Available online at www.HighTechJournal.org

HighTech and Innovation Journal

ISSN: 2723-9535

Vol. 6, No. 3, September, 2025

Simulated Annealing Algorithm for Vehicle Routing with Stochastic Travel Times and Soft Time Windows

M. Jiménez-Carrión [1], Kenin M. Ojeda-Hidalgo [1*]

[1] *School of Industrial Engineering, Universidad Nacional de Piura, Castilla-Piura, 2002, Peru.*

Received 26 May 2025; Revised 13 August 2025; Accepted 19 August 2025; Published 01 September 2025

Abstract

In the context of urban logistics, uncertainty in travel times poses a critical challenge for planning efficient routes. A multi-objective simulated annealing (SA) algorithm is proposed to solve the Vehicle Routing Problem with Stochastic Travel Times and Soft Time Windows (VRPSTTW), prioritizing the minimization of the number of vehicles and travel costs. The methodology follows three phases: (1) calibration of the SA on 12 Solomon instances with 100 customers, achieving an average GAP of 1.9% and a maximum of 3.4%; (2) modeling of travel times using the Box-Muller transformation on 43,200 Google Maps records, segmented into four scenarios according to peak hours and type of day; and (3) parameter tuning through a 3^3 factorial design. With the optimal configuration (T_0 = 1500, α = 0.9, 4000 iterations), the algorithm solved a real instance with 100 customers in 0.5 minutes, achieving 10 vehicles, 614.7 km, 75 minutes of travel time, and a CV of 0.013%; perturbations of ±10% only increased the energy by 0.019%. Compared to recent literature, the distance was reduced by 3.2% without resorting to hybrid algorithms. The main novelty lies in integrating real traffic data and soft windows into a pure SA approach with complexity $O(C \cdot n^2)$, offering a robust, realistic, and scalable tool for dynamic urban environments.

Keywords: Simulated Annealing; VRPSTTW; Stochastic Travel Times; Multi-Objective Optimization; Google Maps.

1. Introduction

The Vehicle Routing Problem (VRP) has been widely studied in the scientific literature due to its relevance in optimizing distribution systems. Over time, various VRP variants have been developed to more accurately represent complex operational scenarios. Among these variants, the VRP with Time Windows and the VRP with Stochastic Travel Times stand out [1]. The combination of both approaches gives rise to the Vehicle Routing Problem with Stochastic Travel Times and Time Windows (VRPSTTW), a formulation that considers not only the time constraints imposed by customers but also the uncertainty in travel times caused by factors such as traffic or road conditions. This variant allows for a more realistic modeling of distribution systems by integrating stochastic elements and flexible time constraints that closely reflect real-world operating conditions.

The choice between hard or soft time windows directly depends on the type of constraints one aims to model, always considering realistic conditions. Unlike hard windows, which impose severe penalties for any deviation from the allowed interval, soft windows introduce a degree of flexibility by applying penalties proportional to the level of non-compliance. This approach is particularly useful in urban contexts with heavy traffic congestion, where strictly meeting each delivery time is often unfeasible. For example, a delivery company in Lima might face unexpected delays due to accidents or

* Corresponding author: 0502018043@alumnos.unp.edu.pe

http://dx.doi.org/10.28991/HIJ-2025-06-03-01

traffic detours; in such cases, imposing hard windows could lead to infeasible or excessively costly solutions, whereas using soft windows enables operational efficiency under such conditions.

Given that these factors play a crucial role in distribution logistics, developing a robust methodology becomes essential to transform and optimize these systems, enabling better connectivity and greater efficiency in meeting demand. To achieve this, it is necessary to assess the costs associated with distribution and apply strategies that improve route planning. In this context, the VRP is consolidated as a key mathematical model to address the challenges in assigning and optimizing routes for a fleet of vehicles, ensuring that services are more efficient, cost-effective, and adaptable to real-world conditions [2, 3].

Recent studies such as those by Abdullahi et al. [4] and Rajabi-Bahaabadi et al. [5] have addressed the VRP under scenarios with uncertainty but limit their focus to theoretical traffic distributions or simulations not connected to real data. In contrast, Muñoz-Villamizar et al. [6] have begun incorporating empirical information through the Google Maps API, applying it to the traditional VRP using a Mixed-Integer Linear Programming (MILP) model. While this approach is often less efficient than metaheuristic methods, it represents a significant advance by opening the possibility of realistically modeling the stochastic variable in more complex problem variants.

However, to date, no study has jointly addressed the use of stochastic travel times derived from real data and soft time windows within a metaheuristic optimization framework. This gap represents a critical omission in the current literature, especially in urban contexts where traffic variability and the need for time flexibility are constant.

This paper proposes a multi-objective algorithm to address the Vehicle Routing Problem with Stochastic Travel Times and Soft Time Windows (VRPSTTW). The approach focuses on minimizing the number of vehicles used and reducing the associated costs, considering travel distance, travel time, and penalties. The implemented methodology integrates stochastic elements and flexible time constraints, providing companies and service providers with an effective tool to optimize logistics planning and minimize operational costs in dynamic and realistic environments.

2. Literature Review

Combinatorial optimization has paid special attention to the Vehicle Routing Problem (VRP) due to its relevance in improving logistics efficiency and goods distribution. With the aim of providing a detailed classification of the various approaches developed, Braekers et al. (2009) [7] carried out an exhaustive taxonomic analysis of the VRP, offering a fundamental reference framework for subsequent research.

Among the different VRP variants, the Vehicle Routing Problem with Time Windows (VRPTW) stands out for its inherent complexity, being classified as an NP-hard problem. This difficulty has encouraged the use of approximate techniques based on search agents, such as metaheuristics. In this regard, Pratiwi et al. (2018) [8] proposed a solution based on nature-inspired algorithms, hybridizing the bat algorithm with simulated annealing to improve performance by replacing the worst generated solutions. Complementarily, Gibbons & Ombuki-Berman (2024) [9] developed a memetic algorithm (MA-BCRCD) for the VRPSPDTW, using real data and combining evolutionary techniques with local search, achieving better results than previous methods across all evaluated instances.

In parallel, uncertainty in travel times has become a critical dimension in the study of stochastic routing problems. The first model to formally address the VRP with Stochastic Travel Times (VRPST) was presented by Laporte et al. (1992) [10]. In their proposal, they considered vehicles without capacity limitations, using a formulation based on chance constraints and simple recourse stochastic programming, which was solved via a branch-and-cut strategy on small networks (10 to 20 nodes and up to five scenarios). Later, Guevara et al. (2025) [11] tackled travel and service time stochasticity through a simheuristic approach combining Tabu Search and Monte Carlo simulation, evaluating solutions under probabilistic scenarios and improving performance compared to deterministic approaches. Meanwhile, Van Woensel et al. (2008) [12] incorporated traffic congestion as a source of randomness in travel times, modeling it using queuing theory and applying optimization techniques based on Tabu Search.

The integration of time windows and stochastic travel times gave rise to the Vehicle Routing Problem with Stochastic Travel Times and Soft Time Windows (VRPSTTW), a variant that allows for more accurate modeling of logistics scenarios where soft time constraints and travel time uncertainty interact significantly. Nguyen et al. (2016) [13] developed a Tabu Search-based algorithm to address this variant, focusing on operational scenarios closer to real conditions. However, their proposal was tested and validated only on the classic Solomon benchmark instances [14], without considering critical factors such as real traffic, weather conditions, or specific operational constraints. This omission limits the applicability of the model in highly dynamic and unpredictable contexts, reducing its effectiveness in complex logistics environments.

