we can summarize the *MOVRPTW* with five objectives addressed in this paper as:

$$\min f = \{f_1, f_2, f_3, f_4, f_5\} \tag{17}$$

subject to constraint conditions defined by 14–16.

3. Related work

In this section, we delve into the previously proposed approaches for addressing *MOVRPTW*. Furthermore, we provide a brief description about the basic concepts of multiobjective optimization problems (MOPs) for better understanding the terminology used in the proposed approach.

3.1. Existing approaches for MOVRPTW

As already mentioned in Section 1, *MOVRPTW* with five objectives was introduced by Castro-Gutierrez et al. (2011). At the same time, it was observed that no problem instance exists that model real-world characteristics. Most of the existing datasets for *MOVRPTW* either do not consider real-world characteristics or are derived from the classic single objective datasets. In Castro-Gutierrez et al. (2011), authors proposed a real-world problem instances derived from the data of a distribution company located in Tenerife, Spain. The *MOVRPTW* instances presented in Castro-Gutierrez et al. (2011) are considered as ideal benchmark instances for *MOVRPTW*. These datasets are having real-world characteristics, for example in real-world the travel distance and travel time are often distinct and asymmetric. One can easily notice the fact that the travel time in urban areas and in rural areas has a significant difference for the same distance. Similarly, travel time from location *A* to *B* need not be same as travel time taken in traveling from *B* to *A*. Furthermore, Castro-Gutierrez et al. (2011) considers the triangle inequality violations for both travel distance and travel time, i.e., $Dist_{ik} \geqslant Dist_{ij} + Dist_{jk}$, which are highly prevalent in real-world scenarios. Rodríguez and Ruiz (2012) and Fleming, Griffis, and Bell (2013) have studied the effects of asymmetry and triangle inequality violations on realistic VRP. The above mentioned points show that the real-world *MOVRPTW* instances presented in Castro-Gutierrez et al. (2011) truly reflect multiobjective nature, and hence, are more desirable to evaluate the performance of multiobjective optimization approaches.

Castro-Gutierrez et al. (2011) used *NSGA-II* approach on newly proposed *MOVRPTW* instances and on well-known Solomon's problem instances (Solomon, 1987). In *NSGA-II* approach, the population initialization uses a constructive method in which the customers farthest from depot are serviced first. It used three crossover operators, viz. one-point crossover, edge crossover and generic crossover. It used four kinds of mutation operators, viz. swap, insertion, inversion and displacement. The goal of using *NSGA-II* approach was to show that the newly proposed instances have truly multiobjective characteristics than commonly used Solomon's problem instances. The authors conducted a pair-wise comparison between all combinations of objectives. Castro-Gutierrez et al. (2011) concluded that the proposed *MOVRPTW* instances have better dependency relationships in pair-wise comparisons of the objectives, whereas dependence relationships between objectives are weak in Solomon's instances. Hence, the proposed *MOVRPTW* instances represent ideal benchmark scenarios for multiobjective VRPTW.

To address *MOVRPTW*, Zhou and Wang (2015) presented a new local search-based multiobjective algorithm (*LSMOVRPTW*). In *LSMOVRPTW* approach, one local search procedure is devised for each objective. Initially an archive of solutions is created in an iterative manner. Five local searches each corresponding to one objective are applied iteratively on a randomly generated solution. The local search for f_1 picks the route with fewest customers and try to insert these customers in some other routes where the constraints remain satisfied. The local search procedures for f_2, f_3, f_4 and f_5 use three neighborhood operators. A solution is selected randomly from the archive and all objective-wise local searches are applied on this solution in an iterative manner. Each objective-wise local search is used ten times in a continuous manner, by randomly selecting a neighborhood operator each time. The newly generated solutions are updated in the archive by using concept of $\in$-dominance (Laumanns et al., 2002; Deb et al., 2005).

The performance of *LSMOVRPTW* is compared with the *NSGA-II* approach presented in Castro-Gutierrez et al. (2011). Zhou and Wang (2015) re-executed this *NSGA-II* approach in their computing environment, as the source code of NSGA-II was available for download (Castro-Gutierrez et al., 2012). Since *LSMOVRPTW* uses a series of local searches and there is no explicit concept of generation in this approach, hence the running time of *NSGA-II* is used as stopping criteria for *LSMOVRPTW* approach. The comparison of results in Zhou and Wang (2015) reveals that the *LSMOVRPTW* approach outperformed the *NSGA-II* approach in almost all the instances.

The *NSGA-II* approach of Castro-Gutierrez et al. (2011) uses generic crossover and mutation operators to produce offspring. These generic

operators do not consider the structure of the problem and the characteristics of the objectives while generating offspring, and thus lack proper exploitation characteristics. This adversely affected the quality of final solutions obtained through this approach. The *LSMOVRPTW* presented in Zhou and Wang (2015) uses objective-wise local search. In *LSMOVRPTW* approach,for each objective, one local search procedure is devised by considering the problem structure of MOVRPTW with corresponding objective. Furthermore, local search approaches are considered as exploitative in nature and lacks the exploration characteristics. In spite of this drawback of *LSMOVRPTW* and the fact that NSGA-II is among the most successful multiobjective optimization techniques, *NSGA-II* approach of Castro-Gutierrez et al. (2011) performed much worse than *LSMOVRPTW*. This created the need for variation operators that can utilize the structure of MOVRPTW and characteristics of its various objectives in a suitable manner. Such variation operators facilitates better exploitation thereby aiding the genetic algorithm in maintaining an adequate balance between exploration and exploitation which is necessary for finding high quality solutions. To bridge this research gap, we have developed a new approach based on NSGA-II for MOVRPTW with desired variation operators. As our variation operators are designed considering the structure of MOVRPTW and characteristics of its various objectives, an exclusive crossover operator and an exclusive mutation operator are designed for each objective leading to a total of five different crossover operators and five different mutation operators. Our proposed NSGA-II approach is able to maintain an adequate balance between exploration and exploitation owing to these variation operators.

3.2. Overview of multiobjective optimization

A multiobjective optimization problem (MOP) can be modeled as:

$$\min F(x) = \{f_1(x), f_2(x), \ldots, f_m(x)\} x \in \Omega \tag{18}$$

where Ω denotes the solution space, x is the solution and $f_i(x)$ is the i^{th} objective function. The number of objectives is m, and in most of the cases these objectives conflict one another, i.e., improvement in value of one objective results in deterioration in values of some others objectives. A solution x dominates another solution y (represented as $x \prec y$), iff $f_i(x) \leqslant f_i(y) \forall i \in \{1, 2, \ldots, m\}$, and $f_i(x) < f_i(y) \exists i \in \{1, 2, \ldots, m\}$. In other words, x dominates y, iff no component of x is larger than the corresponding component of y and at least one component is smaller (Zitzler, Thiele, Laumanns, Fonseca, & Da Fonseca, 2003). A solution x^* is said to be *Pareto optimal* if it is not dominated by any other solution $x \in \Omega$. The set of all Pareto optimal solutions constitute the *Pareto set* and the set of their corresponding objective vectors is known as *Pareto front* (Chen & Chiang, 2014). Any multiobjective algorithm for an MOP intends to find a set of nondominated solutions which perform well in terms of convergence and diversity. That is, a multiobjective algorithm considered as a better approach, if it provides a set of nondominated solutions close to and widely distributed along the Pareto front. Literature contains numerous multiobjective evolutionary algorithms (MOEAs) to address various MOPs, such as Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler, Laumanns, & Thiele, 2002), Multiobjective Evolutionary Algorithm based on Decomposition(MOEA/D) (Zhang & Li, 2007), Multiobjective Particle Swarm Optimization (MOPSO) (Coello & Lechuga, 2002), Generalized Differential Evolution 3 (GDE3) (Kukkonen & Lampinen, 2005), Pareto Archived Evolution Strategy (PAES) (Knowles & Corne, 2000), NSGA II (Deb et al., 2002). So far the most popular MOEA in literature is NSGA-II (Coello, González Brambila, Figueroa Gamboa, Castillo Tapia, & Hernández Gómez, 2020). The reader is referred to Coello et al. (2020) and Zhou et al. (2011) for a comprehensive survey on various multiobjective evolutionary algorithms.

4. Proposed NSGA-II approach for *MOVRPTW*

This paper presents a nondominated sorting genetic algorithm II (NSGA-II) (Deb et al., 2002) approach with objective-specific variation operators to address the *MOVRPTW*. Hereafter, our proposed approach will be referred to as *INSGA-II*. Literature reveals that NSGA-II is most commonly used algorithm to address the multiobjective optimization problems due to its salient features which include fast nondomination sorting procedure, fast crowded distance estimation procedure and simple crowded comparison operator. NSGA-II was proposed by Deb et al. (2002) and the simulation results of NSGA-II on various benchmark instances show its superiority over other multiobjective optimization techniques (Yusoff, Ngadiman, & Zain, 2011).

