Chapter 10
A Discrete Inspired Bat Algorithm for Firetruck Dispatch in Emergency Situations

Dimitra Trachanatzi, Manousos Rigakis, Magdalene Marinaki and Yannis Marinakis

Abstract This research considers the case where a large fire has developed beyond the possibility of suppression and resources need to be deployed to reduce the risk to critical assets. Thus, to determine an optimal deployment of the firetrucks to multiple assets in a large area, a mathematical formulation is proposed, focusing on the maximization of the aggregated value of the protected assets that are critically selected, and on the minimization of the dispatch strategy cost. Moreover, the novelty of the presented formulation is the incorporation of the CO_2 emissions of the firetrucks in the cost function, and, hence, the formulation of the Green-Prize Collecting Vehicle Routing Problem. Moreover, a hybrid Bat Algorithm (BA) is developed for the optimization of the aforementioned problem, namely the Discrete Inspired Bat Algorithm (DIBA). The effectiveness of the proposed algorithmic approach is demonstrated over computational experiments, in comparison with the results of a commercial exact solver.

Keywords Discrete bat algorithm · Prize-collecting vehicle routing problem · CO_2 emissions

10.1 Introduction

In case of an emergency situation, such as a large fire, incidences may result in the loss of or damage to houses, schools, bridges, factories and hospitals, often referred

D. Trachanatzi (✉) · M. Rigakis · M. Marinaki · Y. Marinakis
School of Production Engineering and Management, Technical University of Crete, Chania, Greece
e-mail: dtrachanatzi@isc.tuc.gr

M. Rigakis
e-mail: mrigakis@isc.tuc.gr

M. Marinaki
e-mail: magda@dssl.tuc.gr

Y. Marinakis
e-mail: marinakis@ergasya.tuc.gr

M. Gocić et al. (eds.), *Natural Risk Management and Engineering*, Springer Tracts in Civil Engineering, https://doi.org/10.1007/978-3-030-39391-5_10

to as community assets. In order to prevent the loss of assets, sufficient resources are dispatched to them in a timely manner. The issue that an incident management team (IMT) faces in case of a large fire is the allocation of the available resources. The typical resource units being assigned are fire trucks, commonly referred to as tankers. When dealing with major fire disasters, especially when multiple fire points occurring in one region simultaneously, time is an indispensable and primary factor for each decision-maker. As, Martell (2007) stated: the fire management can be viewed from a supply chain management perspective as *delivering the right amount of the right fire to the right place at the right time at the right cost*. Thus, the objective of an optimal allocation is the maximization of the value of the assets protected and at the same time, the minimization of the response time and cost, subject to several constraints, and among them, the number of the available vehicles. The response time corresponds to the travel duration of a firetruck from the central station to the various points of fire. Moreover, apart from enhancing the cost-effectiveness of the fire suppression activities, the environmental sustainability should be considered in the decision-making and planning of suppression efforts. Thus, fire managers need to be able to incorporate into their response strategies, the consumed fuel amount and the impact of the corresponding CO_2 emissions of the firetrucks.

The focus of the presented research is the formulation of an asset protection problem, in case of a large fire, that takes into account the environmental cost efficiency of the dispatch planning. Thus, the Green-Prize Collecting Vehicle Routing Problem (Green-PCVRP) is proposed. In addition, a hybrid Bat Algorithm (BA) is developed for the optimization of the aforementioned problem, namely the Discrete Inspired Bat Algorithm (DIBA). The effectiveness of the proposed algorithmic approach is demonstrated over computational experiments, in comparison with the results of a commercial exact solver. The rest of the paper is organized as follows: Sect. 10.2 contains a brief literature review; in Sect. 10.3 the mathematical model of the PCVRP is described; in Sect. 10.4 the proposed mathematical formulation of the Green-PCVRP is described; in Sect. 10.5 the original BA is presented; in Sect. 10.6 the DIBA is introduced and described in detail; Sect. 10.7 contains the experimental results; and Sect. 10.8 gives the conclusions.

10.2 Brief Literature Review

The main objective of the presented research is the asset protection optimization in case of a large forest fire, where each node, in the respective problem, is associated with a prize value, i.e. the benefit of protecting the corresponding asset. Van Der Merwe et al. (2014) proposed the Cooperative Orienteering Problem with Time Windows (COPTW), and in their formulation each asset/node needs a predefined number of firetrucks to work simultaneously on it, to be protected. The same authors (Van Der Merwe et al. 2014b) presented a mixed integer programming model formulation for asset protection during escaped wildfires, with the aim of maximizing the total saved asset value. The aforementioned model, solved using CPLEX, incorporates mixed vehicle types with interchangeable capabilities and with vehicle travel times determined by vehicle specific speeds and road network information, while the protection requirements of locations/nodes are defined in terms of the vehicles' capabilities. The Asset Protection Problem (APP) after the initial work of Van Der Merwe et al., was also, solved by an Adaptive Large Neighbourhood Search (ALNS), proposed by Roozbeh et al. (2018), while their two-stage stochastic programming approach was developed to handle the unusual feature of uncertainty in the timing of changes in conditions, such as wind direction and velocity. These changes determine new time windows during which assets must be serviced and hence the optimal deployment schedule and routing of vehicles.

Another formulation was proposed by Tian et al. (2016) who pioneered a bi-objective optimization model to simultaneously optimize the total fire-extinguishing time and the total number of fire engines dispatched. They proposed a Multi-objective Hybrid Differential Evolution Particle Swarm Optimization (MHDP) algorithm to create a set of Pareto solutions for this problem, while the proposed model considers the fire spread speed as a crucial factor. Based on the latter work, Wu et al. (2017) developed an improved bi-objective integer program, containing less variables and constraints, that based on the computational experiments outperforms the work of Tian et al. (2016). Moreover, Wu et al. (2019) presented an emergency scheduling problem for forest fires with limited rescue team resources and priority disaster areas. Such, each fire point is associated with a priority level based on the severity of the fire situation and the formulated integer linear-programming (ILP) model aims to minimize the total travel distance of all rescue teams, which was exactly solved by CPLEX. Consequently, the common practice to simulate an asset protection problem, is the formulation of Vehicle Routing Problem variants that incorporate nodes with prize values, and aim at the maximization of the completed service (as is the aggregated prize from the nodes serviced) and/or at the minimization of total distance travel, i.e. fast response in case of emergency. Thus, we adopt a variation of the Prize-Collecting Vehicle Routing Problem (PCVRP).

In 2006, Tang and Wang formulated the PCVRP to solve the hot rolling production scheduling problem. The proposed mathematical formulation of the PCVRP incorporated a linear combination of three objectives: the total distance minimization; the number of utilized vehicles minimization and the maximization of the collected prizes. Thus, the model can satisfy different requirements by altering the value of the three coefficients, respectively (Tang and Wang 2006). To optimize the PCVRP, they utilized an Iterated Local Search algorithm (ILS) based on Very Large-Scale Neighbourhood (VLSN). In the same field, Zhang et al. (2009) proposed a multi-objective PCVRP-based model for the hot-rolling batch scheduling problem, that included penalty for the non-visited customers and a fixed number of vehicles, and solved it using a Particle Swarm Optimization (PSO) variant. The aforementioned problem with similar mathematical formulation was also solved by Jia et al. (2013) using a Pareto Max-Min Ant System (P-MMAS).

