ISSN 1870-4069

Design and Implementation of a CVRP Simulator Using Genetic Algorithms

Iván Ramírez Cortes[1], José Alberto Hernández Aguilar[2,3], Miguel Ángel Ruiz[1], Marco Antonio Cruz-Chavez[2], Gustavo Arroyo-Figueroa[3]

[1] Polytechnic University of the State of Morelos, Jiutepec, Morelos, Mexico

[2] Autonomous University of the State of Morelos, Cuernavaca, Morelos, Mexico

[3] National Institute of Electricity and Clean Energies, Cuernavaca, Morelos, Mexico

jose_hernandez@uaem.mx

Abstract. We discuss the design and implementation of a CVRP simulator, for this purpose firstly, we discuss briefly CVRP and its taxonomy, later we discuss in depth the genetic algorithm (GA), further we present the design and implementation of a three-tier web-based system to upload a CVRP instance and select a Metaheuristic for its processing. For probing our simulator, we calculate the best route(s), processing ten different .vrp instances, with our GA and compare it with the nearest neighbor algorithm (NNA). Preliminary results show genetic algorithm generates some best routes, but processing time is higher regarding nearest neighbor algorithm.

Keywords: CVRP, simulation, genetic algorithm, nearest neighbor algorithm, client-server technology.

1 Introduction

Research Problem

It has been identified that in the state of Morelos, Mexico, small and medium businesses need a logistics in real time to solve routing problems, which requires the implementation of a plotter of routes to represent real-world instances of the CVRP (Capacitated Vehicle Routing Problem) that meets following features:

Operate Under Three Tiers Client-Server Architecture

Allows the reading of. vrp files from the client-side, send the request to the server, select the algorithm or heuristic goal to use (for example nearest neighbor, ant's colony, genetic). Call the corresponding method and display in real time the results of the progress in the process of optimization of the route. Is required the use of java running

on a web platform for the implementation of the plotter of routes for the CVRP by intelligent algorithms, since the server will be mounted on open source software.

Justification

The research group UAEMOR CA124 "Operations research and computer science", participated in the convocation of the PRODEP 2015 for the creation of academic networks, derived from this convocation was obtained financial support for the implementation of several projects, within these projects was financed the corresponding plotter for the solution of the CVRP, which for its complete solution includes not only the estimate of the calculation for routes but its real-time plotting.

At present, the lack of simulators that graph in real time this type of problems is very wide, and it is even more difficult to find simulators operating via the client-server architecture.

Hypothesis

Hi. Using a graphing tool for problems CVRP type, operating under the client-server architecture, allows determining if genetic algorithms obtained better results than the nearest neighbor algorithm.

Ho. Using a graphing tool for problems CVRP type, operating under the client-server architecture, does not allow determining if genetic algorithms do not obtain better results than the nearest neighbor algorithm.

Scope and Limitations

We analyze the problem and abstract its requirements in four modules:

Module of reading of .vrp instances

- The system will be able to read files of .vrp type previous proper validation.

Module of parameters selection

- Type of Technology to use for the server process: sequential, parallel.
- Select the type of algorithm to simulate (genetic algorithm).

Module of nodes graphing in real time

- Plots the local optimal routes regularly, showing where the user will be able to select the best-desired route.

Module of Graphical Export

- Once calculated the best route, it could be exported in JPG or PNG format, as well as the summary of information (total distance and processing time).

Methodology for Development: Extreme Programming

XP is an agile methodology for software development, which basically consists in strict adherence to a set of rules that are focused on the needs of the client in order to achieve

a good quality product in a short time, focused on enhancing relationships as key to the success of software development.

The philosophy of XP is to satisfy the customer's needs, integrating it as part of the development team. In all the iterations of this cycle, both the client and the program learn.

Structure of Document

In section one we describe the problem at hands, section two discuss related work, including VRP theory, its taxonomy and main approaches to solve it, we focus in genetic algorithm metaheuristics as a method of solution. Later we discuss CVRP mathematical model. In section three, we present the implementation of the CVRP simulator based on the client-server technology; we present pseudo code of main functions and include segments of Matlab code created to implement a genetic algorithm to solve CVRP instances. Finally, we present preliminary results, conclusions, and future work.

