A simulated annealing algorithm for solving a routing problem in the context of municipal solid waste collection

Matías Fermani[1], Diego Gabriel Rossit[1,2][0000-0002-8531-445X]

and Adrián Toncovich[1]

[1] Department of Engineering, Universidad Nacional del Sur, Argentina
[2] INMABB, Universidad Nacional del Sur (UNS)-CONICET, Argentina
mati.ferma@gmail.com, diego.rossit@uns.edu.ar, atoncovi@uns.edu.ar

Abstract. The management of the collection of Urban Solid Waste is a complex task for local governments since it consumes a large portion of their budget. Thus, the use of computer-aided tools to support decision-making can contribute to improve the efficiency of the system and reduce the associated costs. In the present work, a simulated annealing algorithm is proposed to address the problem of designing the routes of waste collection vehicles. The proposed algorithm is compared against two other metaheuristic algorithms: a *Large Neighborhood Search* (LNS) algorithm from the literature and a standard genetic algorithm. The evaluation is carried out on real instances of the city of Bahía Blanca and on benchmarks from the literature. The proposed algorithm was able to solve all the instances, having a performance similar to the LNS, while the standard genetic algorithm showed the worst results.

Keywords: Municipal Solid Waste, Vehicle Routing, Simulated annealing.

1 Introduction

The generation of urban solid waste (MSW) is an inalienable consequence of the development of modern cities. Likewise, its correct management is one of the key elements for the sustainable development of a city [1]. The stages in the reverse logistics chain of MSW are diverse. According to Tchobanoglous et al. [2], they can be classified into waste generation; handling, separation, accumulation and processing of waste at source; collection, transfer and transportation; separation, processing and transformation of solid waste; and finally disposition. In these stages there are a lot of decisions to be made. For this reason, proposing computer-aided tools that allow assisting decision-making agents can contribute to a more efficient use of resources.

This work focuses on the collection, transfer and transportation stage of waste. In particular, a simulated annealing based metaheuristic algorithm is proposed to solve this problem. The proposed algorithm is compared against other metaheuristic algorithms in the literature in order to study its performance. The experimentation is carried out on real instances of the city of Bahía Blanca, Argentina, as well as in well-known benchmarks of the literature.

This article is structured as follows. Section 2 presents the addressed problem and a bibliographic review of the main related works, Section 3 outlines the used resolution

algorithms, Section 4 develops the computational experimentation and, finally, Section 5 presents the main conclusions of this work.

2 Problem addressed

The management of urban solid waste in the city of Bahía Blanca is a crucial activity for local authorities, not only because of the broad environmental and social impact associated with its operation but also because it consumes a large part of the municipality's budgetary resources [33]. Therefore, approaches that allow optimization of collection logistics may be relevant to achieve a more efficient provision of the service [17, 34]. It is within this framework that proposals have been made to migrate from the current door-to-door collection system to a containerized system [17,33]. Continuing with these studies, the next section presents a mathematical model to plan the next stage in the reverse logistics chain of urban solid waste: the design of the collection routes for the waste accumulated in the containers. Likewise, a review of the main related works in the literature is carried out.

2.1 Mathematical model

The problem of collecting the accumulation in waste containers can be represented as a Capacitated Vehicle Routing Problem (CVRP). Thus, it can be modeled as follows. Given a set of containers C and a superset $\underline{C} = C \cup c_0$, where c_0 is the warehouse where the vehicles start and end their trips, the following variables and parameters are defined: binary variable x_{ij} that adopts value 1 if a vehicle uses the path from container $i \in C$ to container $j \in C$ and 0 otherwise; the continuous positive variable u_i that indicates the load of the vehicle before visiting container $i \in C$; parameter Q is defined as the capacity of the vehicle; parameter d_{ij} as the distance between containers $i \in C$ and container $j \in C$; and parameter q_i as the amount of waste to be collected at container $i \in C$. In this way, the following model is proposed, using the two-index formulation developed by Miller et al. [3] in Equations (1) - (6).