Hence, two variants of the PCVRP formulation have been derived from the literature: when the number of vehicles to be utilized is predetermined (PCVRP-P) and when is not predetermined (PCVRP-NP) (Long et al. 2019). In 2015, Tiwari et al.

(2015) proposed a hybrid edge recombination approach for the PCVRP-P. Subsequently, Li and Tian (2016) developed a two-level self-adaptive variable neighbourhood search for the PCVRP-NP, aggregating the multiple objectives in one function through the weighted sum method. Lately, in 2018, Long et al. studied both version of the PCVRP as a multi-objective problem, using a Pareto-based evolutionary algorithm, combined with a local search strategy. In order to handle the PCVRP-NP, they proposed a decomposition strategy that divides the problem into multiple PCVRP-P problems (Long et al. 2019). The proposed research focuses on the solution of a variant of the PCVRP-P, the Green-PCVRP, that takes into account the CO_2 emissions of the vehicles that follow the generated routes, and which, at least in our knowledge, has never been proposed and particularly, incorporating firetrucks and the action of unloading the vehicle that, actually, significantly affects the proposed objective function.

The Bat Algorithm has been utilized in the solution of several variants of the Vehicle Routing Problem (VRP). In 2015, Taha et al. proposed an adapted variant of the BA for the solution of the Capacitated VRP (CVRP). The Adapted BA, developed in that study, utilizes three parameters (frequency, wavelength and direction search), and allowing a diversity of the population, swifts between global and local search. Also, for the solution of the CVRP, in 2016, Zhou et al. (2016) presented a Hybrid Bat Algorithm with path relinking (HBA-PR), which apart from the BA logic, it integrates the Greedy Randomized Adaptive Search Procedure (GRASP) and the path relinking. Moreover, in the aforementioned approach the random sub-sequences and single-point local search are operated with certain loudness/probability. Furthermore, in 2016, Osaba et al. proposed a discrete Bat Algorithm for solving the Traveling Salesman Problem and the Asymmetric Traveling Salesman Problem, in their version bats are endowed with some kind of "intelligence". This intelligence makes the bats follow different patterns of movement depending on the point of the solution space in which they are located, while they adopted the Hamming distance to the velocity equation for bat movement and extend an additional mechanism of 3-opt moves into local search part. Regarding the same application, also in 2016, Saji and Riffi presented a discrete version of the BA that includes the 2-opt local search technique.

Furthermore, in 2017, Taha et al. presented a discrete version of the BA for solving VRP with Time-Windows (VRPTW), which was utilized in combination with the Large Neighborhood Search (LNS), namely the BA-LNS. Their method incorporates operators that perform selective extractions of customers in an attempt to minimize the number of vehicles and the traveling distance, allowing the bat to discover a large part of the solution space. In 2018, Osaba et al. proposed an evolutionary and discrete variant of the Bat Algorithm (EDBA) for solving the VRPTW, combined with diverse heuristic operators that achieve hybridization using selective node extractions and subsequent reinsertions, as an extension of their previous work. Recently, in 2019 Osaba et al. focused on a Rich VRP (RVRP), the Clustered Vehicle Routing Problem with Pickups and Deliveries, Asymmetric Variable Costs, Forbidden Roads and Cost Constraints. They developed a Discrete and Improved Bat Algorithm (DaIBA), which employs two different neighborhood structures, that are explored depending on the bat's distance regarding the best individual of the swarm.

10.3 Prize-Collecting Vehicle Routing Problem

The present paper is focused on the solution of the Prize-Collecting Vehicle Routing Problem (PCVRP), as such a predefined number of vehicles have to be used, i.e. a predefined number of feasible routes M have to be constructed to visit the available assets to be protected. Following the formulation presented in Li and Tian (2016), the Prize-collecting Vehicle Routing Problem can be described through a complete graph $G = (V, A)$, where $V = \{0, \ldots, N\}$ is the node set and $A = \{(i, j)|i, j \in V\}$ is the set of corresponding arcs. Each node i included in the set $\{1, \ldots, N\}$, represents an asset and such, specific values of prize, p_i and demand d_i are associated to it. The depot, as the initial/starting point, is denoted by node 0, and no prize of demand value is attributed to it. In addition, for each pair of nodes i, j, the travelling time between them can be expressed by their Euclidean distance, noted as c_{ij}. The symmetry of the problem defines that $c_{ij} = c_{ji}$. Furthermore, each vehicle has a maximum capacity Q and a large fixed usage cost G associated to it. Additionally, by r the task completion parameter (minimum ratio) is denoted, obtained by dividing the predetermined amount of demand by the total demand of all nodes. Finally, as N_m is considered the set of nodes that are visited by vehicle m, $(m \in 1, \ldots, M)$. The following decision variables are used:

- $x_{ij}(i \neq j) = 1$ when node j is visited immediately after node i in the optimal solution, otherwise $x_{ij} = 0$, $(i, j \in N)$.
- $x_{ii} = 1$ when customer i is visited in the optimal solution, otherwise $x_{ii} = 0$, $(i \in N)$.

The mathematical formulation of the PCVRP is:

$$Minimize: \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} + G * M - \sum_{i=1}^{N} p_i (1 - x_{ii}) \tag{10.1}$$

s.t

$$\sum_{i=1}^{N} x_{i0} = \sum_{i=1}^{N} x_{0i} = M \tag{10.2}$$

$$\sum_{i=1}^{N} x_{ij} \leq 1, \quad j = 1, \ldots, N \tag{10.3}$$

$$\sum_{j=1}^{N} x_{ij} \leq 1, \quad i = 1, \ldots, N \tag{10.4}$$

$$\sum_{i \in N_m} d_i (1 - x_{ii}) \leq Q, \quad m = 1, \ldots, M \tag{10.5}$$

$$\sum_{i \in S} \sum_{j \in S, (j \neq i)} x_{ij} \leq |S| - 1, \quad \forall S \subset N \tag{10.6}$$

$$\frac{\sum_{i=1}^{N} d_i (1 - x_{ii})}{\sum_{i=1}^{N} d_i} \geq ratio_Q \tag{10.7}$$

$$x_{ij} \in \{0, 1\}, \quad i, j \in N \tag{10.8}$$

The goal of the objective function Eq. (10.1) is the minimization of the total cost (travelled distance and fixed vehicle usage) by taking into account the total collected prize from the visited nodes, the protected assets. Constraint (10.2) requires that each vehicle conducts a route that initiates from the depot and returns to it. Constraints (10.3) and (10.4) ensure that each node is visited at most once, while Constraints (10.5) facilitate the capacity restrictions of the vehicle. Additionally, Constraints (10.6) are used to eliminate the sub-tours for each vehicle route. Constraint (10.7) ensures the minimum ratio of demand to be covered. Finally, Constraints (10.8) specify the integrity conditions on the variables.