Nondomination sorting procedure iteratively sorts the population into different nondomination levels. First each solution is compared with other solutions. The solutions not dominated by any other solutions in population are considered as nondominated solutions and they constitute the first nondominated front. There must be some solutions which are dominated by only the solutions in first nondominated front. Hence, if we temporarily remove the solutions in the first front, then we will find the solutions in second nondominated front. Thus the solutions in first front are discounted temporarily, and the above procedure is repeated to find the second nondominated front. The same process continues until all solutions in population are divided into different nondomination fronts. The above mentioned procedure has the overall complexity of $\mathcal{O}(MN^3)$, where M is the number of objectives and N is the size of the population. In NSGA-II (Deb et al., 2002), a fast nondomination sorting procedure is proposed which has the complexity of $\mathcal{O}(MN^2)$. The idea is to compute two entities for each solution ρ: (1) domination count n_ρ, i.e., the number of solutions which dominate the solution ρ and (2) S_ρ, a set of solutions dominated by the solution ρ. The computing of these two entities require $\mathcal{O}(MN^2)$ comparisons. The solutions with domination count as zero will constitute the first nondominated front. In order to find the second nondominated front, the solutions in first front are discounted temporarily. Now, for each solution ρ in first nondominated front, the corresponding domination count of each member of set S_ρ is reduced by one. The solutions which were only dominated by the solutions in first nondominated front, now must have their domination count as zero and these solutions will be selected as second nondominated front. The same process is repeated until all fronts are identified (Deb et al., 2002).

In the original NSGA (Srinivas & Deb, 1995), the sharing function approach was used to maintain diversity in the population. There are two difficulties in using sharing function. It requires a sharing parameter value set by user, and secondly, the complexity of sharing function is $\mathcal{O}(N^2)$ due to involvement of comparison of each solution with all others solutions in the population. To maintain diversity in the population, Deb et al. (2002) proposed the fast crowded distance estimation procedure in NSGA-II. The crowding distance provides the density of solutions surrounding a particular solution in a nondominated set. The solutions in a nondominated front are first sorted according to each objective function value in an ascending order of magnitude. Then, for each objective, the boundary solutions (solutions with smallest and largest values for that objective) are given an infinite distance value. All other intermediate solutions are assigned a distance value equal to the absolute normalized difference in the corresponding objective function values of two adjacent solutions. This computation is done for each objective. The overall crowding-distance value is the sum of individual distance values corresponding to each objective. The value of each objective is normalized before computing the crowding distance. The overall complexity of crowded distance estimation is $\mathcal{O}(MNlogN)$ (Deb et al., 2002).

In this work, we have used the NSGA-II framework with objective-wise crossover and mutation operators, which consider the specific requirements of each objective. The process begins with initial population generation. The probabilistic binary tournament selection (BTS) pro-

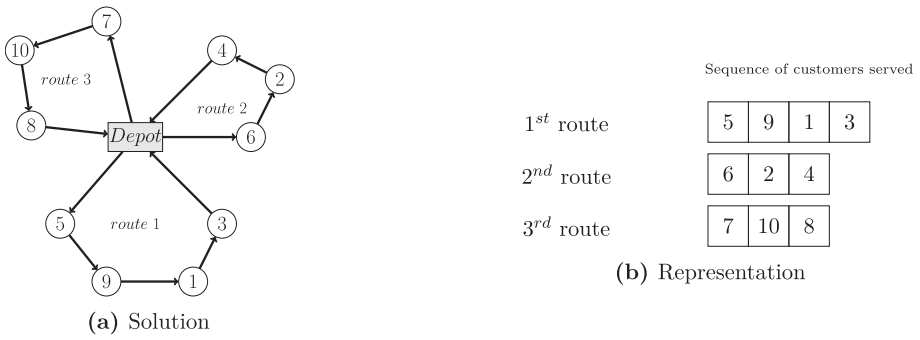

Fig. 1. Solution representation.

cedure has been used to select two parents for crossover. In this procedure, two solutions are selected randomly from the parent population. The better solution is selected to be a parent with probability p_{bt}, otherwise worse one is selected. Here a solution is considered better than the other if it dominates the other one, and in case if no one dominates each other, then the solution having more crowding distance is

Table 1
Average values of IGD, HV, And C-Metric of INSGA-II and LSMOVRPTW.

Instance	IGD		HV		C-metric		Time [a]
	LSMOVRPTW	INSGA-II	LSMOVRPTW	INSGA-II	C(LS,IN) [b]	C(IN,LS) [c]	
50-0-0	0.002849	**0.001374**	0.778693	**0.798471**	0.117421	**0.481708**	17.10
50-0-1	0.005578	**0.004443**	**0.460442**	0.434376	**0.290333**	0.029811	12.73
50-0-2	0.013812	**0.005579**	0.293905	**0.372857**	**0.231333**	0.041352	11.26
50-0-3	0.006679	**0.002525**	0.514643	**0.617789**	0.029333	**0.425595**	8.42
50-0-4	0.008824	**0.005600**	**0.943907**	0.939618	**0.306000**	0.215279	14.33
50-1-0	0.003507	**0.001724**	0.799644	**0.819721**	0.125619	**0.539394**	13.00
50-1-1	0.005708	**0.004427**	**0.460791**	0.433047	**0.288333**	0.030735	13.29
50-1-2	0.013848	**0.005465**	0.298750	**0.365217**	**0.223667**	0.034913	11.96
50-1-3	0.006677	**0.002525**	0.514647	**0.617789**	0.029333	**0.444048**	8.48
50-1-4	0.008454	**0.005473**	0.922378	0.923307	0.247000	0.225724	8.91
50-2-0	0.011147	**0.007107**	0.867581	**0.898339**	0.108445	**0.751587**	5.84
50-2-1	0.011408	**0.004776**	**0.734837**	0.701035	**0.306667**	0.055424	6.20
50-2-2	0.019621	**0.004737**	0.368060	**0.615539**	0.148000	0.186794	6.31
50-2-3	0.007721	**0.003965**	0.655798	**0.729146**	0.037667	**0.384127**	6.03
50-2-4	0.019475	**0.009309**	0.783465	**0.872617**	0.003667	**0.914849**	5.53
150-0-0	0.002900	**0.001820**	0.735135	**0.755004**	0.187478	**0.391211**	134.39
150-0-1	0.017247	**0.006331**	0.267536	**0.292883**	**0.419333**	0.003367	93.95
150-0-2	0.012206	**0.007195**	**0.277424**	0.255240	**0.467333**	0.004369	80.57
150-0-3	0.027004	**0.002912**	0.023876	**0.355080**	**0.169333**	0.114975	53.30
150-0-4	0.011028	**0.008906**	**0.469320**	0.397354	**0.653000**	0.012018	96.48
150-1-0	0.002921	**0.001830**	0.735225	**0.755004**	0.188144	**0.390476**	129.30
150-1-1	0.017019	**0.006302**	0.267405	**0.292916**	**0.419000**	0.003345	86.72
150-1-2	0.012152	**0.007187**	**0.277578**	0.255240	**0.467000**	0.004371	80.13
150-1-3	0.026895	**0.002906**	0.023876	**0.355080**	**0.169000**	0.112892	52.98
150-1-4	0.011262	**0.009006**	**0.467632**	0.397354	**0.651333**	0.011976	87.64
150-2-0	0.004479	**0.002366**	0.787328	**0.821469**	0.141612	**0.508714**	45.99
150-2-1	0.015057	**0.006741**	**0.440961**	0.374096	**0.527333**	0.000741	53.81
150-2-2	0.012309	**0.007315**	**0.271850**	0.246393	**0.484333**	0.001149	54.96
150-2-3	0.025342	**0.002870**	0.029056	**0.361563**	**0.156333**	0.098382	48.86
150-2-4	0.009941	**0.008416**	**0.597763**	0.493563	**0.683000**	0.009872	56.62
250-0-0	0.003846	**0.002675**	0.713256	**0.734973**	**0.353986**	0.142155	338.06
250-0-1	0.022698	**0.007819**	0.183124	**0.244897**	**0.489000**	0.001389	211.74
250-0-2	0.023812	**0.007688**	0.137427	**0.218211**	**0.385667**	0.001190	178.88
250-0-3	0.031870	**0.005041**	0.145107	**0.563146**	0.120333	0.096493	111.10
250-0-4	0.017896	**0.010545**	0.232987	**0.471353**	**0.486000**	0.009488	226.10
250-1-0	0.003866	**0.002687**	0.713344	**0.734973**	**0.354666**	0.140602	339.96
250-1-1	0.022437	**0.007742**	0.183535	**0.244658**	**0.490333**	0.001333	234.87
250-1-2	0.023050	**0.007706**	0.145502	**0.218211**	**0.405000**	0.001190	183.26
250-1-3	0.031775	**0.005035**	0.145077	**0.563146**	0.120000	0.096920	105.61
250-1-4	0.017722	**0.010528**	0.233035	**0.471431**	**0.485333**	0.009429	215.09
250-2-0	0.004514	**0.002327**	0.742472	**0.783369**	0.218530	0.284062	179.09
250-2-1	0.022466	**0.007084**	0.176404	**0.231610**	**0.471000**	0.000000	194.70
250-2-2	0.021257	**0.007408**	0.126384	**0.208450**	**0.342667**	0.000000	178.11
250-2-3	0.031753	**0.005106**	0.145161	**0.563510**	0.121667	0.095974	110.86
250-2-4	0.020832	**0.012327**	0.199133	**0.429170**	**0.470667**	0.008397	193.74
W/S/B	0/0/45		11/1/33		29/6/10		

[a] Seconds
[b] C(LSMOVRPTW,INSGA-II)
[c] C(INSGA-II,LSMOVRPTW)

considered as the better solution.

