Expert Systems With Applications 138 (2019) 112823

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Green vehicle routing problem with queues

G. Poonthalir*, R. Nadarajan

Department of Applied Mathematics and Computational Sciences, PSG College of Technology, Peelamedu, Coimbatore, India

ARTICLE INFO

Article history:
Received 5 December 2018
Revised 14 July 2019
Accepted 17 July 2019
Available online 18 July 2019

Keywords:
Green vehicle routing problem
Chemical reaction optimization
Meta heuristic

ABSTRACT

Organizations spend considerable amount of time and money in distributing goods from production centres to customers who are geographically distributed. When vehicles are engaged to make long trips, along with serving the customers it may halt at facility centres like toll booths, warehouses, refuelling stations etc., for some service. The cost and time associated with these halts will have an impact on the overall cost of the route. But, when organizations plan a route with minimum cost, the costs associated with halts in the facility centres are generally not accounted. When these facility centres are limited in number, vehicles should wait for their turn to get the service which has an impact on the overall cost of the tour. This paper intends to address the issues that are associated with the waiting time at the refuelling station. Green Vehicle Routing Problem with Queues (GVRP-Q) is introduced in this paper. GVRP-Q aims to serve a set of customers, but includes halts in between for refuelling in the refuelling stations. Each refuelling station is modelled as M/M/1 queue model, where vehicle is allowed to wait in the queue. The impact of this wait time on the overall cost of the route is studied in this paper. GVRP-Q is solved using an enhanced Chemical Reaction Optimization (e-CRO) where the performance of CRO is enhanced by a bacterial transformation. Experiments are conducted on bench mark data sets and the results predict the influence of the wait time on the overall route cost.

1. Introduction

Transportation plays a major role in distributing finished goods from production centres to various customers. Logistic companies and distribution centres spend a considerable amount of cost and time to distribute the finished goods to customers. Cost reductions in this distribution are highly beneficial for any production house to realize profit and considerable time savings. Hence an expert and intelligent system is required for designing cost effective routes which addresses the needs of the organization.

Vehicle Routing Problem (VRP) is a NP hard combinatorial optimization problem. There are many variants in defining and delivering the problem. There exists a class of traditional variants of VRP like the Capacitated VRP (CVRP), VRP with Pickup and Delivery (VRPPD), VRP with Time window (VRPTW), Multi Depot Vehicle Routing Problem (MDVRP) etc. Many exact and heuristic solutions are proposed in the literature for solving these problems. An elaborate review is found in Laporte (1992) and also in Toth and Vigo (2002). Apart from these classical variants of VRP, a huge collection of problems emerged that address real world issues related to VRP from a different perspective like the emergency routing problem (He, Zhang, Song, Wen & Wu, 2009), disaster management and routing (Özdamar, Ekinci & Küçükyazici, 2004), routing with environmental constraints like green vehicle routing problem (Erdogan & Miller Hooks, 2012), pollution routing problem (Bektas & Laporte, 2011) etc., which addresses the more practical variants of VRP with real time objectives and constraints. The introduction of new objectives and constraints makes the problem difficult and challenging to solve and requires a more sophisticated design to model and devise routes.

Though many variants of VRP are discussed in literature, little attention is paid to design a VRP with halts at the facility centres. Hence it is required to build an intelligent system that addresses the issue with an elegant approach. While organization/distribution centre is planning a distribution, a hint on the approximate waiting time at the facility centres can be suggested. This paper addresses VRP that halts en route for refuelling apart from serving the customers.

When the facility centres are limited in number, vehicles queue up for the service and many factors like delay in the service, wait time of vehicles to get the service, non availability of the resources in the facility centre will have an impact on the total route cost.

To discuss further and to validate the model with experiments, the problem Green Vehicle Routing Problem (G-VRP) proposed by Erdogan and Miller Hooks (2012) is discussed. The objective of G-VRP is to devise low cost route for a set of geographically

* Corresponding author.
E-mail addresses: poonthalirk@gmail.com (G. Poonthalir), rn@amc.psgtech.ac.in (R. Nadarajan).

https://doi.org/10.1016/j.eswa.2019.112823
0957-4174/

distributed customers with a fleet of homogenous vehicles stationed at a depot. When the vehicle runs out of fuel, they are directed to nearby refuelling station that has artificial fuel for refuelling. All vehicles should reach the depot within the maximum time limit after serving the customers.

A variant of G-VRP is modelled in this paper. When the number of refuelling stations is limited, vehicles should wait for their turn to get the service and a new model Green VRP with Queues is formulated (GVRP-Q) where the refuelling stations are modelled with M/M/1 queuing system. This study is the first to model the facility centre as a queuing system and to enumerate the influence of the queue on the total route cost.

This paper proposes to solve GVRP-Q using Chemical Reaction Optimization (CRO). CRO is a recent meta heuristic introduced by Lam and Li (2010) by simulating the behaviour of chemical molecules. CRO is a varied population based meta heuristic. It is initialized with a set of molecules. It portrays the interaction of molecules in high energy unstable states in a chemical reaction that strives to attain a lower energy stable state. CRO has good exploration and exploitation capabilities. The elementary reactions of CRO are both simple and effective that can be customized for the problem. It is successfully applied to solve various discrete optimization problems.

2. Related work

A review of the works that includes VRP with halts en route is discussed here. An inventory routing problem with satellite facilities is introduced by Bard, Huang, Dror and Jaillet (1998) where vehicles can be reloaded with capacity at facility centres. The problem was solved using randomized Clarke-Wright procedure, GRASP and modified sweep algorithm. Periodic Vehicle Routing Problem (PVRP) with intermediate facility (Angelleli & Speranza, 2002) was studied as an extension of PVRP where vehicle can renew their capacity at intermediate facility centres any number of times. The problem is to collect the goods from facility centres, serve the customers and return to depot when it is finished with the work shift and is modelled as a collection problem and is solved using tabu search. An extension of multi depot VRP where vehicles replenish at intermediate depots was discussed by Crevier, Cordeau and Laporte (2007). VRP designed with flexible start and end location that allow vehicles to start and end the route at any depot which includes intermediate halting station for reloading the vehicle's capacity was proposed by Kek, Cheu and Meng (2008). Tarantilis, Zachariadis and Kiranoudis (2008) discussed VRP with intermediate facility to renew the capacity. Intermediate facilities for waste collection vehicles are addressed in Liu and He (2012), Markov, Varone and Bierlaire (2015) and Mofid-Nakhaee and Barzinpour (2019), etc. A rich vehicle routing problem (Prescott-Gagnon, Desaulniers & Rousseau, 2014) for heating oil distribution was discussed with multiple depots and intra route replenishment centres.

Electric vehicle routing problem (EVRP) with intermediate charging stations for charging vehicles en route was addressed in Bruglieri, Pezzella, Pisacane and Suraci (2015) where they solved the problem using variable neighbourhood search branching meta heuristic. EVRP with partial recharging strategies at intermediate charging stations was studied by Keskin and Catay (2016). EVRP with charging stations are discussed by Desaulniers, Errico, Irnich and Schneider (2016), Hiermann, Puchinger, Ropke and Hartl (2016), Zhang, Gajpal, Appadoo and Abdulkader (2018), Koç et al. (2019) etc.

In literature, problems that are modelled with an emphasis to reduce carbon emissions are also discussed under green vehicle routing problem. But, this section describes the developments in G-VRP proposed by Erdogan and Miller Hooks(2012). Green vehicle routing problem with multiple technology and partial recharges is studied by Felipe, Ortuño, Righini and Tirado (2014) as an extension of G-VRP. They developed a local search heuristic within simulated annealing framework to solve the problem. An electric vehicle routing problem with recharging stations was introduced by Schneider, Stenger and Goeke (2014). A mathematical model for Electric Vehicle Routing Problem with recharging stations (E-VRPTW) was designed as a Mixed Integer Programming (MIP) problem and was solved using tabu search combined with neighbourhood search algorithm. The algorithm is tested with the problem instances of G-VRP and with their own benchmark test instances. This work was extended as VRP with intermediate halts (Schneider, Stenger & Hof, 2015). The problem was solved using Adaptive variable neighbourhood heuristic and tabu search. G-VRP with limited refuelling halts is discussed by Poonthalir, Nadarajan and Geetha (2015). The problem is modelled as a bi objective optimization problem using ε constraint method with objectives as minimum route cost and minimum number of halts. It is solved using particle swarm optimization with greedy mutation operator.

A Heuristic Based Exact Algorithm (HBEA) for G-VRP was studied by Koç and Karaoglan (2016). They extended the work of G-VRP and gave an alternate formulation for the problem. They have adopted various valid in equalities from the literature and solved the problem using branch and cut. Their solution method is similar to Bard et al. (1998) for solving VRP with satellite facilities. An exact algorithm for green VRP is proposed by Andelmin and Bartolini (2017) where G-VRP is modelled as a set partitioning problem. They strengthened the problem by adding valid in equalities where the columns represented the feasible routes. Abdoli, MirHassani and Hooshmand (2019) proposed a LP relaxation bound on G-VRP where the bound is strengthened to be able to run with less time using commercial solvers. But, their model requires a pre processing step where the paths between every vertex should be enumerated before processing. Capacitated G-VRP was proposed by Normasari, Yu and Bachtiyar (2019) where they extended G-VRP with capacity constraint. The problem is solved using Simulated Annealing and a comparative analysis is presented. A detailed survey on G-VRP can be found in Lin, Choy, Ho, Chung and Lam (2014) that include the recent advancements in green routing and future research directions. A comparative study of different models of G-VRP is presented in Table 1.

The contribution of this paper is to introduce GVRP-Q with M/M/1 queue model at the refuelling station, to propose an enhanced chemical reaction optimization with bacterial transformation to solve GVRP-Q and to study the impact of the wait time in the queue which in turn affects the route cost.

3. Problem formulation

This section describes a general frame work for calculating the time spent by any vehicle in a facility centre followed by the formulation of the problem where the facility centres are replaced by refuelling stations.

3.1. Problem definition

Green Vehicle Routing Problem with Queue (GVRP-Q) is defined on an undirected complete graph $G=(V, E)$. It has vertices V and edges E. The vertex set is defined with $C=\{C_1,C_2,...C_N\}$ customers, Facility centres are defined as $S=\{S_1,S_2,...S_K\}, K < N$ and a depot V_0. The vertex set V is defined as $V=C\cup S\cup V_0$. Each customer is visited only once by one vehicle only, but a facility centre can be visited any number of times. Let S' represents several visits to each facility centre and hence $V'=C\cup S'\cup V_0$. Each vehicle visits a set of customers and return to the depot within a time limit T_{limit}. Let t_i be the arrival time at any vertex i, where $i \in V'$. Each edge is an arc

Table 1
Comparative study of the previous work in G-VRP.

Authors	Problem	Category	Solution methodology	Wait time at refuelling station	Population based heuristic
Erdogan and Miller Hooks(2012)	G-VRP	G-VRP	MCWS and DBCA	No	No
Felipe et al., 2014)	G-VRP with Partial recharges	G-VRP	Local search heuristic and simulated annealing	No	No
Schneider et al. (2014)	Electric Vehicle Routing Problem with Time Windows (E-VRPTW)	G-VRP	tabu search with neighbourhood search algorithm	No	No
Schneider et al., 2015	VRP with intermediate halts	G-VRP	Adaptive variable neighbourhood heuristic and tabu search	No	No
Poonthalir et al. (2015)	VRP with limited refuelling halts	G-VRP	Particle Swarm Optimization with Greedy Mutation Operator	No	Yes
Koç and Karaoglan (2016)	G-VRP	G-VRP	Heuristic Based Exact Algorithm (HBEA)	No	No
Andelmin and Bartolini (2017)	G-VRP	G-VRP	Exact Algorithm	No	No
Mancini (2017)	Hybrid VRP (both fuel and battery)	G-VRP	Neighbourhood search based heuristic	No	No
Normasari et al. (2019)	Capacitated G-VRP	G-VRP	Simulated Annealing	No	No
This paper	G-VRP	G-VRP	enhanced Chemical Reaction Optimization	Yes	Yes

connecting any two vertices in the vertex set V. Let c_{ij} be the cost or distance between i and j, τ_{ij} is the time taken to travel from i to j where $i, j \in V'$.

