A Hybrid Model based on Genetic Algorithm and Space-Filling Curve applied to Optimization of Vehicle Routes

Warley Rocha MENDES[1], Flávio Garcia PEREIRA[2], Daniel Cruz CAVALIERI[2]
[1]*Companhia Espirito Santense de Saneamento, Av. Governador Bley 29010-150, Vitória-ES, Brazil*
[2]*Instituto Federal do Espirito Santo, Rodovia ES-010 29173-087, Serra-ES, Brazil*
daniel.cavalieri@ifes.edu.br

***Abstract*—This work is the result of a real problem in the Sanitation Company of Espirito Santo (Companhia Espirito Santense de Saneamento), which owns a Geographic Information System, but lacks a mechanism to build routes to server customers that open in average 2148 services requests per day. Therefore, we propose a Hybrid Optimization Algorithm that combines Genetic Algorithm and Space-Filling Curves to solve the Vehicle Route Problem. We establish the validity of the hybrid algorithm by performing tests in two different benchmarks datasets. Our proposal reached an average result of 12.7 percent and 4.1 percent better than the previous solutions in the first and second datasets respectively. Also, we compare our solution and five other variations of Ant Colony Optimization Algorithm. The results show that our proposal is superior in some simulations and, when it was not superior, presented the second-best results for almost all instances.**

***Index Terms*—vehicle routing, genetic algorithms, fractals, hybrid intelligent systems, computer applications.**

I. INTRODUCTION

The Sanitation Company of Espírito Santo (Companhia Espírito Santense de Saneamento or CESAN) faces the following challenges: a) collecting water samples in the specific locations of the 52 cities served by the company to inspect the quality of the water supplied to houses, local markets, and local companies; b) performing an inspection of hydrants, drains, lift units, and registers, to keep usability and operation of such elements; c) to oversee complaints and possible illegal links to the sanitation sewage network all over the cities. All this within the legal deadlines established by the monitoring agencies, also to keep the excellence of the services provided by the Sanitation Company and the satisfaction of its clients. To perform all these works, it is extremely important to build a route plan that efficiently will guide the external service teams to the sequence of locations where will be accomplished the activities through the day.

In the end of 2014, CESAN performed a massive investment and implemented a corporative Geographic Information System (GIS) that represents in the geographical space different kind of information of the Sanitation Company. Nowadays, daily the planners make by hand different sets of routes to visit based on the geographic position of the occurrences and cartographic layers of GIS system. The knowledge and experience of the planners, besides the information from the maps, are the foundation for this process. Specifically, in the case of CESAN the workers work from 8 am to 6 pm and the routes are daily generated at 7:30 am, i. e., 30 minutes in advance. In addition, we need an algorithm that can generate new routes as quickly as possible, due to the quality of certain routes such as traffic jams or road works. Therefore, very large runtime algorithms (such as optimal solution) cannot be used.

Considering this scenario, we proposed to employ the geographic information of CESAN in a graph algorithm based on the Space-Filling Curves theory and combined with a Genetic Algorithm to build automatically minimal cost vehicle routes. In addition to that, we implement other algorithms based on the ant colony theory to compare with our proposal.

Rest of the paper is organized as follows: Some related work is discussed in section II. A brief description about Single-Depot Multiple Traveling Salesman Problem, Space-Filling Curves and Genetic Algorithm is given in Sections III, IV and V, respectively. A new approach has been proposed keeping in mind the advantages and disadvantages of various techniques, in section VI. Section VII provides results and discussion and the paper has been concluded in section VIII.

II. RELATED WORK

For more than a decade, researchers like [1-3] have been proposed strategies to optimization of routes applied to Vehicle Optimization Problem based on Genetic Algorithm and its variations. The work by [4] stands out due to their comparison between three approximation heuristics: the Clark and Wright's heuristic [5], the Mole and Jameson's heuristic [6], and the Genetic Algorithm [7]. In this work, the authors performed several experiments for each heuristic and compared the quality of solutions and execution time. The conclusion was that Genetic Algorithm performed better than the two other heuristics.

Also, some important works indicate that Genetic Algorithm is a powerful solution to vehicle route problem (VRP), like [8] that claims that Genetic Algorithm can solve a wide variety of VRPs and their variations. Another work, [9] proposed an algorithm that uses an optimized crossover operator, which was designed by a complete non-directed bipartite graph. The strategy then finds an optimal set of routes to deliver that satisfies the constraints and present the

Digital Object Identifier 10.4316/AECE.2018.03007

minimum total cost to the Capacitated Vehicle Problem (CVPR). According to the authors, the algorithm was tested using benchmark examples and compared to previous heuristics. The results achieved were competitive.

Employing a different methodology from the previous ones, the works by [10-12] prove, for the first time, a solution consisting of simple dynamic programming for the traveling salesman problem using Space-Filling Curves. To build a short route between points in the plane, the points are followed as they appear along with a space-filling curve. This heuristic consists essentially of filtering, so, it is easily implemented, and its performance is competitive compared to other fast methods.

Next, John J. Bartholdi [13-15] improved his results presenting new works related to the route optimization topic and described a new proposal of application of the Space-Filling Curves in the route vehicle problem. He showed the positive results and the feasibility of his approach. His work inspired and stimulated the use of two techniques in the resolution of automatic generation of routes in a GIS system, which is the theme of this work.

III. Single-Depot Multiple Traveling Salesman Problem

The proposal of mathematical formulation for the Single-Depot Multiple Traveling Salesman Problem (SD-MTSP) is based on applying the syntactic pattern used in the literature [16]. For the single-depot traveling salesman problem, we used an oriented graph representation, $G=(V,A)$, where V is a set of n nodes and A is a set of edges. The graph has a cost matrix of $C=(c_{ij})$ for each edge $(i,j)\in A$. Let x_{ij} be a binary variable that holds 1 if the edge (i,j) is selected as a candidate solution, and 0 otherwise. u_i is the number of visited nodes in a vehicle route from origin/warehouse to a node i, for any salesman, like, for instance, the position of a node i in a routing.