In contrast, the present proposal aims to take this approach one step further, using the real-world environment as a testing ground to evaluate the algorithm's effectiveness in authentic and global scenarios. In this way, not only is the performance validated on the Solomon instances, but it also pushes towards the development of a VRPSTTW that can be universally applied, leveraging available technology to integrate real-world data and bring the solution closer to more complex operational contexts.

2.1. The Simulated Annealing Algorithm

Simulated annealing has been widely used in combinatorial optimization problems due to its ability to escape local optima by probabilistically accepting worse solutions during the early stages of the process. This probabilistic strategy allows for a more effective exploration of the solution space than deterministic approaches, which is particularly useful in routing problems with complex constraints, such as the VRPSTTW.

Kirkpatrick et al. (1983) [15] proposed a metaheuristic inspired by the physical annealing process in metals, where the metal is heated to a high temperature and then gradually cooled at a controlled rate. During this process, multiple solutions are generated and evaluated using an energy function. As the temperature decreases, the probability of accepting lower-quality solutions also decreases, following a probabilistic function that depends on the current temperature and the change in the objective function. This strategy enables the algorithm to escape local optima and explore potentially better solutions.

Černý (1985) [16] described the simulated annealing algorithm in the following stages:

- Stage 1 (Parameters): In a simulated annealing algorithm, three fundamental parameters are defined: Initial temperature, Cooling schedule, and maximum number of iterations. These parameters determine the behavior of the algorithm during the search and solution adjustment process.
- Stage 2 (Initial Solution): A list is created containing the indices of the data points that should be included in the solution, which are randomly distributed to build the initial solution.
- Stage 3 (Generation of Neighboring Solutions): The current solution is altered through a combinatorial process, generating a neighboring solution that can meet the problem's requirements.
- Stage 4 (Energy Evaluation): The energy is calculated as the value of the problem's objective function, which allows quantifying the current state of the system. This energy value facilitates the evaluation of the new solution, providing a clear metric to compare its quality with the previous solution.
- Stage 5 (Acceptance of New Solutions): If the new solution is better, it is accepted. If it is worse, it is accepted with a probability that depends on the temperature and the change in energy, see Equation 1.

$$P = exp\ (-\Delta E/T) \tag{1}$$

where ΔE is the change in energy and T is the current temperature.

- Stage 6 (Cooling): The temperature is gradually reduced according to a cooling schedule. A common schedule is geometric cooling, where the temperature is reduced by multiplying it by a constant less than 1, see Equation 2.

$$T = \alpha(T)\ with\ 0 < \alpha < 1 \tag{2}$$

- Stage 7 (Repetition): The previous steps are repeated until a stopping criterion is reached, such as a minimum temperature or a maximum number of iterations.

2.2. Stochastic Travel Time Model

In routing problems where travel times show high variability due to factors such as traffic, weather conditions, or unplanned events, stochastic modeling becomes an essential tool to capture this uncertainty and reflect more realistic operational scenarios. In this context, Papacostas & Prevedouros (1993) [17] analyzed how factors affecting travel times, although they may individually exhibit different probability distributions, tend to converge towards a normal distribution when grouped into defined intervals and their means are considered. This aligns with the findings of Mazmanyan & Trietsch (2013) [18], who argue that the sum of multiple independent segments tends to approximate a normal distribution, based on the Central Limit Theorem.

The mathematical representation of the normal distribution, considering the means or sums of historical travel times, is expressed as follows (see Equation 3):

$$f(t) = N(\mu, \sigma) \tag{3}$$

where N is Normal distribution, μ is Mean of the averages of the defined travel time intervals, and σ is Standard deviation of the averages of the defined travel time intervals.

Papacostas & Prevedouros (1993) [17] also explained that, in order to express variability, the Box-Muller transformation must be performed. This method allows generating a pair of normally distributed random numbers from uniformly distributed random numbers, following this methodology:

- Two uniformly distributed random numbers U1 y U2 are generated in the interval (0,1).

- The Box-Muller transformation is then applied to convert these numbers so they follow a standard normal distribution, as shown in Equations 4 and 5:

$$Z_O = \sqrt{-2lnU_1}.cos(2\pi U_2) \quad (4)$$

$$Z_1 = \sqrt{-2lnU_1}sen(2\pi U_2) \quad (5)$$

- To define a 95% confidence interval, Equations 6 and 7 are used, where Z_{95} is the critical value corresponding to a 95% confidence level (generally $Z_{95} \approx 1.96$).

$$LI = \mu - \sigma Z_{95} \quad (6)$$

$$LS = \mu + \sigma Z_{95} \quad (7)$$

- These generated numbers are then scaled using the mean μ and standard deviation σ, as shown in Equations 8 and 9:

$$X = \mu + \sigma Z_O \quad (8)$$

$$Y = \mu + \sigma Z_1 \quad (9)$$

- These two generated numbers represent the variability of the data and help form a normal distribution of travel times according to the independent factors.

3. Research Methodology

This research is methodologically structured into four key phases, progressing from a deterministic model to its stochastic extension. In the first phase, the mathematical formulation of the problem is carried out, specifying the objective function, capacity constraints, and time windows. In addition, stochastic parameters are introduced to model the variability in travel times, laying the foundation for the subsequent implementation of the simulated annealing algorithm.

The second phase involves implementing the SA under a deterministic approach, using the Solomon instances with 100 customers. The objective is to minimize the number of vehicles and the total distance traveled. This stage allows validating the effectiveness of the algorithm in complex environments, establishing a robust starting point for its extension to the stochastic context. In the third phase, stochasticity is incorporated through a model based on the normal distribution, supported by the Central Limit Theorem. The validity of this approximation is verified using the Kolmogorov-Smirnov test, with adjustments to the sample size in cases where the normality hypothesis is rejected.

The fourth phase consolidates the complete algorithm for the VRPSTTW, integrating deterministic and stochastic components into a unified structure. This stage allows evaluating the model's performance on a representative instance built with real coordinates, with the aim of demonstrating its accuracy, robustness, and applicability in scenarios close to real operational contexts. Finally, the findings obtained in this research were compared with previous studies, which allowed validating the proposed contributions. The conclusions present a synthesis of the main results achieved. The procedure followed is shown in Figure 1.

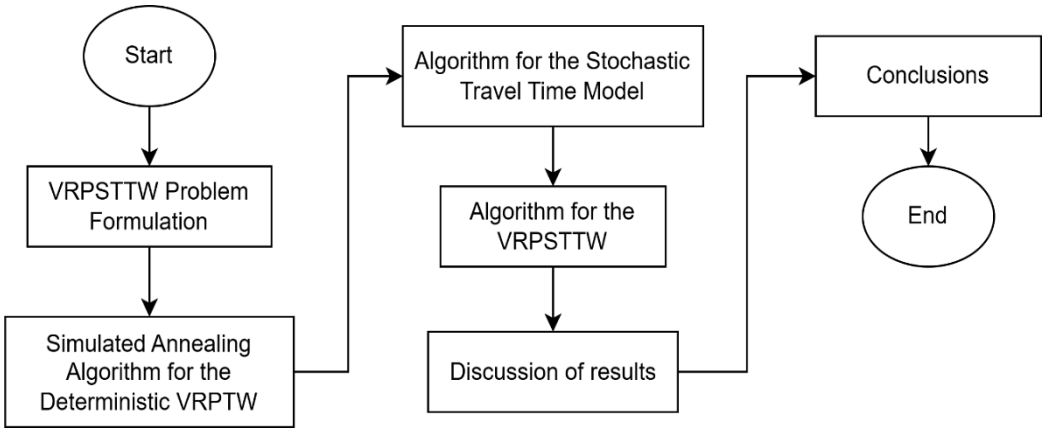

Figure 1. Research procedures

3.1. Formulation of the VRPSTTW Problem

The Vehicle Routing Problem with Stochastic Travel Times and Soft Time Windows (VRPSTTW) is formally defined through a mathematical model that specifies the objective function, constraints, and relevant variables. This

formulation structures the problem as a graph $G = (V,A)$, where $V = \{0, 1, ..., ..., n\}$ represents the set of vertices and A the set of arcs. The vertices $i = 1, ..., n$ correspond to the customers, each with an associated demand $d_i > 0$, while vertex 0 denotes the depot. The cost C is associated with each arc (i, j), representing the cost of traveling between vertices i and j. The full model formulation can be found in Equations 10 to 20.