For each generation, we have used an iterative process which is repeated for 50 iterations. In each iteration, the BTS is used to select two parents from the parent population. The selected parents are passed as input to crossover operator, which produces a child solution. The mutation operator is applied on this child solution and a mutant solution is generated. The child solution and mutant are compared and the better one is added in offspring population. There are two possibilities of being a better solution. First, the solution which dominates other is considered better. The other possibility, if no one dominates each other, then solution which has less objective value for the specific crossover/mutation operator is considered better. It is pertinent to mention that the newly generated solution is added only if it is unique. To check uniqueness, we have compared the objective values of a solution from the existing solutions in both the populations, i.e,. parent population as well as offspring population. If all of the five objective values of a newly generated solution do not match with any existing solution's objectives in the population, then it is considered as unique solution. We have used objective-wise crossover followed by mutation, hence, five different crossovers and mutations are used in this framework and consequently each iteration may add up to five solutions in offspring population. At the end of every generation, the parent and offspring population are combined and a new parent population is created for next generation by following the concepts of nondominated sorting and crowding distance presented in Deb et al. (2002). In calculation of crowding distance, the boundary solutions are assigned a very large value, for each objective. In case, if two or more solutions have equal minimum objective value, the solution with minimum value of normalized sum for all objectives is consider as lowest boundary solution. For example, while calculating crowding distance for first objective, viz. number of vehicles, if more than one solution have same minimum first objective value, then the solution having minimum normalized sum value of all other objectives is selected as lowest boundary solution in this objective. By this method, it is guaranteed that the solution which has minimum value in all objectives will always be included in population, which is also desired.

The proposed approach is executed for N_g generations. Since mutation operator is having better exploitation characteristics in comparison to crossover, hence for the last 25% generations, we have used only

Table 2
Comparison of INSGA-II with LSMOVRPTW in terms of count of the runs as well as in overall 30 runs, on which INSGA-II achieved better (<), equal (=) and worse (>) values in five objectives.

Instance	f1			f2			f3			f4			f5		
	<	=	>	<	=	>	<	=	>	<	=	>	<	=	>
50-0-0	0	30	0	11	10	9	0	30	0	0	30	0	0	30	0
50-0-1	22	7	1	20	0	10	0	30	0	26	0	4	0	30	0
50-0-2	0	30	0	16	0	14	0	30	0	24	1	5	0	30	0
50-0-3	0	30	0	12	0	18	0	30	0	30	0	0	0	30	0
50-0-4	0	30	0	19	2	9	0	30	0	0	30	0	0	30	0
50-1-0	0	30	0	22	0	8	0	30	0	0	30	0	0	30	0
50-1-1	18	10	2	19	1	10	0	30	0	26	0	4	0	30	0
50-1-2	0	30	0	12	0	18	0	30	0	20	4	6	0	30	0
50-1-3	0	30	0	12	0	18	0	30	0	30	0	0	0	30	0
50-1-4	0	30	0	22	1	7	0	30	0	0	30	0	0	30	0
50-2-0	0	30	0	26	0	4	0	30	0	0	30	0	0	30	0
50-2-1	0	30	0	26	0	4	0	30	0	27	0	3	0	30	0
50-2-2	0	30	0	28	0	2	0	30	0	25	0	5	0	30	0
50-2-3	0	30	0	25	0	5	0	30	0	30	0	0	0	30	0
50-2-4	0	30	0	30	0	0	0	30	0	28	0	2	0	30	0
150-0-0	10	18	2	25	0	5	17	12	1	0	30	0	0	30	0
150-0-1	0	30	0	22	0	8	30	0	0	16	0	14	0	30	0
150-0-2	0	30	0	27	0	3	30	0	0	1	1	28	0	30	0
150-0-3	0	30	0	29	0	1	0	30	0	30	0	0	0	30	0
150-0-4	0	6	24	23	0	7	5	25	0	4	2	24	0	30	0
150-1-0	9	19	2	25	0	5	17	12	1	0	30	0	0	30	0
150-1-1	0	30	0	22	0	8	30	0	0	16	0	14	0	30	0
150-1-2	0	30	0	27	0	3	30	0	0	1	1	28	0	30	0
150-1-3	0	30	0	29	0	1	0	30	0	30	0	0	0	30	0
150-1-4	0	6	24	23	0	7	5	25	0	4	2	24	0	30	0
150-2-0	0	30	0	17	0	13	20	10	0	0	30	0	0	30	0
150-2-1	0	30	0	25	0	5	30	0	0	6	0	24	0	30	0
150-2-2	0	30	0	28	0	2	30	0	0	0	0	30	0	30	0
150-2-3	0	30	0	28	0	2	0	30	0	30	0	0	0	30	0
150-2-4	0	30	0	25	0	5	1	29	0	4	1	25	0	30	0
250-0-0	0	30	0	8	0	22	24	5	1	0	30	0	0	30	0
250-0-1	3	22	5	23	0	7	30	0	0	18	1	11	0	30	0
250-0-2	0	30	0	28	0	2	30	0	0	0	0	30	0	30	0
250-0-3	0	30	0	30	0	0	0	30	0	30	0	0	0	30	0
250-0-4	0	3	27	26	0	4	27	3	0	3	0	27	0	30	0
250-1-0	0	30	0	8	0	22	24	5	1	0	30	0	0	30	0
250-1-1	3	22	5	23	0	7	30	0	0	18	1	11	0	30	0
250-1-2	0	30	0	28	0	2	30	0	0	0	0	30	0	30	0
250-1-3	0	30	0	30	0	0	0	30	0	30	0	0	0	30	0
250-1-4	0	3	27	26	0	4	27	3	0	3	0	27	0	30	0
250-2-0	0	30	0	19	0	11	26	3	1	0	30	0	0	30	0
250-2-1	1	24	5	25	0	5	30	0	0	13	1	16	0	30	0
250-2-2	1	29	0	30	0	0	30	0	0	0	0	30	0	30	0
250-2-3	0	30	0	30	0	0	0	30	0	30	0	0	0	30	0
250-2-4	0	3	27	18	0	12	30	0	0	2	1	27	0	30	0
Total	67	1132	151	1027	14	309	583	762	5	555	346	449	0	1350	0
Best value in all 30 runs	1	44	0	36	1	8	13	32	0	21	14	10	0	45	0

mutation operator. The mutation operator at this fag end phase acts as a local search to some extent and improves the solution quality further in the population. In this phase, the parent which has better objective value for the particular mutation operator is selected and copied as child solution and the mutation is applied on this child solution. The following subsection provides the detailed description of different components used in proposed INSGA-II approach. In case, if both parents are same then also we have applied only mutation operator, since the use of crossover is futile under this scenario. Algorithm 1 presents the pseudo code of proposed *INSGA-II* approach.

4.1. Solution representation for MOVRPTW

We have used the similar solution representation as presented in Zhou and Wang (2015). Here, a solution contains several routes and each route is a linear permutation of customers, i.e., if j^{th} route contains $|r_j|$ customers, then route r_j is presented as permutation of $|r_j|$ customers such that the positions of the customers in this route specify the order in which customers are served. It is assumed that by default every route must begin and end with vertex *0*, which represents the depot, and hence, depot is not explicitly included in our solution representation. Fig. 1 illustrates solution representation, by assuming an example consisting of $N = 10$, i.e., 10 customers and $M = 3$ routes, viz. $R = \{r_1, r_2, r_3\}$. Note that 0 is not explicitly included in any route in our representation as shown in Fig. 1(b), but every route starts and ends at 0. Hence, 0 is made part of a route whenever we process that route to compute various values associated with that route. For example, the first route r_1 is $\langle 0, c(1,1), c(2,1), c(3,1), c(4,1), 0\rangle$, though we represented it as $\langle c(1,1), c(2,1), c(3,1), c(4,1)\rangle$. Here, $c(1,1) = 5$, $c(2,1) = 9$, $c(3,1) = 1$ and $c(4,1) = 3$, i.e., customer 5 is the first customer served in this route, customer 9 is the second customer served, and so on. Likewise, r_2 and r_3 have 3 customers each, and, their composition can be explained analogously.

4.2. Initial population generation

To generate initial population, we have used an iterative approach similar to initialization procedure presented in Zhou and Wang (2015). The process begins with creation of a route by selecting a customer randomly. Again the next customer is randomly selected and inserted into next position in the current route. If insertion fails due to constraints violation, then this customer is iteratively tried in previous existing routes, starting from the first route. If a feasible place is found in any route then this customer is inserted there, otherwise a new route is created. This process continues until all customers is inserted into some routes. The generated solution is checked for uniqueness and added in population if it is unique. Using this approach, we have created a population of size p as initial population.