The mathematical formulation of the classic version of the SD-MTSP aims to minimize the total cost/size of the routes and can be defined as integer linear program. Note that this definition of the VRP has some assumptions: a) the node 1 is considered the origin/warehouse; b) the travel of each vehicle should leave from the warehouse and arrive in the warehouse (node 1); c) Each customer should be visited only once. The model of the SD-MTSP can be formulated in the following way:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1}$$

$$\text{s. t.} \sum_{j=2}^{n} x_{1j} = m \tag{2}$$

$$\sum_{j=2}^{n} x_{j1} = m \tag{3}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j=2,\ldots,n \tag{4}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i=2,\ldots,n \tag{5}$$

$$x_{1i}+x_{i1}\le 1, \quad i=2,\ldots,n \tag{6}$$

$$u_i-u_j+(n-m)x_{ij}\le n-m-1, \quad 2\le i\ne j\le n \tag{7}$$

$$x_{ij}\in\{0,1\}, \quad \forall(i,j)\in A \tag{8}$$

The constraints (2) and (3) assure that exactly m salesmen travel and back to the warehouse. The constraints (4) and (5) are the degree constraints, namely, a route will be assigned only to vehicles used in the customer service. The inequality (6) enforces that each salesman visits at least two nodes besides the warehouse. The constraints (7) are typical constraints to sub tours without the origin.

IV. Space-Filling Curves

Space-Filling Curves (SFC) are continuous planar curves without crossed traces used to filling an entire two-dimensional space, as described in [17]. The SFCs were proposed by the Italian mathematician Giuseppe Peano (1890) and, nowadays, due to their properties of spatial discretization are employed in many practical applications such as decomposition algorithms and even solutions for the salesman problem. Inspired by the patterns of growth in nature, such as plant growth and cell growth, SPCs are fractal forms of space filling consisting of continuous curves such that their strokes fill any two-dimensional area (such as a square) or *N*- dimensional (hypercube) in an orderly manner [18].

Although Giuseppe Peano presented for the first time the spatial filled curves, it was the mathematician David Hilbert (1900) who spread this field of geometry. The Hilbert curve is a space-filling curve that visits all points of a square grid [18].

The basic elements of the Hilbert curves are the so-called "cups" (a square with an open side) and "junctions" (a vector that joins two cups). The open side of a glass may be upper, lower, left, or right. In addition, each cup has two ends, and each can be either the entry point or the exit point. So, there are eight possible kinds of cups. In practice, a Hilbert curve uses only four types of cups and has a direction: up, down, left, or right [18].

As shown in Fig. 1, a first-order Hilbert curve is only a single cup (Fig. 1-A) that fills a 2 × 2 space. On the second order curve, this cup is replaced by four smaller ones, which are bound by three junctions (Fig. 1-B). Next, the process is repeated, replacing each cup by four smaller ones interconnected by three new joints (Fig. 1-C).

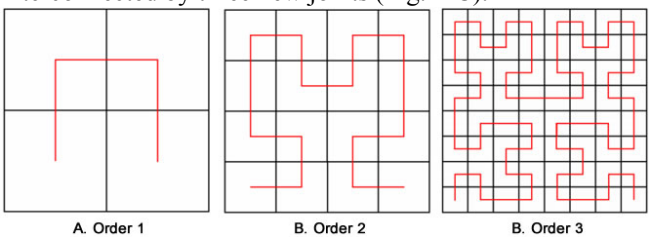

Figure 1. Hilbert curve of different degrees

When Hilbert curves are built, it imposes an order of points in every cell of its matrix and the points are sorted according to the sequence in which the curve visits the cells of the matrix. Thus, the classification of the points is according to the curves formations, where, first, each point

is in a different cell to only establish the order of the points [14]. After the complete formation of the curves in the filling of a certain area, a sequence of the visited cells is generated, consequently allowing the complete mapping of the area.

Fig. 2 shows how the mapping sequence of an area using the space-filling curves proposed by Hilbert is formed. For each level of decomposition, there is a different mapping and the higher the order, the more detailed the mapping level.

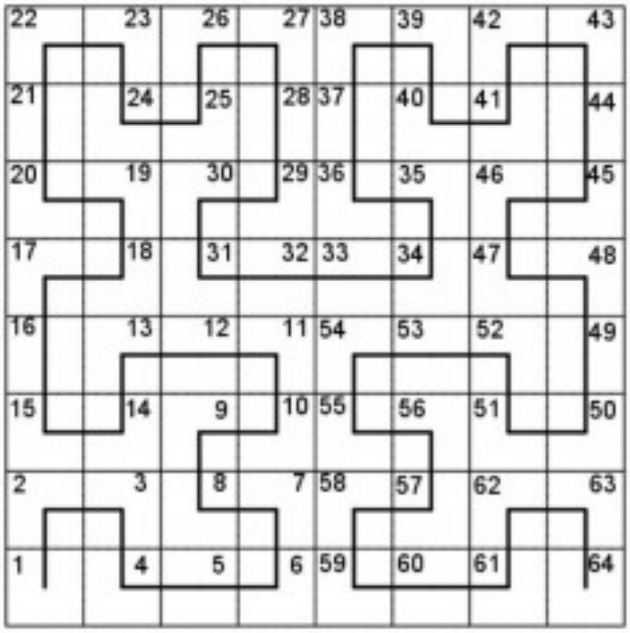

Figure 2. Mapping of the Hilbert curve of order 3

A useful property of a space-filling curve is that it tends to visit all points in a region after it has entered its quadrant. Thus, points that are close to each other in the plane tend to be close to each other in appearance along the curve. Hence, points scattered along an area can be visited in the same sequence as the formation of the space-filling curve. Fig. 3 presents the application of the filling curve proposed by Hilbert for the generation of a service request service route in the city of Vitoria containing 46 points. The result of the algorithm is a route with a cost of 28,740.26 meters.