The units used in the algorithm to represent distance, time, and weight are kilometers (km), minutes (min), and kilograms (kg), respectively. This standardization of parameters facilitates the interpretation of results and ensures consistency throughout the model.

Where:

$x_{ij}: \begin{cases} 1, if\ it\ goes\ from\ city\ i\ to\ city\ j \\ 0, otherwise \end{cases}$

Y: Total travel cost

v: {0, . . ., k} vehicles.

n: Number of customers

D_{ij}: Travel distance from customer i to customer j

t_{ij}: Travel time from customer i to customer j

P_{ij}: Penalty cost incurred when traveling from customer i to customer j

d_i: Demand of customer i

C_i: Vehicle capacity upon arriving at city i.

Q: Vehicle capacity limit

$e_i\ y\ l_i$: Time windows set by customer i

λ: Penalty coefficient

s_i: Service time at customer i

b_i^v: Arrival time of vehicle v at customer i

α: Weight assigned to the number of vehicles in the objective function

β: Weight assigned to costs (distance, time, and penalties) in the objective function

$$MIN\ Y = \alpha\ MIN \sum_{v=1}^{k} X^v + \beta\ MIN \left(\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{v=1}^{k} D_{ij} X_{ij}^v + \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{v=1}^{k} t_{ij} X_{ij}^v + \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{v=1}^{k} P_{ij} X_{ij}^v \right) \quad (10)$$

- Multi-objective function: Minimize the number of vehicles and the costs associated with the route (distance, time and penalties).

Having as constraints:

- Each customer must be served by exactly one vehicle.

$$\sum_{i=0}^{n} \sum_{v=1}^{k} x_{ij}^v = 1;\ \forall\ j = 1, \ldots, n \quad (11)$$

- Each vehicle must start its route at the depot.

$$\sum_{j=1}^{n} x_{0j}^v = 1; \forall\ v = 1, \ldots, k \quad (12)$$

- Each vehicle must return to the depot.

$$\sum_{j=1}^{n} x_{0j}^v = 1; \forall\ v = 1, \ldots, k \quad (13)$$

- Each customer i must be fully served by vehicle v.

$$C_i^v \geq d_i; \forall\ i = 1, \ldots, n;\ v = 1, \ldots, k \quad (14)$$

- Each vehicle must not exceed its capacity limit with respect to total delivery.

$$\sum_{i=0}^{n} C_i^v \leq Q; \forall\ v = 1, \ldots, k \quad (15)$$

- Time window constraints must be satisfied.

$$e_i \leq b_i^v \leq l_i; \forall\, i, j = 0, \ldots, n;\ v = 1, \ldots, k \tag{16}$$

- In problems with soft time windows, penalties are incurred for early or late arrivals and unmet demand

$$P_{ij} = \lambda \,.\max(e_i - b_i^v, 0) + \lambda.\max(b_i^v - l_i, 0) + \lambda \,.\max(d_i - C_i^v, 0) \tag{17}$$

- Vehicle v must start serving customer j only when the sum of travel time from customer i to j, service time at iii, and arrival time at iii is greater than or equal to the arrival time at j.

$$x_{ij}^v\left(b_i^v + s_i + t_{ij} - b_j^v\right) \leq 0; \forall\, i = 1, \ldots, n;\ v = 1, \ldots, k \tag{18}$$

- Define the types of variables to be used.

$$x_{ij}^v \in \{0,1\}\, y\, Y_i^v \geq 0;\ \forall\, i = 1, \ldots, n;\ \forall\, j = 1, \ldots, n;\ v = 1, \ldots, k \tag{19}$$

- Ensure that with a probability of at least α, the vehicle arrives at customer j after departing from i and traveling time t_{ij}, directly incorporating travel time uncertainty into the formulation;

$$\mathbb{P}\left(b_j^v \geq b_i^v + t_{ij} X_{ij}^v\right) \geq \alpha \tag{20}$$

In the context of the VRPSTTW, the primary objective is to minimize the number of vehicles used, as this represents the highest operational cost. Subsequently, distance, travel time, and penalty costs are minimized. To establish this objective hierarchy, the approach of Solomon (1987) [14] and Wei et al. (2024) [19] is adopted, assigning α = 1000 to heavily penalize additional vehicle usage and β = 1 to the costs associated with the route.

3.2. Simulated Annealing Algorithm for the Deterministic VRPTW

In this stage, an algorithm based on the simulated annealing (SA) metaheuristic is implemented to address the deterministic variant of the Vehicle Routing Problem with Time Windows (VRPTW). This methodology builds upon the previously described theoretical framework and is articulated through a set of essential functions designed to manage the algorithm's parameters and guide the iterative search process.

The algorithm begins by collecting the parameters that define the problem instance, including the number of customers, the time windows associated with each customer, and the distance and travel time matrices. Based on this input, the main simulated annealing functions are executed, structured into the following specific phases:

Initial Solution Creation Function: This function generates a feasible initial solution by assigning all customers to routes based on vehicle capacity constraints, using a predefined heuristic. The detailed procedure is shown in Algorithm 1.

Algorithm 1. Initial Solution Creation Function

```
Input: VRPTW Instance
1. Initialize Solution ← []
2. Initialize Used_Vehicles ← 0
3. Sort Customers by descending Demand
4. WHILE Clients is not empty DO
5.     Create New_Route ← []
6.     Assign vehicle to New_Route
7.     FOR each Customer in Customers DO
8.         IF Demand (Customer) + Current_Load <= Vehicle_Capacity THEN
9.             Add Customer to New_Route
10.            Remove Customer from Customer
11         END IF
12.    END FOR
13.    Add New_Route to Solution
14.    Increase Used_Vehicles by 1
15. END WHILE
16. Return Solution, Used_Vehicles
```

- Neighbor Solution Generation Function: This function generates new solutions from the current one by applying neighborhood operators such as swap, relocate, 2-opt and merge routes, thus allowing exploration of the solution space. The procedure is detailed in Algorithm 2.