2 Related Work

2.1 VRP

The Vehicle Routing Problem (VRP), is used to determine a set of routes for a fleet of vehicles that are based on one or more deposits or warehouses, to satisfy the demand of multiple customers geographically dispersed proposed by Hillier and Lieberman [10]. According [14], importance of VRP is due:

> *"The transport of goods in urban environments plays a very important role in the sustainable development of a city, as high levels of movement of goods occur within cities".*

In Figure 1, we show the models that originate the Travel Salesman problem (TSP), from which derived the first VRP models.

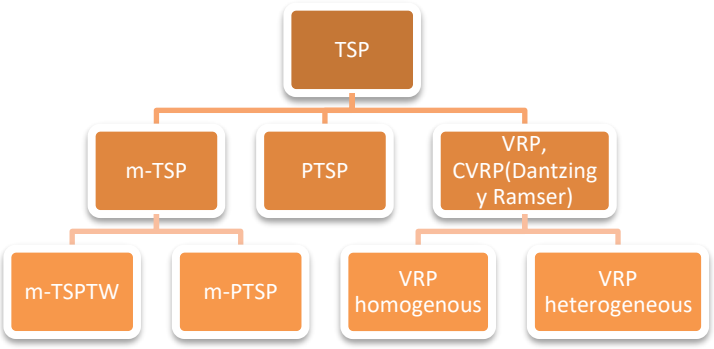

Fig. 1. Models originating VRP problem [13].

Given TSP and VRP are NP-hard problems [8]; there are several approaches to try to solve them. In figure 2, we show the main methods of solution of VRP, this figure shows its taxonomy.

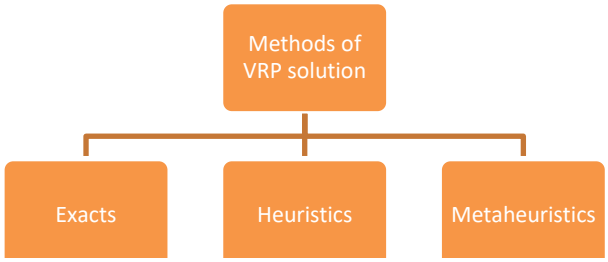

Fig. 2. Methods of solution to the VRP [13].

Exact Methods

In Figure 3, we show the different exact methods to give a solution to the VRP.

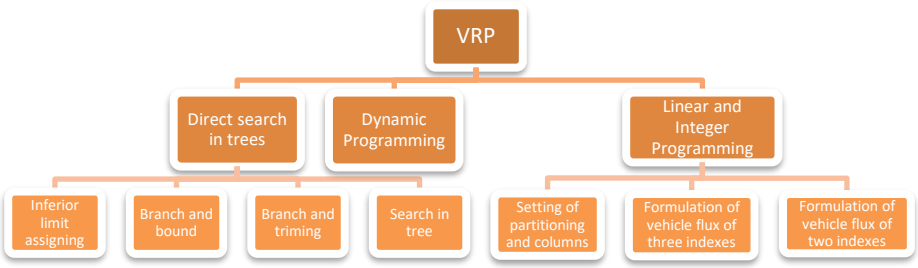

Fig. 3. Classification of exact methods [13].

Heuristics

The heuristics are procedures that provide acceptable quality solutions limited through an exploration of the search space [4]. These methods are based on routes that contain a single node to find the best pair (node, route), that represents the best intersection [11]. Figure 4 shows the classification of the heuristics in constructive methods, methods of two phases and heuristics of improvement.