Figure 3. Georeferenced points in the GIS (1 to 46) and route generated by the sequence of Hilbert curve

As in [13], we can summarize the advantages of using Hilbert Curve for route generation: (i) the algorithm guarantees that all points will be in the produced route, without redundant sub-routes; (ii) the algorithm can be applied in dynamic environments, in which new points are added or removed from the search space and without the need of adjustments in the algorithm; (iii) the algorithm is easy to implement; (iv) the algorithm is fast, namely, it only requires $O(n \log n)$ operations to build a route of n points and only $O(\log n)$ computations to update the solution, adding or removing points.

And the disadvantages: (i) the algorithm is only applied in generation open routes, namely, in routes that the vehicle is unrequired to return to a base, warehouse or starting point; (ii) the algorithm disregards the existing constraints related to routing problems, like time constraints, vehicle capacity or residence time in each customer. It simply generates a sequence of points to visit; (iii) rotating the fractal can produce different results; (iv) the algorithm considers only the points distributions in the plane and disregards the distance (cost) between points in the search space; (v) the algorithm is unsuitable for building routes that are unrequired to visit all the points in the search space; (vi) the algorithm is not guaranteed to build the best route for all route problems; (vii) undesirable points need to be removed from the search space before the execution of the algorithm.

V. Genetic Algorithm

The genetic algorithms (GA) are categorized as evolutionary algorithms that are types of Artificial Intelligence (AI) technique. Evolutionary algorithms are based on the process of natural selection and they pursue to simulate the biological evolution of living beings, then adapting each generation to the environment, ensuring a bigger chance of surviving and to produce descendants. This process was described first by Darwin based on three basic principles: reproduction, natural selection, and diversity of individuals [7].

As described by [19] and [20], large types of route problems can be solved employing Genetic Algorithm. In the classic version of the traveling salesman, there is a list of cities to visit and each city is a mandatory stopping point and the objective function is to minimize the traveling cost between such points.

Let n the number of the cities, a solution is given by a sequence of all n cities to be visited. Fig. 4 depicts an example of a search space in which the cities are represented by the consecutive natural numbers $0,1,2,\ldots,n$. The starting point is also the ending point and is represented by the number 0. To use GA to solve the presented problem is necessary to define the initial population of individuals (chromosomes), which represent a partial solution to the problem. Each gene of the individuals is composed by a point to be visited (customer/city), as shown in Fig. 5. Fig. 6-A represents a route generated by this individual with partial solution. For every reference try to find out (this is not mandatory and not always possible) a permanent on-line link and include it at the end of the respective reference.

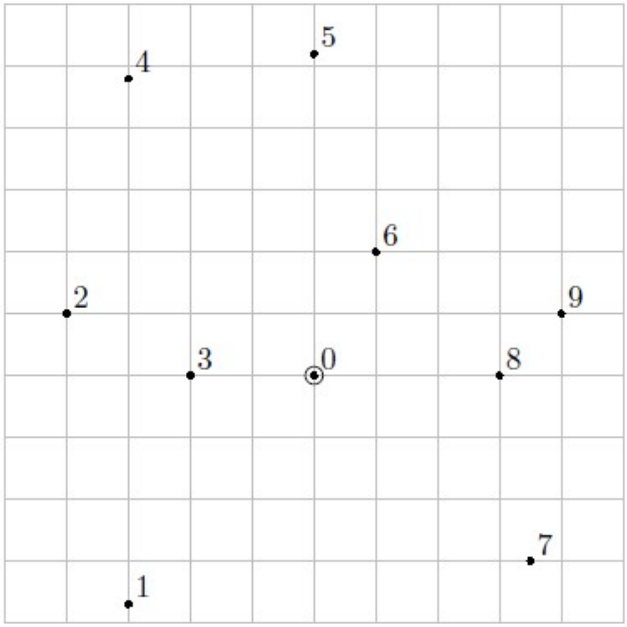

Figure 4. Search space in a traveling salesman problem, where the cities are represented by numbers

0	1	3	2	4	5	9	7	6	8	0

Figure 5. Representation of an GA individual composed by a partial solution

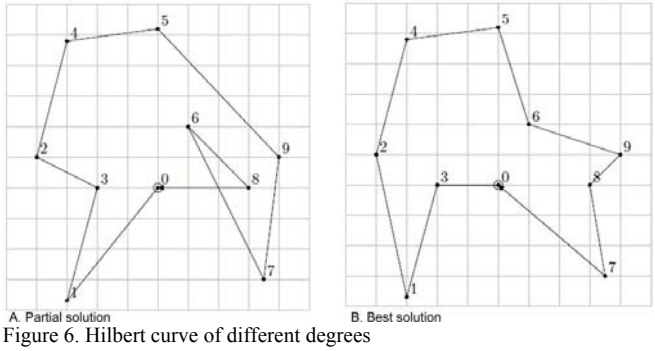

Figure 6. Hilbert curve of different degrees

After the building of the initial population, the algorithm performs the genetic operators: selection of individuals, crossing, and mutation. These operations enable the solution to evolve in each iteration of the algorithm.

When the algorithm reaches the stop criterion, the individual with the lowest cost is selected to represent de final solution of the problem, which is not necessarily the optimal one, but the sub-optimal solution. Fig. 6-B depicts the final solution to the classic traveling salesman problem.