$$\text{Minimize } FO = \sum_{i,j \in \underline{C}} x_{ij} d_{ij} \tag{1}$$

Subject to:

$$\sum_{j \in \underline{C}} x_{ij} = 1, \forall\, i \in C \tag{2}$$

$$\sum_{j \in \underline{C}} x_{ji} = 1, \forall\, i \in C, \tag{3}$$

$$u_j - u_i \leq Q(1 - x_{ij}) - q_j, \forall\, i, j \in \underline{C} \tag{4}$$

$$q_i \leq u_i \leq Q, \forall\, i \in C \tag{5}$$

$$x_{ij} \in \{0,1\}, \forall\, i, j \in \underline{C} \tag{6}$$

The proposed objective is to minimize the total distance traveled expressed in Equation (1). Equations (2) and (3) guarantee that each container is visited only once, having a single successor container and a single predecessor container on the route. Equation (4) prevents the formation of subtours. Equation (5) ensures that the vehicle's capacity is not exceeded. Equation (6) establishes the binary nature of the variable x_{ij}.

2.2 Literature review

The problem of collection of containerized waste in a city has been frequently addressed in the literature. In Beliën et al. [4] and Han y Ponce Cueto [5], extensive reviews of these works are developed.
In the case of Argentina, some studies can be found on applications of VRP models to solve the problem of design of collection routes. For example, in Bonomo et al. [6] and Larrumbe [7] mathematical programming methods are implemented to schedule the collection routes for waste containers in the Southern area of the City of Buenos Aires. On the other hand, in Bonomo et al. [8] a different model is presented for this city, which aims at minimizing the travel distances simultaneously with minimizing wear and tear on vehicles. In the city of Concordia, Bertero [9] presents an application to design the city's collection routes, making an effort to minimize the number of turns to facilitate the implementation of the routes by the authorities. On the other hand, Bianchetti [10] exhibits an algorithm to solve the zoning of the city of San Miguel de Tucumán in order to optimize the use of resources, reassigning trucks to the downtown area of the city. In Braier et al. [11, 12], through mathematical programming models, the collection of recyclable waste is planned for the city of Morón. In Rossit et al. [13], a comprehensive approach is presented that determines the location of the containers, as well as the design of the routes that collect their content in simulated instances of Bahía Blanca. In the same city, Cavallin et al. [14, 15] present vehicle routing models with balancing the distances between the different routes to design the trips of the informal collectors of recyclable waste.

3 Proposed resolution method

As aforementioned, the collection of MSW in containers constitutes an application case of the VRP (Vehicle Routing Problem) problem. In computational complexity theory, this type of problem is classified as NP-hard [16], that is, it is at most as difficult as problems for which efficient solving algorithms that run in polynomial time have not yet been developed, regarding the size of the problem [17]. In this type of problems, metaheuristic tools allow obtaining good solutions in reasonable computational times [35]. The proposed resolution algorithm is based on a simulated annealing (SA) strategy and was adapted from the previous algorithm developed in Toncovich et al. [18,19]. SA is a local search-based method that was developed from an analogy with the physical phenomenon of annealing [20] to solve complex optimization problems. Local search methods search for the solution with the best value of the chosen criteria in the current solution environment, accept it as the current solution, and repeat this procedure until it is not possible to improve the solution in the explored environment. By

systematically applying this procedure, a local optimum for the problem is generally obtained. To avoid being trapped in a local optimum, a diversification mechanism must be incorporated in order to adequately explore the solutions space. In simulated annealing metaheuristics, the diversification strategy allows moves, with some probability, toward solutions that worsen the present value of the objective function.

To get a good approximation to the optimal solution of the problem during the search process, it is necessary to restart the search regularly from one of the solutions accepted during the search process selected at random. The SA algorithm incorporates the classic parameters of simulated annealing. The parameters and variables corresponding to the implemented SA algorithm are indicated below.

- t: current iteration.
- S_0: initial solution.
- S_A: current solution.
- S_c: candidate solution.
- $V(S)$: neighborhood fo solution S, given by the set of solutions that can be obtained from solution S through a basic perturbation.
- T: control parameter that simulates the temperature in the real annealing process. It is a positive value that varies within the interval$T \in [T_f, T_0]$ during the execution of the algorithm, where T_0 is the initial temperature and $T_0 > T_f$.
- N_T: number of iterations performed by the algorithm for a certain temperature value T.
- $\alpha(T)$: function that determines the cooling mechanism, i.e., how T varies during the process. In this case, it is a linear function of the form $\alpha(T) = \alpha T$, with $\alpha \in [0{,}8; 0{,}99]$.
- N_{cont}: the number of iteration without improving the objective function at the iteration t.
- N_{stop}: maximum allowable number of iterations without improvement.