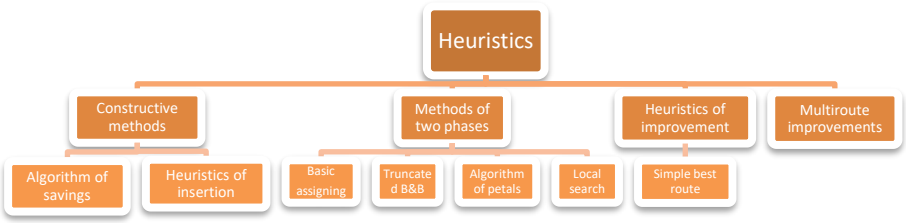

Fig. 4. Classification of Heuristics [13].

Metaheuristics

Most of these methods of solution were developed in the 1990s; one of its features is that procedures of search try to find acceptable solutions [5]. In figure 5, we show the methods of solution that consist of Simulated Annealing, Taboo Search, Neural Networks, Genetic algorithms, ant colony algorithms and search in neighborhoods.

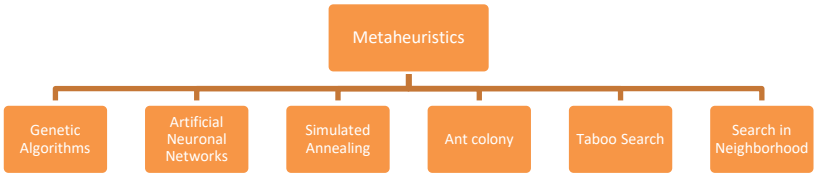

Fig. 5. Classification of Metaheuristics [13].

2.2 Genetic Algorithm

Genetic Algorithms (AGs) are adaptive methods that can be used to solve problems arising from the search, optimization [13] and decision making [15]. They are based in the genetic process in living organisms [2, 7]. It is based on the principles of the laws of the natural life proposed by Darwin. A genetic algorithm according to [9], basically consists of 1) An initial population, which can be generated in a random way, 2) Calculation of the fitness function, 3) Selection based on the fitness of the population, 4) Generate a new population through cross and mutation, 5) Generate a cycle of the above until the function of unemployment is true, and 6) get the best population of individuals.

Pseudo Code of Genetic Algorithm [9]

```
BEGIN A /*Simple Genetic Algorithm */
```

```
        Generate an initial population.
        Compute the fitness function of each individual.
        WHILE NOT Finished DO
      BEGIN to /*produce new generation*/
        FOR POPULATION SIZE / 2 DO
        BEGIN /*Reproductive cycle*/
            Select two individuals of the previous generation,
            for the crossing (probability of selection pro-
            portional the evaluation function of the individ-
            ual).
            Cross with a certain probability the two individ-
      uals obtaining two descendants.
            Mutate The two descendants with a certain proba-
      bility.
            Compute the evaluation function of the two mutated
      descendants.
            Insert the two mutated descendants in the new gen-
      eration.
          END
          IF        the population has converged, THEN
              Completed = True
       END
  END
```

Mathematical Model for the CVRP

The CVRP has as an objective function which purpose is to minimize the costs of the vehicles in the trajectories. Table 1 displays a list of the indexes and variables that occupies the mathematical model of the CVRP.

Table 1. Indexes and variables of CVRP model.

Nomenclature	Description	Nomenclature	Description
I	The node of the departure of the vehicle.	Cij	Cost of transport of the node i to node j
J	The node of the departure of the vehicle.	Dj	Demand in the node j
d	Vehicle to use (1,2,3,…k)	U	Resource capacity k
x_{ij}^k	Customer Demand	N	Number of clients
Y_{ij}	If it is equal to 1, the vehicle k is assigned to the arc from node i to node j. It is equal to 0 otherwise		