We can also summarize the advantages of using GA to solve the vehicle routing problem [21-22]: (i) the Genetic Algorithms are based on the problem codification and support a wide variety of computational program; (ii) the Genetic Algorithms provide solution near to the optimal solution even when there are unknown methods to solve the problem; (iii) the searching for the solution is performed in a population of individuals and not only at one point at a time; (iv) GA is robust and can solve a wide range of hard problems in a reliable and fast way; (v) they are easy to implement and flexible to add modifications; (vi) they are easy to combine with another methods and algorithms.

And the disadvantages: (i) there is no standard definition of the stopping point for all situations. Therefore, the algorithm can perform an unnecessary number of iterations. Or even it could finish before the best possible route could be found; (ii) as the generation of the initial population is random, it can be extremely expensive to build and can even be able to produce the expected number of individuals for the next steps of the algorithm, and the GA would loop infinitely searching for the sufficient number of individuals; (iii) it requires a substantial number of evaluations of fitness functions; (iv) any modification in the search space requires a new execution of the algorithm.

VI. Hybrid Optimization Algorithm

As described by [21], to solve optimization problems, specifically the large-scale ones, the generation of the initial population can be infeasible, expensive and inefficient. Thus, it is necessary to replace the random process for a more appropriate local search that can converge faster and more precisely towards the optimal solution.

To improve the existing results and hasten the convergence of the GA, we propose to employ a hybrid optimization algorithm (Hybrid Optimization Algorithm – HOA), which is composed of Genetic Algorithm and Space-Filling Curves. This combination is performed in the following way:

i. The search space is composed of four clusters, according to the formation of the level 1 of the Hilbert Curve;
ii. After the clustering, the graph of the search space for the generation of the curves is reduced to ¼, and the visit points will be grouped according to their respective regions, which are linked by the point that represents the warehouse (see Fig. 7);
iii. Next, four fractal mappings are generated, one for each cluster. The mapping sequence is built from the central point (depot). It is worth to note that the fractal generation is performed clockwise, namely, in each quadrant, the fractal rotates 90 degrees (see Fig. 8);
iv. Afterwards, the points are framed in the cells of the matrix built by the fractal algorithm, so that each point is in a cell. The same order of the fractal detailing is applied in all quadrants;
v. After placing the points in the cells, it is generated an order of the mapping of the points in each quadrant, according to the sequence that the filling curves visit the matrix cells;
vi. In the GA, by the definition, the initial population is randomly generated. However, in the proposed hybrid algorithm the first individual of the initial population is built according to the sequence of fractal mapping, which already deliver to GA an individual composed by genes with approximate points, building a route without crossing and somehow optimized;
vii. In the generation of the super individual, the first and last gene is represented by the deposit and the other genes are formed by the sequence of points obtained by the fractal mapping, which always starts from the central depot. The first fractal sequence creates an individual of the first-generation GA to compile the first route of the

running cluster. The second fractal sequence is used to create the first-generation GA member for the second route, and so on until all routes are created for the cluster. Thus, for the elaboration of each cluster route, a super individual is created using the fractal mapping sequence;

viii. Like in the GA, the fit individuals are selected to compose the new generation and also, they have a higher chance to cross. It is expected that the individual produced by the fractal sequence continues for future generations and its genes will be transmitted to other individuals in the population, thus contributing to an individual with a more efficient route, close to the optimal solution.

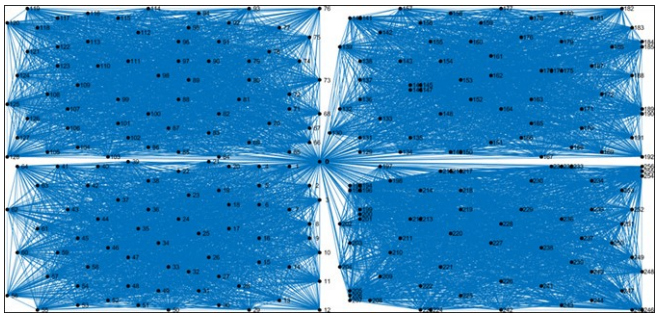

Figure 7. Representation of the graph after division of the search area by Hilbert curve of order 1

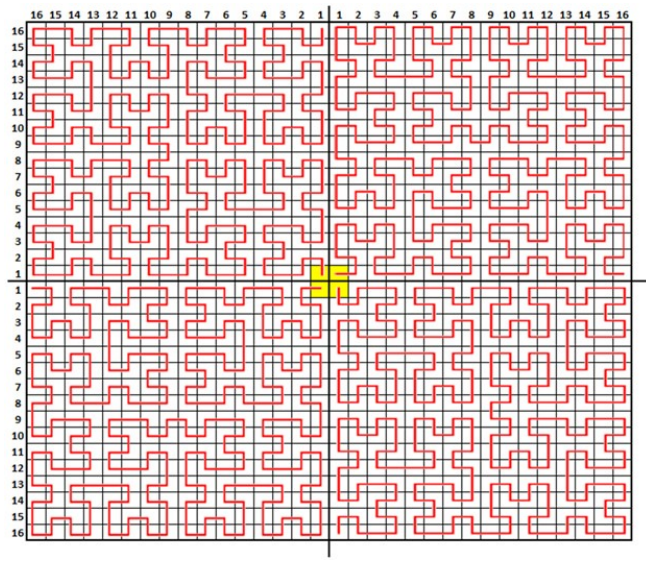

Figure 8. Four fractal mappings generated for each cluster in Fig. 8

Later, to improve the results of the GA, a Random Crossover operation was added to the algorithm after the mutation step. In addition to that, a different version of crossing and mutation were also added. These improvements and their results are detailed in the following paragraphs.

To augment the genetic diversity of the descendants, in this crossing proposal each descendent will be composed of 50% of the genes from each parent. The genes will be selected randomly in the genetic vector of the descendant and will be positioned randomly in the genetic vector of the descendant, which will be restricted to the inexistence of the repetition of the genes in the genetic chain of the new individual. In this crossing, while the genetic vector of the descendant has unfilled positions, the values are chosen randomly from each parent alternately and positioned in the empty spaces of the genetic vector, also selected randomly. As in so far, the values of the genetic vectors from the parents are selected, they become unavailable, and cannot be selected again.

