Biased Random-key Genetic Algorithm for the Hybrid Vehicle-drone Routing Problem for Pick-up and Delivery

1st Anderson Zudio
Institute of Computing
Fluminense Federal University
Niterói, Brazil
azudio@id.uff.br

2nd Igor Machado Coelho
Institute of Computing
Fluminense Federal University
Niterói, Brazil
imcoelho@ic.uff.br

3rd Luiz Satoru Ochi
Institute of Computing
Fluminense Federal University
Niterói, Brazil
satoru@ic.uff.br

***Abstract*—The Hybrid Vehicle drone Routing Problem (HVDRP) was recently introduced as an extension of the classic Vehicle Routing Problem (VRP). In this version, one vehicle is equipped with multiple drones to serve customers with demands for pick-up and delivery. The vehicle travels between stations that serve as parking locations to dispatch drones to attend clients. The drones have limitations in their maximum flight range and carrying capacity. We propose a BRKGA algorithm to solve HVDRP with a decoder component specially tailored to find feasible solutions. The proposed method is empirically analyzed in solution quality through a test set that a mixed-integer programming (MIP) model implementation can optimally solve in reasonable computation time. The computational result shows that the best solution found by BRKGA for each instance of the test set matches the solution quality devised by the MIP implementation. The data also show that the proposed algorithm achieves the best solution consistently through many independent executions. The instance set used and its respective best solutions attained for this work are publicly available.**

***Index Terms*—BRKGA, HVDRP, Metaheuristic**

I. Introduction

The Hybrid Vehicle drone Routing Problem (HVDRP) is an extension of the classic well-known Vehicle Routing Problem (VRP) [11] that belongs to the class of NP-Hard problems [12]. The HVDRP works with a vehicle equipped with multiple drones to serve multiple customers that require pick-up and delivery services. An instance of HVDRP is described by a directed graph $G = \{N, A\}$ that details a multimodal network, a set D that indexes drones by their maximum range r and carrying capacity w, and a set C that defines each client through the pick-up p and deliver q weight of its packages. The set of nodes $N = \{N_d \cup N_v \cup N_c\}$ includes a starting depot N_d, a set of stations N_v that can only be accessed by the vehicle, and a set N_c of clients nodes that drones can only serve. The set $A = (i, j)$ of links, $i, j \in N$, describes the distance between each point and the average travel cost for the vehicle and drones. The objective of HVDRP is to minimize the total cost of the vehicle and the drones to serve all customers.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

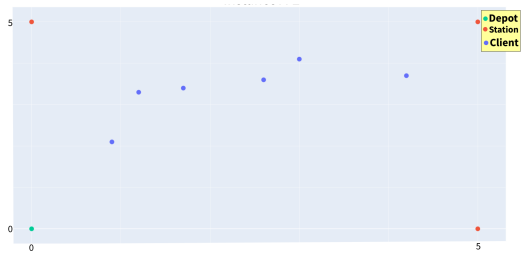
The vehicles only serve as a mobile depot for the drones. Starting from the depot N_d, the vehicle can only visit each station once. Drones can only be dispatched from stations when the vehicle is parked. Each drone can serve multiple customers per dispatch as long as its flight range and load-carrying capacity permit. Drones can return to any station along the vehicle route, which could be the same as or different from the dispatching one. Multiple drones can be dispatched simultaneously from any station. Drones can be dispatched and collected several times from the same station, their batteries are replaced with fully charged ones, and packages are loaded and unloaded each time they are collected. The vehicle cannot move from a station before collecting all drones planned to return to that station. Finally, drones arriving early at a collection station are assumed to wait for the vehicle in idle conditions before being assigned a new tour.

To further summarize the problem, the following list properties reviews the previous paragraph:

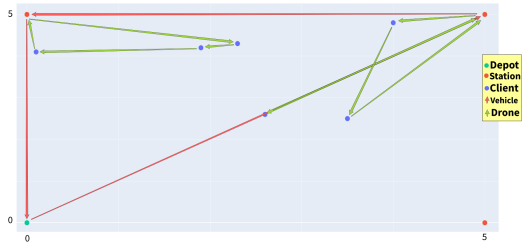
1) The customers are described by their locations, also their delivery and pick-up demands. Each package has an associated weight. Only the drones can serve them.
2) The drones can only be dispatched from stations. They can be dispatched multiple times from the same station. Different drones may be dispatched simultaneously.
3) The vehicle can only travel to stations visiting them only once. It is not required to visit all stations.
4) Each drone is described by its carrying capacity and maximum flight range. Every drone may visit multiple clients by dispatch, but the tour must respect the limits of the drone.
5) Drones can return to any station as long as the vehicle is parked there or it will still be visited at a further point.
6) When the vehicle picks the drone, its battery is immediately replaced and the goods are properly manipulated for the next delivery and stocking.
7) The drone awaits for the vehicle and vice versa when necessary. Consequently, the vehicle can not leave a drone behind.