10.4 Green Prize-Collecting Vehicle Routing Problem

In order to incorporate the environmental impact of the firetrucks dispatch in case of a large forest fire, the objective function of the PCVRP is adjusted accordingly. The linear formulation of emission volume, as presented by El Bouzekri et al. (2013), considering a Heavy Duty Vehicle (HDV) which has average speed of 80 km/h and fully loaded weights 25 tons, is given below, where:

- $E_{ij}(q, d)$: is the CO_2 emissions from a vehicle in kg/km with the variable of load q in ton and d in km,
- e_f: is the CO_2 emissions of a fully loaded vehicle (1.096 kg/km for a HDV truck) and
- e_e: is the CO_2 emissions of an empty vehicle (0.772 kg/km for a HDV truck).

$$E_{ij}(q, d) = c_{ij} * \left[\frac{\left(e_f - e_e\right)}{Q} \left(q_{ij}\right) + e_e \right] \tag{10.9}$$

The above formulation, Eq. (10.9), is based on the percentage that the vehicle is loaded, while according to the PCVRP mathematical formulation the continuous parameter q_{ij} represents the aggregated volume of demand that the vehicle carries as it traverses from node i to node j. However, in case of a tanker firetruck, the vehicle initiates from the depot fully loaded and, progressively, as it protects the assets, the volume that carries decreases. Hence, the element of the objective function that relates to the cost minimization, i.e. the minimization of the total distance travelled, is adjusted as shown in Eq. (10.10).

$$Min: \sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{m=1}^{M} c_{ij}x_{ij}^{m}\left[\frac{(e_f - e_e)}{Q}\left(Q - q_{ij}^{m}\right) + e_e\right] + G * M - \sum_{i=1}^{N}\sum_{m=1}^{M} p_i y_{im} \tag{10.10}$$

Thus, the proposed mathematical formulation of the Green Prize-Collecting Vehicle Routing Problem (Green-PCVRP) emerges, by replacing the objective function and expanding the decision variables with respect to each vehicle/route m. In addition, a binary auxiliary decision variable y_{im} is used to define whether the node i is included into route m. Moreover, the following constraints should be included to control the parameter q_{ij}^{m}, see Constraints (10.18), (10.19) and (10.20). Particularly, Constraints (10.18) indicate the reduced cargo of the vehicle and, also, do not permit any illegal sub-tours.

$$\sum_{i=1}^{N} y_{i0} = \sum_{i=1}^{N} y_{0i} = M \tag{10.11}$$

$$\sum_{m=1}^{M} y_{im} \leq 1, \quad i = 1, \ldots, N \tag{10.12}$$

$$\sum_{i=1}^{N} y_{im} \geq 1, \quad k = 1, \ldots, M \tag{10.13}$$

$$\sum_{j=1, i \neq j}^{N} x_{ij}^{m} = y_{im}, \quad i = 1, \ldots, N \text{ and } m = 1, \ldots, M \tag{10.14}$$

$$\sum_{i=1, i \neq j}^{N} x_{ij}^{m} = y_{jm}, \quad j = 1, \ldots, N \text{ and } m = 1, \ldots, M \tag{10.15}$$

$$\sum_{i=1}^{N} y_{im} * d_i \leq Q, \quad k = 1, \ldots, M \tag{10.16}$$

$$\sum_{i=1}^{N}\sum_{m=1}^{M} y_{im} * d_i \geq Qr \tag{10.17}$$

$$\sum_{i=0, i \neq j}^{N} q_{ji}^{m} - \sum_{i=0, i \neq j}^{N} q_{ij}^{m} = d_j * y_{jm}, \quad j = 1, \ldots, N \text{ and } m = 1, \ldots, M \tag{10.18}$$

$$0 \leq q_{ij}^{m} \leq x_{ij}^{m}, \quad i, j = 1, \ldots, N (i \neq j) \text{ and } m = 1, \ldots, M \tag{10.19}$$

$$q_{1i}^{m} = 0, \quad i = 2, \ldots, N \text{ and } m = 1, \ldots, M \tag{10.20}$$

Constraint (10.11) establish the correct number of routes to be formulated. Constraints (10.12) guarantee that each node is included into a route no more than once and Constraints (10.13) ensure that each route should include at least one node. Constraints (10.14) and (10.15) ensure the continuity of the route. Finally, Constraints (10.16) and (10.17) are established to avoid the overload of a vehicle and ensure that the required ratio of capacity volume is to service by the complete solution, respectively.

10.5 Bat Algorithm

The Bat Algorithm (BA), proposed by Yang in 2010, is a nature-inspired metaheuristic algorithm based on the echolocation behavior of microbats, which can find their prey and discriminate different kinds of insects even in complete darkness. In the nature, bats emit ultrasonic pulses to the surrounding environment to hunt and properly navigate. Bats are listening to the echoes, produced by the emitted pulses and based on them they can locate themselves and also identify and locate preys and obstacles. Moreover, each bat is able to find the most nutritious areas performing an individual search, or moving towards a nutritious location previously found by any other bat of the swarm. Yang established the following approximate or idealized rules:

1. All bats use echolocation to detect the distance, and they have the ability to distinguish between an obstacle and a prey.
2. All bats fly randomly with a velocity v_i at position x_i with a fixed frequency f_{min}, varying wavelength λ and loudness A_i to search for prey. In this idealized rule, it is assumed that every bat can adjust in an automatic way the frequency (or wavelength) of the emitted pulses, and the rate of these pulses emission $r \in [0, 1]$. This automatic adjustment depends on the proximity of the targeted prey.
3. In the real world, the bats emissions loudness can vary in many different ways. Nevertheless, we assume that this loudness can vary from a large positive A_0 to a minimum constant value A_{min}.

Algorithm 1 Bat Algorithm

```
Define the objective function $f(x)$;
Initialize the bat population $X = x_1, \dots, x_n$;
for each $x_i$ in the population do
    Initialize velocity $v_i$, loudness $A_i$ and pulse rate $r_i$;
    Define pulse frequency $f_i$;
end for
repeat
    for each $x_i$ do
        Generate new solutions through Equations 21, 22 & 23;
        if $rand > r_i$ then
            Select one solution among the best ones;
            Generate a local solution around the best;
        end if
        if $rand < A_i$ and $f(x_i) < f(x_*)$ then
            Accept the new solution;
            Increase $r_i$, reduce $A_i$;
        end if
    end for
until termination criterion not reached;
Rank the bats and return the best of the population;
```

Following the pseudo-code of the original BA (see Algorithm 1), every bat of the population (swarm) represents one possible solution of the problem to be optimized. At first, the population is initialized and the respective parameters to each solution are defined, these are the frequency f_i, the velocity v_i, the loudness A_i and the pulse rate r_i. In the main phase, for each generation t, every bat of the population moves by updating its position and velocity following the Eqs. (10.21), (10.22) and (10.23).