Due to its flexibility, one new operation named genetic rearrangement was added to the GA after the mutation phase as depicted in Fig. 9. This new operation checks if the displacement for left or right of one gene diminishes the cost of local solution, without changing the genetic load of the given individual. This operation verifies if the cost of the solution is diminishing when the posterior genes are shifted to the right. Whether the cost decreases, the gene is shifted to the position of verification and the checking is restarted and repeated until there is no decrease in the cost of the solution. Next, a new examination in each gene is performed from right to left, check if the cost of the solution diminishes when the genes are shifted. Using this operator, besides the decreasing in the cost of the local solution, also is expected that the algorithm eliminates the overlaps and crossing of parts in the route, thus creating loops in the route and a more efficient solution.

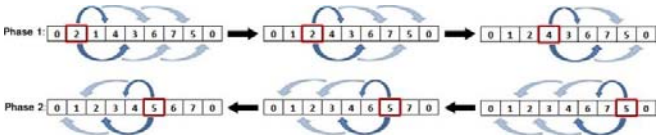

Figure 9. Shifting of the gene and solution verification

After the inclusion of the operator of the genetic rearrangement to the GA, the exchange mutation became obsolete since the rearrangement operator already swaps the genes. For such reason, it was necessary to modify the mutation operator, i. e., instead of swapping the two genes it randomly changes one individual gene for another in the search space [23]. Fig. 10 presents an example of how this mutation operator changed.

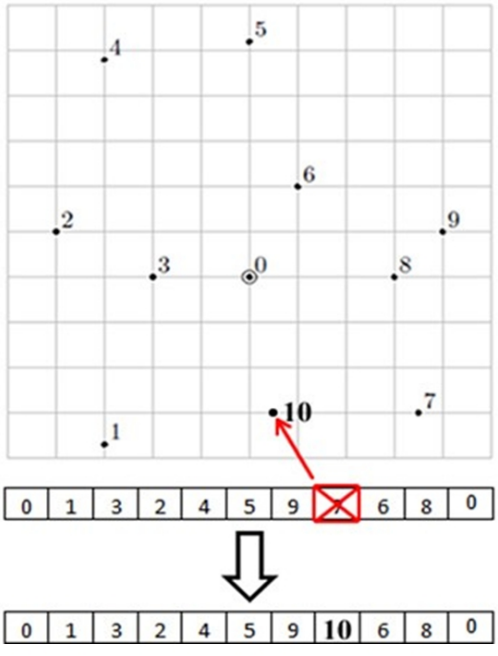

Figure 10. An example of exchange mutation operation

The parameter values used in GA were obtained by means

of an algorithm calibration step, which consists in fixing a set of parameters used and varying only one, performing successive executions of the algorithm and verifying the average cost of the solution for each parameters combination.

VII. Results

As a first test, a classic GA was implemented to solve the vehicle routing problem, following the instructions in [20]. The parameters were set according to following values:

i. Each population is composed of 100 individuals in each generation;
ii. The initial population is randomly built;
iii. In each generation, the elitism method selects the 20 best individuals with lowest cost;
iv. Also, 30 other individuals are drawn by the roulette wheel method, and they will be in the next generation and will cross;
v. 50 individuals selected previously are crossed, generating 50 new individuals, which will be in the next generation;
vi. The crossing employs the Ordered Crossover methodology;
vii. After the crossing, everyone has a chance of 25% to mutate, and the Exchange Mutation is the method used in this step;
viii. The stop points of the algorithm is after 100 iterations.

After setting up the parameters, the algorithm was tested in the 10 instances of the optimization dataset that were adapted for the SD-MTSP and available by [24]. They are the following:

i. A-n32-k5: instance composed of 1 depot, 31 customers, and 5 vehicles;
ii. A-n37-k5: instance composed of 1 depot, 36 customers, and 5 vehicles;
iii. CMT1: instance composed of 1 depot, 49 cities, and 5 vehicles;
iv. CMT3: instance composed of 1 depot, 99 cities, and 8 vehicles;
v. CMT6: instance composed of 1 depot, 49 cities, and 6 vehicles;
vi. CMT8: instance composed of 1 depot, 98 cities, and 9 vehicles;
vii. P-n101-k4: instance composed of 1 depot, 100 cities, and 4 vehicles;
viii. X-n115-k10: instance composed of 1 depot, 114 cities, and 10 vehicles;
ix. X-n139-k10: instance composed of 1 depot, 138 cities, and 10 vehicles;
x. X-n256-k16: instance composed of 1 depot, 255 cities, and 16 vehicles.

For each instance, the algorithm was executed 10 times to achieve reliable results. The final solution of each problem is given by the sum of all found routes, and each route is represented by the best individual of the last generation produced by the algorithm.

Table I presents a comparison between the GA and the IBM ILOG CPLEX, where *n* represents the number of clients plus a depot, and *k* represents the number of vehicles. The CPLEX Optimizer was named for the simplex method as implemented in the C programming language and it is used as benchmark algorithm since it has an optimal solution [25].

It is also important to mention that the hardware parameters used to generate the results obtained by the CPLEX are presented in [24] and the results obtained by the proposed algorithm were generated using a notebook core i7-7700 with NVIDIA GeForce GTX 1060, 16 Gb of RAM and an SSD of 512 Gb.