The following pseudo-code is applied to determine a potentially optimal solution.

Simulated annealing algorithm

i. Initialization

An initial solution S_0 is created randomly.
The solution is evaluated, $FO(S_0)$, using Equation (1).
S_0 is accepted as current solution S_A, and the rest of the parameters are set to $N_{cont} = t = 0$, $T = T_0$.

ii. Iteration t

A random perturbation is performed S_A to create a new solution S_c such that $S_c \in V(S_A)$.
S_c is evaluated $FO(S_c)$.
The value FO is calculated as $FO = FO(S_c) - FO(S_A)$.
If $FO \leq 0$, $S_A = S_c$, $N_{cont} = 0$. *Obs:* S_c is accepted as the new incumbent solution.
Else, $p_A = e^{-\frac{FO}{T}}$. *Obs:* lS_c is accepted as the new incumbent solution with probability p_A.
$\beta = random[0,1]$:
If $\beta \leq p_A$, then $S_A = S_c$, $N_{cont} = 0$;

Else, $N_{cont} = N_{cont} + 1$.
$t = t + 1$
If t is a multiple of N_T, then $T = \alpha(T)$
If $N_{cont} = N_{stop}$ or $T \leq T_f$, then **return** S_A .

3.1 Baseline Metaheuristics

The metaheuristics for comparison are the Large Neighborhood Search (LNS) algorithm developed by Erdoğan [21] and a standard genetic algorithm.

Erdoğan's metaheuristic was adapted from the adaptive LNS heuristic developed by Pisinger and Ropke [22] which extends Shaw's original heuristic [23]. A simplified explanation of its operation and pseudo-code (see Fig. 1) can be found in Cavallin et al. [24].

The operation of the heuristic consists of two main stages. The first is the construction of an initial solution that is obtained by inserting clients into the available routes in accordance with the objective to be minimized (operator Sol-Inicial). Then an enhancement is made using the *Local-Search* operator that applies four local search techniques. The first three are known as *Exchange* (two clients are exchanged), *1-OPT* (a client is extracted from one route and reinserted into another) and *2-OPT* (two route segments are exchanged). These operators are detailed in Groër et al. [25]. The fourth local search operator is *Vehicle-Exchange*. This operator tries to exchange the vehicles assigned to two routes, which is why it is useful when working with a heterogeneous fleet.

The second stage is where you try to get an improvement on the solution from the previous stage. The *Destroy-And-Rebuild* operator is applied, which considerably alters the current solution to explore another region of the search space and thus be able to escape from local optimum. In particular, long stretches of routes are extracted from the current solution and the extracted clients are reinserted in those positions that minimize the objective function (Cost). A solution is accepted if it has a better cost than the best solution found so far or, if a solution does not have a better cost than the previous solution, it can even be accepted with a probability *p*-value. This process is repeated cyclically up to the time limit imposed by the user.

```
1: procedure LNS(input)          ▷ Como input se ingresa la información
   correspondiente a los clientes, vehículos y lugares de descarga (depots)
2:     S ← SolInicial()
3:     S' ← LocalSearch(S)
4:     BestS ← S'
5:     repeat
6:         S' ← DestroyAndRebuild(S')
7:         S' ← LocalSearch(S')
8:         if (Cost(S') <= Cost(S')) then
9:             BestS ← S'
10:        else
11:            p ← Random(0, 1)
12:            if (p <= pvalue) then
13:                S' ← BestS
14:            end if
15:        end if
16:    until Time elapsed > Time limit
17:    return BestS
18: end procedure
```

Fig. 1. LNS algorithm implemented in [21]. Image Source: [24].