$$f_i = f_{min} + (f_{max} - f_{min})\beta, \quad \beta \in rand[0, 1] \tag{10.21}$$

$$v_i^t = v_i^{t-1} + \left[x_i^t - x_*\right] f_i, \quad x_* : \; best\,solution\,in\,the\,swarm \tag{10.22}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{10.23}$$

With respect to the local search phase a new solution is found around one of the best ones, based on a random walk: $x_{new} = x_{old} + rand[-1, 1] * A^t$, where A^t is the average loudness of the swarm at generation t. Finally, the loudness A_i and the rate r_i of each bat are updated following these formulas:

$$A_i^{t+1} = \alpha A_i^t, \quad \alpha \in [0.90, 0.99] \tag{10.24}$$

$$r_i^{t+1} = r_i^0\left[1 - \exp(-\gamma t)\right], \quad \gamma > 0 \tag{10.25}$$

10.6 Discrete Inspired Bat Algorithm

The Bat Algorithm has been initially design for the solution of continuous optimization problems. Such, it could not be directly applied in the solution of the Green-PCVRP, since the representation of the solutions is made by ordinal number encoding, i.e. discrete values that represent a sequence of nodes. As an instance, considering $N = 12$ number of nodes and $M = 3$ vehicles/routes, a feasible solution could be represented in a vector as: $[1, 4, 7, 10, 8, 1, 5, 6, 2, 3, 1, 9, 1]$, where all three routes initiate from and return to node 1, the duplication of node 1 among consecutive routes in a vector is omitted for simplification. Thus, we proposed a discrete version of the BA, namely, the *Discrete Inspired Bat Algorithm* (DIBA), in which similar to previous researches, the movement equations are replaced by heuristic techniques that search the local area around a solution to retrieve a new one, that will be described in the following. Moreover, the frequency parameter f_i, in order to reduce the complexity of the algorithmic scheme, is omitted while the pulse rate r_i and loudness A_i follow the original logic.

Furthermore, the velocity parameter v_i, depends on the position of the best solution in the population, and in our proposed version, since the original formula includes the frequency parameter, it is calculated as the distance among the respective solution and the best solution in the population. As solution distance, we refer to the number of element-wise similarity among the compared solution vectors, i.e. the number of positions at which the corresponding elements in both vectors are the same. Due the selective nature of the Green-PCVRP, the population is not uniform in terms of length, and, thus, other widely-used distance measures, e.g. Hamming Distance, Manhattan Distance could not be utilized. Also, we adopt the proposal of Osaba et al. (2019), that the velocity value could be represented by the number of neighbor solutions to be found, and that the best among them represents the new solution. The complete proposed algorithmic framework, the DIBA, is presented in Algorithm 2 and the respective flowchart is depicted in Fig. 10.1.

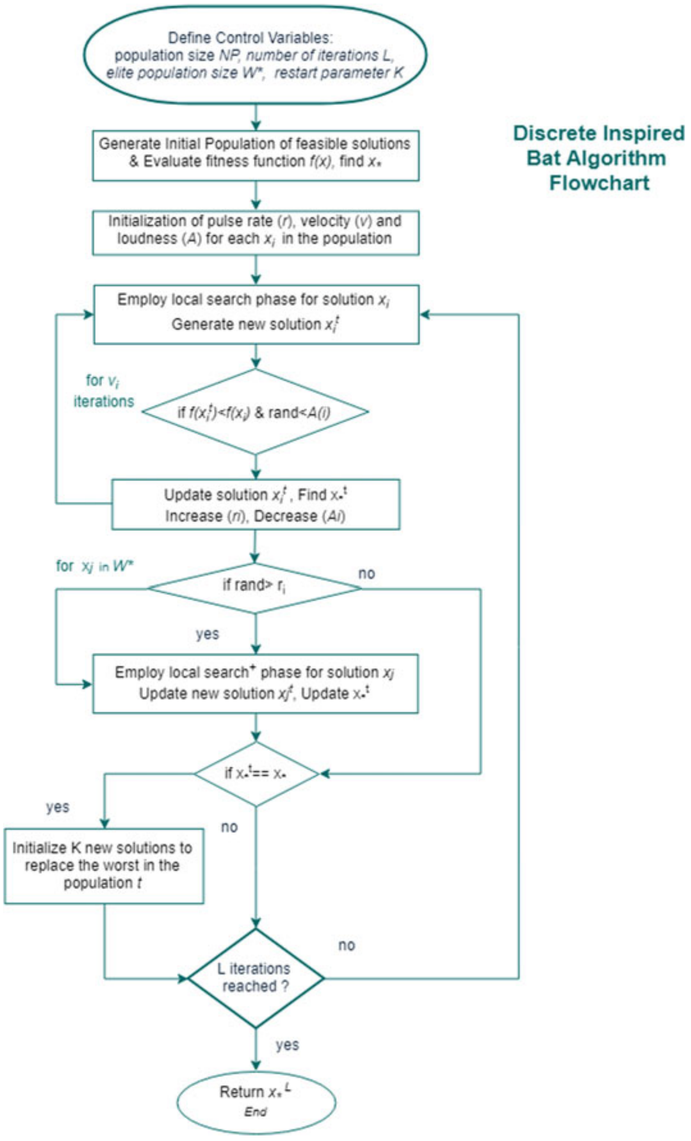
Define Control Variables:
population size NP, number of iterations L, elite population size W*, restart parameter K
Discrete Inspired Bat Algorithm Flowchart
Generate Initial Population of feasible solutions & Evaluate fitness function f(x), find x*
Initialization of pulse rate (r), velocity (v) and loudness (A) for each x_i in the population
Employ local search phase for solution x_i
Generate new solution x_i^t
for v_i iterations
if f(x_i^t)<f(x_i) & rand<A(i)
Update solution x_i^t, Find x*^t
Increase (ri), Decrease (Ai)
for xj in W*
if rand> r_i
no
yes
Employ local search+ phase for solution xj
Update new solution xj^t, Update x*^t
if x*^t== x*
yes
no
Initialize K new solutions to replace the worst in the population t
L iterations reached ?
no
yes
Return x*^L
End

Fig. 10.1 Flowchart of the proposed DIBA

Algorithm 2 Discrete Inspired Bat Algorithm

```
Define: size population $NP$, number of iterations $L$, elite population size $W^*$, restart parameter $K$;
Initialize the solution population via $Initial - Population$;
Evaluate initial population $f(x)$, find best solution $x_*$;
repeat
  for each $x_i$ in the population do
    Calculate velocity $v_i$;
    for a number random number of iterations $\in [1, v_i]$ do
      Employ the $local\ search\ phase$ to generate $x_i^t$:
      Employ $Remove - node$;
      Employ $Reinforcement$;
      Employ $Exchange - node$;
      Evaluate $x_i^t$;
    end for
    if $f(x_i^t) < f(x_i) rand < A_i$ then
      Update $x_i \leftarrow x_i^t$;
      Decrease $A_i$, Equation 24;
      Increase $r_i$, Equation 25;
    end if
  end for
  Sort population, find $x_*^t$;
  Define elite population $W^*$;
  for each solution $j$ in $W^*$ do
    if $rand > r_j$ then
      Employ $local\ search\ phase_+$;
      Update $x_j \leftarrow x_j^t$;
    end if
  end for
  Sort population, find $x_*^t$;
  if $x_*^t == x_*$ then
    Initialize $K$ new solutions via $Initial - Population$;
    Replace $K$ worst solutions;
  end if
until $L$ iterations are reached
Rank the solutions and return the best one;
```