Let L be the total number of vehicles stationed at the depot. Let p_i be the time taken by the vehicle at facility centre or to serve a customer. Let W_i be the time taken to wait for service in station i. Each vehicle waits for its turn to get service where the server in the facility centre is modelled as $M/M/1$ queue model. Each vehicle's arrival is according to Poisson distribution λ and has exponential distribution with rate $\frac{1}{\mu}$ as the mean service time. The service discipline follows *FIFO* (First In First Out) and the customer who arrives first is the one to be served the first. The average waiting time of the customer in the queue is determined by Eq. (1),

$$W_i = \frac{\lambda}{\mu(\mu - \lambda)} \tag{1}$$

The total time spent by the vehicle in the facility centre is given in Eq. (2)

$$time\_spent_i = \mu + W_i \quad i \in S' \tag{2}$$

The time taken by a vehicle to reach a vertex j is given by

$$t_i + (\tau_{ij} + t_\sigma)x_{ijk} - T_{limit}(1 - x_{ijk}) \leq t_j \ \forall i \in V', i \neq j, k \in L$$

where $t_\sigma = \begin{cases} (W_i + \frac{1}{\mu}) & if\ i \in S' \\ p_i & if\ i \in C \end{cases}$

3.2. Mathematical formulation

The mathematical formulation of the problem is defined as below with the objective function as,

$$\min \sum_{i,j \in V' \mathrm{k} \in \mathrm{Li} \neq \mathrm{j}} c_{ij} x_{ijk} \tag{3}$$

where $x_{ijk} = \begin{cases} 1, & if \text{ vehicle } k \text{ travels from i to } j, \quad \mathrm{i,j} \in \mathrm{V}' \\ 0, & \text{otherwise} \end{cases}$

Subject to the following constraints,

$$\sum_{j \in V' \setminus \{V_0\}} x_{V_0 jk} \leq \mathrm{L} \tag{4}$$

$$\sum_{j \in V' \setminus \{V_0\}} x_{jV_0 k} \leq \mathrm{L} \tag{5}$$

$$\sum_{i,h \in V', h \neq i} x_{ihk} - \sum_{j \in V', h \neq i} x_{hjk} = 0, \ \forall \mathrm{k} \in L \tag{6}$$

$$\sum_{i \in C, i \neq j} x_{ijk} = 1, \ \forall j \in \mathrm{V}', \mathrm{k} \in L \tag{7}$$

$$\sum_{j \in V' \setminus S, i \neq j} x_{ijk} = 1 \ \forall i \in S \mathrm{k} \in L \tag{8}$$

$$0 \leq t_i \leq T_{limit} \mathrm{i} \in V' \tag{9}$$

$$t_{V_0} = p_{V_0} = 0 \tag{10}$$

$$0 \leq t_j \leq T_{limit} - (\tau_{jV_0} + p_j) \ \forall j \in V' \setminus \{V_0\} \tag{11}$$

$$t_i + (\tau_{ij} + t_\sigma)x_{ijk} - T(1 - x_{ijk}) \leq t_j \ \forall i \in V' \setminus \{V_0\}, i \neq j, k \in K \tag{12}$$

where

$$t_\sigma = \begin{cases} W_i + \frac{1}{\mu} & if\ i \in S' \\ p_i & if\ i \in C \end{cases} \tag{13}$$

Eq. (3) is the objective function that aims to minimize the overall cost of the tour. Eqs. (4) and (5) are used to check the total number of vehicles that enters and leaves the depot which should be less than the total number of available vehicles. Eq. (6) is used to ensure the flow constraint that the same vehicle enters and leaves a vertex in a tour. From a customer vertex, any vertex can be visited is given in Eq. (7). From a facility centre, either a depot or customer can be visited is given in Eq. (8). The total time limit of each vehicle is specified as constraint in Eq. (9). The initial arrival and service time are specified as a constraint in Eq. (10). Eq. (11) ensures that the vehicle returns to the depot within the maximum allowable time limit. Eq. (12) is used to check the time feasibility to serve another vertex in the tour.

To further elaborate the problem for a specific case, the facility centres are replaced by refuelling stations with the constraints as defined in Erdogan and Miller Hooks (2012) G-VRP. To define the problem, it is assumed that the vehicle has a constant fuel capacity R and drops off at a constant fuel rate g. ω_i is the remaining fuel level at vertex i, $i \in V'$. When the fuel level reaches the minimum and is not able to serve the next customer, it reaches the nearest refuelling station (facility centre) F or depot V_0 for refuelling and continues the tour. It is also assumed that there is enough fuel available for vehicle to reach the refuelling station or the

depot. The following constraints are added along with the constraints from Eqs. (4) to (13).

$$0 \le \omega_j \le \omega_i - g.c_{ij}x_{ijk} + R(1 - x_{ijk}) \ \forall j \in C, i \in V', i \ne j \quad (14)$$

$$\omega_j = R \ \ j \in F \cup V_0 \quad (15)$$

$$\omega_j \ge g.c_{ij} \ \ j \in F \cup V_0 \quad (16)$$

Eq. (14) is used to track the fuel level, Eq. (15) specifies that the capacity of the fuel is R and is replenished in the refuelling station or depot. Eq. (16) ensures that there is enough fuel to reach the refuelling station/depot and the vehicles are not stranded in the middle of the tour.

4. Chemical reaction optimization

To solve GVRP-Q, an enhanced Chemical Reaction Optimization (e-CRO) is used. Chemical reaction optimization is a recent meta heuristic introduced by Lam and Li (2010) inspired by the behaviour of chemical molecules in a chemical reaction. CRO is a varied population based meta heuristic. It portrays the interaction of molecules in high energy unstable states in a chemical reaction that strives to attain a low energy stable state. It is successfully applied to solve discrete optimization problems.

The work carried out using CRO is reviewed here. CRO for task assignment problem in grid computing was given by Xu, Lam and Li (2011) where they compared the efficiency of CRO with other meta heuristic algorithms. An Artificial Chemical Reaction Optimization (ACRO) inspired by chemical reaction was proposed by Alatas (2011) and they demonstrated its efficiency using experiments. CRO with greedy strategy (2013) was used to solve 0/1 knapsack problem. The features of CRO are combined with a repair operator using greedy strategy to achieve the result. CRO for multi objective optimization problem was studied by Bechikh, Chaabani and Ben Said (2014). An oppositional real coded chemical reaction optimization was used by Bhattacharjee, Bhattacharya and Nee Dey (2014) for solving short term hydrothermal scheduling problems. Their contribution is to propose a new quasi-linear average time complexity non dominated sorting algorithm to make efficient multi objective algorithm. CRO is known for its exploration and exploitation capability. CRO with unified tabu search is applied to solve capacitated VRP (Dam, Li & Fournier-Viger, 2016), A decomposition based CRO is used for solving simultaneous pick up and delivery problem and is found to be competitive with other algorithms was shown in Li, Wang, Hei, Li and Jiang (2017).

The literature on CRO as a solution methodology for solving VRP is limited. CRO with unified tabu search is used to solve capacitated VRP (Dam et al., 2016) and a decomposition based CRO is used for solving simultaneous pickup and delivery problem which is found to be competitive with other algorithms (Li et al., 2017). Since the use of CRO was not extensively available to solve VRP and, to study the behaviour of CRO on VRP, it is used as a solution methodology to solve GVRP-Q.

4.1. Chemical reaction optimization framework

Chemical reaction optimization is inspired by the reaction of molecules and follows the first two laws of thermodynamics. Chemical reaction is a process wherein one or more chemicals involve, react and produce a different chemical (product). The substances that are involved in a reaction are called reactants. The reactants undergo a series of change through elementary reactions to form a product. Chemical reactions are of two types endothermic, which takes in energy and exothermic, which releases energy. There are many categories of chemical reaction and the most prominent being decomposition and synthesis. In decomposition, a molecule is broken down into smaller chemical pieces as specified below,

$$2H_2O \rightarrow 2H_2 + O_2$$

In Synthesis, two or more molecules combine to produce a complex molecule as shown below,

$$8Fe + S_8 \rightarrow 8FeS$$

These properties of a chemical reaction are brought into the meta heuristic CRO. It combines computation and chemistry. It has superior exploration ability and has a good combination of exploration (diversification) and exploitation (intensification) capabilities.

In CRO, each solution is called a molecule. A molecule (X) can be imagined as a point in the search space. Each molecule wanders in the Potential Energy Space (*PES*) and seeks to attain a lower potential state with minimum energy through chemical reactions.

Each molecule has two types of energy namely Potential Energy (*PE*) and Kinetic Energy (*KE*). *PE* is the energy stored in the molecule and corresponds to the objective function value of the considered molecule. *KE* measures the tolerance level of a molecule and corresponds to situations where a molecule moves to a less favourable structure. It is a measure for accepting the worst solution in the *PES*. Higher the *KE*, greater is the possibility of a new molecular structure with higher *PE*. In short, *KE* of a molecule is for exploration and *PE* is for exploitation.

The profile of a molecule includes its *PE,KE*, number of times it experiences a chemical reaction, the optimal structure and the minimum objective value attained. Chemical reaction is a sequence of collisions that occur either between the molecules or between the molecule and the walls of the container. Based on these characterizations, four elementary chemical reactions are included in the CRO process. They are on-wall ineffective collision, decomposition, inter-molecular ineffective collision and synthesis.

4.2. Elementary reaction of chemical reaction optimization

Chemical reaction can happen when molecules collide with each other or with the walls of the container. Based on the collision, the reactions are classified as uni molecular collision when the molecule hits the walls of the container and multi molecular collision when two or more molecules collide. Based on the nature of collision, they are classified as given in Fig. 1.

The two ineffective collisions search for better positions in the neighbourhood and attain a state close to the current state and exploit *PES*. On the other hand, decomposition and synthesis explore a different *PES* and get a solution that is distinct from the current state. A proper balance of this exploration and exploitation leads to a good solution in the *PES*. CRO is a varied population based meta heuristic as decomposition increases the size of the population and synthesis decreases the size of the population.

4.2.1. On-wall ineffective collision

When molecules collide on a wall, it results in collision and the structure of the molecule changes. Let X be a molecule represented as a vector that stores the positions of the customers and PE_X be its objective function value. Let X' be the molecule obtained using on wall in effective collision which is created using swap mutation that will be described in Section 4.3.4. It's objective function value is $PE_{X'}$. The change in the molecular structure is allowed if,

$$PE_X + KE_X \ge PE_{X'}$$

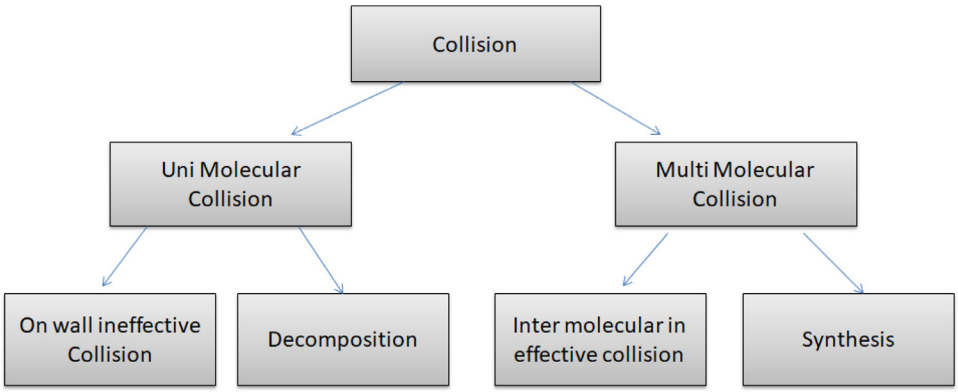

Fig. 1. Elementary reactions of chemical reaction optimization.

The procedure for on-wall ineffective collision is described in Algorithm 1.

Algorithm 1 On-wall ineffective collision.

Let $X=\{x_1,x_2,...x_n\}$ be the selected molecule for on-wall ineffective collision and PE_X be its objective function value
Let $X'=\{x'_1,x'_2,....,x'_n\}$ be the molecule in the neighbourhood of X and $PE_{X'}$ be its objective function // *The molecule X'is generated using swap mutation*
Calculate $PE_{X'}$
if $PE_X+KE_X \geq PE_{X'}$ then
  Get a random number q in [*KElossrate*, 1] // *KElossrate is system parameter*
  $KE_{X'}=(PE_X+KE_X-PE_{X'})\times q$ // *calculate KE of X'*
  Update *buffer*=*buffer*+$(PE_X+KE_X-PE_{X'})\times(1\text{-}q)$ // *store released KE in buffer*
  Update the profiles of *X*as $X=X'$, $PE_X=PE_{X'}$ and $KE_X=KE_{X'}$
end if
end

During the reaction, the *KE* lost (*KElossrate*) is saved in central buffer and $(1-q)$ represents the amount of *KE* lost in the environment.