The mathematical model of CVRP routing, according to [1, 12], is:

$$Minimize \sum_{(i,j)\in A} c_{ij} * y_{ij}. \quad (1)$$

Subject to:

$$R1 \sum_{1 \le k \le K} x_{ij}^{k} = y_{ij} \; ; \; \forall \; \boldsymbol{i, j},$$

$$R2 \sum_{1 \le j \le n} \boldsymbol{y_{ij} = 1} \; ; \; \forall \; \boldsymbol{i},$$

$$R3 \sum_{1 \le i \le n} \boldsymbol{y_{ij} = 1} \; ; \; \forall \; \boldsymbol{j},$$

$$R4 \sum_{1 \le j \le n} \boldsymbol{y_{0j} = k},$$

$$R5 \sum_{1 \le j \le n} \boldsymbol{y_{i0} = k},$$

$$R6 \sum_{1 \le i \le n} \sum_{1 \le j \le n} \boldsymbol{d_i * x_{ij}^{k} \le u} \; ; \; \forall \; \boldsymbol{k},$$

$$R7 \sum_{i \in Q} \sum_{j \in Q} \boldsymbol{y_{ij} \le |Q| - 1}; \; \forall \; \boldsymbol{subset \; of \; Q \; of} \; \{1, 2, \dots, n\},$$

$$\boldsymbol{R8 \; k \le K},$$

$$R9 \; \boldsymbol{y_{ij}} \in \{0, 1\} \; ; \; \forall \; (\boldsymbol{i, j}) \; \in \boldsymbol{A},$$

$$R10 \; \boldsymbol{x_{ij}^{k}} \in \{0, 1\} \; ; \; \forall \; (\boldsymbol{i, j}) \; \in \boldsymbol{A}, \forall \; \boldsymbol{k}.$$

A is a set, which is defined as A={(i,j) : yij=1}, i.e. the set of edges of the graph with the possibility of making a single travel from node i to node j. Paths should begin in the node 0 and conclude at the same. Each arc (i, j)∈A, i≠j, the set has a cost Cij. Each of the vehicles has the same charge q, and each customer has a demand di, i ∈ C, for each of the clients, also is to assume that all data are integers known and not negative [6]. The restrictions in table 2 are intended to design a set of Minimum cost routes, one for each vehicle, so that: a) Be met exactly once to each client, b) Each path starts and ends in the tank, and c) Respect the capacity constraints of the vehicles.

Table 2. Description of elements.

Numbering	Description
F.O.	The objective function you want to minimize the total cost of the sum of all the travel, i.e. the distance traveled. Each customer must have an assigned vehicle as well as a sequence to achieve a minimum cost.
R1	In this restriction is indicated if the path was already covered or not will be on the path, the variable x helps to indicate whether or not to use the vehicle k in the arc i,j in the case of x=1.
R2 R3	Indicate the activation of the arc i,j by means of the variable, and what determines a path between nodes i,j, also ensures that every customer is an intermediate node of any route. That is to say, that ensures that each client is visited once by a vehicle.
R4 R5 R6	Indicate that k is the number of vehicles used in the solution and that all those who depart from the deposit must be returned to the same.
R7	Notes that each vehicle does not exceed its capacity.
R8	Monitors that the solution does not contain cycles using 1.2 nodes, …n. Otherwise, the arches to contain any cycle passing through a set of nodes Q and

	the solution would violate the constraint because the left side of the restriction would be at least |Q|.
R9	Limits the maximum number of vehicles to use up to a maximum amount.
R10	Indicate that the variables x, y are binary

3 Implementation of CRVP Simulator

The following are the modules implemented within the simulator and shown in Figure 6. System includes a three-tier architecture as the used in [17].

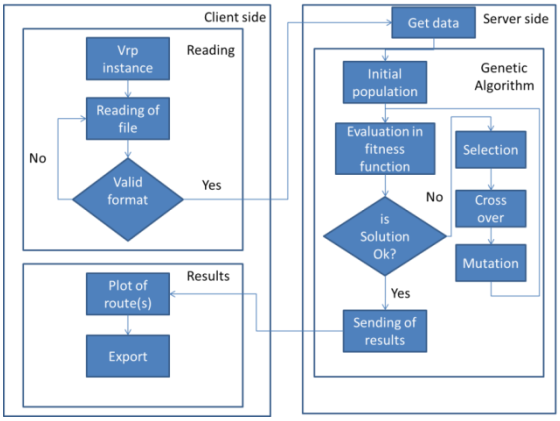

Fig. 6. Functional description of CVRP Simulator.