Following the proposed DIBA algorithm, after the parametrization step, the initial population should be generated and evaluated according to the fitness function, see Eq. 10.10. For that, the *Initial-Population* heuristic method is employed, see Algorithm 3. A predefined number of routes are initialized by a random node and combined to form a single vector. In order to satisfy the task completion constraint, a ratio of demand volume should be covered by the complete solution. Thus, until the aforementioned constraint is satisfied, non-included nodes in the initial solution, are inserted based on their most efficient position, in terms of distance travelled, when the rest of the imposed constraints of the PCVRP are not violated. In order to produce solutions of sufficient quality, the non-included nodes are previously sorted based on their prize value.

In the following, the local search phase is conducted, and such for a specific number of iterations the local search techniques *Remove-node*, *Reinforcement* and *Exchange-node* are implemented to each solution in the current population. The number of iterations for this phase is a random value, ranging from one to v_i, while the velocity of the solution i is calculated as described earlier.

Algorithm 3 *Initial − Population*

```
repeat
  Create $M$ initial routes: [1 node 1];
  Combine the initial routes to vector;
  while Total capacity $< Q_{low}$ do
    Create stack: sorted non-included nodes;
    for Each node $k$ in stuck do
      Calculate the efficient position of node $k$;
      Correlate position to route $m$;
      if including node $k$ in route $m$ does not violate the capacity constraint Q then
        Include $k$ and update the solution;
      end if
    end for
  end while
  Calculate the solutions' value in the objective function;
until Population $NP$ is constructed
```

With respect to the embedded local search techniques of this phase, the *Reinforcement* aims to amplify the solution by inserting non-included, so far, nodes into the solution vector, taking into account the feasibility of the solution and the improvement of the solution's quality, while nodes to be inserted are randomly chosen, aiding to the increase of the exploration capability of the procedure. The previously employed one, is the *Remove-node* technique, during which nodes with low prize value are extracted from the solution vector, with a dual objective: (1) the fitness value of the solution could be improved, when the prize decrease offsets the new travel distance and emissions; and (2) the solution is reduced in terms of capacity Q per route, and when the *Reinforcement* is employed, different nodes could be inserted, exploiting that capacity gap.

Algorithm 4 *Remove − node*

```
repeat
  Find route $m$ with the highest value of total distance travelled;
  Find the node $k$ with the smaller prize in route $m$;
  if Reducing the capacity of route $m$ by the prize of node $k$ does not violate the constraints then
    Calculate new distance travelled, connecting the nodes immediate before and after $k$;
    Calculate the objective function of the new formation;
    if New objective function value is smaller than the initial one then
      Update the solution accordingly;
    end if
  end if
until $itermax$ iterations reached
```

Algorithm 5 *Reinforcement*

```
Find all nodes $n$ non-included in the solution vector;
$S \cup S + n$;
Randomly perturb $S$ set;
for each node $j$ in $S$ do
  Find efficient position $k$ for $n_j$;
  Insert $n_j$ in position $k$ in the solution vector;
  Check feasibility and update solution vector accordingly;
end for
```

Regarding the *Exchange-node* local search technique, a simple iterative exchange of nodes between randomly selected routes is conducted, to further exploit the solution, with respect to all imposed constraints of the problem.

Algorithm 6 *Exchange − node*

```
repeat
  Randomly select routes: $m_1$ and $m_2$;
  Randomly select position in $m_1$, node: $i$;
  Compute distance vector of route $m_2$;
  Select position with the greatest distance value, node $j$;
  Compute total demand of both routes after the exchange;
  if New total demand values do not exceed the capacity limit then
    Calculate new distance travelled, exchanging the position of nodes $i$, $j$;
    Calculate the objective function of the new formation;
    if New objective function value is smaller than the initial one then
      Update the solution accordingly;
    end if
  end if
until $itermax$ iterations reached
```

After the completion of the local search phase, the best neighbor solution is selected, from all generated over the conducted iterations. According to the original BA scheme, the neighbor solution is accepted following a probability based on the loudness parameter of the initial solution, even when the newly formed one is better. When a solution is accepted and replaces the initial one, the loudness parameter of the respective solution is decreased and the pulse ration is increased. All the above described processes should be followed for every member of the population. Then, the current best fitness value is retrieved. Subsequently, the best fraction of the current population is selected, denoted by W^*, and the solutions included in this elite population are randomly selected with a probability based on their respective pulse ratio value to undergo a second local search phase. The aforementioned phase is enriched with one more heuristic technique, in which a number of permutations of the nodes included into the solution of interested is tested, aiming to obtain a more efficient node sequence, in terms of distance traveled and emissions minimization. Nevertheless, it is possible that over the evolution of the population, following the local search phases, the algorithmic process could not obtain a more effective solution i.e. smaller fitness value. Thus, in the proposed DIBA, a restart procedure is applied, that introduces to the current population, newly generated solutions, that should replace the worst fraction of the current population and by that avoiding the fast converge of the solution method.

10.7 Computational Experiments

This section presents the experimental results of this study, solving the Green-PCVRP, in order to prove the solution quality of the proposed DIBA. The benchmark instances are drawn form the recent literature (Long et al. 2019), while there is a lack of comparison with other publish algorithms, since the research focuses on a proposed mathematical formulation, as a variant of the PCVRP. Thus, the obtained results of DIBA are compared with the solutions of the Gurobi exact solver on the same instances.

Table 10.1 Parameter setting

Parameter	Description	Expression
NP	Population size	80
L	Number of iterations	200
W^*	Elite population size	$0.2 * NP$
A^0	Initial loudness	$rand[0.7, 0.1]$
r^0	Initial pulse ratio	$rand[0.1, 4]$
γ, α	BA constants	0.98
$itermax$	Local search iterations	100
K	restart parameter	$0.1 * NP$

10.7.1 Benchmark Instances

As the PCVRP has not been studied extensively, previous publications used the benchmark instances of the Capacitated Vehicle Routing Problem (CVRP), incorporating a prize value to each customer and a minimum demand served ratio $ratio_Q$, both generated uniformly from [1, 80] or [1, 100] and [0.6, 0.8], respectively. However, recently, Long et al. publish a set of 120 instances for the PCVRPP, accessible at https://github.com/longjianyuGH/PCVRP.git (2019). In this study, we consider a set that consists in total of 120 benchmark problems, as for each of the 24 CVRP instances (group: A, B, E, M), 5 versions were generated, by changing the ratio $ratio_Q$: {0.60, 0.65, 0.70, 0.75, 0.80}. The different variants include problems with number of nodes from 32 to 200, and number of vehicles from 4 to 17. The parameter setting used for the conducted computational experiments is presented in Table 10.1.