4.2.2. Decomposition

In decomposition, a molecule X hits the wall and break down into two molecules X'_1 and X'_2 which is created using modified circular shift described in Section 4.3.4. The collision is so influential that the resultant molecular structures are entirely different from the original structure. The change is allowed if the original molecule has enough energy to enrich the *PE* of the resultant molecules otherwise,

$$PE_X+KE_X \geq PE_{X'_1}+PE_{X'_2} \tag{17}$$

is satisfied. If this condition is not satisfied, the energy drawn in the central buffer is used as given below,

$$PE_X+KE_X+buffer \geq PE_{X'_1}+PE_{X'_2}$$

So, to enable decomposition the molecule should have enough energy to complete the procedure else the energy needed for decomposition is drawn from central buffer. The new values of *KE* for the molecules X_1' and $X_2'$are,

$$KE_{X'_1}=(temp1+buffer)\times m1\times m2$$

$$KE_{X'_2}=(temp1+buffer)\times m3\times m4$$

$$buffer=buffer+temp1-KE_{X'_1}-KE_{X'_2}$$

where $temp1=PE_X+KE_X-PE_{X'_1}-PE_{X'_2}$ and $m1$, $m2$, $m3$ and $m4$ are random numbers generated between [0, 1]. To generate $KE_{X'_1}$ and $KE_{X'_2}$, two random numbers are multiplied to ensure that the resultant *KE* are not too large.

4.2.3. Synthesis

Synthesis occurs when two molecules X_1 and X_2 collide and result in a single molecule X''. The new molecule X'' is generated using the procedure unite_molecule which will be explained in Section 4.3.4. The collision is so violent that the resultant molecule is different from the original molecule. Algorithm 2 specifies the steps carried out using synthesis.

Algorithm 2 Synthesis.

Obtain X'' from molecules X_1 and X_2
Calculate $PE_{X''}$
if $PE_{X_1}+PE_{X_2}+KE_{X_1}+KE_{X_2} \geq PE_{X''}$ then
  $KE_{X''}=PE_{X_1}+PE_{X_2}+KE_{X_1}+KE_{X_2}-PE_{X''}$
  assign $PE_{X''}$,$KE_{X''}$to the profile of molecule X''
end

4.2.4. Inter molecular ineffective collision

This collision occurs when two molecules X_1 and X_2 collide with each other and then bounces away. It is similar to on-wall ineffective collision but no *KE* is drawn to central energy buffer. The changes in the structure of the molecules are not prominent, but they attain a structure with slight deviation from the current structure. So, X_1 and X_2 create two new molecules X'_1 and X'_2 in the neighbourhood of X_1 and X_2. The two new molecules X'_1 and X'_2 are created using the procedure bounce_collide that will be defined in Section 4.3.4. The change is accepted when Eq. (18) is satisfied (Algorithm 3).

$$PE_{X_1}+PE_{X_2}+KE_{X_1}+KE_{X_2} \geq PE_{X'_1}+PE_{X'_2} \tag{18}$$

In addition to these procedures, CRO is also equipped with some attributes that are used to keep track of the reactions. These attributes are,

➢ *Number of hits*: Used to keep track of the number of times a molecule participated in a chemical reaction.

Algorithm 3 Inter molecular ineffective collision.

Create two molecules X_1' and X_2' from X_1 and X_2
Calculate PE_{X_1},PE_{X_2} and $PE_{X'_1}$,$PE_{X'_2}$
Let $temp2 = (PE_{X_1} + PE_{X_2} + KE_{X_1} + KE_{X_2}) - (PE_{X'_1} + PE_{X'_2})$
If $temp2 \geq 0$ then
  Get p randomly in the interval [0,1]
  $KE_{X_1} = temp2 \times p$
  $KE_{X_2} = temp2 \times (1 - p)$
update the profiles of molecules with new *KE* and *PE*
end
End

➢ *Minimum hit number*: Used to track the number of times a molecule attained a minimum structure with minimum *PE*.

4.3. Proposed enhanced chemical reaction optimization algorithm for GVRP-Q

An enhanced chemical reaction optimization (e-CRO) is used to solve the green vehicle routing problem with queues. During chemical reactions catalysts are used to enhance the reactions. Similarly, to enhance the capability of CRO a bacterial transformation is proposed that acts as a catalyst to increase the exploration and convergence of the molecules. The details of the bacterial transformation will be presented in Section 4.4.

Each molecule represents a string of customers. To construct a route, refuelling stations and depot are included which transforms a molecule into a route. The overall steps of the algorithm e-CRO for solving GVRP-Q are depicted as a flow chart in Fig. 2.

4.3.1. Molecule representation and initialization

Each molecule X is represented with N customers as $\{C_1, C_2, \ldots, C_N\}$. Let *popsize* denotes the total number of molecules initialized in *molecule pool* where 70% of the molecules are generated at random and the remaining 30% of molecules are generated through the nearest neighbour heuristic starting from a random initial customer. To facilitate each generated customer string as a route (solution), depot and if necessary refuelling stations are included. A procedure called *generate route* is used to convert each molecule to a feasible route.

4.3.2. Generate route

Generate route is used to convert all molecules to routes. Depot is inserted at the start and end of each molecule. A vehicle starts from depot and visits a set of customers, if necessitates it halts at the nearby fuelling station, waits in the queue for service and departs the fuelling station. Hence, the main constraint is the maximal allowable time limit on each vehicle and the remaining fuel level. If the time limit exceeds, a new vehicle is introduced. If fuel is not sufficient to serve the next customer, the vehicle continues the tour by visiting the nearby refuelling station and can serve more customers within the time limit.

The total time taken $Total\_time_k$ by each vehicle k to serve customers is calculated using Eq. (19).

$$Total\_time_k = \tau_{V_0 i} + \sum_{i,j \in V' \setminus \{V_0\}, i \neq j} (t_\sigma + \tau_{ij}) x_{ijk} + \tau_{jV_0} \tag{19}$$

The time taken τ_{ij} is calculated using Eq. (20)

$$\tau_{ij} = \frac{d_{ij}}{speed} \quad where \ \ i, j \in V' \tag{20}$$

d_{ij} is the distance between i and j and t_σ is calculated as specified in Eq. (20). If $Total\_time_k \geq T$ then a new vehicle is introduced and the vehicle is guided to the depot within the time limit. Refuelling stations are introduced when vehicle's fuel level is not sufficient to reach the next customer. At the end of the *generate route* procedure, the molecules are converted to feasible routes where depot and refuelling stations are included in the routes.

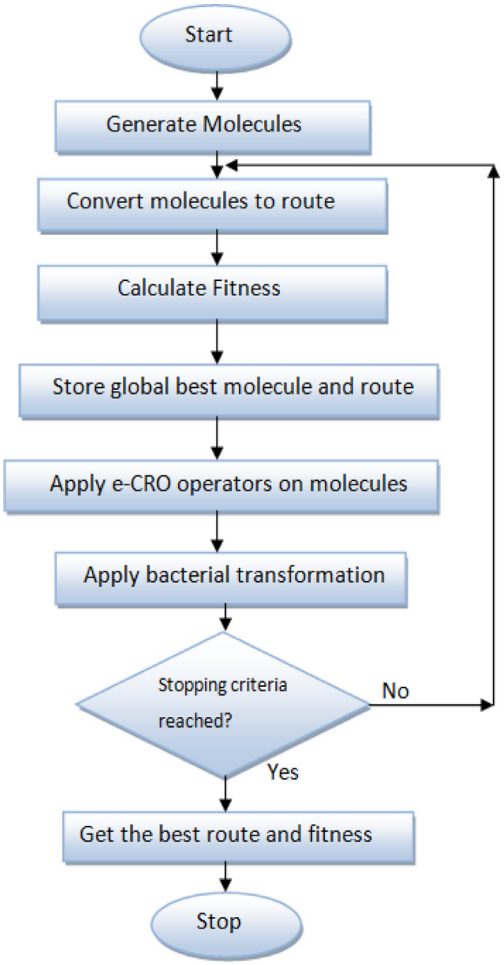

Fig. 2. Sequence of steps used to solve GVRP-Q using e-CRO.

4.3.3. Fitness function

The *PE* of a molecule is calculated using Eqs. (21) and (22) respectively which provides the *route_cost* for a single vehicle and *total_cost* for all the vehicles.

$$route\_cost = d_{V_0 i} + \sum_{i,j \in V' \setminus \{V_0\}} d_{ij} + d_{jV_0} \quad \forall i, j \in V' \setminus \{V_0\}, \ i \neq j \tag{21}$$

$$total\_cost = \sum_{k \in K} route\_cost \tag{22}$$

where *route_cost* is used to calculate the cost of a single route by a vehicle and *total_cost* is used to find the total cost incurred by k vehicles and is stored in *PE* which represents the overall fitness. *PE* of all the molecules are calculated and stored. Then the best *PE* is stored in PE_{best}, and the corresponding molecule and route are stored in g_{mole} and g_{best} respectively which represents the global best molecule and global best route.

4.3.4. Operators used in e-CRO

To carry out the four elementary reactions of CRO, carefully tailored operators are used which are specific to the problem under study. All the operations are carried out on the molecule. Since there are four chemical reactions, four different operators are used

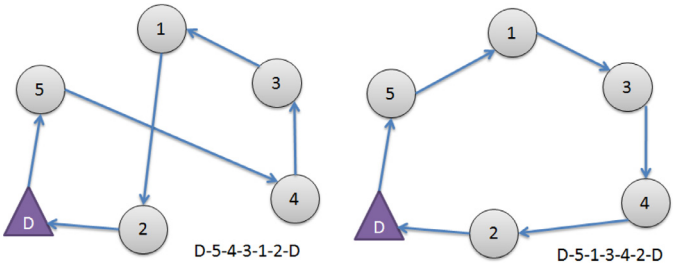

Fig. 3. Swap mutation.

to carry out these reactions. The procedures for the chemical reactions are already described under Section 4.2. The following operators are used to convert a selected molecule(s) to new molecule(s).

4.3.4.1. Swap mutation for on-wall ineffective collision. The unimolecular collision is not so severe hence there is not much difference between the generated molecule and the original molecule. To carry out this, swap mutation is used as described in Algorithm 4.

Algorithm 4 Swap_mutation (X).

Select two random numbers $r1$, $r2$ in the interval [0, 1]
Let N be the size of the molecule
Let $\omega_1 = \lceil N \times r1 \rceil$
Let $\omega_2 = \lceil N \times r2 \rceil$ // *get the customer position values within N*
Swap $X(\omega_1)$ and $X(\omega_2)$
End

Fig. 3 represents the swap mutation on a molecule and is shown with the depot for better understanding. The positions of customers 1 and 4 are swapped.

4.3.4.2. Modified circular shift for decomposition. Decomposition is used to break down a single molecule into two different molecules which is carried out using modified circular shift. It proceeds by shifting the molecule's position that is recommended by $rr1$ as defined in Algorithm 5. This number suggests whether the shift proceeds with the original molecule or reversed molecule. If the random number is negative, the shift continues with the original molecule otherwise, the shift is carried out with the reversed molecule. The procedure is detailed in Algorithm 5.

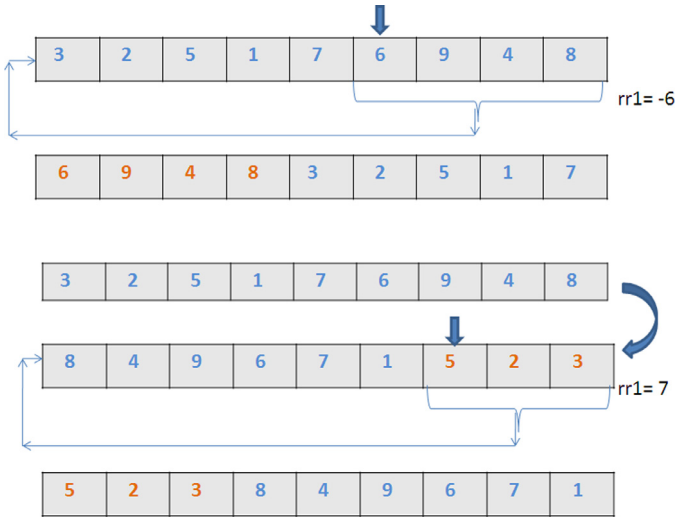

Fig. 4. Modified circular shift operator.

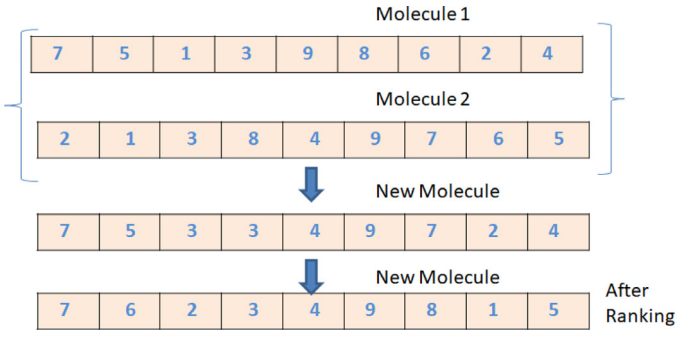

Fig. 5. Unite molecule operator.

in Fig. 4. The same procedure is repeated with another set of random numbers to generate the second molecule as decomposition produces two different molecules.