Algorithm 2. Neighboring Solutions Generation Function

```
Input: Solution
1. Clone Solution to Neighbor
2. Select a random operator from {swap, relocate, 2-opt, merge_routes}
3. Apply selected operator to Neighbor
4. IF operator ∈ {"swap", "relocate", "2-opt"} THEN
5.     Reorder clients within same route or between routes
6.     Check if the new structure reduces Total_Distance or Total_Time
7.     Verify Neighbor Feasibility
8. ELSE IF operator == "merge_routes" THEN
9.     Select two routes r1 and r2
10.    Check whether they can be merged without exceeding capacity
11.    If possible, Combine r1 and r2 into a single route
12.    Decrease Used_Vehicles by 1
13. END IF
14. Return Neighbor, Used_Vehicles
```

- Energy Calculation Function: This function evaluates the quality of a solution by calculating its energy using the problem's objective function, see Equation 21.

$$F(\text{Energy}) = 1000\ (\text{Used\_Vehicles}) + \text{Total\_Distance} + \text{Total\_Time} + \text{Penalties} \quad (21)$$

This evaluation considers the two main components of the multi-objective function: the number of vehicles used and the associated travel costs, which include the total travel distance (in kilometers), total travel time (in minutes), and penalties for time window violations and capacity overruns. The resulting energy is a dimensionless quantity that allows for unified solution comparison. The detailed procedure is presented in Algorithm 3.

Algorithm 3. Energy Calculation Function

```
Input: Solution, Used_Vehicles
1. Initialize Total_Distance ← 0
2. Initialize Total_Time ← 0
3. Initialize Penalty ← 0
4. FOR each Route in Solution DO
5.     Initialize Current_Capacity ← Vehicle_Capacity
6.     FOR each customer in route DO
7.         Total_Distance ← Total_Distance + Distance (Prev_Customer,
           Curr_Customer)
8.         Total_Time ← Total_Time + Time (Prev_Customer, Curr_Customer)
9.         IF b_i < e_i OR b_i > l_i THEN
10.            Penalty ← Penalty + CalculateTimeWindowPenalty (Customer)
11.        END IF
12.        IF Demand (Customer) > Current_Capacity THEN
13.            Penalty + CalculateCapacityPenalty (Customer)
14.            Current_Capacity ← Current_Capacity - Demand (Customer)
15.        END IF
14.     END FOR
15. END FOR
16. Calculate Energy ← 1000* Vehicles_Used + 1*(Total_Distance + Total_Time +
    Penalty)
17. Return Energy
```

Once the functions are defined, the main function is implemented, integrating and executing each of the previously described functions. This function manages the simulated annealing process, continuously evaluating and updating the best solution found throughout the search. The detailed procedure can be found in Algorithm 4.

Algorithm 4. Simulated Annealing Main Function

```
Input: VRPTW Instance
1. Define SA parameters: T_max, T_min, Cooling_Rate, Max_Iter
2. Load data from the VRPSTTW instance
3. Current_Solution, Current_Used_Vehicles ← Initial Solution Creation Function
4. Current_Energy ← Energy Calculation Function(Current_Solution, Current_Used_Vehicles)
5. Best_Solution ← Current_Solution
6. Best_Energy ← Current_Energy
7. T ← T_max
8. Iterations ← 0
9. WHILE Max_Iter > Iterations and T > T_min DO
10.    Neighbor_Solution, Neighbor_Used_Vehicles ← Neighbor Solution Generation Function
11.    Neighbor_Energy ← Energy Calculation Function (Neighbor_Solution,
        Neighbor_Used_Vehicles)
12.    ΔE ← Neighbor_Energy - Current_Energy
13.    IF ΔE < 0 (better solution) OR exp (-ΔE / Temp_Initial) > random (0,1) THEN
14.        Current_Solution ← Neighbor_Solution
15.        Current_Used_Vehicles ← Neighbor_Used_Vehicles
16.        Current_Energy ← Neighbor_Energy
17.    END if
18.    IF Current_Energy < Best_Energy THEN
19.        Best_Solution ← Current_Solution
20.        Best_Energy ← Current_Energy
21.    END IF
22.    T ← T * Cooling_Rate
23. END WHILE
24. RETURN Best_Solution, Best_Energy
```

To validate the proposed algorithm in solving the VRPTW, the set of instances introduced by Solomon (1987) [14], composed of 100 customers, is used. This validation aims to establish a standardized benchmark to evaluate the effectiveness of simulated annealing in highly complex scenarios due to the large number of customers. In this way, both the algorithm's ability to handle complex instances and its efficiency in approximating optimal solutions are verified.

This evaluation is fundamental as it provides a solid foundation on which the stochastic travel time model will later be integrated, allowing any performance variations to be attributed exclusively to the incorporation of uncertainty and not to deficiencies in the base VRPTW structure.

For the analysis, three instances from each of the classes C1, C2, R1, and R2 are randomly selected, allowing for a balanced evaluation across different problem scenarios. The results obtained are presented in Table 1.

Table 1. Results in the Solomon Instances

Instance	Iterations			
C101	10V, 828.94	10V, 828.94	10V, 828.94	10V, 828.94
C104	10V, 867.89	10V, 824.78	10V, 824.78	10V, 824.78
C108	10V, 828.94	10V, 828.94	10V, 828.94	10V, 828.94
C202	3V, 591.56	3V, 591.56	3V, 591.56	3V, 591.56
C204	3V, 604.93	3V, 604.63	3V, 590.60	3V, 590.60
C205	3V, 588.88	3V, 588.88	3V, 588.88	3V, 588.88
R101	19V, 1730.75	19V, 1682.69	19V, 1652.44	19V, 1648.09
R103	13V, 1338.49	13V, 1324.44	13V, 1316.20	13V, 1292.68
R107	10V, 1139.40	10V, 1127.16	10V, 1107.50	10V, 1104.66
R201	4V, 1326.03	4V, 1318.75	4V, 1253.23	4V, 1252.37
R204	2V, 899.51	2V, 849.57	2V, 825.52	2V, 825.52
R205	3V, 1041.35	3V, 1027.22	3V, 1027.08	3V, 1018.15

To assess solution quality, the relative Gap (%) between the average distance of the solutions generated by the algorithm and the best-known solution is calculated, see Equation 22.

$$Gap(\%) = 100 * \left(\frac{Average\ Distance - Best\ Known\ Distance}{Best\ Known\ Distance}\right) \quad (22)$$

The results are shown in Table 2.

Table 2. Gap Analysis for each Instance

Instance	Average Vehicles	Average Distance	Best Known	Gap
C101	10	828.94	10V, 828.94	0
C104	10	835.55	10V, 824.78	1.31
C108	10	828.94	10V, 828.94	0
C202	3	591.56	3V, 591.56	0
C204	3	597,69	3V, 590.60	1.2
C205	3	588.88	3V, 588.88	0
R101	19	1678.49	19V, 1645.79	1.98
R103	13	1317.95	13V, 1292.68	1.95
R107	10	1119.68	10V, 1104.66	1.35
R201	4	1287.59	4V, 1251.37	2.89
R204	2	850.03	2V, 825.52	2.96
R205	3	1028.45	3V, 994.42	3.42

As a result, the simulated annealing algorithm has shown favorable performance in solving the VRPTW, reaching the best-known solution in several instances, particularly in type C cases, where the Gap is 0% in multiple scenarios. However, in more complex instances, such as type R, the Gap ranges from 1.35% to 3.42%, showing slight deviations from the best-reported solutions. Despite these differences, the algorithm achieves results close to optimal values; in 33% of the evaluated instances, it matches the best-known solution, while in the remaining 67%, the generated solutions are within 3.42% of the optimal value.

The observed variability in the GAP values suggests that algorithm performance may depend on problem structure, highlighting that the analyzed instances are limited to 100 customers. Nevertheless, the results obtained in this phase not only confirm the effectiveness of simulated annealing in solving the VRPTW but also provide an essential comparative baseline for the subsequent incorporation of the stochastic travel time model.