4.3. Objective-specific crossover operators

We have developed greedy variation operators which are not only problem specific, but also objective-specific, i.e., our crossover and mutation operators are designed as per the characteristics of *MOVRPTW* and the characteristics of each objective. Hence, we have a dedicated crossover operator and a dedicated mutation operator for each objective, thereby leading to five different crossover operators and five different mutation operators. The crossover operators used in our approach are inspired from the crossover operators used in Singh and Baghel (2009) and Singh and Gupta (2007). These crossover operators create a child solution in two phases.

The first phase starts with an empty solution and iteratively constructs the child solution. In each iteration, it chooses one of the parent randomly and finds the most promising feasible route in this parent. The most promising route is copied to child solution and number of routes in child is incremented by one. This route is deleted from the parent from which it was taken. In case of other parent, the customers belonging to the most promising route is deleted from their respective routes, by connecting the predecessor of each such customer to its successor and updating the various quantities associated the affected routes accordingly. The process is repeated k times. It is pertinent to mention that, first phase need not always produce a partial solution with k routes, since it is also possible that after few iterations, some or all of the updated routes may become infeasible, and hence, in this case none of the route from the selected parent satisfy the most promising feasible route criteria. In this case, a child solution with less than k routes is produced.

Clearly, after first phase, some customers remain unassigned and need to be inserted in the child solution. The second phase inserts the unassigned customers into best position in an iterative manner, and, a complete child solution is created. During each iteration, it selects an unassigned customer randomly and inserts it at the best position which is defined as per the objective under consideration. To find the best position, the crossover operator begins from the first route and try to insert at every position between the customers in the route. Similarly other routes are also considered one-by-one. If a position is found which satisfy all the constraints, then this position is considered as a feasible position and then all feasible positions are evaluated to find the best position. If the selected unassigned customer cannot be inserted into any existing route due to the violation of any constraint, then a new route is created by inserting the selected unassigned customer at first position.

Since we have used objective-wise crossover operator, and hence, the definition of most promising route, the definition of best position and the value of k vary as per the objective and are specified below for each objective.

Let MIN_R and MAX_R are the minimum and maximum value of the number of routes between both parents. For example, let 8 and 10 are the number of routes in parent 1 and parent 2 respectively. Then, $MIN_R = 8$ and $MAX_R = 10$, for the assumed example. The value of k depends on the MIN_R and MAX_R.

f_1: The most promising route is the route which has maximum number of customers. Since f_1 seeks a solution which has minimum number of vehicles (minimum number of routes), hence selecting a route with more number of customers may create a child solution with fewer routes in comparison to both the parents. Here $k = MIN_R - 1$, due to reason that we may find a child solution which has less number of routes in comparison to both the parents. Here the best position is the first feasible position which satisfy all the constraints.

f_2: Here, the route which has smallest ratio of the distance traveled to the number of customers in the route, is considered as most promising route. If in a route more number of customers are covered by traveling less distance, then selecting such routes may provide a better solution as per objective f_2. This is the idea behind selecting the smallest ratio of the distance traveled to the number of customers in the route. Since we already have a feasible solution (parent 1 or parent 2) with number of routes equal to MIN_R, hence k is assigned here MIN_R. The best position for f_2 is the position which yields least increment in the total traveled distance.

f_3: The objective f_3 is to minimize the longest travel time of a route, and hence, the most promising route is considered as the route which has the minimum travel time. Here, $k = MAX_R + (\lfloor u_{01} \times 10 \rfloor)$, where u_{01} is a uniform random number between [0,1). The initial solution generation procedure provides a solution in which all customers are tried to be accommodated in some existing route instead of creating a new route, and hence, in general, most of the routes have large travel times. Here the value of k is set to a larger value than the number of routes in both parents because the crossover operator used here intends to provide more opportunity to both parents so that a route with less travel time is selected in child solution. Also a large value of k may create more number of routes in child solution, and, it is evident that if a solution has more number of routes then the travel time of each route will be reduced. The best position in a route is considered as a position which leads to least increment in travel time of the route. In case, if there are more than one best position then the route which has less travel time is

selected for insertion. It is also possible the least increment travel time position in a route may result in highest makespan, i.e., inserting the customer at this position may makes the concerned route to have longest travel time in comparison to inserting it at respective best positions in all other routes. Hence, in such scenario the second least increment position is considered. If this also result in same highest makespan then third least increment position is considered and so on. If inserting in any route lead to same makespan then instead of creating a new route, we inserted the unassigned customer to the first route considered, so that f_1 should not get deteriorated at the same time.

Objectives f_4 and f_5 are sharing similarity with objective f_2, since all f_2,f_4 and f_5 seek to minimize a overall quantity in a solution. The quantity is total traveled distance, total waiting time and total delay time for f_2,f_4 and f_5 respectively. Hence the crossover operators for f_4 and f_5 are similar to crossover operator of f_2.

f_4: The most promising route here is the route which has smallest ratio of the waiting time to the number of customers in the route and $k = MIN_R$. The best position is the position which yields least increment in the total waiting time.

f_5: The most promising route here is the route which has smallest ratio of the delay time to the number of customers in the route and $k = MIN_R$. The best position is the position which yields least increment in the total delay time.

Algorithm 1. Pseudo-code of *INSGA-II for MOVRPTW*

```
Input: p is the population size
Output: Set of nondominated solutions (Pareto set)
Population ← Initial population generation(p);
Nondominated Sort(population);
Crowding distance(population);
// Crowding distance is calculated as per the solutions in the particular front
N ← 0;          // N is the generation count & N_g is the total number of generations
while (N < N_g) do
    if (N is less than 75% of N_g) then
        flag ← 1;
    else
        flag ← 0;
    // flag variable is used to perform only mutation for last 25% generations
    // P is the current population of size p
    p_x ← p+1;
    // p_x is the place where newly generated offspring solution is added in population
    for (i := 1 to 50) do
        Par_1 ← BTS(); // Binary tournament selection
        Par_2 ← BTS();
        // Select parents Par_1 & Par_2 using binary tournament selection
        for (obj := 1 to 5) do
            if (flag is equal to 1) then
                if (Par_1 and Par_2 are not same) then
                    C ← Crossover_obj(Par_1, Par_2);
                else
                    C ← Par_1;
                M ← Mutation_obj(C);
            else
                C ← Better_obj(Par_1, Par_2);
                // The parent which is better in particular objective
                M ← Mutation_obj(C);
            if (M ≺ C)    // M dominates C then
                X ← M;
            else if (C ≺ M) then
                X ← C;
            else
                // Solutions C and M do not dominate each other
                if (f_obj(M) < f_obj(C)) then
                    // If Solution M is better than C in the objective obj
                    X ← M;
                else
                    X ← C;
            if (X is unique in population) then
                Add X in population at p_x;
                p_x ← p_x + 1;
    P ← Select p solutions from the union of parent (p) and offspring (p_x) population using nondomination sorting and
    crowding distance;
    N ← N + 1;
return (set of nondominated solutions);
```

4.4. Objective-specific mutation operators

Our mutation operators are based on destruction and reconstruction strategy, where the solution is partially destroyed and then again reconstructed by using an objective-specific approach which is an appropriate mix of greediness and randomness. These mutation operators use Select_route procedure to select one or more routes and create a pool of unassigned customers by removing some customers randomly, from the selected route(s). There are two versions of Select_route procedure. The mutation operators for f_1 and f_3 use only first version of Select_route procedure, whereas mutation operators for f_2,f_4 and f_5 use both versions of Select_route procedure, in a mutually exclusive manner. In first version of Select_route procedure, a route is selected as per the requirement of objective and some or all customers from this route are

removed and considered as unassigned customers. If after removal of customers, the selected route become infeasible, then the route is made feasible by breaking the route into two or more feasible routes. The second version of Select_route procedure works in an iterative manner and used only in mutation operators for f_2, f_4 and f_5. In each iteration, one route is picked randomly and a random customer is selected in this route. If the removal of the selected customer does not result in an infeasible route then this customer is removed from this route and added in pool of unassigned customer.

The complete solution is created in iterative manner. In each iteration, a customer is selected randomly, from the pool of unassigned customers and inserted at best position in some existing route. If no best position is found due to violation of constraints, then a new route is created by inserting the selected unassigned customer at first position in this newly created route.