File Validation

For a correct operation of programs is mandatory to validate that the .vrp file has a correct structure. This function validates a .vrp extension, and description of file content: name, comment, type, dimension, edge type, weight type, and capacity.

Module of File Reading

This module is intended to validate the reading of the files of type .vrp taking into account: a) At the time of actually read a file; b) the extension is correct (.vrp); c) Correct structure, d) titles are spelled correctly.

Reading of File

As a first step, the simulator reads a file .vrp. To be able to read the file is implemented the next pseudo code using java.

Setting of Instances for Java and Matlab

This function creates instances from Java to be able to be used in Matlab. Through the use of functions to read from and write to files, the file includes an identifier, coordinates x & y and the demand, the function creates a new .m extension file. We

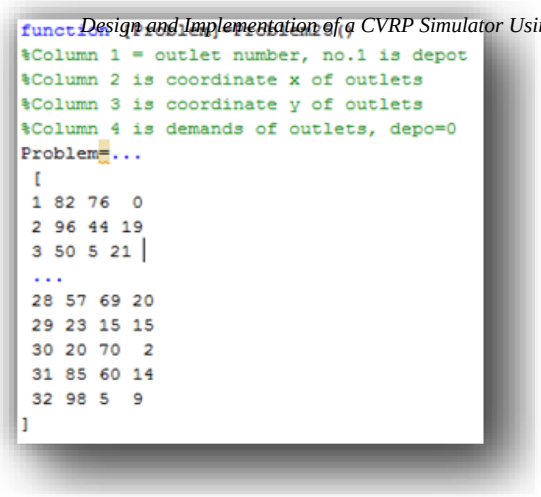

Fig. 7. Example of m file created in Java and compatible with Matlab.

decide to use Matlab due its flexibility to code algorithms and capacity for doing simulations as shown in [18, 19].

Server Side Module

In figure 6 is shown in the diagram corresponding to the server side, in which Matlab receives the .m file generated in Java, read the coordinates, and uses a genetic algorithm to perform calculations of routes and produce its graph interacting with Matlab.

Implementation of the Genetic Algorithm

In order to obtain the optimal routes, we implemented a genetic algorithm in Matlab, next are displayed the most important functions.

Selection by tournament

For the implementation of the selection was used tournament, Matlab code is shown next:

```
Selection by tournament Matlab code
T = round (random (2*N, S) * (5) +1;
% Tournament
[d, idx] = max (F(t),[],2); % index to determine winners
W = T(sub2ind (size(t), (1:2*N)', idx)); %winners
```

Crossover

We used one-point crossover. Matlab code is shown below:

```
One point crossover (Matlab code)
Pop2 = Pop (W(1:2: FIN),1:)% First winners variable of Pop2
P2A = Pop (W(2:2:FIN),:)% Second winners variable of Pop2
```

```
Lidx = sub2ind (size (Pop), round (random(N,1)*(5)+1)
```

Next Matlab code shows the selection process to perform the crossover:

```
Selection process to perform crossover (Matlab code)
% Selection of one point
vLidx= P2A(Lidx)*(1,G) % Point of two winners
[r,c] = find (Pop2 == vLidx) %winners 1
[d, Ord] = sort( r ) %Sort linear index
r = r (Ord); c = c (ord) % Reorder index
Lidx2 = sub2ind(size(Pop), r, c) % Convert to one index
Pop2(Lidx2) = Pop2(Lidx) % Half of crossover 1
```

Mutation

Finally, there is the stage of mutation which is shown next:

```
Mutation (Matlab code)
Idx = rand (N,1) < Muta % Selection of individuals to permute
Loc1 = sub2ind(size(Pop2), 1: N, round(random(1,N) *(5) +1)) %
Interchange of index 2
Loc 2 = sub2ind(size(Pop2), 1: N, round(random(1,N) *(5) +1)) %
Interchange of index 2
Loc2 (idx == 0) = Loc1 (idx == 0) % Probability of mutation
[Pop2 (Loc1), Pop2(Loc2) ] = deal (Pop2(Loc2), Pop2(Loc1)) % Mu-
tation
```

Plotting

Once the optimal routes are calculated is possible to plot them, by using Matlab functions, plotting each path to a different color, each graph will have a reservoir called "Depo" as well as *n* number of customers called *"C"*, a summary to display the number of routes, a total distance of travel and customers of each route.