The generic genetic algorithm was developed using the Distributed Evolutionary Algorithms in Python (DEAP) framework [26]. This has the following characteristics: Representation of solutions. Solutions are encoded as a permutation of integers of length equal to the number of containers n. Each index in the vector represents the visit order in the tour and the corresponding integer value represents one of the containers.
Initialization. The population of size *#P* is initialized by applying a random procedure to generate the permutations with a uniform distribution.
Genetic operators. The recombination operator is the Partially Mapped Crossover (PMX) applied on two selected individuals with pc probability. The mutation operator is based on Swap Mutation and swaps two elements of the permutation. The mutation applies to an individual with probability pm. The proposed operators guarantee the feasibility of the solution.
Selection. The selection is made through a binary tournament, from which the one with the best fitness is selected.
Replacement. In each iteration, the new population is made up of the best 10% (with better fitness) of the previous population and the rest of new individuals generated through genetic operators.
Fitness assessment. The fitness function is decoded by reading alleles from left to right and inserting a depot visit when necessary.
Parametric analysis. The parameters that were analyzed through a statistical analysis were the size of the population *#P* - the values 100 were considered; 150 and 200 - the probability of mutation p_m - values 0.05 were considered; 0.1; 0.15; 0.20 and 0.25- and p_c crossover -values 0.5; 0.6; 0.7; 0.8; 0.9 and 1-. Parametric analysis were carried out on the instances P-n16-k8, P-n19-k2 and P-n20-k2 proposed in [27]. For each instance and for each parametric configuration, 50 independent executions were carried out. The Shapiro-Wilk test was applied to study whether the fitness distribution had a normal distribution. Since many executions did not fit a normal distribution, the medians were

evaluated through the Friedman test, selecting the following configuration: $\#P$ = 200, p_c = 0.9 and p_m = 0.25.

Cut due to stagnation. A deadlock cut-off mechanism was incorporated. The algorithm stops when no improvement in the fitness of the best individual is obtained during a time interval equal to 50% of the maximum execution time (Eq. (7)) of the instance. Otherwise, it continues until the generation - evaluation, selection, crossing, mutation and replacement cycle - which has exceeded the maximum execution time at the end.

4 Results

The results obtained correspond to the resolution of four real scenarios of the MSW collection logistics of Bahía Blanca constructed in Herran Symonds [28] and Signorelli Nuñez [29]. Likewise, the experimentation corresponding to the metaheuristic approaches was extended with the resolution of problems of a benchmark of the literature.

4.1 Tests carried out on real scenarios

The scenarios presented in Herran Symonds [28] and Signorelli Nuñez [29] constitute real cases designed based on the existing MSW problem in the city of Bahía Blanca.

Two different plans for the disposal of the containers are studied, considering the collection of both waste fractions: both dry and wet. Said plans contemplate the option of separating the containers into wet and dry waste while considering the option of using different collection frequencies. The plans differ fundamentally in terms of the frequency with which it is proposed to collect dry waste, thus conforming to the following four scenarios:

Scenario 1:
- Collection frequency per week: 6.
- Type: humid waste.

Escenario 2:
- Collection frequency per week: 4.
- Type: dry waste.

Escenario 3:
- Collection frequency per week: 6.
- Type: humid waste.

Escenario 4:
- Collection frequency per week: 3.
- Type: dry waste.

It is necessary to clarify that the differences indicated between the different scenarios essentially affect the number of containers that will be used, however, the explanation of this fact is not part of the objectives set for this work.

For the exact resolution, it was implemented using CPLEX software version 12.6.0.0 in a GAMS environment, using the opportunistic parallelism version of said software, with the addition of the valid inequality of the minimum number of trips to accelerate convergence. A time limit was set for each run of 15,000 seconds.

Both the SA and LNS algorithms were programmed using the Visual Basic for Applications (VBA) interface of MS Excel. While, to carry out the GA algorithm, the DEAP library was used and Python was used as the programming language.

Each algorithm was run on different personal computers, performing 50 runs for each instance in order to obtain a more representative sample of the performance of the algorithms.

The results obtained are shown below in Tables 1 and 2. In the first one, the best distance -FO according to Eq. (1) - obtained for each MSW scenario with each of the proposed resolution methods can be observed. For the CPLEX solution, the gap calculated by the software is reported, which indicates that the optimal value was not found in scenarios 2 and 4. In the second table, three indicators of the sets of solutions obtained for each metaheuristic are contrasted, a measure of centralization, such as the average value of the results obtained, a measure of dispersion, that is, the standard deviation, and the average gap. The gap is calculated as the difference between the average value and the value obtained by CPLEX, divided by the latter.