Like in case of our crossover operators, here also the definition of Select_route and best position are different for various objectives and their descriptions are as follows:

f_1: The Select_route finds a route with least number of customers. If more than one route is having least number of customers then a route is randomly selected from among them. All of the customers in the selected route is considered as unassigned customers. In each iteration, a customer is randomly selected from this pool of unassigned customers and inserted at the best position. Here, the best position is the first feasible position which do not delay the starting time of other customers. The starting time of a customer means the vehicle arrival time to the customer before insertion. If an unassigned customer is inserted at a position which do not delay the starting time of other customers, then such insertion will create possibilities to insert more customers in the route, as the chances of the violation of second and third constraints (15 and 16) will be less. If no such position is found then the unassigned customer will be inserted at last feasible position. If no feasible position found then the mutation operator simply discard the solution and the mutant will be same as child solution generated by the crossover, since mutation operator fails to decrease the number of routes. Please note that travel times do not follow triangle inequality, and hence, insertion of a new customer may decrease the starting time of subsequent customers. If mutation operator successfully reduces the number of routes by one then next route with least number of customers is again tried for merging with other routes. The mutation operator is stopped when a customer cannot be inserted into any other route.

f_2: The mutation operator for f_2 has used two types of Select_route procedures. The first one which is used with probability 0.7, selects a route randomly and then removes random number of customers from this selected route. For example, if the selected route has 10 customers then a random number of customers between [1, 10] can be selected and a pool of unassigned customers is created using these removed customers. The second type of Select_route procedure is used with remaining probability 0.3 and iteratively removes a random customer from a randomly selected route. It iterates for ν times. Here, the best position is same as used in crossover operator used for objective f_2 i.e, the position which has least increment in the total distance traveled.

f_3: The Select_route finds a route with maximum travel time. If there exist more than one route corresponding to maximum travel time, then a route is randomly selected from among them. A random number $\in [1, \nu]$ of customers is removed randomly from this selected route and a pool of unassigned customers is created. If ν is greater than the number of customers in this route then ν is reset to the number of customers in this route. Since travel times do not satisfy the triangle inequality, so removal of customers from the selected route may result a route with more travel time than the previous maximum travel time. In this case the maximum travel time is considered as the new travel time value. Here, the best position is the place where the increment in travel time is least after insertion. Like crossover operator here also, if there are more than one best position then the route which has less travel time is selected for insertion. We consider only those positions for insertion which can yield a travel time of route strictly less than the previous maximum travel time.

Here also the mutation operators for f_4 and f_5 work in a similar manner as mutation operator for f_2, due to the reasons mentioned in Section 4.3.

f_4: Two types of Select_route procedure is used with different probabilities. First one which is used with probability 0.6, selects a route randomly and then remove random number of customers from this selected route. The second Select_route procedure is used with remaining probability 0.4 and iteratively removes a random customer from a randomly selected route. The number of iterations is ν. The best position is same as used in the crossover operator used for objective f_4, i.e., the position which has least increment in the total waiting time.

f_5: Two types of Select_route procedure is used with different probabilities. First one which is used with probability 0.5, selects a route randomly and then remove random number of customers from this selected route. The second Select_route procedure is used with remaining probability 0.5 and iteratively removes a random customer from a

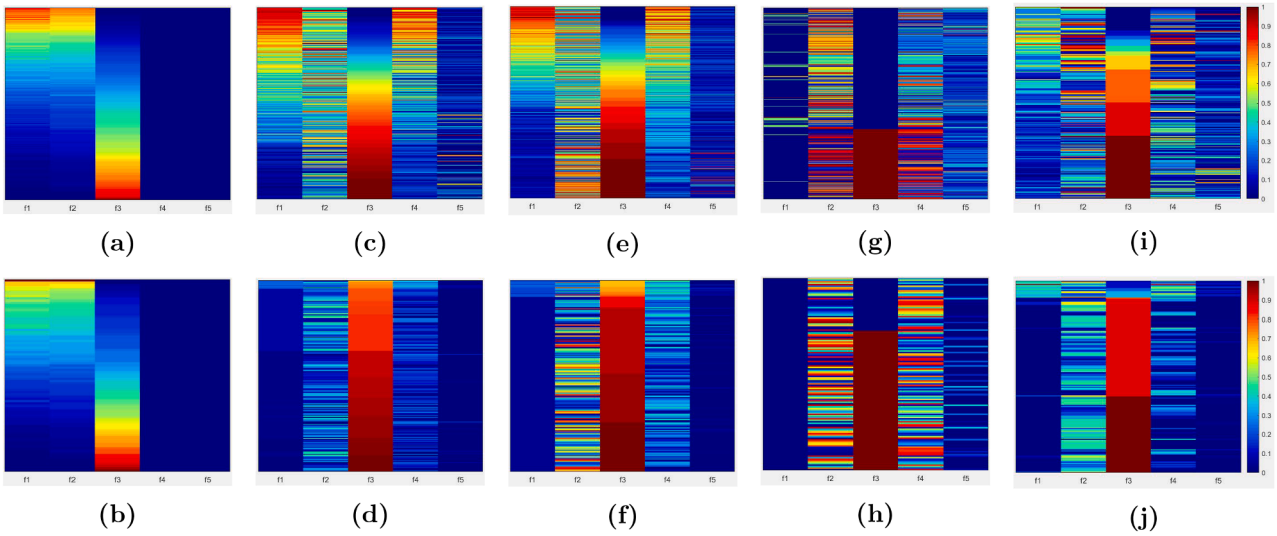

Fig. 2. Heatmaps of nondominated solutions obtained by INSGA-II and LSMOVRPTW on selected instances (The rows of each heatmap have been rearranged according to f_3 in ascending order): (a) INSGA-II on 250-2-0, (b) LSMOVRPTW on 250-2-0, (c) INSGA-II on 250-2-1, (d) LSMOVRPTW on 250-2-1, (e) INSGA-II on 250-2-2, (f) LSMOVRPTW on 250-2-2, (g) INSGA-II on 250-2-3, (h) LSMOVRPTW on 250-2-3, (i) INSGA-II on 250-2-4, (j) LSMOVRPTW on 250-2-4.

randomly selected route. The number of iterations is ν. The best position is same as used in crossover operator used for objective f_5, i.e., the position which has least increment in the total delay time.

5. Computational results

We have implemented the *INSGA-II* approach in C language. A Linux based 2.50 GHz Intel Core i7 processor based system with 8 GB of RAM is used to perform all computational experiments. To evaluate the performance of our approach, we have used the same test datasets as used in Zhou and Wang (2015) and Castro-Gutierrez et al. (2011). Since Solomon's instances do not represent the ideal benchmark scenarios for multiobjective VRPTW. Hence, we have not tested our approach on Solomon's dataset. In order to assess the relative performance of our proposed approach with the state-of-the-art approach viz. LSMOVRPTW (Zhou & Wang, 2015), we have implemented the *LSMOVRPTW* approach in C language and executed this approach under the same computing environment as our approach.

The proposed *INSGA-II* approach is executed for 2000 generations i.e., $N_g = 2000$. The population size p is set to 100. The parameter p_{bt} in binary tournament selection is set as 0.7, and the value of ν in mutation operators is set to 4. The proposed approach has been executed 30 independent times on each instance like the LSMOVRPTW approach of Zhou and Wang (2015). All the parameter values are determined empirically after executing the proposed approach a number of times. These values fetch better results on most of the instances, but they cannot regarded as optimal parameter values for all the instances.

5.1. Description of MOVRPTW instances

The real-world *MOVRPTW* dataset (Castro-Gutierrez et al., 2011) consists of a total of 45 instances, generated by considering combinations of three different sizes of customers, viz. 50, 150 & 250, five different time windows profiles, viz. 1, 2, 3, 4 & 5, and three kinds of vehicles capacities, each corresponds to a particular value of the parameter δ. The three values of δ are 60, 20 and 5. The depot has 8 h working time. The five time window profiles represent the availability of customers, and, are based on the experience of the delivery company. These five profiles are described as follows:

1. All the customers are available for all day, i.e., for entire 8 h (480 min).
2. Three kinds of customers are considered, viz. early customers, mid-day customers and late customers. The total time 480 min is divided into three equal part, viz. 160 min time slots for each kind of customer. The early customers will be served in [0, 160) minutes time slot, mid-day customers will be served in [160, 320) minutes time slot, and late customers are served in [320, 480] minutes time slot.
3. The length of each time window is decreased by 30 min. Thus the time slot for early customers is [0, 130], for mid-day customers is [175, 305], and for late customers is [350, 480].
4. The length of each time window is further decreased by 30 min. Thus the time slot for early customers is [0,100], for mid-day customers is [190,290] and for late customers is [380,480].
5. Here, a customer can belong to any of the aforementioned time windows profiles.

In order to serve each customer, the minimum capacity of vehicle ($\underline{D}$) must be equal to the maximum demands among the customers. Evidently, the maximum capacity of vehicle ($\overline{D}$) should not be greater than the sum of demands of all the customers. Hence, the capacity of each vehicle C must lie in between $[\underline{D},\overline{D}]$. The capacity of each vehicle is set as $C = \underline{D} + \delta \times \frac{\left(\overline{D}-\underline{D}\right)}{100}$, where $\delta \in \{60, 20, 5\}$ as mentioned already. Hence, the capacity of vehicle is directly proportional to δ. Each customer i has a demand g_i and requires service time s_i. The demand and service time both are assigned from the set $\{10, 20, 30\}$ with a probability of $\frac{1}{3}$. A maximum delay of 30 min is allowed for each customer after the end of their time window, i.e., $m_d = 30$ min (Zhou & Wang, 2015; Castro-Gutierrez et al., 2011).