4.3.4.3. Unite molecule for synthesis. To unite two molecules into a single molecule, synthesis is used. To enable this, an operator used by Truong, Li and Xu (2013) is used. Two molecules are chosen and the position values of the molecules are compared. A new

Algorithm 5 Modified circular shift (X).

*mole*1 represents the new molecule and X represents the current molecule and X' is reversed molecule
Let $a = -N$, $b = N$ // *a and b define the range to generate random number*
Select a random number r from [0, 1]
Let $rr1 = \lceil a + (b-a)^* r \rceil$ // *rr*1 *defines the random number within the range*
Select a molecule X at random
if $rr1 < 0$ then
  Let $rr2 = -rr1$
    $num1 = N - (rr2) + 1$ // *num*1 *determines the number of position values to be copied*
    $mole1(1{:}\ num1) = X(rr2{:}\ N)$ // *customer values from rr*2 *to N is copied to*1: *num*1
    $mole1(num1 + 1,\ N) = X(1{:}\ rr2 - 1)$ // *remaining customer positions copied to mole*1
else
    $num1 = rr1$
    $num2 = N - num1 + 1$ // *num*2 *specifies the number of positions to be copied*
    $mole1(1{:}\ num2) = X'(rr1{:}\ N)$ //*customer positions are copied into mole*1
    $mole1(num2 + 1{:}\ N) = X'(1{:}\ rr1 - 1)$ //*the remaining positions are copied into new molecule*
end
end

When $rr1 = -6$, the shift is for the original molecule. When $rr1 = 7$, the shift proceeds with the reversed molecule as depicted

molecule is generated based on this position value. The resultant molecule is different from the original molecules. The process is represented in Fig. 5 and Algorithm 6 explains the procedure.

Algorithm 6 Unite Molecule (X_1,X_2).

```
Select two molecules X1 and X2 at random to produce new molecule X3
for i = 1 to N
  Select a random number r1
  If r1 > 0.5 then
    X3(i) = X1(i)
  else
    X3(i) = X2(i)
  end
End
```

Fig. 6. Bounce collide operator for inter molecular in effective collision.

Then, a ranking procedure is run on the resultant molecule X_3 to get a molecule without duplicates as shown in Fig. 5.

4.3.4.4. Bounce collide operator for inter molecular ineffective collision. Inter molecular ineffective collision is a bi molecular collision. To construct this, a new operator is devised called as *bounce_collide*. Two molecules are chosen namely X_1 and X_2 at random. Then the best molecule g_{mole} is chosen. Similar to a two point crossover in genetic algorithm, two cut points are chosen at random in g_{mole} and the position values of customers between the cut points are copied in to two new molecules *mole*1 and *mole*2 respectively. To create *mole*1, the positions of customers out of the cut point in X_2 are compared with the customer positions within the cut point of *mole*1, if the corresponding customer positions are same, the customer position of *mole*1 is replaced with X_1 and checked. When the customer position is not available between the cut points, then the position of the customer from X_1 is inserted into *mole*1. Then the duplicates from the new molecule are removed. The same procedure is repeated to create another molecule *mole*2. The procedure is detailed in Algorithm 7.

After each molecular reaction, the duplicates are removed to get the resultant molecules. The given algorithm is repeated by inter changing X_2 with X_1 and *mole*2 with *mole*1 to get another new molecule. The working of the algorithm is demonstrated in Fig. 6.

4.4. Bacterial transformation

The energy required to assemble a reaction is called activation energy. Energy is required for molecules to move around in the *PES*. When a catalyst is added, molecular structure can change releasing some amount of energy. This energy can invoke a lot of reactions. Catalyst can rapidly transform the course of reactions.

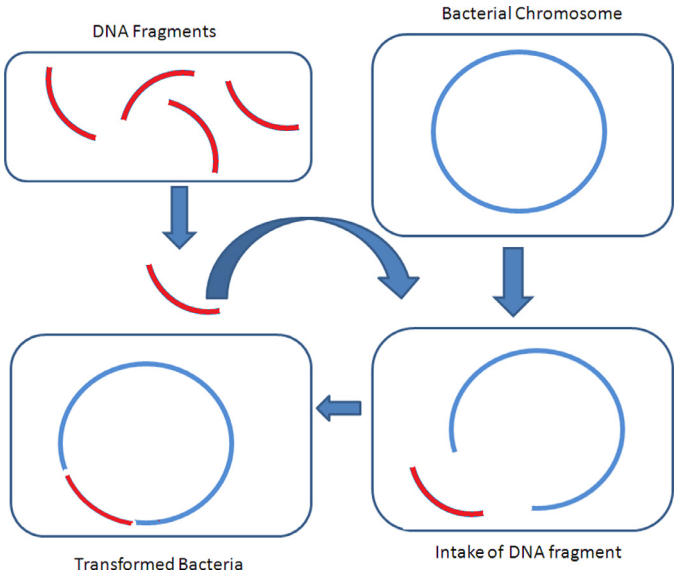

Fig. 7. Bacterial transformation.

Hence a new operator is designed that act as a catalyst to enhance the reaction inspired by bacterial transformation.

Nature supports genetic diversity using crossover and mutation. Apart from these known techniques, genetic inversion, transduction and transformation, conjugation, transposition and translocation are some other mechanisms (Gould & Keeton, 1996) used by nature. Instead of crossover and mutation, some other transformations, if introduced could improve the performance of evolutionary algorithms as suggested by Mitchell and Forrest (1994) and Banzhaf, Nordin, Keller and Francone (1998). Based on these studies bacterial transformation is included as a new mechanism for increasing the molecular diversity.

Definition 4.1. In molecular biology, transformation is the genetic alteration of a cell resulting from the direct update and incorporation of exogenous genetic material from its surroundings taken up through cell membrane.

Bacteria take fragments of DNA from the environment and use it for transformation. Based on these ideas, bacterial transformation as described in Clark and Russell (1997) is used in this paper. Fig. 7 illustrates the bacterial transformation where bacteria acquires DNA fragment from the environment which is transformed into a new structure.

To facilitate this bacterial transformation in CRO, two sets of pools are constructed as shown in Fig. 8.

First pool called as the *molecule pool* has the set of molecules, the second pool, called as the *segment pool* has a set of molecular fragment / slice of molecules that are created using nearest neighbour heuristic during the initial iteration. During the iteration a molecule is selected at random from both *segment pool* and *molecule pool*. A random point is chosen in the recipient molecule from the *molecule pool* and the segment/fragment from the *segment pool* is inserted at that point. This corresponds to the biological process where the gene segment is incorporated into the recipients' chromosome and changes the gene ordering as shown in Fig. 7 and a new molecule is formed. As the iteration progresses, *segment pool* are updated with best molecular segments from the best molecules.

A binary coding of these bacterial transformation is studied by Simoes et al. (2001) to solve 0/1 knapsack problem. They initialized a random pool of binary strings and incorporated these strings in the solution string. The transformed gene is found to be promis-

Algorithm 7 Bounce_collide(X_1,X_2).

```
Choose two molecules X1, X2 and g_mole
Let mole1 and mole2 be two molecules formed
Let c1 and c2 be two random cut points chosen where c1 < c2
mole1(c1: c2) = g_mole(c1: c2)      //copy positions between two cut points to mole1
mole2(c1: c2) = g_mole(c1: c2)      //copy the positions between two cut points to mole2
for i ≥ 1 and i <c1      // loop from 1 till cut point c1
  Let temp = X2(i)      // copy the contents of molecule X2 into temp
    for j ≥ c1 and j ≤ c2      // loop from cut point c1 to cut point c2
      if temp = mole1(j)      //check whether temp is available between the cut points
      temp = X1(i)      // copy the content of X1 specified by the given position into temp
      end
    end
mole1(i) = temp      // copy the content of temp into molecule mole1
end
for i = c2 + 1: cust      // loop from c2 till end of molecule
  temp = X2(i)      // copy content of X2 at ith position into temp
    for j = c1: c2      // loop from cut point 1 to cut point 2
      if temp = mole1(j)      // check if temp is available within the cut point in mole1
        temp = X1(i)      // if available copy content of X2 into temp at that position
      end
    end
mole1(i) = temp      // copy content of temp into the specified position of mole1
End
```

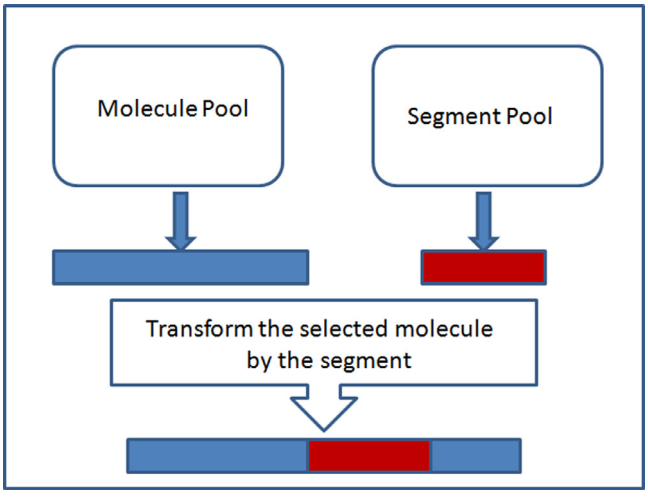

Fig. 8. Depiction of molecule and segment pool.

ing and is used to enhance the solution quality. But, the proposed method uses the *segment pool* that has molecular segments created using nearest neighbour heuristic and is an integer encoding. Moreover, it is updated with the best molecules available in the *segment pool* in the search space across the iterations. The procedure describing this bacterial transformation is explained in Algorithm 8.

Algorithm 8 Bacterial transformation (X, X_s).

```
Let seg_size be the size of the segment pool
popsize is the size of molecule pool
Select a molecule X_s at random from segment pool
Select a molecule X at random from molecule pool
Choose a random point in X and insert the segment X_s
```

The total number of segments (*seg_size*) residing in the *segment pool* is 30% of the size of the population($0.3 \times popsize$) in *molecule pool*. During the iteration, the molecules with best fitness are selected and the segment is extracted within the chosen random positions. The length of the segment is not uniform in the *segment pool*. The segment does not include depot / refuelling stations. After each iteration, a random set of segments are moved out of the *segment pool* to preserve the diversity. Let *no_s* be the number of segments moved out of segment pool which is calculated using eqn. (23)

$$no\_s = \lceil 0.2 \times seg\_size \rceil \tag{23}$$

4.5. Procedure e-CRO for GVRP-Q

The complete procedure of e-CRO for GVRP-Q is explained in Algorithm 9. The algorithm starts with an initialization phase where the parameters *molecol, KElossrate,*α, β, *buffer, initKE* are initialized. The parameter *molecol* is used to decide unimolecular or inter molecular collision, *initKE* specifies the initial kinetic energy of a molecule. $\alpha$and β are the intensification and diversification parameters respectively. Initially the *molecule pool* and *segment pool* are populated. During the iteration phase, uni or inter molecular collision is decided based on the random number *rand* taken in the interval[0, 1]. If *rand* > *molecol* or there is a single molecule, uni molecular collision is chosen, otherwise inter molecular collision is selected.

For uni molecular collision, a molecule is randomly chosen and checked for the decomposition criterion (*number of hits* − *minimum hit number*) $\geq \alpha$ where α is the tolerance duration level for the molecule without getting any new minimum solution. If the decomposition criterion is satisfied, it is carried out otherwise on wall in effective collision is done.

For inter molecular collision, two random molecules are chosen and the synthesis criterion $KE \leq \beta$ is checked, where β specifies the minimum *KE* possessed by a molecule. If the condition is satisfied, synthesis is performed otherwise inter molecular ineffective collision is carried out.

4.6. Theoretical analysis

This section presents the time complexity analysis of e-CRO which provides a polynomial run time on the input. This section also demonstrates that bacterial transformation aids in faster convergence.

4.6.1. Time complexity of e-CRO

The *molecule pool* has *popsize* number of molecules each with dimension *N*. As 30% of the molecules are initialized with nearest neighbour heuristic, the complexity is O($popsize.N^2$), *generate route* procedure has a computational complexity of O($popsize.N$).