Table 1. Results of the RSU problem: best solutions of each metaheuristic.

Scenario	Exact solver CPLEX		Best result of FO for each metaheuristic		
	FO	CPLEX *gap* %	SA	LNS	GA
Scenario 1	22613.90	0.00	22814.00	22877.00	36712.00
Scenario 2	41533.28*	26.35	43763.00	43491.00	48561.00
Scenario 3	27150.00	0.00	27707,00	27648.00	49463.00
Scenario 4	62800.63*	18,33	65995,00	65938.00	71185.00

* Since the optimal solution for these scenarios was not found, the values correspond to the best value found by CPLEX for the execution time (Upper Bound).

Table 2. Results of the RSU problem: average performance of each metaheuristic.

Scenario	SA			LNS			GA		
	Avg. FO	Std. dev.	*gap* %	Avg. FO	Std. dev.	*gap* %	Avg. FO	Std. dev.	*gap* %
Scenario 1	23310.12	216.82	3.08	23542.56	246.68	4.11	39041.70	1164.59	72.65
Scenario 2	44294.64	291.85	6.65	44229.92	303.77	6.49	51047.52	1247.72	22.91
Scenario 3	28143.00	296.63	3.66	28032.34	223.30	3.25	53408.00	1396.83	96.99
Scenario 4	66366.31	202.49	5.68	66356.19	180.61	5.66	73290.82	1096.94	16.70

The maximum execution time of all metaheuristics was established proportionally to the number of nodes in the instance -excluding the repository- and it comes from Equation (7).

$$Maximum\ Execution\ Time\ [sec] = \frac{(n°\ of\ nodes * 15 * 60)}{50} \quad (7)$$

It should be taken into account that both the SA and the GA described here have cut-off mechanisms due to stagnation that can lead to a real execution time that is less than the maximum. This is not the case for the LNS whose execution time will be equal to the maximum.
Although at the time of executing the RSU scenarios in CPLEX the time was limited to 15,000 seconds, in scenarios 1 and 3, in which the optimal value was reached, a cut-off was made earlier, having consumed 1892 seconds in the first scenario and 4367 seconds in scenario 3.
In order to evaluate its performance, Table 3 records the number of iterations required by each metaheuristic in the time established by the previously mentioned equation.

Table 3. Average time and iterations required for MSW scenarios.

Scenario	Maximum Execution Time (sec)	Average number of iterations		
		SA	LNS	GA
Scenario 1	1242	76860.12	33700.22	30739.22
Scenario 2	1242	211455.16	16724.16	50439.62
Scenario 3	1422	265653.78	11563.00	51479.40
Scenario 4	1422	1516847.40	6129.72	63967.20

4.2 Tests carried out on the benchmark of the literature

To extend the metaheuristic comparison, some instances of the benchmark set proposed in Christofides and Eilon [30], called Set E. Both the instances and the optimal solutions were downloaded from the VRP-REP digital repository [32].
Next, Table 4 shows the best results found in each instance by the algorithms under study, while Table 5 shows the performance of the metaheuristics through the same parameters used to evaluate the performance in RSU.

Table 4. Results of set E: best known solutions of each metaheuristic.

Instance	Optimal solution	Best result of each metaheuristic		
		SA	LNS	GA

E-n13-k4	247,00	247.00	247.00	247.00
E-n22-k4	375.00	375.00	375.00	375.00
E-n23-k3	569.00	569.00	619.00	569.00
E-n30-k3	534.00	503.00*	534.00	529.00*
E-n31-k7	379.00	379.00	379.00	427.00
E-n33-k4	835.00	835.00	835.00	859.00
E-n51-k5	521.00	521.00	521.00	582.00

* The solutions found have a lower value of FO, but a greater number of routes (k = 4) than the reference solution, a fact that is also observed in Table 5.

Table 5. Results of set E: Average performance of each metaheuristic.