10.7.2 Computational Results

In the following, the conducted experimental results on the benchmark instances are presented in Tables 10.2, 10.3, 10.4 and 10.5. For each of the 120 considered instances, the fitness function value of the best solution obtained by the Gurobi solver, is presented, denoted by g_{min}. In addition, the minimum (w_{min}) and average w_{avg} achieved values of the objective function, over five algorithmic executions of the proposed DIBA can be seen in the aforementioned tables. Moreover, the columns labeled as rpe and rpe_{avg} present the average percentage error of the best and the average solution, from the corresponding g_{min}, for each tested instance.

It is important to highlight that the exact solution method is able to find the optimal results within 10 min, with respect to the small instances, i.e. small number of nodes and routes, while regarding the larger ones the solution deviation from the optimal (MIPgap) is significant, while in some cases the solver could not return a feasible solution within 20 min of execution time, as seen in Table 10.5, when the number of nodes ranges between [101, 200] and the number of routes between [7, 17]. Overall,

Table 10.2 Group A: computational results

Instance	Gurobi solver	DIBA			
	g_{min}	w_{min}	rpe (%)	w_{avg}	rpe_{avg} (%)
A-n32-k5-1	4.357E+03	4.396E+03	−0.89	4.410E+03	−1.20
A-n32-k5-2	4.436E+03	4.459E+03	−0.52	4.473E+03	−0.84
A-n32-k5-3	4.337E+03	4.404E+03	−1.55	4.411E+03	−1.72
A-n32-k5-4	4.367E+03	4.410E+03	−0.98	4.415E+03	−1.09
A-n32-k5-5	4.644E+03	4.703E+03	−1.26	4.713E+03	−1.48
A-n37-k6-1	5.419E+03	5.490E+03	−1.30	5.501E+03	−1.50
A-n37-k6-2	5.450E+03	5.520E+03	−1.28	5.525E+03	−1.38
A-n37-k6-3	5.385E+03	5.454E+03	−1.28	5.459E+03	−1.37
A-n37-k6-4	5.428E+03	5.511E+03	−1.53	5.515E+03	−1.61
A-n37-k6-5	5.181E+03	5.239E+03	−1.11	5.241E+03	−1.16
A-n44-k6-1	5.053E+03	5.115E+03	−1.23	5.117E+03	−1.26
A-n44-k6-2	4.884E+03	4.928E+03	−0.91	4.943E+03	−1.22
A-n44-k6-3	5.249E+03	5.316E+03	−1.26	5.328E+03	−1.50
A-n44-k6-4	4.828E+03	4.887E+03	−1.21	4.892E+03	−1.31
A-n44-k6-5	5.032E+03	5.122E+03	−1.80	5.123E+03	−1.83
A-n48-k7-1	5.941E+03	6.008E+03	−1.13	6.009E+03	−1.16
A-n48-k7-2	6.217E+03	6.277E+03	−0.96	6.282E+03	−1.04
A-n48-k7-3	6.074E+03	6.112E+03	−0.61	6.117E+03	−0.69
A-n48-k7-4	6.056E+03	6.103E+03	−0.77	6.108E+03	−0.85
A-n48-k7-5	6.033E+03	6.125E+03	−1.53	6.128E+03	−1.57
A-n53-k7-1	5.948E+03	6.013E+03	−1.08	6.016E+03	−1.13
A-n53-k7-2	5.704E+03	5.790E+03	−1.50	5.795E+03	−1.59
A-n53-k7-3	5.716E+03	5.811E+03	−1.67	5.826E+03	−1.93
A-n53-k7-4	5.553E+03	5.627E+03	−1.34	5.642E+03	−1.61
A-n53-k7-5	6.008E+03	6.065E+03	−0.94	6.075E+03	−1.11
A-n60-k9-1	7.969E+03	8.083E+03	−1.43	8.093E+03	−1.56
A-n60-k9-2	8.031E+03	7.971E+03	0.75	7.976E+03	0.68
A-n60-k9-3	9.923E+03	8.211E+03	17.25	8.219E+03	17.17
A-n60-k9-4	7.801E+03	7.926E+03	−1.60	7.931E+03	−1.66
A-n60-k9-5	8.069E+03	8.157E+03	−1.10	8.173E+03	−1.30
A-n65-k9-1	7.848E+03	7.661E+03	2.39	7.761E+03	1.11
A-n65-k9-2	9.680E+03	7.599E+03	21.50	7.649E+03	20.98
A-n65-k9-3	8.931E+03	7.448E+03	16.61	7.598E+03	14.93
A-n65-k9-4	9.049E+03	7.692E+03	14.99	7.697E+03	14.94
A-n65-k9-5	7.546E+03	7.355E+03	2.53	7.405E+03	1.87

(continued)

Table 10.2 (continued)

Instance	Gurobi solver	DIBA			
	g_{min}	w_{min}	rpe (%)	w_{avg}	rpe_{avg} (%)
A-n69-k9-1	9.410E+03	7.599E+03	19.25	7.634E+03	18.88
A-n69-k9-2	9.785E+03	7.607E+03	22.26	7.647E+03	21.85
A-n69-k9-3	9.450E+03	7.492E+03	20.72	7.542E+03	20.19
A-n69-k9-4	9.877E+03	7.395E+03	25.13	7.410E+03	24.98
A-n69-k9-5	9.819E+03	7.503E+03	23.58	7.511E+03	23.50
A-n80-k10-1	1.099E+04	9.009E+03	18.05	9.059E+03	17.59
A-n80-k10-2	1.121E+04	8.951E+03	20.16	8.976E+03	19.93
A-n80-k10-3	1.102E+04	8.789E+03	20.27	8.814E+03	20.05
A-n80-k10-4	1.130E+04	9.078E+03	19.64	9.118E+03	19.29
A-n80-k10-5	1.097E+04	8.606E+03	21.57	8.621E+03	21.43

the proposed DIBA approach performs efficiently since, allowing 3 min of max execution time, it obtained 58/120 solutions better than the corresponding of the exact solver, and for 62/120 instances resulted in a worst solution. Nevertheless, the average relative percentage error over these 62 instances does not exceed the 0.98%, which taking into account the different allowed execution time of the two utilized methods, is an effective and competitive result.

With respect to the instance groups A, B and E, for which a comparison among the proposed DIBA and the exact solver can be made, the experimental results are summarized as follows, while the main attribute of the DIBA behaviour is that performs better over the exact solver for instances with more than 60 number of nodes N:

- Group A: with number of nodes $N \in \{32, 37, 44, 48, 53, 60, 65, 69, 80\}$ and number of routes $M \in \{5, 6, 7, 9\}$, the DIBA performed worst with -1.21% and better with 14.94% average deviation from the Gurobi exact solver.
- Group B: with number of nodes $N \in \{39, 41, 50, 56, 63, 78\}$ and number of routes $M \in \{5, 6, 7, 10\}$, the DIBA performed worst with -0.52% and better with 16.81% average deviation from the Gurobi exact solver.
- Group E: with number of nodes $N \in \{23, 33, 51, 76, 101\}$ and number of routes $M \in \{3, 4, 5, 10, 14\}$, the DIBA performed worst with -1.20% and better with 7.14% average deviation from the Gurobi exact solver.