Algorithm 9 e-CRO for GVRP-Q.

```
Initialize the parameters of e-CRO molecol, KElossrate, α, β, buffer, initKE
t is the iteration number
Initialize the population of molecules in molecule pool
Initialize the segments in segment pool
Let N be the number of customers
for i = 1 to popsize
  Call procedure generate route
  Calculate the PE using Eqs. (21) and (22).
  Assign initial KE in initKE
end
Store the best molecule in g_mole and store the best route in g_best
let t = 1
while(t ≤ max_iterations)      //iterate till maximum number of iterations
  get rand in the interval [0,1]
  if(rand > molecol)      // branch for uni molecular collision
    select a molecule X at random from molecule pool
    if (number of hits − minimum hit number) ≥ α
      Create new molecules using modified circular shift
      Call the procedure for decomposition
      update PE and KE
    else
      Create new molecule using swap mutation
      Call the procedure on-wall ineffective collision
      update PE, KE and buffer
    end
  else      //branch out for inter molecular collision
  select two molecules X1 and X2 at random
  if KE(X) ≤ β
    Create new molecule using Unite molecule
    Call procedure Synthesis
    Calculate PE and modify KE according to the procedure
  else
    Select the best molecule g_mole
    Create new molecule using bounce collide
    Call procedure inter molecular ineffective collision
    Calculate PE and modify KE
  end
end
  Call procedure bacterial_transformation
end
end while
```

The fitness of a single route is calculated with a complexity of $O(N+n)$where $(N+n)$ is the length of the route with the inclusion of depots and refuelling station for a single route. The procedures on-wall ineffective collision, decomposition, synthesis, inter molecular ineffective collision each has a complexity of $O(N+n)$. Swap mutation requires$O(1)$and modified circular shift has $O(N)$. For bounce collide, let $\varphi_1 = (c2 - c1 + 1)$, $\varphi_2 = c1$ and $\varphi_3 = (N - c2 + 1)$, then the complexity is $\max(O(\varphi_1.\varphi_2)O(\varphi_1.\varphi_3))$. Unite molecule has $O(N^2)$ as the complexity. The best molecule is found with time $O(popsize)$. The cost of bacterial transformation is $O(seg\_length)$ where *seg_length* is the length of the segment, and updation of *segment pool* takes $O(seg\_size)$. The total time complexity of the procedure e-CRO is $O(popsize.N^2)$for a single iteration.

Lemma. *Bacterial Transformation aids in faster convergence in e-CRO.*

Proof. Let $X = \{x_1, x_2, ..., x_N\}$ be a random molecule chosen from *molecule pool*. Let $X_s = \{x_{s1}, x_{s2}, ..., x_{sl}\}$ be the segment selected at random from the *segment pool*. Let the fitness value of X be $f(X)$. Let $\tilde{X}$ be the molecule obtained by incorporating the segment X_s into X. This transformation is said to be successful if $f(\tilde{X}) < f(X)$ or the molecule lies in an improved search space.

Let σbe the number of transitions required to transform X to desired state $\tilde{X}$ where $\tilde{X}$ is in improved search space. Let π be the number of transitions required to transform X to $\tilde{X}$ using e-CRO. To better portray the working of e-CRO, if the search space is depicted as a tree, with each level representing iteration and each vertex as a solution state with / without any change in the molecular structure. Then, when a transition to a molecule is done based on the chemical reactions the level of the tree increases as it explores many possible reactions in the structure before reaching the stable molecular structure.

But, when a molecular segment is used, the levels are not exhaustive, since with fewer reactions the molecule can attain the stable state as the segment is already a part of a molecule with reduced cost. Hence there is more chance of X transformed to $\tilde{X}$ with fewer transitions, say π which implies that $\pi < \sigma$. Also, during the initial iterations, exploration is carried out by transforming the molecules to increase the search space and as iteration increases, exploitation is done which provide faster convergence. When the best molecules from *segment pool* are copied, the count of best molecules are increased in the *molecule pool* which further leads to good molecule selection and hence molecules converge fast with fewer iterations.

Table 2
Parameter Setting of e-CRO.

Parameter	Value
popsize	60
KElossrate	0.8
molecol	0.2
initKE	800
α	500
β	25
buffer	0

5. Results and discussion

Two different experiments are conducted on G-VRP data set, first set of experiments are conducted to assess the performance of e-CRO and second set of experiments are conducted to evaluate the impact of the wait time at the refuelling station which is modelled with M/M/1 queuing system. The results obtained are summarized in this section.

5.1. Experimental setup

The proposed algorithm e-CRO is executed on a computer with Intel Core i3-3220 processor at 3.30 GHz and 4 GB RAM and is implemented using MatlabR2011a. Experiments are conducted using G-VRP data set proposed by Erdogan and Miller Hooks (2012).

Table 2 describes the parameter setting of e-CRO algorithm. It is very difficult to tune the parameters and there is no standard available for tuning. The parameters *molecol, KElossrate, buffer* are initialized based on Truong et al. (2013) whereas *initKE*, α and β are tuned for performance.

5.2. Comparison of proposed algorithm e-CRO with other algorithms

To assess the performance of e-CRO, it is compared against six other algorithms proposed in the literature, MCWS and DBCA (Erdogan & Miller Hooks, 2012), Heuristic Based Exact Solution (HBEA) (Koç & Karoaglan, 2016), Local search and Simulated Annealing (Felipe et al., 2014), Variable Neighbourhood Search/Tabu Search (VNS/TS) (Schneider et al. 2014) and Adaptive VNS/TS (AVNS/TS) (Schneider et al., 2015). Tables 3–6 projects the results obtained. The tables also projects the Number of Customers (NC), Number of Routes (NR), cost of each route (cost) and the time taken to run the program in seconds (CT). All data instances are run for 5000 iterations and the total time taken is recorded. In almost many cases, the instance is able to get the solution with less than 5000 iterations.

When compared with the algorithms MCWS, DBCA and Local Search, e-CRO is able to get better results. It is competitive with

Table 3
Results obtained for uniformly distributed customer data set.

Dataset	MCWS			DBCA			HBEA		VNS/TS		AVNS/TS		Local Search			e-CRO			
	NR	NC	Cost	NR	NC	Cost	NC	Cost	NC	Cost	NC	Cost	NR	NC	Cost	NR	NC	Cost	CT secs
20c3sU1	6	20	1818.35	6	20	1797.51	20	1797.49	20	1797.49	20	1797.49	6	20	1805.41	6	20	1797.5	354.4
20c3sU2	6	20	1614.15	6	20	1613.53	20	1574.78	20	1574.77	20	1574.77	6	20	1574.78	6	20	1574.02	465.1
20c3sU3	7	20	1969.64	7	20	1964.57	20	1704.48	20	1704.48	20	1704.48	6	20	1704.48	6	20	1704.59	426.3
20c3sU4	5	20	1508.41	5	20	1487.15	20	1482.00	20	1482.00	20	1482.00	6	20	1482.00	5	20	1482.02	330.2
20c3sU5	6	20	1752.73	6	20	1752.73	20	1689.37	20	1689.37	20	1689.37	6	20	1689.37	6	20	1689.25	315.8
20c3sU6	6	20	1668.16	6	20	1668.16	20	1618.65	20	1618.65	20	1618.65	6	20	1618.65	6	20	1620.83	338.8
20c3sU7	6	20	1730.45	6	20	1730.45	20	1713.66	20	1713.66	20	1713.66	6	20	1713.67	6	20	1713.7	423.1
20c3sU8	6	20	1766.36	6	20	1718.43	20	1706.50	20	1706.50	20	1706.50	6	20	1722.78	6	20	1706.5	312.6
20c3sU9	6	20	1714.43	6	20	1714.43	20	1708.82	20	1708.81	20	1708.82	6	20	1708.82	6	20	1708.82	327.5
20c3U10	5	20	1309.52	5	20	1309.52	20	1181.31	20	1181.31	20	1181.31	4	20	1181.31	5	20	1181.3	426.3
Sum	59	200	16,852.2	59	200	16,756.48	200	16,177.06	200	16,177.06	200	16,177.06	57	200	16,201.27	58	200	16,178.5	3720.1
Average	5.9	20	1685.22	5.9	20	1675.64	20	1617.70	20	1617.70	20	1617.70	5.7	20	1620.12	5.8	20	1617.85	372.01

Table 4
Results obtained for clustered customer data set.

Dataset	MCWS			DBCA			HBEA		VNS/TS		AVNS/TS		Local Search			e-CRO			
	NR	NC	Cost	NR	NC	Cost	NC	Cost	NC	Cost	NC	Cost	NR	NC	Cost	NR	NC	Cost	CT secs
20c3sC1	5	20	1300.36	5	20	1300.62	20	1173.57	20	1173.57	20	1173.57	4	20	1178.97	4	20	1173.42	312.45
20c3sC2	5	19	1553.53	5	19	1553.53	19	1539.97	19	1539.97	20	1539.97	5	19	1539.97	6	19	1540	272.5
20c3sC3	4	12	1083.12	4	12	1083.12	12	880.20	12	880.20	20	880.20	3	12	880.97	3	12	879.38	289.5
20c3sC4	5	18	1135.90	5	18	1091.78	18	1059.35	18	1059.35	20	1059.35	4	18	1059.35	5	18	1059.42	295.1
20c3sC5	7	19	2190.68	7	19	2190.68	19	2156.01	19	2156.01	20	2156.01	7	19	2156.01	6	19	2157.34	286.7
20c3sC6	9	17	2883.71	9	17	2883.71	17	2758.17	17	2758.01	20	2758.01	8	17	2758.17	9	17	2758.17	274.2
20c3sC7	5	6	1701.40	5	6	1701.40	6	1393.99	6	1393.99	20	1393.99	4	6	1393.99	5	6	1394	211.8
20c3sC8	10	18	3319.74	10	18	3319.74	18	3139.72	18	3139.72	20	3139.72	9	18	3139.72	7	18	3139.74	299.6
20c3sC9	6	19	1811.05	6	19	1811.05	19	1799.94	19	1799.94	20	1799.94	6	19	1799.94	6	19	1799.9	241.6
20c3sC10	8	15	2648.84	8	15	2644.11	15	2583.42	15	2583.42	20	2583.42	8	15	2583.42	8	15	2583.42	297.3
Sum	64	163	19,628.33	64	163	19,579.74	163	18,484.34	163	18,484.18	200	18,484.18	58	163	18,490.51	59	163	18,484.8	2780.7
Average	6.4	16.3	1962.83	6.4	16.3	1957.97	16.3	1848.43	16.3	1848.42	20	1848.42	5.8	16.3	1849.05	5.9	16.3	1848.48	278.07

Table 5
Results obtained for uniform and clustered customer data set.

Dataset	MCWS			DBCA			HBEA		VNS/TS		AVNS/TS		Local Search			e-CRO			
	NR	NC	Cost	NR	NC	Cost	NC	Cost	NC	Cost	NC	Cost	NR	NC	Cost	NR	NC	Cost	CT secs
S1_2i6s	6	20	1614.15	6	20	1614.15	20	1578.12	20	1578.12	20	1578.12	6	20	1578.12	6	20	1588.74	385.5
S1_4i6s	5	20	1567.30	5	20	1541.46	20	1397.27	20	1397.27	20	1397.27	5	20	1413.97	5	20	1402.89	369.7
S1_6i6s	6	20	1616.20	6	20	1616.20	20	1560.49	20	1560.49	20	1560.49	6	20	1571.30	6	20	1573.14	398.1
S1_8i6s	6	20	1902.51	6	20	1882.54	20	1692.32	20	1692.22	20	1692.22	6	20	1692.33	6	20	1692.19	376.8
S1_10i6s	5	20	1309.52	5	20	1309.52	20	1173.48	20	1173.48	20	1173.48	4	20	1173.48	4	20	1173.5	412.5
S2_2i6s	6	20	1645.80	6	20	1645.80	20	1633.10	20	1633.10	20	1633.10	6	20	1645.80	6	20	1645.8	399.3
S2_4i6s	6	19	1505.06	6	19	1505.06	19	1505.07	19	1532.96	19	1505.07	6	19	1505.07	6	19	1505.44	387.5
S2_6i6s	10	20	3115.10	10	20	3115.10	20	2431.33	20	2431.33	20	2431.33	8	20	2660.49	8	20	2491.94	399.0
S2_8i6s	9	16	2722.55	9	16	2722.55	16	2158.35	16	2158.35	16	2158.35	7	16	2175.66	7	16	2174.12	401.5
S2_10i6s	6	16	1995.62	6	16	1995.62	16	1585.46	17	1958.46	16	1585.46	5	16	1585.46	7	17	1673.84	399.2
Sum	65	163	18,993.81	65	191	18,948	191	16,714.99	192	17,115.78	191	16,714.89	59	191	17,001.68	61	192	16,921.6	3929.1
Average	6.5	16.3	1899.38	6.5	19.1	1894.8	19.1	1671.49	19.2	1711.57	19.1	1671.48	5.9	19.1	1700.16	6.1	19.2	1692.16	392.91

Table 6
Impact of refuelling stations on uniform and clustered customer data sets.