5.2. Performance metrics

In multiobjective optimization, the performance of an algorithm is evaluated in terms of both convergence and diversity. In Zitzler et al. (2003) various performance metrics (quality indicators) are analyzed. It is concluded that no single performance metric is able to provide a comprehensive measure on the performance of a multiobjective optimization algorithm, and thus, it is better to use a combination of quality indicators. In this paper, we have used three metrics, viz. inverted generational distance (IGD), hypervolume (HV), coverage metric (C-metric). The first two performance metrics are among most commonly used performance metrics for multiobjective optimization problems (Riquelme, Lücken, & Barán, 2015), whereas the last one was chosen because it is used in Zhou and Wang (2015) to evaluate the relative performance of *LSMOVRPTW*. These three metrics are described below:

1. Inverted generational distance (IGD): IGD (Zhang & Li, 2007) provides a measure to both convergence and diversity of the nondominated solutions obtained by an algorithm in MOP. Let P^* be a set of uniformly distributed solutions in the Pareto front (PF), and X is the set of obtained nondominated solutions. The minimum Euclidean distance between a solution v and the solutions in X is denoted by $d(v, X)$. The IGD approximation of X with respect to P^* is defined as,

$$IGD\left(P^*, X\right) = \frac{\sum_{v \in P^*} d\left(v, X\right)}{|P^*|} \tag{19}$$

   Smaller IGD value represents the closeness of set X towards the Pareto front, and hence, considered as a better set of solutions in terms of convergence and diversity.
2. Hypervolume (HV): HV metric is proposed by Zitzler et al. (2002). It represents the n-dimensional volume in the objective space that is contained by nondominated solutions with respect to a reference point. A set with higher HV value represents a better set of nondominated solutions approximating the Pareto front, from the point of view of the convergence and diversity. To calculate hypervolume, we have used the HV approach presented in Fonseca, Paquete, and López-Ibánez (2006), and its associated software tool available in Fonseca, López-Ibánez, Paquete, and Guerreiro (2006). The point (1.01, 1.01, 1.01, 1.01, 1.01) is used as reference point to calculate the hypervolume.
3. Coverage metric (C-metric): C-metric (Zitzler & Thiele, 1999) is used to compare two sets of nondominated solutions obtained by two different approaches. Let A and B are two sets of nondominated solutions, then $C(A, B)$ represent C-metric of A with respect to B, and defined as the percentage of the solutions in B that are Pareto dominated by at least one solution in A. $C(A, B) = 1$ means that all nondominated solutions in B are Pareto dominated by solutions in set A. It is important to mention that the sum of $C(A, B)$ and $C(B, A)$ need not be equal to 1, as some solutions in A and B may not Pareto dominate one another.

As the true Pareto front can not be determined, the P^* is approximated by considering all unique nondominated solutions obtained by *LSMOVRPTW* and *INSGA-II* in all 30 runs. Since the range of objectives are different, we have normalized all objective values to calculate HV and IGD.

5.3. Experimental results

The performance of *INSGA-II* approach is compared with *LSMOVRPTW* approach (Zhou & Wang, 2015). Table 1 presents the comparison of results in terms of *IGD*, *HV*, and *C-metric*. In this table, the first column represents the name of the instances in the form "$num_1 - num_2 - num_3$", where num_1 shows the number of customers, num_2 shows the index of δ type with respect to capacity of vehicle, and num_3 indicates the index of time window profile. The second, third, and fourth columns represent the average values over 30 runs of the *IGD*, *HV* and *C-metric* respectively. The values for each metric are reported for *LSMOVRPTW* and *INSGA-II* approaches both. The last column represents the average execution time (in seconds) over 30 runs taken by *INSGA-II* approach. Since *LSMOVRPTW* uses a series of local searches and there is no explicit concept of generation in this approach, we have used the running time of *INSGA-II* as stopping criteria for *LSMOVRPTW* approach for a fair comparison. Similar termination criteria was used in Zhou and Wang (2015) to compare *LSMOVRPTW* with *NSGA-II* (Castro-Gutierrez et al., 2011). The performance of *NSGA-II* presented in Castro-Gutierrez et al. (2011) was found to be much worse than *LSMOVRPTW* (Zhou & Wang, 2015), and hence, we have not included *NSGA-II* (Castro-Gutierrez et al., 2011) in this comparison.

Due to the limitation of space, standard deviation of the metrics are not presented in the table. Furthermore, we have used Wilcoxon signed-rank test (Wilcoxon, 1945; Derrac, García, Molina, & Herrera, 2011) at 5% significance level, to show the statistically significant difference between the results obtained by our approach *INSGA-II* and *LSMOVRPTW*. An online calculator[1] is used to perform Wilcoxon signed-rank test. The significantly better values obtained by both the approaches are represented in bold font for ease of identification. The last row of table provides the overall comparison in the format W/S/B, which indicates that the performance of *INSGA-II* is worse than, similar to and better than the *LSMOVRPTW* in W, S and B instances.

The *INSGA-II* approach significantly outperforms *LSMOVRPTW* approach for all 45 instances in terms of *IGD*. In terms of *HV*, *INSGA-II* significantly outperformed the *LSMOVRPTW* in 33 instances and significantly outperformed by *LSMOVRPTW* in 11 instances. This shows that *INSGA-II* performs better than *LSMOVRPTW* in terms of both convergence and diversity. C-metric is calculated as the ratio of number of solutions in latter approach that are Pareto dominated by at least one solution in former approach. The *LSMOVRPTW* approach is a local search based approach in which a series of local searches is performed on solutions present in archive, and hence, the chance of getting some highly converged solutions is high. While the *INSGA-II* is a population based approach without any local search procedure, and, use of objective-specific crossover & mutation operators can provide a better set of nondominated solutions in terms of both convergence and diversity. In terms of C-metric, the *INSGA-II* significantly outperformed the *LSMOVRPTW* in 10 instances and significantly outperformed by *LSMOVRPTW* in 29 instances. This can be attributed to the local search based approach in *LSMOVRPTW* which provides some highly converged solutions, thereby enabling better performance of *LSMOVRPTW* in terms of C-metric.

In the instances with first time window profile presented as $num_3 = 0$ (such as "250-2-0"), the customers are always available, and hence, the objectives f_4 and f_5 are always zero and the problem in this particular time window profile has only three significant objectives, viz. f_1, f_2 and f_3. Evidently, the hardness of the problem increases with increase in number of customers and decrease in capacity of vehicles. This is due to the fact that the problem with more number of customers and with vehicle of smaller capacity will have solution with more number of routes, and hence, to converge on all objectives will be tougher. The instances with fourth time window profile presented as $num_3 = 3$ (such as "250-2-3"), the length of time window which presents the availability of customers is smallest. In this case, no customer is available for a total of 178 min (two durations of 89 min each) out of total 480 min (all day duration). Thus the objective f_4, viz. total waiting time of vehicles due to early arrivals at customer place is highest in comparison to all other time window profiles, and hence, the objective f_4 is more significant in these instances. Clearly, all five objectives are equally significant in the instances with fourth time window profile ($num_3 = 3$).

The values of HV in Table 1 reveals that the performance of *INSGA-II* is always better than the performance of *LSMOVRPTW*, in first time window profile ($num_3 = 0$) and with 250 customers ($num_1 = 250$), in all instances. Also the difference in HV values of *INSGA-II* and *LSMOVRPTW* are marginally more in all instances (particularly with more number of customers) with fourth time window profile presented as $num_3 = 3$. This shows that the performance of *INSGA-II* is better than *LSMOVRPTW* especially on harder instances.

It is pertinent to mention that any solution which has minimum value in any one objective is a nondominated solution irrespective of the values of other objectives. In a real-world scenario, the preference for one objective might be more in comparison to other objectives in MOPs. For example, in *MOVRPTW* with five objectives, the objective f_2, viz. total travel distance might be more important in some perspective than other objectives as distance traveled is directly proportional to consumption of fuel, and hence, it has direct impact on environmental pollution as well as important from monetary considerations. Also for some industrial organizations the objective f_5, i.e., total delay time may be more important than other objectives as delay in service adversely impacts the customers' goodwill, thereby reducing the volume of business. Hence, finding minimum values in all the objectives by an approach is crucial in *MOVRPTW*, since in most of the cases the decision maker's preference is not known a priori. An analysis in this regard is performed in Table 2.