The Figures 1a, 1b, 1c, 1d illustrates two instances and two possible solutions of HVDRP, where each red vertex is a station for the vehicle, each blue vertex represents a customer, and the green point is the depot. In the solution, the path denoted by the red arrows represents the vehicle tour, and the green arrows the drone visitation path. It is worth noting that the examples do not use the property that the drones can return to stations ahead of the vehicle.

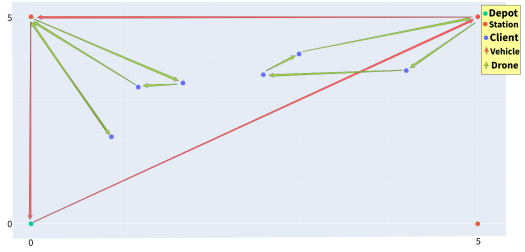
(a) Instance A-1 of HVDRP



(b) Instance A-2 of HVDRP



(c) Solution example for A-1



(d) Solution example for A-2

With the evolution of drones, they can be deployed to perform several tasks related to healthcare [13], public safety [14], forest preservation [15], and transportation [16]. Particularly, drone usage for logistics that involves product delivery has raised substantially [20], [18], and [19]. This tendency will continue to grow significantly due to several contributing factors: the continued expansion of online retail, high competition for pick-up and delivery service providers, and drones' improved capabilities and cost-effectiveness. The HVDRP provides a problem model that differs from other systems that use a single drone, like Flying Sidekick Traveling Salesman Problem [17], and the Two-Echelon Vehicle Routing Problem [21], [22], [23].

Several industrial applications can take advantage of such a model. In this context, [3] and [24] describe the HVDRP as a model to study use cases that involve a single vehicle with multiple drones system at a high realism. The latter work shows different heuristics to approach the HVDRP and provides a mixed-integer programming (MIP) model to solve small instances.

Even with the progress of techniques to optimal solutions and diffusion of computational environments capable of high parallelization power, it is not easy to efficiently solve practical sized HVDRP instances. Therefore, this work proposes a BRKGA algorithm for HVDRP that uses an encoded solution divided by four sets of random keys. The decoder process uses each set of random keys to dictate which order of stations the vehicle will visit, the sequence in which clients will be served, choose what drone will visit which client, and decide what customers will be served from each station. An implementation of the proposed BRKGA was used in an empirical test with ten instances of HVDRP. To evaluate each devised solution of the proposed algorithm, a second application that uses the MIP model from [3] was submitted to the same test instance data to generate the optimal solution. The empirical result shows that the best solution found by BRKGA for HVDRP matches the quality of the optimal cost found by the MIP. The computational results also show that the average solution quality attained by BRKGA stays consistent with the best through many independent executions of the method in relatively low computation time.

The remainder of this work is structured as follows: Section II describes the BRKGA metaheuristic briefly and its specialization for HVDRP, Section III reports the computational results, and Section IV concludes the work.

II. BRKGA Algorithm

Biased Random-Key Genetic Algorithms is an extension of the evolutionary algorithm presented by Bean J. C [6] that was introduced by [1] and [2]. In [5] Gonçalves and Resende formally described the basic concepts of BRKGA alongside a survey of applications of that time. Since them, a considerable number of state-of-the-art applications for optimization problems used BRKGA [7] [8] [9], [10]. The BRKGA metaheuristic was chosen for HVDRP due to the ease of implementation and because it only has one problem dependant component. This section briefly describes the BRKGA metaheuristic and details the decoder component for the HVDRP.