Export

By means of the Matlab is possible to export resulting graph to two files, one in .PDF format and the other in .PNG format.

Client-side displaying

In this stage, the results are displayed on the client side, that is to say, the graphics in image format as well as exported in .PDF format.

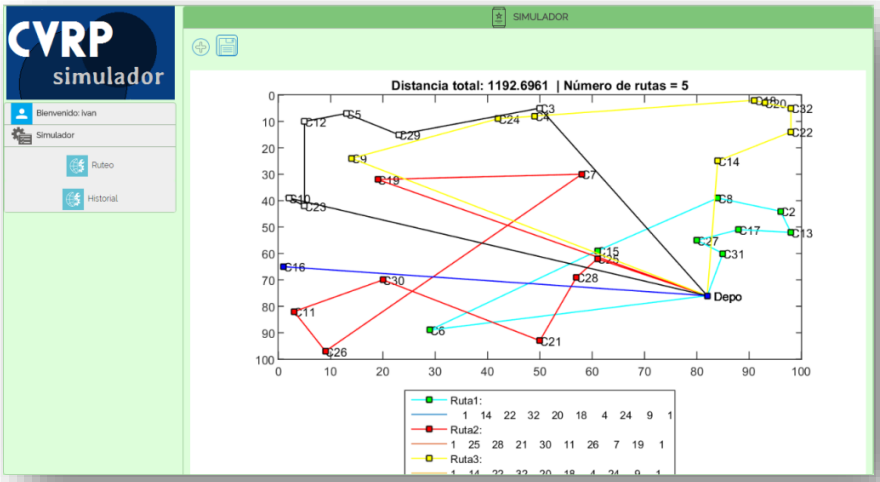

Fig. 8. Displaying of optimal routes in text and graph in client side.

4 Results and Discussion

We compared the performance of genetic algorithm approach regarding nearest neighbor algorithm for ten different .vrp instances (32, 33, 34, 36, 37, 38, 39, 44, 45 & 46). We compare resulting total distance and execution time for each instance.

Table 3. Genetic Algorithm versus nearest neighbor algorithm.

			Genetic Algorithm		Nearest Neighbor		Differences	
#	**Instance size**	**# of routes**	**Total distance**	**Execution time**	**Total distance**	**Execution time**	**Total distance**	**Execution time**
1	32	5	1192,6961	0,22122	1.042,8645	0,18527	-150	-0,03595
2	33	5	1046,4215	0,56151	968,8926	0,19729	-77,5289	-0,36422
3	34	5	1066,7455	0,21822	1035,0417	0,29952	-31,7038	**0,0813**
4	36	5	1112,45	0,21977	1047,6603	0,21634	-64,7897	-0,00343
5	37	5	1155,9072	0,25358	1080,5174	0,22331	-75,3898	-0,03027
6	38	5	1078,3103	0,22807	893,8056	0,23601	-184,5047	**0,00794**
7	39	5	1264,3359	0,13557	1181,4545	0,24701	-82,8814	**0,11144**
8	44	6	1300,0512	0,31112	1347,1924	0,29582	**47,1412**	-0,0153
9	45	6	1342,6026	0,31926	1406,3867	0,31129	**63,7841**	-0,00797
10	46	7	1350,0706	0,31857	1308,7354	0,32642	-41,3352	0,00785
						Average	-59,70398	-0,024861

For instances 44 and 45 Genetic Algorithm obtained better results in distance than the nearest neighbor, in all the other cases nearest neighbor was better. Regarding execution time, genetic algorithm performance was lower regarding NNA.