There are 45 instances in total and both the approaches have been executed for 30 independent times on each instance. Hence, we have 30 sets of nondominated solutions for each instance and a total 1350 sets of nondominated solutions, generated by each approach. For a particular instance, in each run the five minimum values of each objective obtained by *INSGA-II* and *LSMOVRPTW* are compared. The comparison results are presented in Table 2. In this table, each row except the last two represents the summarized comparison over 30 runs for an instance, in terms of number of runs where the minimum value of each objective found by *INSGA-II* is better ($<$), equal ($=$) or worse ($>$) than the minimum value of corresponding objective found by *LSMOVRPTW*. Here, the first column represents the name of the instances in the form "$num_1 - num_2 - num_3$". The second, third, fourth, fifth and sixth columns provides the comparison of five objective values respectively, in all 30 runs. For example, second row (instance "50-0-1") and second column (objective f_1) shows that *INSGA-II* is able to find 22 times better, 7 times equal and 1 time worse values for objective f_1 in 30 runs in comparison to *LSMOVRPTW*. The second last row provides the distribution of 1350 comparisons in better, equal and worse categories. It shows that the minimum objective values obtained by *INSGA-II* is superior than *LSMOVRPTW*, for objectives f_2, f_3 and f_4, and, equal in objective f_5, but slightly inferior in objective f_1. The last row provides the summary of comparison in terms of number of instances between two approaches for each objective by taking the minimum value of each objective over 30 runs of an approach. The *INSGA-II* is able to find better values on 36 instances, equal value on 1 instance, and worse values on 8 instances for objectives f_2, as compared to *LSMOVRPTW*. Similarly, *INSGA-II* performs better for objectives f_3 and f_4, and, equal performance in objectives f_1 & f_5 (except for one instance for f_1 where it performed better). Hence, last two rows of Table 2 clearly show that the performance of *INSGA-II* is better than *LSMOVRPTW* in individual runs as well as in best of all 30 runs.

As already mentioned in Section 3.2, one multiobjective algorithm is considered to be better than another, if the set of nondominated

[1] https://mathcracker.com/wilcoxon-signed-ranks.php.

solutions found by the former algorithm is better than latter in terms of convergence and diversity both. Visual representation is often used to compare the solution sets obtained by different approaches in terms of convergence and diversity as it provides ease of understanding. To visually represent the convergence and diversity of the nondominated solutions obtained by *LSMOVRPTW* and *INSGA-II*, we have used heatmap visualization technique (Walker, Everson, & Fieldsend, 2013; Pryke, Mostaghim, & Nazemi, 2007). Heatmap provides the visual representation of solution sets to many-objective problems and facilitates observing of trade-off among various objectives in a clear manner. A heatmap shows the data as a grid of pixels whose colors represent values on a scale from maximal (hot) to minimal (cold) (Walker et al., 2013). In heatmap representation, each row is a solution and each column represents an objective. The color of the cells represents the value of an objective for a particular solution. The cooler colors shows the convergence of solution and the distribution of full range of colors shows the diversity of solution. The nondominated solutions obtained by an approach can be considered as good solutions, if the heatmap visualization of the solutions shows the cooler colors as well as distribution of full range of colors.

In order to use heatmaps for visual representation, all objectives must be on the same scale (Walker et al., 2013), hence, we have normalized all objective values to the range of [0, 1]. Each heatmap is a visual representation of nondominated solutions obtained by an approach on a particular instance in all 30 runs. Each objective value is shown in particular color which is in the range of cold (blue) to hot (red).

As explained in Section 5.3, the instances with more number of customers and smallest capacity vehicles are harder than others. Therefore, we have selected instance with $num_1 = 250$ (i.e, 250 customers) and $num_2 = 2$ (i.e, third category of vehicle which has least capacity in comparison to other categories) to present the visual comparison of the nondominated solutions obtained by both the approaches. Furthermore, it is observed that the performance of an approach on an instance depends on its time window profile, and hence, all five time window profiles instances with $num_1 = 250$ and $num_2 = 2$ are selected for comparison of these two approaches.

Fig. 2 presents the convergence and diversity of the nondominated solutions obtained by both the approaches using heatmaps on the five instances, viz. 250-2-0, 250-2-1, 250-2-2, 250-2-3 and 250-2-4. The first row and second row of heatmaps represent the nondominated solutions obtained by *INSGA-II* and *LSMOVRPTW* respectively on the selected instances. The rows of each heatmap have been rearranged according to f_3 in ascending order. The heatmaps of *INSGA-II* show cooler colors than the heatmaps of *LSMOVRPTW* particularly for objective f_3, on all five instances. Hence *INSGA-II* has better convergence in objective f_3 than *LSMOVRPTW* in all five instances.

The instance "250-2-0" has only three significant objectives, viz. f_1, f_2 and f_3, as explained in Section 5.3. The Fig. 2 (a) and (b) represents the heatmaps of both the approaches on instance "250-2-0", which clearly show that the *INSGA-II* has cooler colors than *LSMOVRPTW* in objectives f_1 and f_2, and similar color in objective f_3. Hence, it generates better solutions in terms of convergence and diversity. This is also supported by the values of HV and IGD metrics on instance "250-2-0" in Table 1.

In instance "250-2-3", the objective f_4, viz. total waiting time of vehicles due to early arrivals at customer place is highest in comparison to all other time window profiles, and hence, the objective f_4 is more significant in these instances, as explained in Section 5.3. The Fig. 2 (g) and (h) represents the heatmaps of both the approaches on instance "250-2-3", which clearly show that the *INSGA-II* has cooler colors than *LSMOVRPTW* in objective f_4, and hence, able to find more converged solutions with respect to objective f_4.

The instances with fifth time window profile (presented as $num_3 = 4$ in Section 5.3, such as "250-2-4") have all variety of customers from different time window profiles, and hence, they represent more realistic scenarios. The Fig. 2 (i) and (j) are the heatmaps of both the approaches on instance "250-2-4", which reveals that the nondominated solutions generated by *INSGA-II* distribute more dispersedly with in full range of color than those generated by *LSMOVRPTW*. Hence, the solutions generated by *INSGA-II* have better diversity.

From all pair of comparisons, it can be inferred that the *LSMOVRPTW* always lacks diverse solutions and it is more apparent in objectives f_1 and f_5. Also heatmaps of *INSGA-II* show the better convergence in terms of objective f_3 than *LSMOVRPTW*. Evidently, the objectives conflict each other, and hence, an approach is better if the nondominated solutions yielded by an approach has better diversity, i.e., if the nondominated solutions have a wider spread in full range of color for all the objectives. Thus, the heatmaps visualization reveal that *INSGA-II* performs better than *LSMOVRPTW* from the perspective of convergence and diversity.

6. Conclusions

In this paper, a nondominated sorting genetic algorithm II (INSGA-II) based approach with objective-specific variation operators is presented to address the *MOVRPTW*. This problem is a variant of multiobjective vehicle routing problem (MOVRP). *MOVRPTW* has more relevance in real-world applications and consists of five conflicting objectives. In the proposed *INSGA-II* approach, the crossover and mutation operators are designed by exploiting the problem specific knowledge as well as the characteristics of each objective. The experimental results show that the set of nondominated solutions obtained by the proposed approach is better than *LSMOVRPTW* approach in terms of both convergence and diversity, thereby clearly demonstrating the effectiveness of the proposed *INSGA-II* approach in comparison to the state-of-the-art approach. Performance of our approach, particularly in terms of C-metric, can be improved further at the expense of increased execution times if the objective-specific local searches like those in Zhou and Wang (2015) are incorporated in our approach.

The performance of the proposed approach provides inspiration for developing similar approaches for other variants of multiobjective vehicle routing problem. In particular, the proposed approach can be applied to similar problems presented in Wang, Weng, and Zhang (2019) and Wang et al. (2016). Similar multiobjective evolutionary algorithm approaches utilizing objective-specific greedy variation operators can be designed for a wide range of MOPs.

CRediT authorship contribution statement

Gaurav Srivastava: Methodology, Software, Formal analysis, Investigation, Writing - original draft, Visualization. **Alok Singh:** Methodology, Validation, Writing - review & editing, Supervision, Resources. **Rammohan Mallipeddi:** Conceptualization, Validation, Formal analysis, Writing - review & editing, Resources, Project administration.

Declaration of Competing Interest

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

Acknowledgement

Authors would like to thank two anonymous reviewers for their valuable comments and suggestions that have helped in improving the quality of this manuscript. The first author acknowledges the Senior Research Fellowship received from the Council of Scientific & Industrial Research, Government of India. This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology under the Grant NRF-2018R1A1A1A05079524.

References

Baldacci, R., Mingozzi, A., & Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research, 218*, 1–6.

Benjamin, A., & Beasley, J. (2010). Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities. *Computers & Operations Research, 37*, 2270–2280.

Bent, R., & Van Hentenryck, P. (2004). A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science, 38*, 515–530.

Bräysy, O., & Gendreau, M. (2005). Vehicle routing problem with time windows, part I: Route construction and local search algorithms. *Transportation Science, 39*, 104–118.

Bräysy, O., Hasle, G., & Dullaert, W. (2004). A multi-start local search algorithm for the vehicle routing problem with time windows. *European Journal of Operational Research, 159*, 586–605.

Buhrkal, K., Larsen, A., & Ropke, S. (2012). The waste collection vehicle routing problem with time windows in a city logistics context. *Procedia – Social and Behavioral Sciences, 39*, 241–254.

Castro-Gutierrez, J. (2012). Multi-objective tools for the vehicle routing problem with time windows. Ph.D. thesis University of Nottingham.

Castro-Gutierrez, J., Landa-Silva, D., & Pérez, J. M. (2011). Nature of real-world multi-objective vehicle routing with evolutionary algorithms. In 2011 IEEE International Conference on Systems, Man, and Cybernetics (pp. 257–264). IEEE.