10.8 Conclusions

The presented research introduces the Green-Prize Collecting Vehicle Routing Problem, to optimize the resource of firetrucks, allocation in case of a large fire, where

Table 10.3 Group B: computational results

Instance	Gurobi Solver	DIBA			
	g_{min}	w_{min}	rpe (%)	w_{avg}	rpe_{avg}
B-n39-k5-1	4.015E+03	4.027E+03	−0.30	4.042E+03	−0.67
B-n39-k5-2	4.085E+03	4.112E+03	−0.64	4.114E+03	−0.69
B-n39-k5-3	3.975E+03	4.001E+03	−0.65	4.011E+03	−0.90
B-n39-k5-4	4.058E+03	4.065E+03	−0.17	4.070E+03	−0.29
B-n39-k5-5	3.944E+03	3.969E+03	−0.62	3.984E+03	−1.00
B-n41-k6-1	5.100E+03	5.112E+03	−0.24	5.127E+03	−0.54
B-n41-k6-2	5.090E+03	5.132E+03	−0.83	5.182E+03	−1.81
B-n41-k6-3	4.914E+03	4.919E+03	−0.09	4.924E+03	−0.19
B-n41-k6-4	4.807E+03	4.821E+03	−0.29	4.851E+03	−0.91
B-n41-k6-5	5.009E+03	5.014E+03	−0.11	5.064E+03	−1.10
B-n50-k7-1	6.054E+03	6.060E+03	−0.10	6.079E+03	−0.41
B-n50-k7-2	5.509E+03	5.528E+03	−0.34	5.538E+03	−0.53
B-n50-k7-3	5.718E+03	5.787E+03	−1.20	5.807E+03	−1.55
B-n50-k7-4	5.747E+03	5.780E+03	−0.58	5.790E+03	−0.74
B-n50-k7-5	5.815E+03	5.823E+03	−0.14	5.838E+03	−0.40
B-n56-k7-1	5.311E+03	5.352E+03	−0.76	5.357E+03	−0.86
B-n56-k7-2	5.317E+03	5.350E+03	−0.62	5.395E+03	−1.47
B-n56-k7-3	5.289E+03	5.366E+03	−1.45	5.371E+03	−1.55
B-n56-k7-4	5.478E+03	5.518E+03	−0.73	5.549E+03	−1.31
B-n56-k7-5	5.468E+03	5.500E+03	−0.5	5.570E+03	−1.87
B-n63-k10-1	9.320E+03	9.170E+03	1.61	9.210E+03	1.18
B-n63-k10-2	9.287E+03	9.003E+03	3.06	9.053E+03	2.52
B-n63-k10-3	1.104E+04	9.415E+03	14.70	9.465E+03	14.25
B-n63-k10-4	1.088E+04	9.096E+03	16.36	9.146E+03	15.90
B-n63-k10-5	1.022E+04	9.436E+03	7.65	9.446E+03	7.55
B-n78-k10-1	1.092E+04	7.892E+03	27.72	7.895E+03	27.69
B-n78-k10-2	1.070E+04	8.037E+03	24.90	8.087E+03	24.43
B-n78-k10-3	1.104E+04	8.357E+03	24.31	8.362E+03	24.27
B-n78-k10-4	1.059E+04	8.069E+03	23.84	8.104E+03	23.51
B-n78-k10-5	1.109E+04	8.440E+03	23.92	8.465E+03	23.70

multiple points/assets need to be protected. The problem includes selective attributes, capacity constraints and concerns a predefined number of vehicles. The objective of the Green-PCVRP, is the maximization of the collected prize from the assets protected and the cost-efficiency in terms of fuel-consumption emissions of the operating vehicles, based on the formulated dispatch plan. Considering the optimization of the aforementioned problem, the Discrete Inspired Bat Algorithm has been developed, as

Table 10.4 Group E: computational results

Instance	Gurobi solver	DIBA			
	g_{min}	w_{min}	rpe (%)	w_{avg}	rpe_{avg} (%)
E-n23-k3-1	2.597E+03	2.593E+03	0.15	2.613E+03	−0.62
E-n23-k3-2	2.670E+03	2.693E+03	−0.87	2.713E+03	−1.60
E-n23-k3-3	2.659E+03	2.707E+03	−1.80	2.752E+03	−3.49
E-n23-k3-4	2.490E+03	2.515E+03	−1.00	2.535E+03	−1.81
E-n23-k3-5	2.509E+03	2.544E+03	−1.38	2.547E+03	−1.50
E-n33-k4-1	3.751E+03	3.820E+03	−1.85	3.855E+03	−2.79
E-n33-k4-2	3.491E+03	3.479E+03	0.35	3.489E+03	0.07
E-n33-k4-3	3.377E+03	3.416E+03	−1.14	3.466E+03	−2.62
E-n33-k4-4	3.396E+03	3.443E+03	−1.38	3.493E+03	−2.85
E-n33-k4-5	3.347E+03	3.413E+03	−1.99	3.513E+03	−4.98
E-n51-k5-1	3.354E+03	3.382E+03	−0.84	3.383E+03	−0.88
E-n51-k5-2	3.503E+03	3.541E+03	−1.09	3.541E+03	−1.09
E-n51-k5-3	3.439E+03	3.473E+03	−0.99	3.508E+03	−2.01
E-n51-k5-4	3.416E+03	3.457E+03	−1.20	3.482E+03	−1.93
E-n51-k5-5	3.540E+03	3.589E+03	−1.37	3.596E+03	−1.57
E-n76-k10-1	1.002E+04	8.135E+03	18.84	8.165E+03	18.54
E-n76-k10-2	9.686E+03	7.710E+03	20.40	7.710E+03	20.40
E-n76-k10-3	9.447E+03	7.720E+03	18.28	7.720E+03	18.28
E-n76-k10-4	9.982E+03	7.811E+03	21.75	7.811E+03	21.75
E-n76-k10-5	1.004E+04	7.935E+03	20.93	7.935E+03	20.93
E-n101-k14-1	1.336E+04	1.097E+04	17.92	1.147E+04	14.17
E-n101-k14-2	1.399E+04	1.143E+04	18.33	1.168E+04	16.54
E-n101-k14-3	1.390E+04	1.109E+04	20.21	1.154E+04	16.97
E-n101-k14-4	1.356E+04	1.095E+04	19.21	1.095E+04	19.21
E-n101-k14-5	1.401E+04	1.134E+04	19.00	1.134E+04	19.00

a hybridization of the original BA, omitting only the frequency parameter and altering the original algorithmic scheme by enhancing the local search phase and introducing new solution in the population. To demonstrate the effective performance of the proposed DIBA, computational experiments have been conducted, in comparison with the results of a commercial exact solver.