3.3. Stochastic Travel Time Model Algorithm

In this stage, the stochastic travel time model is implemented using data provided by Google Maps due to its broad coverage and high accuracy in representing the global road network. This data source allows access to historical and real-time traffic information, which is key to modeling travel time variability. Based on this information, a stochastic model is structured using empirical data, integrating probabilistic components that more realistically reflect fluctuations in travel durations.

3.3.1. Determination of Time and Distance

Based solely on the coordinates provided by the user, a function was developed to obtain route distances and corresponding travel times under normal traffic conditions — that is, when traffic is neither particularly light nor heavy — using data retrieved from Google Maps. This process is illustrated in the pseudocode presented in Algorithm 5

Algorithm 5. Time and Distance Matrix Function

```
Input: List of locations
1. Initialize Time_Matrix ← [], Distances_Matrix ← []
2. FOR each Origin in Locations DO
3.     Initialize Row_Times ← [], Row_Distances ← []
4.     FOR each Destination in Locations DO
5.         IF Origin = Destination THEN
6.             Travel_Time ← 0, Distance ← 0
7.         ELSE
8.            Retrieve Travel_Time and Distance as a response from Google Maps API
9.         END IF
10.        Add Travel_Time to Row_Times and Distance to Row_Distances
11.    END FOR
12.    Add Row_Times to Time_Matrix and Row_Distances to Distances_Matrix
13. END FOR
14. Return Time_Matrix, Distances_Matrix
```

3.3.2. Scenario Definition and Historical Data Collection

In this stage, traffic scenarios are characterized considering both peak periods and day types. Peak hours are associated with commuting movements at the start and end of the workday, a recurring pattern in urban contexts [20]. Global studies, such as those by the Institute of Transportation Engineers (ITE), place these periods between 7:00–9:00 a.m., 12:00–2:00 p.m., and 5:00–7:00 p.m., ranges commonly used to analyze traffic in different cities [21].

However, vehicle congestion does not depend solely on the hour but also on the day type. During weekdays, work and school activities significantly increase vehicle demand, whereas on non-working days, traffic patterns vary, maintaining certain peaks in commercial and recreational areas [20, 22].

Therefore, the scenarios defined in this study combine both parameters to capture travel time variability:

- Scenario 1: Non-peak hour – Weekday
- Scenario 2: Peak hour – Weekday
- Scenario 3: Non-peak hour - Non-working Day
- Scenario 4: Peak Hour - Non-Working Day

This structure facilitates observing travel time fluctuations under different contexts, establishing a solid foundation for implementing the stochastic model and subsequently evaluating it. Based on the scenario definitions, a function was developed to collect historical travel time data, enabling analysis of behavior and characterization of variability in each context.

- Travel time collection: Using the origin and destination points provided by the user, historical travel time data from Google Maps is accessed. Over a 30-day range prior to the routing date, travel times were recorded every minute, generating a total of 43,200 data points.
- Data segmentation: The 43,200 data points are segmented according to the four defined scenarios, organized by time intervals and day types. The distribution per scenario is as follows: Peak hour/weekday: 8,400 data points; Peak hour/non-working day: 4,200; Non-peak hour/weekday: 20,400; Non-peak hour/non-working day: 10,200.
- Hourly average calculation: Within the data segmented into the four scenarios, the data is divided into 1-hour intervals, and the average travel time in each interval is calculated. This hourly grouping allows capturing stochastic variability homogeneously within each scenario. The number of hourly averages obtained per scenario is as follows: non-peak hour/weekday: 340 averages (20,400 data points divided into 60-minute blocks), peak hour/weekday: 140 averages (8,400 points), non-peak hour/non-working day: 170 averages (10,200 points), and peak hour/non-working day: 70 averages (4,200 points).
- Kolmogorov-Smirnov test: This procedure is fundamental, as it validates the normality assumption underlying the stochastic model. Although the Central Limit Theorem states that a sample composed of more than 30 averages is generally sufficient to approximate a normal distribution, the Kolmogorov-Smirnov test is used to empirically verify this condition. This test is applied to the hourly averages obtained in each scenario, evaluating whether they follow a normal distribution. If any scenario does not meet this criterion, the algorithm issues an alert, indicating the need to increase the sample size to ensure statistical validity, and restarts the process.
- Statistical Analysis: Once normality is validated, the hourly averages are analyzed per scenario to calculate the mean and standard deviation, which are then used to represent their characteristic variability.
- Data insertion: The results obtained from the statistical analysis are organized into four matrices, each corresponding to a specific scenario. These matrices are consolidated and saved in a CSV file named Combined_data.csv, remaining available for later use in the model.

The entire process is detailed in the pseudocode of algorithm 6.

Algorithm 6. Historical Data Analysis Function

```
Input:
- List of Locations (Origin and Destination)
- Start date of the routing plan
Now ← get current date and time
Dates ← generate list of datetime values in 1-minute increments, going back 30 days from the start date
Create empty matrices: Weekday_Peak, Weekday_NonPeak, NonWorkingday_Peak, NonWorkingday_NonPeak
FOR each pair (Origin, Destination) where Origin <> Destination DO
    Data ← get travel times for each datetime value in Dates using Google Maps API
    IF Data is empty THEN continue to the next pair.
    FOR each record in Data DO
        Hour ← extract hour of record
        Day ← extract day of record
        Peak_Hour ← "Yes" IF ∈ {7-9, 12-14, 17-20}, ELSE "No"
        Weekday ← "Weekday" IF day ∈ {Monday-Friday}, ELSE "NonWorkingday"
        IF Peak_Hour = "Yes" AND Weekday = "Weekday" THEN
            Add record to Weekday_Peak
        ELSE IF Peak_Hour = "Yes" AND Weekday = "NonWorkingday" THEN
            Add record to NonWorkingday_Peak
        ELSE IF Peak_Hour = "No" AND Weekday = "Weekday" THEN
            Add record to Weekday_NonPeak
        ELSE
            Add record to NonWorkingday_NonPeak
        END IF
    END FOR
END FOR
FOR each matrix in {Weekday_Peak, Weekday_NonPeak, NonWorkingday_Peak, NonWorkingday_NonPeak} DO
    Divide matrix data into 1-hour intervals
    Calculate the average for each interval
    Store results in corresponding average matrix
END FOR
FOR each average matrix DO
    Perform Kolmogorov-Smirnov test for normality
    IF distribution is not normal THEN
        Stop the process and print: "Increase number of samples to ensure normality"
    END IF
END FOR
FOR each average matrix DO
    Calculate mean and standard deviation of hourly averages
END FOR
SAVE all results to 'Combined_Data.csv' and RETURN
```

3.3.3. Stochastic Travel Time Model

In this stage, the stochastic travel time model function is implemented, whose purpose is to estimate the variability in vehicle travel times, considering both the day and the estimated arrival time at each destination. This model allows simulating real traffic conditions, incorporating uncertainty elements that reflect the inherent fluctuations of the road environment.

Therefore, ensuring a robust design and precise operation of the stochastic model is fundamental to properly represent temporal variability and evaluate the impact of different defined scenarios. In this case, the Box-Muller transformation (using Equations 4 to 9 defined previously) was implemented directly, instead of using integrated normal generators from modern libraries. This decision allows for detailed mathematical control over the simulation process and ensures precise traceability of the model. The complete procedure is detailed in the pseudocode of Algorithm 7.