A. BRKGA Overview

The main idea of BRKGA is to work with only one problem-dependent component named *decoder*. Every component works with solutions represented as collections of real numbers between zero and one called *random-keys*. One of the main advantages of BRKGA is that the user only has to implement the decoder to transform random-keys into an actual solution representation that can be evaluated.

Similar to an evolutionary algorithm framework, BRKGA uses a calibratable parameter P_{max} to dictate the maximum number of individuals that compose a population, and the stopping criteria can be a maximum target generation, a quality target measure for the best solution found, or a time limit. The size of each random-key collection used to represent a solution depends on the decoder devised by the user. The first generation starts with an initial population composed of random individuals. BRKGA works with elitism through another calibratable parameter $\frac{1}{2} < \epsilon < 1$ called TOP. In each generation, the population is sorted based on the fittest. In the initial population and through the evolutionary process of the method, the $100\epsilon\%$ top individuals compose the elite set. The rank procedure makes use of the decoder to define the fitness of each individual. After ranking, the evolutionary process starts.

After the first, the following generations of BRKGA start with a copy procedure, where each elite is maintained with no changes in a new population. Then a mutation step is applied so that introduces $100\omega\%$ random individuals in this new population, where $0 < \omega < \frac{1}{2}$ is a calibratable parameter called BOT. Next, a reproduction step generates the remaining individuals as offspring through crossover. The crossover works by electing two random parents in the previous population with one elite and one non-elite. For each random-key of the offspring, the probability of copying the respective random-key of the elite parent is $100\rho\%$. Otherwise, the respective random-key from the non-elite parent is copied instead. Thus, justifying the method's name with an analogy to a biased coin toss where each elite has a higher probability of being used multiple times compared to non-elite ones, and its random-keys are propagated with a higher probability. The method stops its evolutionary process when the stopping criterion set by the user is met.

B. Decoder

Each encoded solution S_e of an HVDRP instance should have $l = s+3c$ random-keys, where s is the number of stations for the vehicle without the depot and c the number of clients. Thus, each encoded solution is composed of four parts.

- The first part is called *station sequence* $SS = \{r_1, \cdots, r_s\}$ that have the first s random keys of S. It is used to determine in what order the vehicle will visit the stations to dispatch drones.
- The second part is called *client sequence* $CS = \{r_{s+1}, \cdots, r_{s+c}\}$ that has one random key for each client. These random-keys will be used to decide in what order the clients should be served from each visited station.
- The third part is called *drone decider* $DD = \{r_{s+c+1}, \cdots, r_{s+2c}\}$, which is used to determine what drone should serve each client.
- The final part, called *station decider*, is composed of the remaining $SD = \{r_{s+2c+1}, \cdots, r_l\}$ random-keys. The decoder uses those random keys to dictate from what station each client should be served from.

The decoder starts with a sorting procedure using the station sequence. Each station $1 \leq i \leq s$ has a corresponding random key $r_i \in SS$. The station $1 \leq j \neq i \leq s$ will be visited before the station i if $r_j < r_i$. Therefore, the ascending sort of SS will give a permutation P_s of stations to visit.

The decoder will construct a new solution for the actual instance iterating over P_e. Each iteration begins with a vehicle movement to the next station of P_e, where the first is the vehicle moving to the first station of P_e starting from the depot. The next step is a procedure that sends the drones to each client to be served from the actual park location of the vehicle. The decoder decides what clients will receive a drone from the current location based on the random keys of SD. Each station has a non-overlapping interval in $[0, 1]$ that is inversely proportional to s, giving an equal chance to be the base location to serve each client. Therefore, the k-th client will be served from the vehicle's current parking location if its respective random-key $r_k \in SD$ matches its interval. In this manner, we have a selection of clients to serve in each iteration of the decoder. The decoder uses a similar procedure to decide which drone to assign to each client, using the random keys of DD instead. The idea is that each station and drone pair will have an equal chance of serving a client.

The send drone procedure will iterate over the list of clients to be served from the current vehicle park location. The order of customer service is based on the random keys of CS. The client i of the list will be visited before the client j if its respective $r_i \in CS$ is smaller than the $r_j \in CS$ random-key, $1 \leq i \neq j \leq c$. If the serving drone is on the vehicle, it will move directly to the client. Otherwise, the method uses three checks described below.

1) A check to see if the drone can be sent from its current client location to the next and back to the vehicle without reaching its flight limit.
2) A check to see if the drone's capacity can support the pickup package from the next client.
3) A check to see if the drone's capacity can support the sum of delivery package weight when going out of the vehicle.