Table I. Comparison of the Results achieved by GA and the Optimal Results from the Benchmark

Instance Name	*n*	*k*	Cost		
			CPLEX	GA	Difference (%)
A-n32-k5	32	5	784	1003	27.9
A-n37-k5	37	5	669	857	28.1
CMT1	50	5	524	667	27.3
CMT3	100	8	555	712	28.3
CMT6	50	6	826	1053	27.5
CMT8	100	9	865	1109	28.2
P-n101-k4	101	4	681	867	27.3
X-n115-k10	115	10	12747	16125	26.5
X-n139-k10	139	10	13590	17292	27.2
X-n256-k16	256	16	18839	24156	28.2
Average					27.7

Notice from Table I that, in comparison with the optimal solution from CPLEX, the algorithm found an average result 27,7% higher, which demonstrates that it is insufficient to use only the genetic algorithm to solve the problem. Then, as a second test, our hybrid algorithm was implemented using the parameters according to the following values:

i. An initial population of 50 individuals in each generation;
ii. The initial population composed by one individual produced by the fractal algorithm and other 49 individuals randomly produced by the points of the current cluster;
iii. In each generation, 10 individuals of the lowest cost are selected by the elitism method to be in the new generation and to be crossed;
iv. Additionally, 15 individuals are selected by the roulette wheel method and they also will be in the next generation and will be crossed;
v. The 25 individuals selected before are crossed, so generating 25 new individuals that will be in the new generation;
vi. The crossing is performed by the Random Crossover method;
vii. After the crossing, each individual has a 15% chance to mutate, and mutation method applied is the Exchange Mutation Adaptation;
viii. The stopping point is in the 60-th iteration.

Table II shows the results achieved by the hybrid algorithm, which is 12.7% higher compared to the optimal solution, proving the viability of the proposed methodology.

As a final test, to verify the results of the proposed hybrid algorithm, comparing with other hybrid algorithms, we used the work proposed by [26], in which the authors made available a new benchmark dataset and perform a complete evaluation of five variations of the Ant Colony Optimization (ACO) applied to the Single-Depot Multiple Traveling Salesman Problem. The variations of the ACO developed by the authors were:

TABLE II. COMPARISON OF THE PROPOSED ALGORITHM WITH OPTIMAL WORKS FROM THE LITERATURE

Instance Name	*n*	*k*	Cost		
			CPLEX	Our Approach	Difference (%)
A-n32-k5	32	5	784	892	13.8
A-n37-k5	37	5	669	776	16.0
CMT1	50	5	524	595	13.5
CMT3	100	8	555	628	13.2
CMT6	50	6	826	937	13.4
CMT8	100	9	865	974	12.6
P-n101-k4	101	4	681	774	13.7
X-n115-k10	115	10	12747	14128	10.8
X-n139-k10	139	10	13590	14932	9.9
X-n256-k16	256	16	18839	20728	10.0
Average					12.7%

i. Problem decomposition with k-Means followed by ACS for TSP (kM-ACS);
ii. ACS with global-solution pheromone update (g-ACS);
iii. ACS with sub-tour pheromone update (s-ACS);
iv. ACS with global-solution pheromone update and bounded tours (gb-ACS);
v. ACS with sub-tour pheromone update and bounded tours (sb-ACS).

In the comparison between our proposal and the works [26-27], we executed the algorithms in the following instances:

i. eil51: one depot and 50 cities;
ii. eil76: one depot and 75 cities;
iii. rat99: depot and 98 cities.

Also, each instance was tested with 2, 3, 5, and 7 vehicles to travel the route, and the goal is to minimize the total cost to visit all the cities.

Reference [26] describes the running time, i.e. the total duration needed to take a certain route, of CPLEX for the instance eil51 with 7 vehicles and for the instance rat99 with 3, 5 and 7 vehicles. These details are described in Table III, which also shows a comparison of the running time and cost between CPLEX and our hybrid algorithm.

TABLE III. COMPARISON OF THE EXECUTION TIME AND COST BETWEEN CPLEX AND OUR PROPOSED ALGORITHM

Instance Name	#Vehicles	CPLEX		Hybrid Algorithm	
		Run-time (hrs)	Cost	Run-time (sec)	Cost
eil51	7	120	605.21	642	631.59
rat99	5	96	1519.49	560	1632.61
	5	168	1855.83	684	2055.74
	8	216	2291.82	701	2703.38

As expected, notice from Table III that the execution time of CPLEX is higher than the proposed algorithm (a maximum value of 216 hours against only 701 seconds of the proposed algorithm considering 8 vehicles). Even with a lower cost, these results make the use of CPLEX not feasible for cases where the quality of certain routes must be taken into consideration.

Table IV presents a comparison between the results obtained by the combination of Genetic Algorithms and Space-Filling Curves and the five variations of the ACO presented in [26-27]. Again, *k* represents the number of vehicles used. Notice that the proposed algorithm presented a superior result in some problems and, when it was not superior, presented the second-best results for almost instances. For example, in the instance eli51-m2 the result of the hybrid algorithm is only 2.1% higher than the optimal solution against 2.2% of gb-ACS, which was the best result achieved by [26] for this instance. For the instance eli76-m3 our algorithm is only 2.3% higher than optimal against 6.9% of sb-ACS and for the instance rat99-m3 our algorithm achieved a cost of 1519.49 (8.2% of optimum) against 1651.68 (9.7% of optimum) of sb-ACS algorithm. On the other hand, for some instances, the kM-ACS algorithm was better than our proposal, achieving results with lower costs, for example, 7.6% higher than optimal solution for the instance rat99-m5 and only 3.0% for the instance rat99-m7.

The graphs presented in Fig. 11 depict a comparison of the solution founded by [26] and our hybrid algorithm using some routes: rat99 with 7 vehicles; eil76 with 2 vehicles and eil76 with 7 vehicles. When analyzing the graphs, it is possible to observe that in the proposed algorithm the variation was significantly lower, due to the ability of the algorithm to generate more balanced routes, with a small difference between the numbers of points to visit for each route. This restriction was added to the algorithm to enable its use in a commercial application, balancing the work of each employee who will carry out the courses avoiding overloading in customer service.