Algorithm 7. Stochastic Model Function

```
Input: Routing solution (sequence of locations)
1. Time_Matrix, _ ← Call Time and Distance Matrix Function
2. Call Historical Data Analysis Function
3. Initialize Adjusted_Matrix ← []
4. FOR each Origin location in Routing_Solution DO
5.     FOR each Destination in Routing_Solution DO
6.         IF Origin = Destination THEN
7.             Adjusted_Time ← 0
8.         ELSE
9.             Departure_Hour ← get departure hour from Origin to Destination
10.            Working_Day ← determine if the date is a working day
11.            Nominal_Time ← Time_Matrix [Origin][Destination]
12.            Identify scenario based on Departure_Hour and Weekday
13.            FOR each row in 'Combined_Data.csv' DO
14.                IF Origin and Destination match the row THEN
15.                    Mean, StdDev ← extract values from the r
16.                    LI ← Mean - (1.96 * StdDev)
17.                    LS ← Mean + (1.96 * StdDev)
18.                    REPEAT:
19.                        X ← Random (0, 1)
20.                        Y ← Random (0, 1)
21.                        Z0 ← root(-2ln(X)) * cos(2πY)
22.                        Z1 ← root(-2ln(X)) * sin(2πY)
23.                        Var1 ← Mean + (Deviation * Z0)
24.                        Var2 ← Mean + (Deviation * Z1)
25.                    UNTIL (LI <= Var1 <= LS) and (LI <= Var2 <= LS)
26.                    Chosen_Var ← Select randomly between Var1 and Var2.
27.                    Adjusted_Time ← Nominal_Time + Chosen_Var
28.                END IF
29.            END FOR
30.        END IF
31.        Add Adjusted_Time to Adjusted_Time_Matrix
32.    END FOR
33. END FOR
34. Return Adjusted_Time_Matrix
```

To illustrate the procedure, considering the origin coordinates (-5.1965, -80.6328) and destination (-5.179587, -80.677845), the algorithms described in Algorithms 5 and 6 were applied. The data collected for this route is presented in Table 3, which summarizes descriptive statistics segmented by scenario.

Table 3. Compilation of the Pathway Analysis

Time	Day Type	Sample Averages	Media	Standard Deviation	Minimum	Maximum	Origin	Destination
Non-peak	Weekday	340	14.94	0.36	14.37	15.62	-5.1965, -80.6328	-5.179587, -80.677845
Peak	Weekday	140	15.5	0.74	14.48	17.02	-5.1965, -80.6328	-5.179587, -80.677845
Non-peak	Non-Working day	170	14.35	0.54	13.4	15.1	-5.1965, -80.6328	-5.179587, -80.677845
Peak	Non-Working day	70	14.33	0.73	13.23	15.33	-5.1965, -80.6328	-5.179587, -80.677845

Based on the previously defined scenarios, travel times were adjusted using the stochastic function implemented in Algorithm 7. The results of the stochastic adjustment, considering a 95% confidence level, are presented in Table 4, allowing us to observe how traffic fluctuations impact estimated travel times.

Table 4. Adjusted Travel Times

					95% Confidence Level				
Scenario	U1	U2	Z0	Z1	Lower Limit	Upper Limit	Travel time (Z0)	Travel Time (Z1)	Adjusted Travel Time
Non- peak	0.69217	0.31844	-0.35763	0.77971	14.23440	15.64560	14.81125	14.23440	14.23440
Peak	0.47192	0.33866	-0.64796	1.04022	14.04960	16.95040	15.02051	14.04960	14.04960
Non-peak	0.05269	0.74479	-0.07935	-2.42493	13.29160	15.40840	14.30715	13.29160	14.30715
Peak	0.42669	0.98008	1.29494	-0.16292	12.89920	15.76080	15.27530	12.89920	15.27530

3.4. Algorithm for VRPSTTW

After completing the first three stages, we proceed to the final phase, in which the stochastic model is integrated into the main simulated annealing function developed in section 3.1. This integration allows the incorporation of travel time variability while maintaining consistency in the algorithm's flow.

In this final stage, all previously developed components are consolidated to form the complete VRPSTTW algorithm, combining the stochastic travel time model with the simulated annealing metaheuristic. This results in a robust and adaptable algorithm capable of addressing dynamic scenarios where traffic conditions fluctuate according to the defined scenarios.

This implementation represents a clear methodological transition from the deterministic approach to the stochastic one. It is important to note that the deterministic model was used only as a theoretical reference, based on the classic Solomon instances. It was not applied to real-world settings due to its structural limitation: it assumes constant travel times, which is inadequate for representing the uncertain nature of urban traffic. In contrast, the stochastic model was designed to operate with empirical data, incorporating time and contextual variability. Therefore, all practical validations of the algorithm were performed exclusively under the stochastic approach, as it was the only methodologically coherent alternative aligned with the study's objectives.

The entire process is summarized in Algorithm 8, which structures each step from parameter initialization to obtaining the best route.

Algorithm 8. VRPSTTW Development

```
Input: VRPTW Instance
1. Define SA parameters: T_max, T_min, Cooling_Rate, Max_Iter
2. Load data from VRPSTTW instance
3. Call Time and Distance Matrix Function
4. Generate Current_Solution, Current_Used_Vehicles ← Initial Solution Creation Function
5. Apply Stochastic Model Function to Current_Solution
6. Current_Energy ← Energy Calculation Function (Current_Solution,
    Current_Used_Vehicles) using the Adjusted_Time generated by the stochastic model
7. Best_Solution ← Current_Solution
8. Best_Energy ← Current_Energy
9. T ← T_max
10. Iterations ← 0
11. WHILE Max_Iter > Iterations and T > T_min DO
12.    Neighbor_Solution, Neighbor_Used_Vehicles ← Neighbor Solution Generation Function
13.    Apply Stochastic Model Function to Neighbor_Solution
14.    Neighbor_Energy ← Energy Evaluation Function (Neighbor_Solution,
        Neighbor_Used_Vehicles) using adjusted travel times
15.    ΔE ← Neighbor_Energy - Current_Energy
16.    IF ΔE < 0 or exp (-ΔE / Temp_Initial) > random (0,1) THEN
17.        Current_Solution ← Neighbor_Solution
18.        Current_Used_Vehicles ← Neighbor_Used_Vehicles
19.        Current_Energy ← Neighbor_Energy
20.    END YES
21.    IF Current_Energy < Best_Energy THEN
22.        Best_Solution ← Current_Solution
23.        Best_Energy ← Current_Energy
24.    END YES
25.    T_max ← T_max * Cooling_Rate
26. END WHILE
27. Print Best_Solution, Best_Energy
```

4. Results

After completing the four implementation phases of the VRPSTTW algorithm, it was evaluated using a stochastic instance with 100 customers, generated from real coordinates. Since the algorithm's capability to solve complex scenarios had already been validated during the deterministic phase, it was not deemed necessary to evaluate multiple instances at this stage. The objective here was to verify compliance with the multi-objective function — minimizing the number of vehicles and associated costs (including total distance, travel time, and time window penalties) — as well as to establish guidelines for future improvements and methodological adjustments.

To this end, different experimental configurations were assessed using a factorial design, considering three levels for each of the following parameters: initial temperature (500, 1000, and 1500), cooling rate (0.3, 0.5, and 0.9), and maximum number of iterations (1000, 2000, and 4000). These levels were selected due to their frequent use in previous simulated annealing studies and their ability to adequately represent the extremes of the algorithm's configuration space [15].