Dataset	MCWS			DBCA			HBEA		VNS/TS		AVNS/TS		Local Search			e-CRO			
	NR	NC	Cost	NR	NC	Cost	NC	Cost	NC	Cost	NC	Cost	NR	NC	Cost	NR	NC	Cost	CT secs
S1_4i2s	6	20	1582.2	6	20	1582.2	20	1582.21	20	1582.21	20	1582.21	6	20	1598.91	6	20	1582.2	396.9
S1_4i4s	6	20	1580.52	6	20	1580.52	20	1460.09	20	1460.09	20	1460.09	5	20	1483.19	6	20	1461.54	402.5
S1_4i6s	5	20	1561.29	5	20	1541.46	20	1397.27	20	1397.27	20	1397.27	5	20	1413.97	5	20	1397.23	398.8
S1_4i8s	6	20	1561.29	6	20	1561.29	20	1397.27	20	1397.27	20	1397.27	6	20	1397.27	6	20	1397.3	399.1
S1_4i10s	5	20	1536.04	5	20	1529.73	20	1396.02	20	1396.02	20	1396.02	5	20	1396.02	6	20	1401.99	399.7
S2_4i2s	5	18	1135.89	5	18	1117.32	18	1059.35	18	1059.35	18	1059.35	4	18	1059.35	5	18	1059.35	392.5
S2_4i4s	6	19	1522.72	6	19	1522.72	19	1446.08	19	1446.08	19	1446.08	5	19	1446.08	6	19	1446.08	382.4
S2_4i6s	6	20	1786.21	6	20	1730.47	20	1434.14	20	1434.14	20	1434.14	5	20	1434.14	5	20	1434.14	389.1
S2_4i8s	6	20	1786.21	6	20	1786.21	20	1434.14	20	1434.14	20	1434.14	5	20	1434.14	5	20	1434.14	392.6
S2_4i10s	6	20	1783.63	6	20	1729.51	20	1434.13	20	1434.13	20	1434.13	5	20	1434.13	5	20	1434.13	398.6
Sum	57	197	15,836	57	197	15,681.43	197	14,040.7	197	14,040.7	197	14,040.7	51	197	14,097.2	55	197	14,048.1	3952.2
Average	5.7	19.7	1583.6	5.7	19.7	1568.143	19.7	1404.07	19.7	1404.07	19.7	1404.07	5.1	19.7	1409.72	5.5	19.7	1404.81	395.2

the other algorithms HBEA, VNS/TS and AVNS/TS. For data sets 1 and 2, the average route cost produced by e-CRO is almost the same with HBEA, VNS/TS and AVNS/TS. For data sets 3 and 4 there is an increase in the average route cost by 1.2% and 0.3% respectively from the best results produced by AVNS/TS and HBEA. For data set 3, e-CRO is able to serve one customer more than the algorithms HBEA and AVNS/TS which attribute to an increase in the route cost and it almost did proportionately well for data set 4 with a minimum difference in the route cost which proved the competitiveness of e-CRO against the published results.

The algorithm AVNS/TS is able to generate better route costs for all the data instances and is an improved version of algorithm VNS/TS. Both VNS/TS and AVNS/TS use an extensive set of local search heuristics to build the routes. Both the procedures are the same but AVNS/TS use more number of local search procedures than VNS/TS. AVNS/TS include 6 sequence relocation operators, 18 cyclic operators while constructing the route. To improve the routes, an improvement procedure is carried out with 2-opt and OR-opt procedures. The method also includes penalty functions as infeasible solutions are introduced while building the routes. Separate penalty functions for fuel capacity violation and time constraint violation are used in the algorithm. It also includes the use of redundant facility centres which introduces infeasibility in the routes. While constructing the routes, the number of vehicles used by the algorithm is specified as an input and the algorithm is run for that specified number of vehicles. After 500 iterations, if it is found that the specified number of vehicles is not able to serve all customers, then the vehicle count is incremented and iterated. It is one of the main drawbacks of VNS/TS and AVNS/TS.

HBEA combines the advantages of exact algorithm branch and cut and simulated annealing. For the success of branch and cut an initial solution is required which is obtained using Clarke and Wright heuristic and is improved using simulated annealing to get an initial best solution with the help of a set of local exchange operators. For the success of this method, a better initial feasible solution is essential and a considerable amount of time is spent in deciding the initial solution. HBEA combines the essence of exact algorithm branch and cut enumeration and the use of heuristic to search for a better solution and hence is an elaborate procedure to obtain a route and is not suitable for solving large size data instances.

The difference between e-CRO and all the other algorithms is e-CRO does not rely on the use of local search procedures. All the solutions produced by e-CRO are feasible solutions. It does not employ any penalty function where infeasible solutions are allowed and then penalized. In e-CRO, the total number of vehicles used is obtained while constructing the route and hence separate procedures are not necessary for determining the total number of vehicles. These features make e-CRO a better algorithm than other algorithms.

The sample routes obtained using e-CRO is depicted in Figs. 9 and 10 for the data instances 20c3sC3 and 20c3sU9 respectively. A new best solution for data instance 20c3sC3 is obtained using e-CRO. The results also project the total number of routes which represents the number of vehicles used. HBEA, VNS/TS and AVNS/TS did not provide total number of routes, hence it is not possible to compare the total routes and hence the number of vehicles used. The total number of vehicles used by e-CRO is less when compared to MCWS and DBCA.

Table 7 projects the results obtained for large data sets. The iteration is limited to 5000 for large data set. But, when compared to the solutions obtained using MCWS, DBCA and Local Search method, the solution obtained using e-CRO is good. The Relative Percentage Deviation (RPD) of e-CRO with VNS/TS and AVNS/TS is 0.64% and 1.09% respectively which exhibits the competitiveness of e-CRO against these algorithms for large datasets. The solutions for

Table 7
Results obtained for large data sets.

Data Set	MCWS			DBCA			VNS/TS		AVNS/TS			Local Search			e-CRO		
	NR	NC	Cost	NR	NC	Cost	NC	Cost	NC	Cost	Average	NR	NC	Cost	NR	NC	Cost
111	20	109	5626.64	20	109	5626.64	109	4797.15	109	4770.47	4791.53	18	109	4960.60	19	109	4801.25
200	35	190	10,428.59	36	191	10,413.59	192	8963.46	192	8886.00	8970.14	32	192	9276.63	32	192	8964.31
300	49	281	14,242.56	49	282	14,229.92	283	12,594.77	283	12,374.49	12,514.78	46	283	12,869.17	46	283	12,620.22
350	57	329	16,471.79	57	329	16,460.30	329	14,323.02	329	14,103.66	14,271.56	54	329	14,954.83	54	329	14,323.02
400	67	378	19,472.10	66	373	19,099.04	378	16,850.21	378	16,697.21	16,839.23	61	378	17,351.92	62	378	16,852.48
450	75	424	21,854.17	75	424	21,854.19	424	18,521.23	424	18,310.60	18,512.47	68	424	19,215.38	68	424	18,522.54
500	84	471	24,527.46	84	471	24,517.08	471	21,170.90	471	20,609.67	20,874.50	76	471	21,636.59	78	471	21,171.56
Sum	387	2182	112,623.3	387	2179	112,200.7	2186	97,220.7	2186	95,752.1	96,774.2	355	2186	100,265.1	359	2186	97,255.38

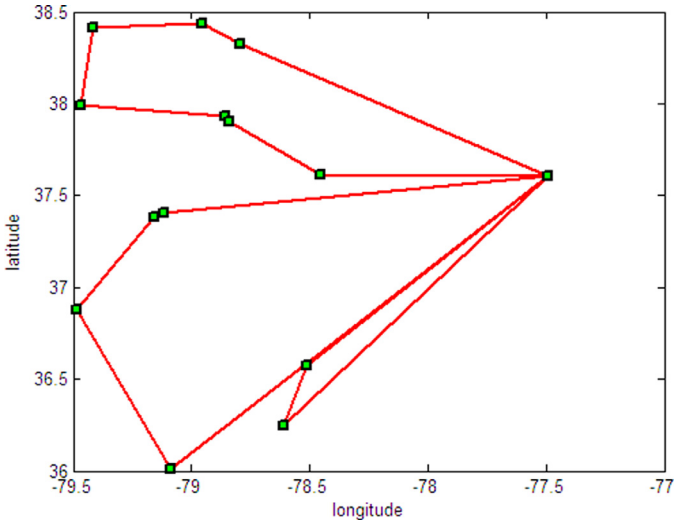

Fig. 9. Route for 20c3sC3.

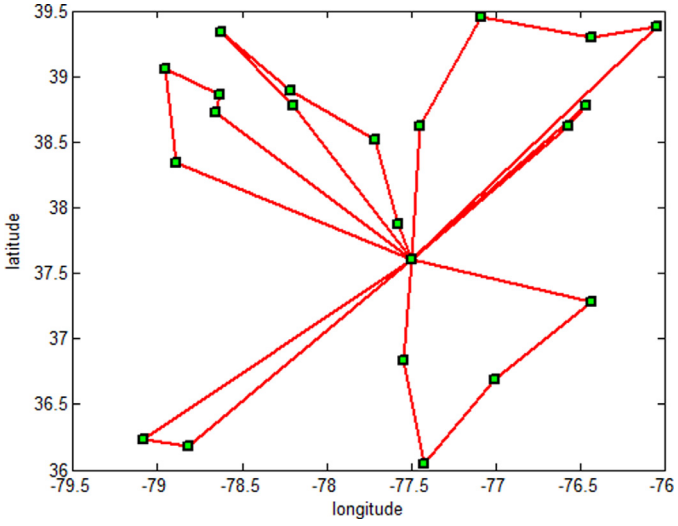

Fig. 10. Route for 20c3SU9.

large data sets are not available using HBEA as the time required to build the solution is more as it use the exact algorithm.

5.3. Green vehicle routing problem with queues

The first set of experiments conducted on G-VRP using e-CRO projects it's efficiency in comparison with all the other algorithms published in the literature. The second set of experiment is conducted by modelling the refuelling station with a single server queue model. The proposed e-CRO is used to solve GVRP-Q and the results are projected in Table 7 where e-CRO is run for 5000 iterations.

To study the behaviour of the system, it is assumed that the arrival rate of vehicles follows Poisson distribution at the facility centre with an arrival rate (λ) of 3 vehicles per hour and the service follows exponential distribution with service rate (μ) given by 4 vehicles per hour. Each vehicle has a service time of 15 min and a new vehicle arrives every 20 min and hence, the wait time is calculated as 45 min using Eq. (1).

To solve this problem, the routes are constructed as before using e-CRO, but the procedure includes an additional wait time in the queue. Since wait time has a considerable influence on the total time, the direct consequence is an increase in the use of number of vehicles. It is not possible to serve all customers with the same number of vehicles as vehicles wait for service in queue in addition to serving time and should return to the depot within the maximum allowable time.

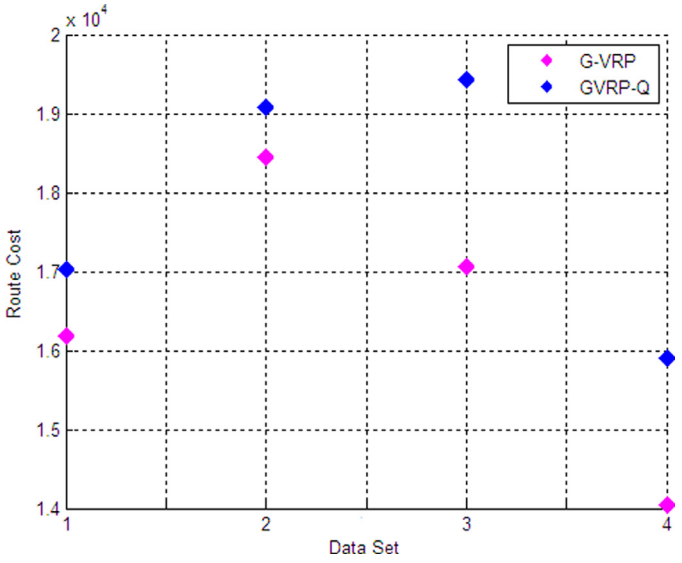

Fig. 11. Comparison obtained for G-VRP with and without queues.

The results obtained for GVRP-Q are projected in Table 8 for 20 customer data instances. It is evident from the results that there is an increase in the route cost because of the wait time constraint. Since the performance of GVRP-Q is dependent on the parameters $\lambda$and μ the route cost for large data sets are not analyzed and the route cost will increase with wait time and the proportion in increase depends on the values of $\lambda$and μ.

The main objective of this paper is to demonstrate the influence and importance of the wait time on the route cost which is generally ignored while calculating the overall route cost. The results projected in Table 8 are based on the values of λ and μ and the route cost varies depending on these values.

Table 9 reports the Relative Percentage Deviation (RPD) of the total route cost with queues against the results obtained without queues using e-CRO and is calculated using Eq. (24)

$$\frac{obtained\ \ solution - optimal\ \ solution}{optimal\ \ solution} \times 100 \tag{24}$$

It is observed that on an average there is an 8% increase in the total route cost. Hence, the experiments substantiate the influence of wait time on the route cost accompanied by an increase in the use of number of vehicles by 34 in GVRP-Q than G-VRP using e-CRO.