To further explore the capabilities of DIBA, as future research, we propose the exploration of the Green-PCVRP-NP, where the number of vehicles to be utilized is not a specified value. Finally, in order to enrich the capabilities of the mathematical formulation, by capturing real-life circumstances, time windows and service time for each asset should be taken into consideration.

Table 10.5 Group M: computational results

Instance	Gurobi solver	DIBA			
	g_{min}	w_{min}	rpe	w_{avg}	rpe_{avg}
M-n101-k10-1	–	7.043E+03	–	7.074E+03	–
M-n101-k10-2	–	7.211E+03	–	7.238E+03	–
M-n101-k10-3	–	7.167E+03	–	7.167E+03	–
M-n101-k10-4	–	7.006E+03	–	7.075E+03	–
M-n101-k10-5	–	7.072E+03	–	7.089E+03	–
M-n121-k7-1	–	3.195E+03	–	3.196E+03	–
M-n121-k7-2	–	3.483E+03	–	3.495E+03	–
M-n121-k7-3	–	3.391E+03	–	3.397E+03	–
M-n121-k7-4	–	3.151E+03	–	3.156E+03	–
M-n121-k7-5	–	3.346E+03	–	3.349E+03	–
M-n151-k12-1	–	7.573E+03	–	7.585E+03	–
M-n151-k12-2	–	7.013E+03	–	7.039E+03	–
M-n151-k12-3	–	7.360E+03	–	7.365E+03	–
M-n151-k12-4	–	6.974E+03	–	6.989E+03	–
M-n151-k12-5	–	6.726E+03	–	6.738E+03	–
M-n200-k17-1	–	6.726E+03	–	8.693E+03	–
M-n200-k17-2	–	1.094E+04	–	1.095E+04	–
M-n200-k17-3	–	1.045E+04	–	1.045E+04	–
M-n200-k17-4	–	1.030E+04	–	1.031E+04	–
M-n200-k17-5	–	1.044E+04	–	1.045E+04	–

Acknowledgements This research is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning" in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research" (MIS-5000432), implemented by the State Scholarships Foundation (IKY).

References

El Bouzekri, E. I. A., Elhassania, M., & Alaoui, A. E. H. (2013). A hybrid ant colony system for green capacitated vehicle routing problem in sustainable transport. *Journal of Theoretical and Applied Information Technology, 54*(2), 198–208.

Jia, S. J., Yi, J., Yang, G. K., Du, B., & Zhu, J. (2013). A multi-objective optimisation algorithm for the hot rolling batch scheduling problem. *International Journal of Production Research, 51*(3), 667–681.

Li, K., & Tian, H. (2016). A two-level self-adaptive variable neighborhood search algorithm for the prize-collecting vehicle routing problem. *Applied Soft Computing, 43,* 469–479.

Long, J., Sun, Z., Pardalos, P. M., Hong, Y., Zhang, S., & Li, C. (2019). A hybrid multi-objective genetic local search algorithm for the prize-collecting vehicle routing problem. *Information Sciences, 478,* 40–61.

Martell, D. L. (2007). Fifty years of OR in forestry preface to the special forestry issue of INFOR. *INFOR: Information Systems and Operational Research*, *45*(1), 5–7.

Osaba, E., Carballedo, R., Yang, X. S., Fister Jr, I., Lopez-Garcia, P., & Del Ser, J. (2018). On efficiently solving the vehicle routing problem with time windows using the bat algorithm with random reinsertion operators. In *Nature-Inspired Algorithms and Applied Optimization* (pp. 69–89). Cham: Springer.

Osaba, E., Yang, X. S., Fister, I., Jr., Del Ser, J., Lopez-Garcia, P., & Vazquez-Pardavila, A. J. (2019). A discrete and improved bat algorithm for solving a medical goods distribution problem with pharmacological waste collection. *Swarm and Evolutionary Computation, 44,* 273–286.

Osaba, E., Yang, X. S., Diaz, F., Lopez-Garcia, P., & Carballedo, R. (2016). An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems. *Engineering Applications of Artificial Intelligence, 48,* 59–71.

Roozbeh, I., Ozlen, M., & Hearne, J. W. (2018). An adaptive large neighbourhood search for asset protection during escaped wildfires. *Computers & Operations Research, 97,* 125–134.

Saji, Y., & Riffi, M. E. (2016). A novel discrete bat algorithm for solving the travelling salesman problem. *Neural Computing and Applications, 27*(7), 1853–1866.

Taha, A., Hachimi, M., & Moudden, A. (2015). Adapted bat algorithm for capacitated vehicle routing problem. *International Review on Computers and Software (IRECOS), 10*(6), 610–619.

Taha, A., Hachimi, M., & Moudden, A. (2017). A discrete Bat Algorithm for the vehicle routing problem with time windows. In *2017 International Colloquium on Logistics and Supply Chain Management (LOGISTIQUA)* (pp. 65–70). IEEE.

Tang, L., & Wang, X. (2006). Iterated local search algorithm based on very large-scale neighborhood for prize-collecting vehicle routing problem. *The International Journal of Advanced Manufacturing Technology, 29*(11–12), 1246–1258.

Tian, G., Ren, Y., & Zhou, M. (2016). Dual-objective scheduling of rescue vehicles to distinguish forest fires via differential evolution and particle swarm optimization combined algorithm. *IEEE Transactions on Intelligent Transportation Systems, 17*(11), 3009–3021.

Tiwari, A., Chang, P. C., Elangovan, G., & Annadurai, S. P. (2015). A hybrid edge recombination approach to solve price collecting vehicle routing problem. In *2015 International Conference on Control, Automation and Robotics (ICCAR)* (pp. 200–203). IEEE.

Van Der Merwe, M., Minas, J., Ozlen, M., & Hearne, J. (2014). *The cooperative orienteering problem with time windows*. Optimization-online.org.

Van der Merwe, M., Minas, J. P., Ozlen, M., & Hearne, J. W. (2014b). A mixed integer programming approach for asset protection during escaped wildfires. *Canadian Journal of Forest Research, 45*(4), 444–451.

Wu, P., Cheng, J., & Feng, C. (2019). Resource-constrained emergency scheduling for forest fires with priority areas: An efficient integer-programming approach. *IEEJ Transactions on Electrical and Electronic Engineering, 14*(2), 261–270.

Wu, P., Chu, F., Che, A., & Zhou, M. (2017). Bi-objective scheduling of fire engines for fighting forest fires: New optimization approaches. *IEEE Transactions on Intelligent Transportation Systems, 19*(4), 1140–1151.

Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)* (pp. 65–74). Berlin, Heidelberg: Springer.

Zhang, T., Chaovalitwongse, W. A., Zhang, Y. J., & Pardalos, P. M. (2009). The hot-rolling batch scheduling method based on the prize collecting vehicle routing problem. *Journal of Industrial and Management Optimization, 5*(4), 749–765.

Zhou, Y., Luo, Q., Xie, J., & Zheng, H. (2016). A hybrid bat algorithm with path relinking for the capacitated vehicle routing problem. In *Metaheuristics and Optimization in Civil Engineering* (pp. 255–276). Cham: Springer.