4.1. Factorial Design and Analysis of Variance

To evaluate the impact of design factors on the stability of solutions generated by the algorithm, an analysis of variance (ANOVA) was performed, considering the obtained energy as the response variable. The factors included were initial temperature (A), cooling rate (B), and maximum number of iterations (C), as well as their interactions. The results are presented in Table 5.

Table 5. Analysis of Variance

Source of Variation	Degrees of Freedom	Sum of Squares	Mean Squares	F-Statistic	Significance (* at 0.05 and ** at 0.01)
Treatments	26	26718227.2561			
A	2	2134.8176	1067.4088	2.62393	
B	2	4295.3208	2147.6604	5.27942	*
C	2	26701252.3001	13350626.1500	32818.77686	**
AB	4	1178.8409	294.7102	0.72446	
AC	4	2057.2887	514.3222	1.26432	
BC	4	4054.3008	1013.5752	2.49159	
ABC	8	3254.3873	406.7984	1.00000	
ERROR	81	20877.9172	257.7521		
TOTAL	107	26739105.1733			
		Coefficient of Variation (CV)	**0.15%**		

The analysis of variance shows that both the cooling rate (B) and the maximum number of iterations (C) have a significant impact on the obtained energy. The cooling rate shows an F-statistic of 5.28 with a significance level below 0.05, while the number of iterations shows an F of 32,818.78, with a significance below 0.01, consolidating it as the most determining factor in optimizing the simulated annealing process.

In contrast, the interactions AB, AC, and BC did not show significant effects, suggesting that the combined effects between these factors do not notably influence the solution energy.

The coefficient of variation (CV) was 0.15%, reflecting low variability in the analyzed data, confirming the stability of the algorithm after calibration.

A more detailed analysis was then performed using Duncan's multiple range test at a 1% significance level and with 81 error degrees of freedom, to identify the optimal factor levels. The results are presented in Table 6.

Table 6. Duncan's Multiple Range Test

KEY	Cooling Rate	Subset (Mean)	
B1	0.3	11047.43	-
B2	0.5	11043.52	11043.52
B3	0.9		11032.53
KEY	**Maximum Iterations**	**Subset (Mean)**	
C1	1000	11055.05	-
C2	2000	-	11042.14
C3	4000	-	11035.29

Regarding the cooling rate (B), the level corresponding to 0.9 recorded the lowest mean value, 11,032.53, indicating better algorithm performance under higher rates. Although the ANOVA showed significance for this factor at the 5% level, Duncan's test, applied with a stricter 1% threshold, confirmed that level 0.9 has a significantly lower mean than level 0.3. This validates that the choice of the optimal level remains solid even with a more stringent statistical criterion.

Meanwhile, the maximum number of iterations (C) showed a decreasing trend in means as its value increased, reaching a minimum of 11,035.29 with 4,000 iterations. This behavior suggests that the algorithm tends to stabilize with a higher number of iterations, favoring solutions with lower energy.

After defining the optimal levels for the factors — Cooling Rate (B = 0.9) and Maximum Iterations (C = 4000) — the energy behavior throughout the iterations was evaluated, recording values on an Energy vs. Iterations graph. The results reveal a progressive trend toward energy stabilization, evidencing the convergence of the algorithm toward near-optimal solutions. This evolution can be observed in Figure 2.

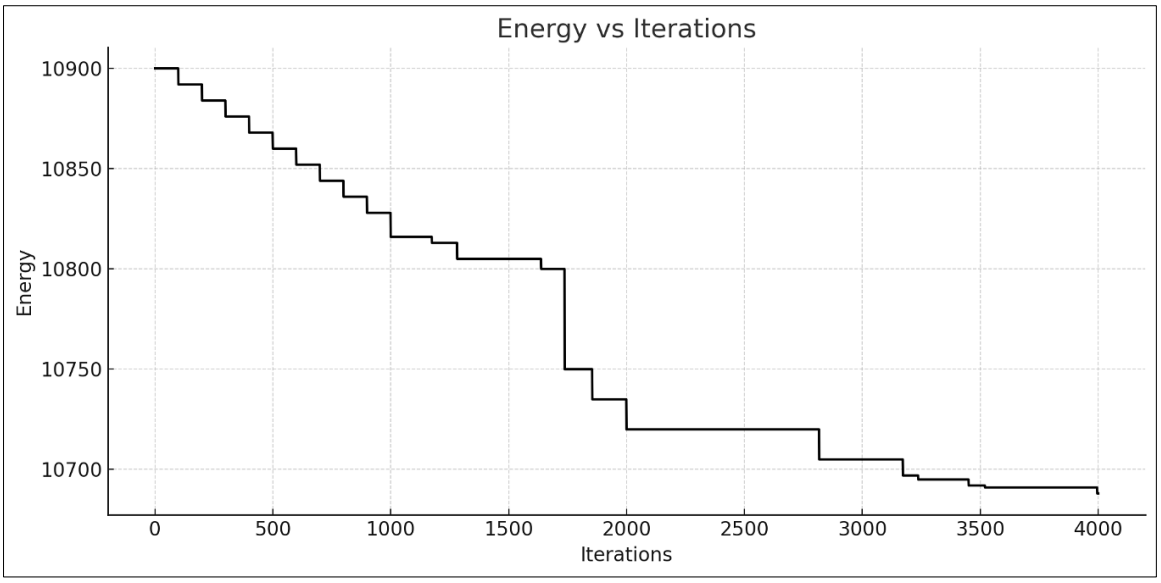

Figure 2. Energy vs Iterations Graph

4.2. Algorithm Precision

Using the optimal parameters previously determined, 10 runs of the algorithm were carried out on the test instance composed of 100 clients. The objective was to evaluate the consistency and precision of the results obtained by analyzing the variability in the solutions generated in each random run. This procedure provided a comparative framework for assessing the precision of the algorithm in a controlled environment. The results obtained are presented in Table 7.

Table 7. Algorithm Precision

Run	Energy	Number of Vehicles	Total distance (km)	Total Travel Time (min)	Penalties	Execution time (min)
1	10692.7	10	617.23	75.47	0	0.49
2	10690.97	10	616.15	74.82	0	0.47
3	10689.16	10	613.82	75.34	0	0.59
4	10688.95	10	611.97	76.98	0	0.7
5	10689.38	10	613.52	75.86	0	0.46
6	10692.18	10	617.47	74.71	0	0.47
7	10690.04	10	614.81	75.23	0	0.48
8	10688.61	10	613.98	74.63	0	0.46
9	10690.35	10	614.87	75.48	0	0.49
10	10688.78	10	613.5	75.28	0	0.46
Average	10690.112	10	614.732	75.38	0	0.507
Standard Deviation	1.437	0	1.757	0.68	0	
Minimum	10688.61	10	611.97	74.63	0	
Maximum	10692.7	10	617.47	76.98	0	

The developed algorithm recorded an average execution time of 0.507 minutes, demonstrating efficient performance even on large instances. In terms of energy, a mean of 10,690.112 with a standard deviation of 1.437 was obtained, indicating low variability in the generated solutions. Regarding the objective function, defined to minimize both the number of vehicles and the associated costs (distance, time, and penalties), it was observed that in all runs the number of vehicles remained constant at 10, reaching the required minimum. This demonstrates the algorithm's effectiveness in optimizing routes without needing to increase the fleet size.

As for the associated costs, the traveled distance showed a standard deviation of 1.757 km, while the total travel time exhibited a dispersion of 0.68 minutes; both indicators reflecting low variability in the results.