Instance	SA			LNS			GA		
	Avg. FO	Std. dev.	*gap* %	Avg. FO	Std. dev.	*gap* %	Avg. FO	Std. dev.	*gap* %
E-n13-k4	247.08	0.27	0.03	247.00	0.00	0.00	247.72	1.09	0.29
E-n22-k4	376.14	3.24	0.30	375.00	0.00	0.00	393.94	14.90	5.05
E-n23-k3	569.04	0.20	0.01	619.00	0.00	8.79	625.08	41.06	9.85
E-n30-k3	503.04	0.28	-5.80	535.08	1.45	0.20	583.86	36.37	9.34
E-n31-k7	394.34	13.11	4.05	379.64	2.16	0.17	485.44	31.18	28.08
E-n33-k4	836.44	2.21	0.17	835.00	0.00	0.00	903.42	23.11	8.19
E-n51-k5	522.48	3.81	0.28	521.84	2.76	0.16	645.56	29.05	23.91

* The gap was calculated as the difference between the average and the optimal value, divided by the latter.

s in Table 3, the average of iterations reached in each metaheuristic in the time given by Equation (7) were recorded.

Table 6. Average time and iterations performed per instance.

Instance	Maximum Execution Time (sec)	Average number of iterations		
		SA	LNS	GA
E-n13-k4	216	76860.12	33700.22	30739.22
E-n22-k4	378	211455.16	16724.16	50439.62

E-n23-k3	396	265653.78	11563.16	51479.40
E-n30-k3	522	1516847.40	6129.72	63967.20
E-n31-k7	540	2390054.58	9992.40	6800.86
E-n33-k4	576	3614994.88	4502.92	68233.54
E-n51-k5	900	6460189.50	2788.66	91230.70

4.3 Analysis of resutls

From the analysis of the results it is verified that the three metaheuristic procedures are valid for use in solving CVRP-type problems.

It is necessary to clarify that both SA and GA find a value lower than the optimal value reported in the bibliography for the E-n30-k3 instance given that, since they do not have a restriction indicating the number of routes, they present a solution with an additional route. It is also for this reason that a negative value of the gap is observed in this instance within Table 5 for the SA.

The experience carried out on the benchmark and later on the RSU scenarios, marked a good performance both of the proposed algorithm SA and the LNS of the literature by yielding considerably small values of the average gap. On the other hand, GA showed a low performance compared to SA and LNS, which is again reflected in the average gap, possibly due to the use of a standard genetic algorithm with a simple decoding function taken from applications in unrestricted problems. as the Traveler Agent Problem.

Regarding the average iterations carried out, it is observed that SA and GA reach a greater number of iterations as the number of nodes grows, the same does not happen in the case of LNS for which where the number of iterations decreases with increasing number of nodes.

5 Conclusions

Finding new tools to make urban solid waste (MSW) logistics more efficient is a pressing concern in today's societies. This work is focused on solving waste collection problems for the Bahía Blanca area. In particular, it addresses a problem found in previous works where the exact resolution of waste collection problems was inefficient, because the partitioning of the scenarios into smaller portions was required to achieve convergence to an acceptable feasible solution.

Therefore, in this work a comparative study is presented between the exact resolution and three metaheuristic resolution tools for the problem of MSW collection in Bahía Blanca. The exact resolution is based on a mathematical programming formulation of the CVRP model using the CPLEX software. On the other hand, as regards metaheuristic resolution, a simulated annealing algorithm is proposed, which is

compared against an algorithm taken from the literature based on Large Neighborhood Search (LNS) and a standard genetic algorithm. In addition, the comparison of metaheuristics with some instances of a well-known benchmark from the literature is extended. The proposed simulated annealing algorithm has a similar performance to the LNS algorithm in the literature - obtaining values close to the optimal solution on several occasions - and markedly exceeds the standard genetic algorithm.

After the experimental work carried out, it can be affirmed that the application of these algorithms results in the obtaining of potentially acceptable and competent results, counting the metaheuristics with the advantage of the less computational effort required to reach the solution.

As lines of work in the future, it is planned to experiment with larger scenarios of the city of Bahía Blanca, using the proposed metaheuristics, which were validated in the present work. On the other hand, it is proposed to continue improving the mathematical programming formulation, implemented through CPLEX, by including additional valid inequalities that allow reducing resolution times. Furthermore, it is proposed to develop the coding function of the genetic algorithm with an approach that is better adapted to the type of CVRP problem addressed.