Fig. 11 shows the comparison of the route cost of G-VRP and GVRP-Q. As seen, there is a steep increase in the route cost of all the data sets, particularly for data set 3 which is attributed by the use of more number of vehicles, since the wait time has a considerable influence on the overall time taken by the vehicle.

Some of the limitations of e-CRO as observed from this experiment are, since it executes a single chemical reaction in an iteration, the convergence is dependent on the type of reaction being executed which results in a bit slower convergence. Also, CRO has more parameters to tune for better performance. Among the parameters, *buffer* and *molecol* are important parameters which decide the course of the reactions. Around 500 iterations are conducted to fix the parameters α, β and *initKE* for e-CRO. As CRO is a varied population based meta heuristic, the initial population should be determined effectively. When compared to the performance of e-CRO with small and large data set, it has performed

Table 8
Route cost obtained for GVRP-Q.

Dataset	e- CRO for GVRP-Q			Dataset	e- CRO for GVRP-Q			Dataset	e- CRO for GVRP-Q			Dataset	e- CRO for GVRP-Q		
	NR	NC	Cost		NR	NC	Cost		NR	NC	Cost		NR	NC	Cost
20c3sU1	7	20	1794.32	20c3sC1	5	20	1347.05	S1_2i6s	6	20	1604.74	S1_4i2s	7	20	1594.74
20c3sU2	7	20	1613.30	20c3sC2	6	19	1574.11	S1_4i6s	7	20	1623.24	S1_4i4s	7	20	1623.24
20c3sU3	7	20	1875.61	20c3sC3	4	12	1084.33	S1_6i6s	7	20	1727.97	S1_4i6s	6	20	1585.38
20c3sU4	6	20	1551.43	20c3sC4	5	18	1130.28	S1_8i6s	7	20	1970.19	S1_4i8s	7	20	1624.15
20c3sU5	7	20	1802.44	20c3sC5	6	17	1671.76	S1_10i6s	5	20	1309.6	S1_4i10s	6	20	1530.93
20c3sU6	7	20	1707.93	20c3sC6	10	17	2654.21	S2_2i6s	6	20	1702.75	S2_4i2s	5	18	1127.29
20c3sU7	7	20	1788.80	20c3sC7	6	6	1625.14	S2_4i6s	6	19	1574.23	S2_4i4s	6	19	1515.49
20c3sU8	7	20	1799.83	20c3sC8	8	18	3399.41	S2_6i6s	10	17	2851.54	S2_4i6s	7	20	1768.42
20c3sU9	7	20	1775.19	20c3sC9	7	18	1815.24	S2_8i6s	10	14	2753.9	S2_4i8s	6	20	1768.42
20c3U10	5	20	1313.45	20c3sC10	9	15	2781.11	S2_10i6s	7	17	1921.84	S2_4i10	6	20	1768.42
Average	6.7	20	1702.23		6.6	16	1908.26		7.1	18.7	1904.0		6.3	19.7	1590.64

Table 9
Relative Percentage Deviation in total Route Cost in GVRP and GVRP-Q.

Data Set	Total route cost of GVRP-Q using e-CRO	Total route cost of G-VRP using e-CRO	Relative percentage deviation
Data Set 1	16,178.5	17,022.3	5.21
Data Set 2	18,484.8	19,082.6	3.23
Data Set 3	16,921.6	19,040.0	12.51
Data Set 4	14,048.1	15,906.4	13.22

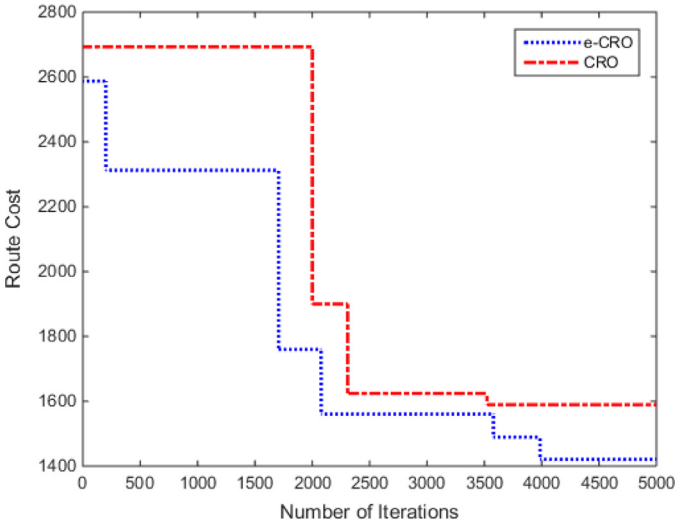

Fig. 12. Convergence of e-CRO and CRO for S2_4i4s data set.

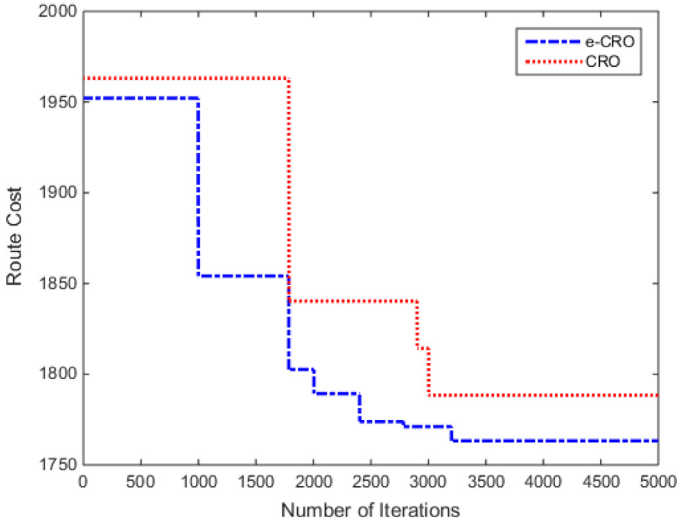

Fig. 13. Convergence of e-CRO and CRO data set 20c3sc9.

well with small data set and has done proportionately well for large data set. Since only a single chemical reaction occur in each iteration, the trajectory expected for exploration is slowed down which has an influence on the number of iterations.

5.4. Comparing the convergence of CRO and e-CRO

To test the performance of e-CRO against CRO, the convergence of both the methods are tested against each other. Figs. 12–14 displays the convergence of e-CRO against CRO for ten runs on the problem instances with the same parameter setting. It can be observed that e-CRO has good exploration combined with bacterial transformation and hence it is able to obtain optimal solution with less number of iterations. Within the same number of iterations, CRO is not able to deduce an optimal solution. Consequently, CRO combined with bacterial transformation has better exploration and is able to find good result within the specified number of iterations. Hence, the convergence rate is better in e-CRO when compared to CRO for the same number of iterations.

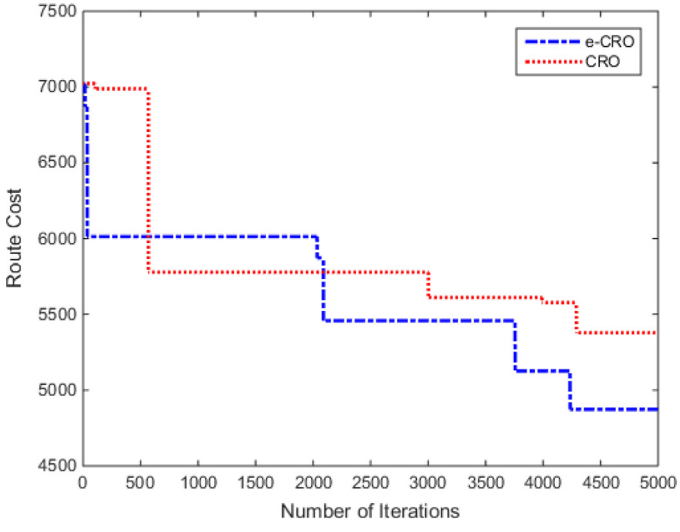

Fig. 14. Convergence of e-CRO and CRO for 111 customers' data set.

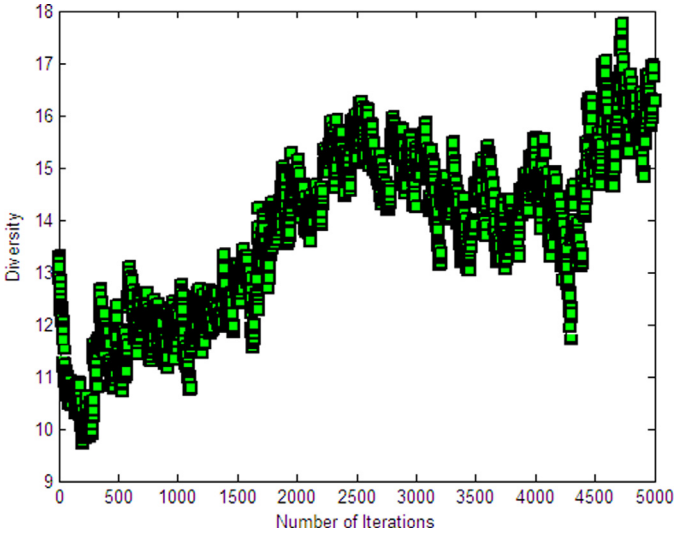

Fig. 15. Diversity for 20c3sC6 with population size 30.

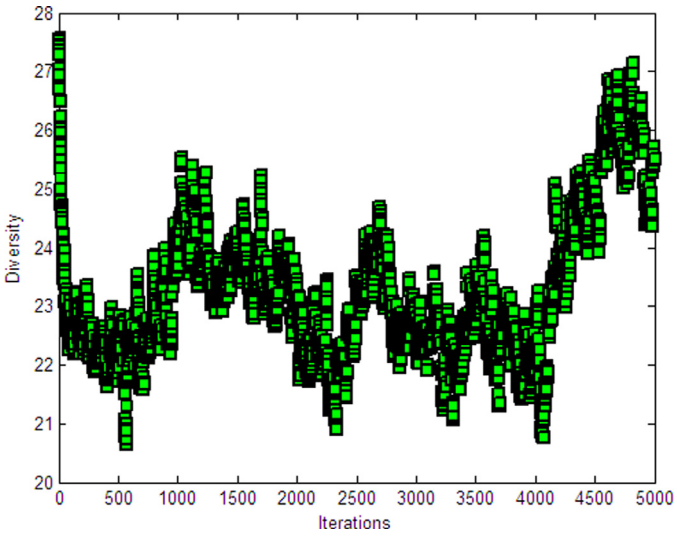

Fig. 16. Diversity for 20c3sC6 with population size 60.

5.5. Diversity in population

When the molecule's diversity is high, there will be good exploration. For better convergence of the algorithm an initial diversity should be high and towards convergence, the diversity should be minimal or zero. To better portray this on the molecules, a population diversity measure is used. The molecule's diversity is affected by various factors like decomposition, synthesis, collision and bacterial transformation.

In general, it is difficult to examine the effects of diversity on the operators. With the introduction of custom designed operators for the molecular collision and with bacterial transformation, the diversity of the molecules on various instances is analyzed. The molecules exhibit a high degree of diversity on initialization and in each iteration it experiences some reactions and finally the bacterial transformation is applied which demonstrates a high degree of diversity on the molecules.

To compute the similarity between the molecules, the Normalized Hamming Distance (*Norm_HD*) is used and is given in Eq. (25)

$$Norm\_HD = \frac{1}{popsize \times N(popsize-1)} \sum_{i=1}^{popsize} \sum_{j=1}^{popsize} HD(X_i, X_j) \quad (25)$$

Here *popsize* is the population size and *N* is the size of the molecule. X_i and X_j are the *i*th and *j*th molecule respectively and $HD(X_i,X_j)$ is given in Eq. (26)

$$HD(X_i, X_j) = \begin{cases} 1 & \text{if } X_i(p) \neq X_j(p) \\ 0 & \text{otherwise} \end{cases} \quad (26)$$

In Eq. (26) *p* specifies the dimension of the molecule. To represent the diversity of the molecules over the iterations an average hamming distance value is used. Figs. 15–18 shows the diversity exhibited by the molecules for the same data set 20c3sC6 on different population sizes like 30, 60, 90 and 120 and the diversity measure is computed for 5000 iterations. At the start of the iterations, diversity is high as molecules are initialized with some randomness, then with a series of operations the diversity of the molecules are observed. When diversity is more, it explores the search space better and there is a good chance to attain a better solution. There is a positive correlation between population size and initial diversity, as population size increases, diversity is more because there is an increased search space.