If any of the checks above has a negative response, the drone is sent back to the vehicle before going to the next customer. Once every client of the current list is visited, the procedure sends back to the vehicle every drone located at some client, and the procedure finishes. The vehicle will only move again if every drone is back at its location.

Once all clients have been served, the vehicle goes back

to the starting depot, thus not needing to visit any remaining stations that do not have any clients assigned. One may note that the decoder may generate infeasible solutions that may have at least one drone path with a flight distance beyond its limit or has an accumulation of package weight that surpasses its capacity. In these cases, the decoder applies a penalty using a high positive constant value M during the fitness measurement of the individual proportional to the number of infeasible paths. The idea is that any infeasible solution has a lower fitness evaluation than any feasible one, scoring low on the ranking step of BRKGA.

The Algorithm 1 summarizes the decoder, where the function $sort$ returns the permutation P_s of stations based on $SS \subset S$, $list\_client$ gives the list of clients to visit in the current station based on $SD \subset S$, and $drone\_to\_client$ dictates which drone should go to c based on $DD \subset S$.

Algorithm 1 HVDRP Decoder

Require: Encoded solution $S_e = \{SS \cup CS \cup DD \cup SD\}$
  P_s = sort(SS)
  The vehicle starts at the depot
  for Each element $s \in P_s$ **do**
    Move the vehicle to S
    L = list_client(s, SD)
    for Each $c \in L$ sorted by CS **do**
      d = drone_to_client(c, DD)
      if d is not in the vehicle and can not move to c **then**
        Move the drone to the vehicle
      else
        Move d to c
      end if
    end for
    for Every drone d that is not in the vehicle **do**
      Move the drone to the vehicle
    end for
    if All clients were served **then**
      Move the vehicle back to the depot
      End the loop
    end if
  end for
Ensure: all the movements are registered to output S.

III. Computational Results

This section shows the results attained with BRKGA for HVDRP using the decoder detailed in Section II-B. To the best knowledge of the authors, the work [3] is the only one that proposes methods to solve HVDRP. Unfortunately, the original authors the cited work does not provide the instances used or any implementation of the methods[1]. Therefore, an implementation of the MIP detailed in [3] and a set of ten instances was used to assess the solution quality devised by the proposed algorithm[2]. The cited work classifies seven categories identified by the letters A to G with ten instances each. The test set of ten instances used in this work was generated based on a similar setup originally classified as *A*.

The ten instances of the set "*A*" are organized as a grid of size five in a euclidean plane. The depot is at the origin, and each vehicle station is positioned at the intersections of the grid. The clients are randomly located around each station, with pickup and delivery varying in $[0, 5]$ intervals. Each instance has two drones to serve the customers with a maximum flight range of seven units and a capacity of ten. The vehicle cost for transversing an arc is the length itself, and the drone's flight cost is two times the distance of the arc.

The BRKGA for HVDRP implementation was made in *C++17* and compiled with *GCC* from the *GNU Compiler collection* using the flag *-O3*. This application uses the Mersenne Twister algorithm to generate the pseudorandom numbers. The MIP model was implemented using the *C++* API of Gurobi version 9.1.2. The computational environment used to devise all the results has an Intel Core i7-10700f @2.9 GHz CPU, 32 GB of RAM, and Ubuntu 20.4 LTC (x64) OS.

The BRKGA application was executed 30 times independently for each instance using distinct seeds for the pseudorandom engine in the interval $[1, 30]$ to control the test environment. The metaheuristic parameters used for all runs are as follows: maximum population size $P_{max} = 100$, TOP $\epsilon = 0.2$, BOT $\omega = 0.15$, and the elitism rate $\rho = 0.7$. The stopping criteria is set to stop at 5 seconds to return the best solution found during the evolving process. The second application, which solves the MIP model using the Gurobi optimizer, was executed only once for each instance with no time limit.

Table I shows the result of this test battery. The first column has the instance name. The second and third column has the optimal cost found for the objective function of each instance using the Gurobi application and the total CPU execution time. The third and fourth columns show the best and the average cost found with BRKGA. Finally, the last column shows the execution time to find the best solution. The Gurobi application used 16 threads of the target CPU, and the BRKGA implementation used only one thread.