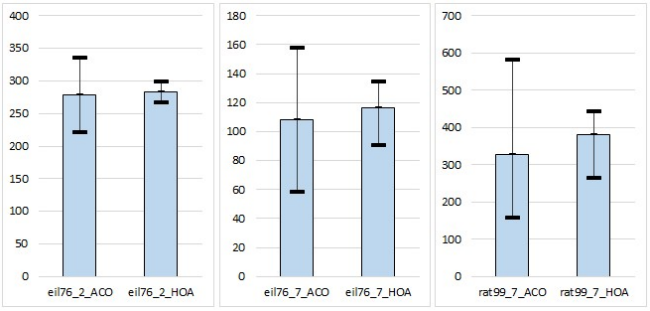

Figure 11. Comparison of the solution founded by [26] and our hybrid algorithm using some routes

VIII. CONCLUSION

The experiments performed in the two benchmark datasets proves that the proposed solution achieves good results. In the first dataset the hybrid algorithm achieved results close to the optimal solutions: 10.7% higher than optimal solution for the instance P-n101-k4 and only 9.2% for the instance X-n139-k10. Compared to other hybrid algorithms for the Single Depot VRP, it was shown that our approach was superior in the instance eil76 and presents the second-best results in almost all instances from eil51 and rat99. Also, the hybrid algorithm achieved solutions close to the optimal solutions in the instances eli51-M2 and eli76-M3 (only 0.1% and 0.9% close to the optimal solution). Moreover, our approach presents a very low execution time, which allows the rapid generation of new routes, if necessary.

As the main motivation of this work was related to the real problem of the Sanitation Company of Espirito Santo, the outcome of this work resulted in a new module of automatic generation of routes with lower cost using georeferenced information.

As future work, we suggest applying another space-filling curve, like Sierpinski curve, replacing the Hilbert curve.

TABLE IV. COMPARISON OF THE PROPOSED HYBRID APPROACH WITH FIVE VARIATIONS OF ACO FROM LITERATURE

Instance Name	*k*	Optimal	kM-ACS		g-ACS		s-ACS		gb-ACS		sb-ACS		Our Approach	
			Cost	Diff	Cost	Diff	Cost	Diff	Cost	Diff	Cost	Diff	Cost	Diff
eil51	2	442.32	454.30	2.7%	452.66	2.3%	454.96	2.9%	452.22	2.2%	453.81	2.6%	451.60	**2.1%**
	3	464.11	500.00	7.7%	485.73	4.7%	489.64	5.5%	479.51	**3.3%**	483.39	4.2%	482.14	3.9%
	5	529.70	563.58	**6.4%**	582.36	9.9%	590.63	11.5%	585.76	10.6%	598.61	13.0%	570.73	7.7%
	7	605.21	634.47	**4.8%**	674.78	11.5%	680.38	12.4%	688.26	13.7%	699.47	15.6%	642.54	6.2%
eil76	2	558.59	594.21	6.4%	580.77	4.0%	583.41	4.4%	579.68	3.8%	578.96	3.6%	572.64	**2.5%**
	3	579.30	642.89	11.0%	622.91	7.5%	630.67	8.9%	613.76	5.9%	619.19	6.9%	592.83	**2.3%**
	5	680.67	740.35	8.8%	747.49	9.8%	760.05	11.7%	734.61	7.9%	744.94	9.4%	707.67	**4.0%**
	7	759.90	820.35	8.0%	873.65	15.0%	883.63	16.3%	894.70	17.7%	911.06	19.9%	818.57	**7.7%**
rat99	2	1350.73	1485.56	10.0%	1398.01	3.5%	1398.01	3.5%	1391.89	3.0%	1382.05	**2.3%**	1407.40	4.2%
	3	1519.49	1672.11	10.0%	1691.56	11.3%	1707.20	12.4%	1661.04	9.3%	1651.68	8.7%	1643.74	**8.2%**
	5	1855.83	1996.04	**7.6%**	2260.74	21.8%	2297.05	23.8%	2286.73	23.2%	2337.94	26.0%	2074.27	11.8%
	7	2291.82	2361.55	**3.0%**	2859.98	24.8%	2878.97	25.6%	3004.37	31.1%	2984.42	30.2%	2727.41	19.0%

Additionally, we suggest testing the combination of the fractal algorithm and other searching heuristics, like Ant Colony or Particle Swarm Optimization [28]. This work also can be evolved to deal with other types of VRPs problems, like the VRP with Time Windows and the Pickup-and-Delivery VRP, in addition to other search problems, like lower-cost routes for robots and obstacle avoidance.

ACKNOWLEDGMENT

We would like to thank our associates at CESAN and Federal Institute of Espirito Santo for their help during our work on this research.

REFERENCES

[1] B. M. Baker and M. A. Ayechew, "A genetic algorithm for the vehicle routing problem," Computers & Operations Research, vol. 30, no. 5, pp. 787–800, Apr. 2003. doi:10.1016/S0305-0548(02)00051-5

[2] O. Braysy, W. Dullaert, and M. Gendreau, "Evolutionary Algorithms for the Vehicle Routing Problem with Time Windows," Journal of Heuristics, vol. 10, no. 6, pp. 587–611, Dec. 2004. doi:10.1007/s10732-005-5431-6

[3] A. Kadar, M. Shahjalal, M. Faisal, and M. Iqbal, "Solving the Vehicle Routing Problem using Genetic Algorithm," International Journal of Advanced Computer Science and Applications, vol. 2, no. 7, 2011. doi:10.14569/IJACSA.2011.020719

[4] F. T. Hanshar and B. M. Ombuki-Berman, "Dynamic vehicle routing using genetic algorithms," Applied Intelligence, vol. 27, no. 1, pp. 89–99, Jun. 2007. doi:10.1007/s10489-006-0033-z

[5] G. Clarke and J. W. Wright, "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," Operations Research, vol. 12, no. 4, pp. 568–581, Aug. 1964. doi:10.1287/opre.12.4.568

[6] R. H. Mole and S. R. Jameson, "A Sequential Route-Building Algorithm Employing a Generalised Savings Criterion," Operational Research Quarterly (1970-1977), vol. 27, no. 2, p. 503, 1976. doi:10.2307/3008819

[7] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning", pp. 1-25, Reading, Mass: Addison-Wesley Pub. Co, 1989.