5 Conclusions and Further Work

The simulators are one of the most important parts of the optimization process, allowing the user to perform tests and avoid generation of extra expenses due errors, allowing the user to generate optimal solutions and carry them out in the industry.
A genetic algorithm allows the generation of optimal routes acceptably, however, there are other artificial intelligence algorithms that can compete, this is the reason why this simulator has open the possibility of testing different A.I. techniques.

Our future work is to implement the simulator not only for working with an algorithm, it means, if you do not get results the expected results, it will be possible to try different algorithms like ant colony, swarm particle optimization, among others. On the other hand, it aims to make the simulator to operate under a mobile platform such as Android and/or IOS. One of the most important challenging future work is making the change from graph nodes to real-world environment, that is to say, that plot really the streets where would the fleet of vehicles being working. We would to include time windows analysis in routing problems as described in [16].

References

1. Ahuja, R., Magnati, T., Orlin, J.: Network flows: theory, algorithms, and applications. Prentice Hall (1993)
2. Alander, J.: On optimal population size of genetic algorithms. In Proceedings CompEuro´92, Computer systems and Software Engineering, pp. 65–70 (1992)
3. Bremermann, H.: Optimization through evolution and recombination. In: Self-Organizing systems. Spartan Books (1962)
4. Butrón, A. A.: Asignación de rutas de vehículos para un sistema de recolección de residuos sólidos en la acera. Revista de Ingeniería, 13(5), pp. 5–11 (2001)
5. Contardo, C.: Formulación y solución de un problema de ruteo de vehículos con demanda variable en tiempo real, trasbordos y ventanas de tiempo. Universidad de Chile (2005)
6. Dorigo, M., Gambardella, L.: Biosystems. ELSEVIER, 43(2), pp. 73–81 (1997)
7. Fraser, A.S.: Simulation of genetic systems by automatic digital computers. I. Introduction. Aust. J. Biol. Sci., 10, pp. 484–491 (1957)
8. Garey, M., Johnson, D.: Computers and intractability. A Guide to the Theory of NP-Completeness, W. H. Freeman & Co. (1990)
9. Goldberg, D.: Genetic algorithms in Search Optimization and Machine Learning. Addison-Wesley (1989)
10. Hillier, F., Lieberman, G. J.: Introducción a la Investigación de Operaciones 8th ed. McGraw-Hill/Interamericana Editores (2006)
11. Laporte, G., Gendreau, M., Hertz, A.: An approximation algorithm for the traveling salesman problem with time windows. Operations Research, 45(4), pp. 639–641 (1998)
12. Olivera, A.: Heurísticas para problemas de ruteo de vehículos. Uruguay (2004)

13. Rocha, L., González, C., Orjuela, J.: Una revisión al estado del arte del problema de ruteo de Ingeniería: Evolución histórica y métodos de solución. Ingeniería 16(2), pp. 35–55 (2011)
14. Thompson, R.G., Van-Duin, J.H.: Vehicle routing and scheduling. Innovations in freight transportation, pp. 47–63 (2003)
15. Leyva-López, J.C.: A genetic algorithm application for individual and group multicriteria decision making. Computación y Sistemas, 4(2), pp. 183–188 (2000)
16. Pérez, N., Cuate, O., Schütze, O., Alvarado, A.: Integración de las preferencias de los usuarios en el proceso de toma de decisiones para problemas de optimización de muchos objetivos discretos. Computación y Sistemas, 20(4), pp. 589–607 (2016)
17. Hernández-Aguilar, J. A., Burlak, G., Lara, B.: Design and Implementation of an Advanced Security Remote Assessment System for Universities Using Data Mining. Computación y Sistemas, 13(4) (2010)
18. García-Mendoza, C.V., Juárez, M., et. al: Filtro estimador por deconvolución y pseudoinversa: descripción e implementación recursiva. Computación y Sistemas, 18(4), pp. 833–840 (2014)
19. Naredo, E., Duarte-Villaseñor, M.A., García-Ortega, M.D.J., Vázquez-López, C. E., Trujillo, L., Siordia, O.S.: Novelty Search for the Synthesis of Current Followers. Computación y Sistemas, 20(4), pp. 609–621 (2016)