The average versus the best solution found by the BRKGA application shows that the method described by this work consistently finds the best solution for each tested instance through independent runs. The data also shows that the best solution found equals the optimal cost found by the solver, suggesting that the devised algorithm achieves the optimal

[1]The authors of this work contacted the original author of [3] to no avail.

[2]The instances used in this work and the best solution found are publicly available at https://github.com/AndersonZM/hvdrp.

TABLE I: Best and average solution cost found by BRKGA compared to the optimal cost found by the MIP.

Instance	MIP		BRKGA		
	Z	Execution time (s)	Z_{Best}	Z_{Avg}	Time to best (s)
A-1	54.3	100	54.3	54.3	<1
A-2	48.1	92	48.1	48.1	<1
A-3	42.9	4	42.9	42.9	<1
A-4	46.3	81	46.3	46.4	<1
A-5	41.7	12	41.7	41.9	<1
A-6	54.8	1129	54.8	55.1	<1
A-7	65.7	10887	65.7	66.1	<1
A-8	58.8	1077	58.8	59.2	<1
A-9	60.8	699	60.8	61.3	<1
A-10	83.1	4890	83.1	85.9	<1

solution for the test set. The data also shows that the BRKGA implementation attains the best solution in a relatively short execution time than the MIP implementation.

IV. Conclusion

We proposed a BRKGA method to solve the Hybrid Vehicle-drone Routing Problem for Pick-up and Delivery, which was recently introduced by [3]. The proposed decoder receives encoded solutions divided into four parts. Each piece decides what order the vehicle should visit the stations, what customer sequence should be used, what drone should serve which client, and what location each drone should be sent from. The decoder uses a penalty system to ensure that infeasible solutions score lower than feasible ones at the ranking procedure of BRKGA. An implementation of the proposed BRKGA was compared against a lower bound attained by solving the MIP model detailed in [3] using Gurobi. The computational results show that the proposed BRKGA is capable of finding the best solution consistently for the testing instances across many executions with a small time limit. The data also shows the best solution cost found by the BRKGA equals the optimal solution found by the solver, suggesting that the proposed method works well with the instance set used in this work. Finally, the data also shows that the BRKGA attains the best solution in a relatively short execution time than the MIP implementation.

For future works, we aim to extend results and the test set to use all ten instances of each setup detailed by [3]. We also intend to propose a novel algorithm that makes use of local search heuristics. Finally, we strive to improve further the algorithm described in this work.

Acknowledgment

The computational results presented in this work were attained using the resources of Laboratório de Inteligência Computacional (LABIC) of the Institute of Computing of Federal Fluminense University (IC-UFF). The concepts used in each implementation used for this work was based on the OptFrame[3] optimization framework open source code [4].

[3]OptFrame is available at https://github.com/optframe/optframe

References

[1] Ericsson, M, Resende, Mauricio GC, and Pardalos, P. M., "A genetic algorithm for the weight setting problem in OSPF routing," Journal of combinatorial optimization, ed. Springer, vol. 6, pp. 299–333, 2002.

[2] Gonçalves, José Fernando and Almeida J. R., "A hybrid genetic algorithm for assembly line balancing" Journal of heuristics, ed. Springer, vol. 8, n. 6, pp. 629–642, 2002.

[3] Karak, A. and Abdelghany, K., "The hybrid vehicle-drone routing problem for pick-up and delivery services," Transportation Research Part C: Emerging Technologies, pub. Elsevier, vol. 102, pp 427–449, 2019.

[4] Coelho I. M. ; Coelho V. N. ; Zudio A.; Araujo R. P. et al. "Microbenchmark Studies in OptFrame: a 10-Year Anniversary." In: ANAIS DO LII SIMPÒSIO BRASILEIRO DE PESQUISA OPERACIONAL, João Pessoa, Campinas, ed. Galoá, 2020.

[5] Gonçalves, J. F. and Resende, M., "Biased random-key genetic algorithms for combinatorial optimization," Journal of Heuristics, ed. Springer, vol. 17, n. 15, pp 487–525, october 2011.

[6] Bean, J. C, "Genetic Algorithms and Random Keys for Sequencing and Optimization, " ORSA Journal on computing, pub. INFORMS, vol. 6, n. 2, pp 154–160, january 1994.