Castro-Gutierrez, J., Landa-Silva, D., & Pérez, J. M. (2012). MOVRPTW dataset. Available online: https://github.com/psxjpc/.

Chen, H.-K., Chou, H.-W., Hsueh, C.-F., & Ho, T.-Y. (2011). The linehaul-feeder vehicle routing problem with virtual depots. *IEEE Transactions on Automation Science and Engineering, 8*, 694–704.

Chen, S.-W., & Chiang, T.-C. (2014). Evolutionary many-objective optimization by MO-NSGA-II with enhanced mating selection. In 2014 IEEE Congress on Evolutionary Computation (CEC) (pp. 1397–1404). IEEE.

Chiang, W.-C., & Russell, R. A. (2004). A metaheuristic for the vehicle-routeing problem with soft time windows. *Journal of the Operational Research Society, 55*, 1298–1310.

Coello, C. C., & Lechuga, M. S. (2002). MOPSO: A proposal for multiple objective particle swarm optimization. In Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (pp. 1051–1056). Vol. 2. IEEE.

Coello, C. A., González Brambila, S., Figueroa Gamboa, J., Castillo Tapia, M. G., & Hernández Gómez, R. (2020). Evolutionary multiobjective optimization: Open research areas and some challenges lying ahead. *Complex & Intelligent Systems, 6*, 221–236.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science, 6*, 80–91.

Deb, K., Mohan, M., & Mishra, S. (2005). Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. *Evolutionary Computation, 13*, 501–525.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*, 182–197.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*, 3–18.

Eksioglu, B., Vural, A. V., & Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering, 57*, 1472–1483.

Fachini, R. F., & Armentano, V. A. (2020). Logic-based benders decomposition for the heterogeneous fixed fleet vehicle routing problem with time windows. *Computers & Industrial Engineering, 148*, Article 106641.

Fleming, C. L., Griffis, S. E., & Bell, J. E. (2013). The effects of triangle inequality on the vehicle routing problem. *European Journal of Operational Research, 224*, 1–7.

Fonseca, C. M., López-Ibánez, M., Paquete, L., & Guerreiro, A. P. (2006). Computation of the hypervolume indicator. Available online: http://lopez-ibanez.eu/hypervolume.

Fonseca, C. M., Paquete, L., & López-Ibánez, M. (2006). An improved dimension-sweep algorithm for the hypervolume indicator. In 2006 IEEE international conference on evolutionary computation (pp. 1157–1163). IEEE.

Fu, Z., Eglese, R., & Li, L. Y. (2008). A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society, 59*, 663–673.

Garcia-Najera, A., & Bullinaria, J. A. (2011). An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows. *Computers & Operations Research, 38*, 287–300.

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co.

Ghoseiri, K., & Ghannadpour, S. F. (2010). Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Applied Soft Computing, 10*, 1096–1107.

Gmira, M., Gendreau, M., Lodi, A., & Potvin, J.-Y. (2021). Tabu search for the time-dependent vehicle routing problem with time windows on a road network. *European Journal of Operational Research, 288*, 129–140.

Gong, Y., Zhang, J., Liu, O., Huang, R., Chung, H. S., & Shi, Y. (2012). Optimizing the vehicle routing problem with time windows: A discrete particle swarm optimization approach. *IEEE Transactions on Systems, Man, and Cybernetics Part C (Applications and Reviews), 42*, 254–267.

Hashimoto, H., Ibaraki, T., Imahori, S., & Yagiura, M. (2006). The vehicle routing problem with flexible time windows and traveling times. *Discrete Applied Mathematics, 154*, 2271–2290.

Hoff, A., Andersson, H., Christiansen, M., Hasle, G., & Løkketangen, A. (2010). Industrial aspects and literature survey: Fleet composition and routing. *Computers & Operations Research, 37*, 2041–2061.

Hsu, W.-H., & Chiang, T.-C. (2012). A multiobjective evolutionary algorithm with enhanced reproduction operators for the vehicle routing problem with time windows. In *2012 IEEE Congress on Evolutionary Computation* (pp. 1–8). IEEE.

Ip, W. H., Wang, D., & Cho, V. (2013). Aircraft ground service scheduling problems and their genetic algorithm with hybrid assignment and sequence encoding scheme. *IEEE Systems Journal, 7*, 649–657.

Jozefowiez, N., Semet, F., & Talbi, E.-G. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research, 189*, 293–309.

Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation, 8*, 149–172.

Kukkonen, S., & Lampinen, J. (2005). GDE3: The third evolution step of generalized differential evolution. In 2005 IEEE Congress on Evolutionary Computation (pp. 443–450). Vol. 1.

Laumanns, M., Thiele, L., Deb, K., & Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation, 10*, 263–282.

Lenstra, J. K., & Kan, A. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks, 11*, 221–227.

Lim, A., & Zhang, X. (2007). A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. *INFORMS Journal on Computing, 19*, 443–457.

Molina, J. C., Salmeron, J. L., & Eguia, I. (2020). An acs-based memetic algorithm for the heterogeneous vehicle routing problem with time windows. *Expert Systems with Applications, 157*, Article 113379.

Müller, J. (2010). Approximative solutions to the bicriterion vehicle routing problem with time windows. *European Journal of Operational Research, 202*, 223–231.

Oliveira, H. C. B., & Vasconcelos, G. C. (2010). A hybrid search method for the vehicle routing problem with time windows. *Annals of Operations Research, 180*, 125–144.

Ombuki, B., Ross, B. J., & Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence, 24*, 17–30.

Osaba, E., Yang, X.-S., & Del Ser, J. (2020). Is the vehicle routing problem dead? An overview through bioinspired perspective and a prospect of opportunities. In Nature-Inspired Computation in Navigation and Routing Problems (pp. 57–84). Springer.

Prescott-Gagnon, E., Desaulniers, G., Drexl, M., & Rousseau, L.-M. (2010). European driver rules in vehicle routing with time windows. *Transportation Science, 44*, 455–473.

Pryke, A., Mostaghim, S., & Nazemi, A. (2007). Heatmap visualization of population based multi-objective algorithms. In Evolutionary Multi-Criterian Optimization, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds. (pp. 361–375). Springer-Verlag volume 4403.

Repoussis, P. P., Tarantilis, C. D., & Ioannou, G. (2009). Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *IEEE Transactions on Evolutionary Computation, 13*, 624–647.

Riquelme, N., Lücken, C. V., & Barán, B. (2015). Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)* (pp. 1–11). IEEE.

Rodríguez, A., & Ruiz, R. (2012). A study on the effect of the asymmetry on real capacitated vehicle routing problems. *Computers & Operations Research, 39*, 2142–2151.

Singh, A., & Baghel, A. S. (2009). A new grouping genetic algorithm approach to the multiple traveling salesperson problem. *Soft Computing, 13*, 95–101.

Singh, A., & Gupta, A. K. (2007). Two heuristics for the one-dimensional bin-packing problem. *OR Spectrum, 29*, 765–781.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research, 35*, 254–265.

Srinivas, N., & Deb, K. (1995). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*, 221–248.

Taillard, É., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science, 31*, 170–186.

Tan, K. C., Chew, Y. H., & Lee, L. (2006). A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows. *Computational Optimization and Applications, 34*, 115–151.

Taş, D., Dellaert, N., Van Woensel, T., & De Kok, T. (2013). Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research, 40*, 214–224.

Ursani, Z., Essam, D., Cornforth, D., & Stocker, R. (2011). Localized genetic algorithm for vehicle routing problem with time windows. *Applied Soft Computing, 11*, 5375–5390.

Walker, D. J., Everson, R. M., & Fieldsend, J. E. (2013). Visualizing mutually nondominating solution sets in many-objective optimization. *IEEE Transactions on Evolutionary Computation, 17*, 165–184.

Wang, J., Weng, T., & Zhang, Q. (2019). A two-stage multiobjective evolutionary algorithm for multiobjective multidepot vehicle routing problem with time windows. *IEEE Transactions on Cybernetics, 49*, 2467–2478.

Wang, J., Zhou, Y., Wang, Y., Zhang, J., Chen, C. P., & Zheng, Z. (2016). Multiobjective vehicle routing problems with simultaneous delivery and pickup and time windows: Formulation, instances, and algorithms. *IEEE Transactions on Cybernetics, 46*, 582–594.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics, 1*, 80–83.

Yusoff, Y., Ngadiman, M. S., & Zain, A. M. (2011). Overview of NSGA-II for optimizing machining process parameters. *Procedia Engineering, 15*, 3978–3983.

Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation, 11*, 712–731.

Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., & Zhang, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation, 1*, 32–49.

Zhou, Y., & Wang, J. (2015). A local search-based multiobjective optimization algorithm for multiobjective vehicle routing problem with time windows. *IEEE Systems Journal, 9*, 1100–1113.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the strength pareto evolutionary algorithm. In *Evolutionary Methods for Design, Optimization and Control* (pp. 95–100). CIMNE: Barcelona, Spain.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation, 3*, 257–271.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Da Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation, 7*, 117–132.