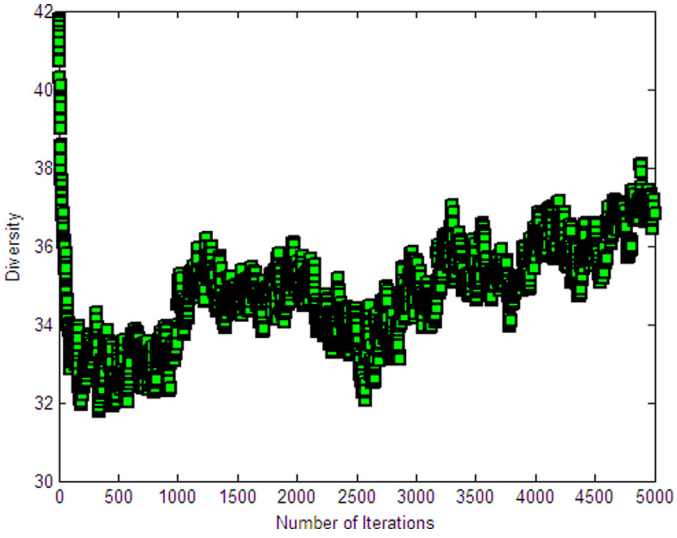

Fig. 17. Diversity 20c3sC6 with population size 90.

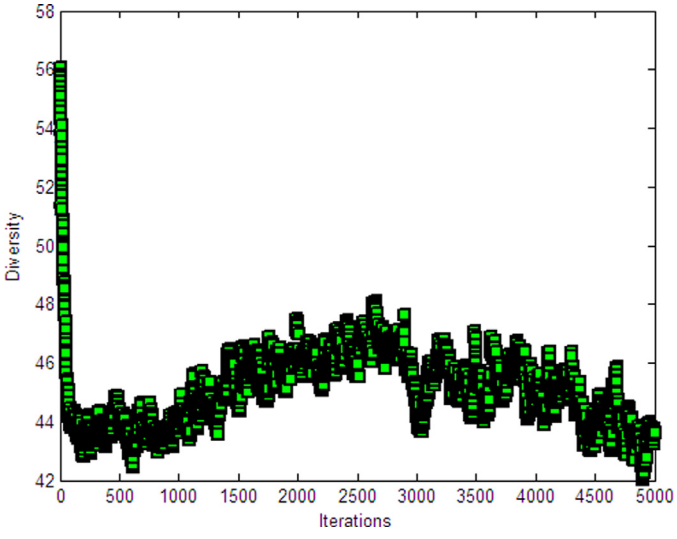

Fig. 18. Diversity for 20c3sC6 with population size 120.

For most instances of the data sets, as the population size is small, the initial diversity is low but as iteration increases, there is high diversity. But for large population size, the initial diversity is high and as iteration increases, the diversity decreases which shows the convergence of the population towards a solution.

When population size is taken as 30, it takes a little longer time to converge. As seen in Fig. 15 the initial diversity is less, after some iterations, the diversity is more compared to initial iterations. But, even with less population size, the algorithm is able to explore the search space for a solution, but the number of iterations may increase. The experiments conducted on the data instance for diversity depicts the influence of population size on the diversity of the molecules which may lead to greater exploration towards an optimal solution.

6. Conclusion and future work

When vehicles are allowed to take long trips, it includes halts in various facility centres. Vehicles may queue up for service at the facility centres which have an impact on the overall cost of the tour. In line with this, a new model GVRP-Q is introduced as an extension of GVRP. It includes halts at refuelling station which are modelled with M/M/1 queuing models.

The results obtained for GVRP-Q demonstrates the influence of wait time on the overall route cost. In GVRP-Q, number of vehicles increase with wait time as vehicles are not able to serve more customers within the maximum allowable time. The experiments conducted on G-VRP data sets projects the influence of the time spent by the vehicles in the facility centre where an increase in the use of number of vehicles lead to an increase in the route cost.

The problem GVRP-Q is solved using e-CRO. To demonstrate the efficiency of e-CRO, G-VRP without queues is solved and the results are compared against six algorithms to prove the competitiveness of e-CRO. To ensure the enhanced behaviour of e-CRO, it is compared against CRO and is proved that within the stipulated number of iterations, the performance of e-CRO is better than CRO. The performance of e-CRO is further tested for diversity and convergence which proved that for small data sets the initial diversity is high and convergence of molecules for e-CRO is better than CRO.

Some of the insights observed from the experiments are the population size of e-CRO has a considerable effect on the convergence. If e-CRO has a sequence of decompositions on the initial stage of the iterations a better exploration of the molecules can be attained which is dependent on the parameter α. Tuning α is an important step for the success of e-CRO. Apart from increasing the route cost, wait time increases the use of number of vehicles as vehicles spend more time in the station which lead to introduction of more vehicles.

The main contribution of the paper is to study the influence of wait time on the route cost hence future research can be focussed to address these waiting time issues. An immediate extension of this research work is to introduce different queuing models in the refuelling station and to explore the behaviour of the model. Based on the length of the queue, suggestions can be provided to increase or decrease the number of servers at the refuelling station. The model can be extended to include the availability of refuelling station for service as an additional constraint. Similarly, each refuelling station can have a predefined upper bound on the capacity of fuel which can be taken as a constraint in the model. As an extension of the work, placement of new refuelling station, removal of existing refuelling station can also be suggested based on the length of the queue. The paper can also be extended to include heterogeneous vehicles in the model.

Conflict of interests

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.

Credit authorship contribution statement

G. Poonthalir: Conceptualization, Writing - original draft. **R. Nadarajan:** Supervision, Validation, Writing - review & editing.

References

Abdoli, B., MirHassani, S. A., & Hooshmand, F. (2019). On different formulations of green vehicle routing problem. *Journal of Information and Optimization Sciences, 40*(4), 883–903.

Alatas, B. (2011). ACROA: Artificial chemical reaction optimization algorithm for global optimization expert systems with applications 38(10), 13170–13180.

Andelmin, J., & Bartolini, E. (2017). An exact algorithm for the green vehicle routing problem. *Transportation Science, 51*(4), 1288–1303.

Angelelli, E., & Speranza, M. G. (2002). The periodic vehicle routing problem with intermediate facilities. *European Journal of Operational Research, 137*(2), 233–247.

Banzhaf, W., Nordin, P., Keller, R. E., & Francone, F. D. (1998). *Genetic programming - An introduction- On the automatic evolution of computer programs and its applications*. Morgan Kaufmann.

Bard, J. F., Huang, L., Dror, M., & Jaillet, P. (1998). A branch and cut algorithm for the VRP with satellite facilities. *IIE Transactions, 30*(9), 821–834.

Bhattacharjee, K., Bhattacharya, A., & Nee Dey, S. H. (2014). Oppositional real coded chemical reaction based optimization to solve short-term hydrothermal scheduling problems. *International Journal of Electrical Power & Energy Systems, 63*, 145–157.

Bechikh, S., Chaabani, A., & Ben Said, L. (2014). An efficient chemical reaction optimization algorithm for multi-objective optimization. *IEEE Transaction on Cybernetics, 45*(10).

Bektaş T Laporte, G. (2011). The pollution-routing problem. *Transportation Research Part B: Methodological, 45*(8), 1232–1250.

Bruglieri, M., Pezzella, F., Pisacane, O., & Suraci, S. (2015). A matheuristic for the electric vehicle routing problem with time windows. arXiv preprint arXiv:1506.00211.

Clarke, D., & Russell, L. (1997). *Molecular biology made simple and fun*. Cache River Press.

Crevier, B., Cordeau, J. F., & Laporte, G. (2007). The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research, 176*(2), 756–773.

Dam, T. L., Li, K., & Fournier-Viger, P. (2016). Chemical reaction optimization with unified tabu search for the vehicle routing problem. *Soft Computing*, 1–13.

Desaulniers, G., Errico, F., Irnich, S. &, & Schneider, M. (2016). Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research, 64*, 1388–1405.

Erdoğan, S., & Miller-Hooks, E. (2012). A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review, 48*(1), 100–114.

Felipe, Á., Ortuño, M. T., Righini, G., & Tirado, G. (2014). A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review, 71*, 111–128.

Gould, J. L., & Keeton, W. T. (1996). *Biological science*. W. W. Norton & Company.

He, S., Zhang, L., Song, R., Wen, Y., & Wu, D. (2009). Optimal transit routing problem for emergency evacuations. *Transportation research board 88th annual meeting (no. 09-0931)*.

Hiermann, G., Puchinger, J., Ropke, S. &, & Hartl, R. F. (2016). The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal Operational Research, 252*, 995–1018.

Kek, A. G., Cheu, R. L., & Meng, Q. (2008). Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots. *Mathematical and Computer Modelling, 47*(1), 140–152.

Keskin, M., & Çatay, B. (2016). Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies, 65*, 111–127.

Koç, Ç., & Karaoglan, I. (2016). The green vehicle routing problem: A heuristic based exact solution approach. *Applied Soft Computing, 39*, 154–164.

Koç, Ç., Jabali, O., Mendoza, J. E., & Laporte, G. (2019). The electric vehicle routing problem with shared charging stations. *International Transactions in Operational Research, 26*(4), 1211–1243.

Lam, A., & Li, V. O. (2010). Chemical-reaction-inspired metaheuristic for optimization. *IEEE Transactions on Evolutionary Computation, 14*(3), 381–399.

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research, 59*(3), 345–358.

Li, H., Wang, L., Hei, X., Li, W., & Jiang, Q. (2017). A decomposition-based chemical reaction optimization for multi-objective vehicle routing problem for simultaneous delivery and pickup with time windows. *Memetic Computing*, 1–18.

Lin, C., Choy, K. L., Ho, G. T., Chung, S. H., & Lam H, Y. (2014). Survey of green vehicle routing problem: Past and future trends. *Expert Systems with Applications, 41*(4), 1118–1138.

Liu, J., & He, Y. (2012, October). A clustering-based multiple ant colony system for the waste collection vehicle routing problems. In *2012 Fifth International Symposium on Computational Intelligence and Design: 2* (pp. 182–185). IEEE.

Mancini, S. (2017). The hybrid vehicle routing problem. *Transportation Research Part C: Emerging Technologies, 78*, 1–12.

Markov, I., Varone, S., & Bierlaire, M. (2015). The waste collection VRP with intermediate facilities, a heterogeneous fixed fleet and a flexible assignment of origin and destination depot (No. REP_WORK).

Mitchell, M., & Forrest, S. (1994). Genetic algorithms and artificial life. *Artificial Life, 1*(3), 267–289.

Mofid-Nakhaee, E., & Barzinpour, F. (2019). A multi-compartment capacitated arc routing problem with intermediate facilities for solid waste collection using hybrid adaptive large neighborhood search and whale algorithm. *Waste Management & Research, 37*(1), 38–47.

Normasari, N. M. E., Yu, V. F., & Bachtiyar, C. (2019). A simulated annealing heuristic for the capacitated green vehicle routing problem. *Mathematical Problems in Engineering*, 2019.

Özdamar, L., Ekinci, E., & Küçükyazici, B. (2004). Emergency logistics planning in natural disasters. *Annals of Operations Research, 129*(1–4), 217–245.

Poonthalir, G., Nadarajan, R., & Geetha, S. (2015). Vehicle routing problem with limited refueling halts using particle swarm optimization with greedy mutation operator. *RAIRO-Operations Research, 49*(4), 689–716.

Prescott-Gagnon, E., Desaulniers, G., & Rousseau, L. M. (2014). Heuristics for an oil delivery vehicle routing problem. *Flexible Services and Manufacturing Journal, 26*(4), 516–539.

Schneider, M., Stenger, A., & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science, 48*(4), 500–520.

Schneider, M., Stenger, A., & Hof, J. (2015). An adaptive VNS algorithm for vehicle routing problems with intermediate stops. *OR Spectrum, 37*(2), 353–387.

Simões, A., & Costa, E. (2001). On biologically inspired genetic operators: Transformation in the standard genetic algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 584–591).

Tarantilis, C. D., Zachariadis, E. E., & Kiranoudis, C. T. (2008). A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities. *INFORMS Journal on Computing, 20*(1), 154–168.

edited by Toth, P., & Vigo, D. (2002). An overview of vehicle routing problems. In P. Toth, & D. Vigo (Eds.), *The vehicle routing problem*. edited bySIAM monographs on discrete mathematics and applications.

Truong, T. K., Li, K., & Xu, Y. (2013). Chemical reaction optimization with greedy strategy for the 0–1 knapsack problem. *Applied Soft Computing, 13*(4), 1774–1780.

Xu, J., Lam, A., & Li, V. O. (2011). Chemical reaction optimization for task scheduling in grid computing. *IEEE Transactions on Parallel and Distributed Systems, 22*(10), 1624–1631.

Zhang, S., Gajpal, Y., Appadoo, S. S., & Abdulkader, M. M. S. (2018). Electric vehicle routing problem with recharging stations for minimizing energy consumption. *International Journal of Production Economics, 203*, 404–413.