[7] Fadel, A. C., Ochi, L. S., Moura B. J. A., and Semaan, G. S., "Microaggregation heuristic applied to statistical disclosure control", Information Sciences, pub. Elsevier, vol. 548, pp 37–55, 2021.

[8] Andrade, C. E, Toso, R. F., Gonçalves, J. F. and Resende, M. GC, "The Multi-Parent Biased Random-Key Genetic Algorithm with Implicit Path-Relinking and its real-world applications," European Journal of Operational Research, pub. Elsevier, 2019.

[9] Pinto, B. Q., Ribeiro, C. C., Rosseti, I., and Plastino, A., "A biased random-key genetic algorithm for the maximum quasi-clique problem," European Journal of Operational Research, pub. Elsevier, vol. 271, n. 3, pp. 849–865, 2018.

[10] Faria, H., Resende, M. GC, and Ernst, D., "A biased random key genetic algorithm applied to the electric distribution network reconfiguration problem," Journal of Heuristics, ed. Springer, vol. 23, n. 6, pp. 533–550, 2017.

[11] Golden, Bruce L and Raghavan, Subramanian and Wasil, Edward A, "The vehicle routing problem: latest advances and new challenges," pub. Springer Science & Business Media, vol. 43, 2008.

[12] Solomon, M. M., "Algorithms for the vehicle routing and scheduling problems with time window constraints," Operations research, pub. Informs, vol. 35, n. 2, pp. 254–265, 1987.

[13] Kim, S. J., Lim, G. J., Cho, J., and Côté, Murray J., "Drone-aided healthcare services for patients with chronic diseases in rural areas," Journal of Intelligent & Robotic Systems, ed. Springer, vol. 88, n. 1, pp 163–180, 2017

[14] Chowdhury, S., Emelogu, A., Marufuzzaman, M., Nurre, S. G., and Bian, L., "Drones for disaster response and relief operations: A continuous approximation model," International Journal of Production Economics, pub. Elsevier, vol. 188, pp 167–184, 2017.

[15] "Assessing biodiversity in forests using very high-resolution images and unmanned aerial vehicles," Getzin, S., Wiegand, K., and Schöning, I., Methods in ecology and evolution, pub. Wiley Online Library, vol. 3, n. 2, pp 397–404, 2012.

[16] Menouar, H., Guvenc, I., Akkaya, K., Uluagac, A. S., Kadri, A., Tuncer, A., "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges," IEEE Communications Magazine, pub. IEEE, vol. 55, n. 3, pp 22–28, 2017.

[17] Murray, C. C., and Chu, A. G., "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," Transportation Research Part C: Emerging Technologies, pub. Elsevier, vol. 54, pp. 86–109, 2015.

[18] Taniguchi, E., Thompson, R. G., and Qureshi, A. G., "Modelling city logistics using recent innovative technologies," Transportation Research Procedia, pub. Elsevier, vol. 46, pp 3–12, 2020.

[19] Moshref-Javadi, M., and Lee, S., and Winkenbach, M., "Design and evaluation of a multi-trip delivery model with truck and drones," Transportation Research Part E: Logistics and Transportation Review, pub. Elsevier, vol. 136, 2020.

[20] Murray, C. C. and Raj, R., "The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones," Transportation Research Part C: Emerging Technologies, pub. Elsevier, vol. 110, pp. 368–398, 2020.

[21] Luo, Z., Liu, Z., and Shi, J. “A two-echelon cooperated routing problem for a ground vehicle and its carried unmanned aerial vehicle,” Sensors, pub. Multidisciplinary Digital Publishing Institute, vol. 17, n. 5, 2017.
[22] Kitjacharoenchai, P., Min, Byung-Cheol, and Lee, S., “Two echelon vehicle routing problem with drones in last mile delivery,” International Journal of Production Economics, pub. Elsevier, vol. 225, 2020.
[23] Liu, Yao and Liu, Zhong and Shi, Jianmai and Wu, Guohua and Pedrycz, Witold, “Two-echelon routing problem for parcel delivery by cooperated truck and drone,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, pub. IEEE, 2020.
[24] Karak, A., “Hybrid Vehicle-drone Routing Problem For Pick-up And Delivery Services Mathematical Formulation And Solution Methodology, ” Civil and Environmental Engineering Theses and Dissertations, n. 9, 2020.