[8] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins, "A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows," Computers & Operations Research, vol. 40, no. 1, pp. 475–489, Jan. 2013. doi:10.1016/j.cor.2012.07.018

[9] H. Nazif and L. S. Lee, "Optimised crossover genetic algorithm for capacitated vehicle routing problem," Applied Mathematical Modelling, vol. 36, no. 5, pp. 2110–2117, May 2012. doi:10.1016/j.apm.2011.08.010

[10] J. J. Bartholdi, L. K. Platzman, R. L. Collins, and W. H. Warden, "A Minimal Technology Routing System for Meals on Wheels," Interfaces, vol. 13, no. 3, pp. 1–8, Jun. 1983. doi:10.1287/inte.13.3.1

[11] N. Biggs, "THE TRAVELING SALESMAN PROBLEM A Guided Tour of Combinatorial Optimization," Bulletin of the London Mathematical Society, vol. 18, no. 5, pp. 514–515, Sep. 1986. doi:10.1112/blms/18.5.514

[12] J. J. Bartholdi and L. K. Platzman, "Heuristics Based on Spacefilling Curves for Combinatorial Problems in Euclidean Space," Management Science, vol. 34, no. 3, pp. 291–305, Mar. 1988. doi:10.1287/mnsc.34.3.291

[13] L. K. Platzman and J. J. Bartholdi, "Spacefilling curves and the planar travelling salesman problem," Journal of the ACM, vol. 36, no. 4, pp. 719–737, Oct. 1989. doi:10.1145/76359.76361

[14] J. J. Bartholdi and P. Goldsman, "Vertex-labeling algorithms for the Hilbert spacefilling curve," Software: Practice and Experience, vol. 31, no. 5, pp. 395–408, Apr. 2001. doi:10.1002/spe.376

[15] J. J. Bartholdi and P. Goldsman, "Continuous indexing of hierarchical subdivisions of the globe," International Journal of Geographical Information Science, vol. 15, no. 6, pp. 489–522, Sep. 2001. doi:10.1080/13658810110043603

[16] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," Omega, vol. 34, no. 3, pp. 209–219, Jun. 2006. doi:10.1016/j.omega.2004.10.004

[17] R. C. Mittal, "Space-filling curves," Resonance, vol. 5, no. 12, pp. 26–33, Dec. 2000. doi:10.1007/BF02840392

[18] H. Sagan, "Space-Filling Curves", pp. 9-30, New York, NY: Springer New York, 1994.

[19] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," Computers & Industrial Engineering, vol. 99, pp. 300–313, Sep. 2016. doi: 10.1016/j.cie.2015.12.007

[20] S. Karakatič and V. Podgorelec, "A survey of genetic algorithms for solving multi depot vehicle routing problem," Applied Soft Computing, vol. 27, pp. 519–532, Feb. 2015. doi:10.1016/j.asoc.2014.11.005

[21] Y. Yue, T. Zhang and Q. Yue, "Improved Fractal Space Filling Curves Hybrid Optimization Algorithm for Vehicle Routing Problem," Computational Intelligence and Neuroscience, 2015, doi:10.1155/2015/375163.

[22] A. S. Ruela, F. G. Guimaraes, R. A. R. Oliveira, B. Neves, V. P. Amorim, and L. M. Fraga, "A Parallel Hybrid Genetic Algorithm on Cloud Computing for the Vehicle Routing Problem with Time Windows", 2013, pp. 2467–2472. doi:10.1109/SMC.2013.421

[23] J. Berger and M. Barkaoui, "A parallel hybrid genetic algorithm for the vehicle routing problem with time windows," Computers & Operations Research, vol. 31, no. 12, pp. 2037–2053, Oct. 2004. doi:10.1016/S0305-0548(03)00163-1

[24] E. Uchoa, D. Pecin, A. Pessoa, M. Poggi, T. Vidal, and A. Subramanian, "New benchmark instances for the Capacitated Vehicle Routing Problem," European Journal of Operational Research, vol. 257, no. 3, pp. 845–858, Mar. 2017. doi:10.1016/j.ejor.2016.08.012

[25] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," Mathematical Programming, vol. 91, no. 2, pp. 201–213, Jan. 2002. doi:10.1007/s101070100263

[26] R. Necula, M. Breaban, and M. Raschip, "Performance Evaluation of Ant Colony Systems for the Single-Depot Multiple Traveling Salesman Problem," in Hybrid Artificial Intelligent Systems, vol. 9121, E. Onieva, I. Santos, E. Osaba, H. Quintián, and E. Corchado, Eds. Cham: Springer International Publishing, 2015, pp. 257–268. doi:10.1007/978-3-319-19644-2_22

[27] R. Necula, M. Breaban, and M. Raschip, "Tackling the Bi-criteria Facet of Multiple Traveling Salesman Problem with Ant Colony Systems," 2015, pp. 873–880. doi:10.1109/ICTAI.2015.127

[28] S.-H. Xu, J.-P. Liu, F.-H. Zhang, L. Wang, and L.-J. Sun, "A Combination of Genetic Algorithm and Particle Swarm Optimization for Vehicle Routing Problem with Time Windows," Sensors, vol. 15, no. 9, pp. 21033–21053, Aug. 2015. doi: 10.3390/s150921033