Acknowledgements

The authors of this work wish to acknowledge the funding received from the Universidad Nacional del Sur for the development of the projects PGI 24 / J084 and PGI 24 / ZJ35. In addition, the first author of this work is grateful for the funding received from the *Consejo Interuniversitario Nacional* of Argentina (CIN) through a scholarship *Becas de Estímulo a las Vocaciones Científicas (Becas EVC – CIN)*.

References

1. Hoornweg, D.; Bhada-Tata, P. (2012). “What a waste: a global review of solid waste management”. World Bank. Vol. 15, pág. 116. Washington DC, EUA.
2. Tchobanoglous, G., Kreith, F.; Williams, M. E. (2002). “Introduction”. En: Handbook of solid waste management. G. Tchobanoglous, F. Kreith (Editores). McGraw-Hill, EUA. Segunda edición. Capítulo 1.
3. Miller, C. E., Tucker, A. W.; Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. Journal of the ACM (JACM), 7(4), 326-329.
4. Beliën, J., De Boeck, L.; Van Ackere, J. (2012). “Municipal solid waste collection and management problems: a literature review”. Transportation Science. Vol. 48, no. 1, págs. 78-102.
5. Han, H.; Ponce Cueto, E. (2015). “Waste collection vehicle routing problem: literature review”. PROMET - Traffic & Transportation. Vol. 7, no. 4, págs. 345-358.
6. Bonomo, F., Durán, G., Larumbe, F.; Marenco, J. (2009). “Optimización de la Recolección de Residuos en la Zona Sur de la Ciudad de Buenos Aires”. Revista de Ingeniería de Sistemas. Vol. 23.
7. Larrumbe, F. (2009). Optimización de la Recolección de Residuos en la Zona Sur de la Ciudad de Buenos Aires. Tesis de Grado. Universidad de Buenos Aires. Buenos Aires, Argentina.

8. Bonomo, F., Durán, G., Larumbe, F.; Marenco, J. (2012). “A method for optimizing waste collection using mathematical programming: a Buenos Aires case study”. Waste Management & Research. Vol. 30, no. 3, págs. 311-324.
9. Bertero, F. (2015). Optimización de recorridos en ciudades. Una aplicación al sistema de recolección de residuos sólidos urbanos en el Municipio de Concordia. Tesina de Grado. Universidad Nacional de Rosario. Rosario, Argentina.
10. Bianchetti, M. L. (2015). Algoritmos de zonificación para recolección de residuos. Tesis de Grado. Universidad de Buenos Aires. Buenos Aires, Argentina.
11. Braier, G., Durán, G., Marenco, J.; Wesner, F. (2015). “Una aplicación del problema del cartero rural a la recolección de residuos reciclables en Argentina”. Revista de Ingeniería de Sistemas. Vol. 29.
12. Braier, G., Durán, G., Marenco, J.; Wesner, F. (2017). “An integer programming approach to a real-world recyclable waste collection problem in Argentina”. Waste Management & Research. Vol. 35, no. 5, págs. 525-533.
13. Rossit, D. G., Broz, D., Rossit, D. A., Frutos, M.; Tohmé, F. (2015). “Modelado de una red urbana de recolección de residuos plásticos en base a optimización multi-objetivo”. Proceedings of the XXVI EPIO and VIII RED-M. Bahía Blanca, Argentina.
14. Cavallin, A., Vigier, H. P.; Frutos, M. (2015a). “Aplicación de un modelo CVRP-RB a un caso de logística inversa”. Proceedings of the XXVI EPIO and VIII RED-M. Bahía Blanca, Argentina.
15. Cavallin, A., Vigier, H. P.; Frutos, M. (2015b). “Logística inversa y ruteo en el sector de recolección informal de residuos sólidos urbanos”. In: Avances en Gestión Integral de Residuos Sólidos Urbanos 2014-15. Instituto Nacional de Tecnología Industrial, Buenos Aires, Argentina. Pps. 37-49.
16. Lenstra, J. K.; Kan, A. R. (1981). “Complexity of vehicle routing and scheduling problems”. Networks. Vol. 11, no. 2, págs. 221-227.
17. Rossit, D. G., Toutouh, J., y Nesmachnow, S. (2020). Exact and heuristic approaches for multi-objective garbage accumulation points location in real scenarios. Waste Management, 105, 467-481.
18. Toncovich, A., Burgos, T.; Jalif., M. (2017). “Planificación de la logística de recolección de miel en una empresa apícola”. In proceedings of the X Congreso Argentino de Ingeniería Industrial. Ciudad Autónoma de Buenos Aires, Argentina. Pages. 397-406.
19. Toncovich, A., Rossit, D. A., Frutos, M.; Rossit, D. G. (2019). “Solving a multi-objective manufacturing cell scheduling problem with the consideration of warehouses using a Simulated Annealing based procedure”. International Journal of Industrial Engineering Computations. Vol. 10, no. 1, págs. 1-16.
20. Kirkpatrick, S.; Gelatt, C. D.; Vecchi, M. P. (1983). “Optimization by simulated annealing”. Science. Vol. 220, no. 4598, págs. 671-680.
21. Erdoğan, G. (2017). “An open source spreadsheet solver for vehicle routing problems”. Computers & Operations Research. Vol. 84, págs. 62-72.
22. Pisinger, D.; Ropke, S. (2007). “A general heuristic for vehicle routing problems”. Computers & Operations Research. Vol. 34, no. 8, págs. 2403-2435.
23. Shaw, P. (1998). “Using constraint programming and local search methods to solve vehicle routing problems”. Actas de la International Conference on Principles and Practice of Constraint Programming. Springer. Berlin, Heidelberg. Págs. 417-431.
24. Cavallin, A., Rossit, D. G., Savoretti, A. A., Sorichetti, A. E.; Frutos, M. (2017). “Logística inversa de residuos agroquímicos en Argentina: resolución heurística y exacta”. Actas del XV Simposio en Investigación Operativa - 46 Jornadas Argentinas de Informática e Investigación Operativa. Córdoba, Argentina.
25. Groër, C., Golden, B.; Wasil, E. (2010). “A library of local search heuristics for the vehicle routing problem”. Mathematical Programming Computation. Vol. 2, no. 2, págs. 79-101.

26. Fortin, F. A., De Rainville, F. M., Gardner, M. A. G., Parizeau, M., y Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. The Journal of Machine Learning Research, 13(1), 2171-2175.
27. Augerat, P. (1995). Approche polyèdrale du problème de tournées de véhicules. Tesis Doctoral, Institut polytechnique de Grenoble. Grenoble, Francia (1995).
28. Herran Symonds, V. (2019). Ubicación de contenedores diferenciados de RSU. Tesis Final de Carrera. Universidad Nacional del Sur. Bahía Blanca, Argentina.
29. Signorelli Nuñez, M. (2019). Análisis del sistema actual de recolección de RSU en el Barrio Universitario de la ciudad de Bahía Blanca. Tesis Final de Carrera. Universidad Nacional del Sur. Bahía Blanca, Argentina.
30. Christofides, N.; Eilon, S. (1969). "An algorithm for the vehicle-dispatching problem". Journal of the Operational Research Society. Vol. 20, no.3, págs. 309-318.
31. Christofides, N., Mingozzi, A.; Toth P. (1979). "The vehicle routing problem". En: Combinatorial optimization. N. Christofides, A. Mingozzi, P. Toth, C. Sandi (Editores). John Wiley, Chichester. Capítulo 14.
32. Mendoza, J.; Hoskins, M.; Guéret, C.; Pillac, V.; Vigo, D. (2014). "VRP-REP: a vehicle routing community repository". Actas del Third meeting of the EURO Working Group on Vehicle Routing and Logistics Optimization (VeRoLog'14). Oslo, Noruega.
33. Cavallin, A.; Rossit, D. G.; Herrán Symonds, V.; Rossit, D. A.; Frutos, M. (2020). Application of a methodology to design a municipal waste pre-collection network in real scenarios. Waste Management & Research, 38(1_suppl), 117-129.
34. Rossit, D. G.; Nesmachnow, S.; Toutouh, J. (2019). A bi-objective integer programming model for locating garbage accumulation points: a case study. Revista Facultad de Ingeniería Universidad de Antioquia, (93), 70-81.
35. Nesmachnow, S.; Rossit, D. G.; Toutouh, J. (2018). Comparison of multiobjective evolutionary algorithms for prioritized urban waste collection in Montevideo, Uruguay. Electronic Notes in Discrete Mathematics, 69, 93-100.