Regarding penalties, it is worth noting that no violations were recorded in any of the runs, which reaffirms the algorithm's capacity to generate feasible solutions without violating constraints, thus consolidating its precision, stability, and efficiency.

4.3. Algorithm Robustness

In the robustness analysis, energy is used exclusively as the central indicator, since it integrates the multi-objective function in a consolidated manner. This allows for a more coherent evaluation of the impact of variations in the algorithm's parameters, avoiding analytical dispersion that might arise when considering indicators separately.

Following this premise, 36 additional runs were performed using the test instance with 100 customers, applying small modifications to the previously determined optimal parameters. The results obtained are presented in Table 8.

Table 8. Robustness of the Algorithm (Energy Harvesting)

	Initial Temperature	**1500**		
	Cooling Rate	**0.9**	**0.95**	**0.99**
Maximum Iterations	3500	10694.52	10693.78	10691.87
		10693.45	10691.45	10692.54
		10697.45	10694.57	10695.45
		10693.25	10693.25	10691.23
	3750	10691.45	10691.64	10690.87
		10690.87	10690.56	10690.78
		10692.78	10691.45	10690.38
		10693.45	10692.34	10692.25
	4000	10692.31	10691.75	10689.24
		10690.05	10689.74	10688.79
		10689.31	10688.98	10689.01
		10689.47	10689.56	10688.74
Average		10691.63		
Standard Deviation		2.048		

The mean energy recorded in these additional runs was 10,691.63, representing an increase of 1.518 units compared to the value obtained in Table 7. This increase, accompanied by a standard deviation of 2.048 (higher than the initial 1.437), reflects slightly greater dispersion in the results, consistent with the modifications applied to the algorithm parameters.

In percentage terms, the standard deviation corresponds to 0.019% of the mean, indicating low variability, although slightly higher than that recorded with the optimal parameters. Despite this increase, the algorithm maintains an acceptable level of consistency, demonstrating its robustness and ability to generate stable solutions even with slight parameter modifications.

Additionally, this practical analysis supports the validity of the conclusions drawn from the factorial design, showing that the algorithm's performance remains stable without relying on exact configurations or a single strict statistical significance threshold. Moderate variations in the cooling rate and the number of iterations did not substantially alter the quality of the solutions, reinforcing the solidity of the recommendations obtained.

4.4. Computational Complexity of the Algorithm

The estimation of the computational complexity of the proposed algorithm was carried out based on the structural analysis of the pseudocode and the implemented functional modules. Table 9 details the main blocks of the algorithm and their respective asymptotic complexity order, considering the number of customers as the dominant variable n and the number of iterations C as the control parameter of simulated annealing.

Table 9. Algorithm Complexity

Block / Function	Description	Complexity	Remarks
Initial Solution Generation	Create a feasible solution with multiple vehicles and clients	$O(n^2)$	Involves the distance/time matrix between all clients
Stochastic Model	Adjusts travel times between all customer pairs based on the scenario	$O(n^2)$	Can be optimized with a dictionary for constant-time access
Neighbor Generation	Alter the current solution using move operators	$O(n)$	Depends on the type of move (swap, 2-opt, relocate, etc.)
Stochastic Model on Neighbor	Repeated for each generated neighbor	$O(n^2)$	Dominant component of each SA iteration
Energy Calculation (Total Cost)	Sum of distances, times, penalties	$O(n)$	Route is traversed to compute the objective function
Acceptance Condition (Metropolis)	Checks whether to accept the new solution	$O(1)$	Compares energy values and a random number
Main SA Loop	Runs the process over "Max_Iter" iterations	$O(C)$	Loop body executed until temperature decreases

Overall, the total algorithm complexity is obtained by identifying the heaviest computational block within the iterative cycle. The stochastic model, which runs for each evaluated neighbor solution, introduces a cost of $O(n^2)$ per iteration. As the number of iterations is constant and parameterized by C, the total complexity of the algorithm is formally analyzed in Equation 23.

$$T(n) = O(C.n^2) \tag{23}$$

This expression reflects quadratic behavior relative to the number of customers, which is consistent with the combinatorial nature of the problem and the structure of the adjusted time evaluation model.

The average execution time recorded for an instance of 100 customers was 0.5 minutes. Assuming that the algorithm's complexity remains stable and the number of iterations does not vary, it is possible to estimate the execution time for other input scales using the projection shown in Equation 24.

$$T(n) = T(100)\left(\frac{n^2}{100}\right) \rightarrow T(n) = 0.5\left(\frac{n^2}{100}\right) \tag{24}$$

This estimate will be used to generate the projected time graph as a function of the number of customers, see Figure 3.

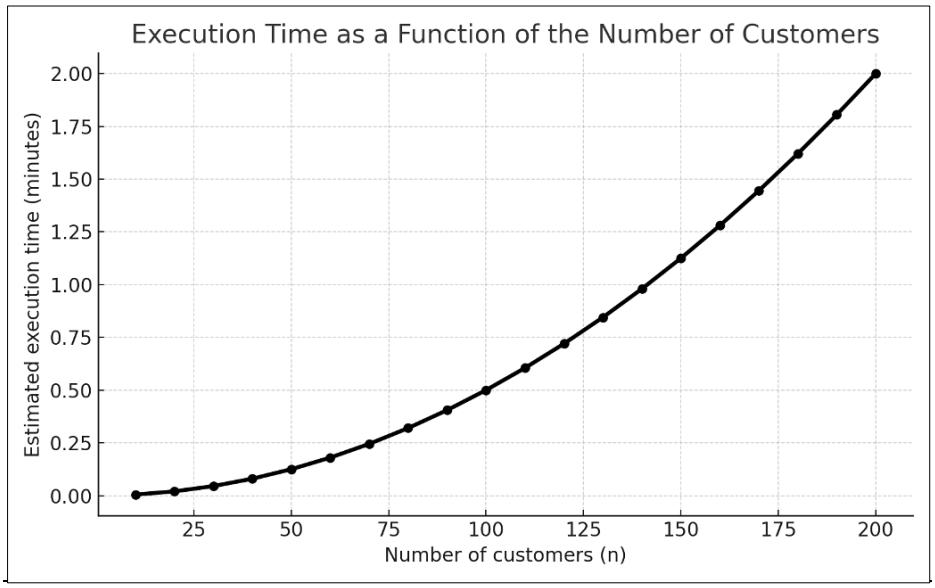

Figure 3. Time as a Function of the Number of Customers

5. Discussion

The results obtained in this research confirm the effectiveness of simulated annealing (SA) as a competitive metaheuristic for solving complex VRP variants, as also shown in previous studies such as Pratiwi et al. (2018) [8], where SA was used within a hybrid proposal. Unlike recent approaches such as Gibbons & Ombuki-Berman (2024) [9], which opt for memetic algorithms combining evolution and local search, this study demonstrates that a classic, properly tuned SA can achieve high-quality solutions on instances with up to 100 customers without the need for more complex hybrid structures.

The fine-tuning of the initial temperature, cooling rate, and number of iterations allowed energy to stabilize with a standard deviation of 1.437 (CV = 0.013%). This level of variability is even lower than that reported by Abdullahi et al. (2025) [4], whose study presented a CV = 0.03% on comparable instances, revealing that a rigorous factorial design is sufficient to achieve similar stability without additional reliability layers.

When the optimal parameters were perturbed, the deviation rose to 2.048 (0.019%), an increase still below the 0.05% observed by Rajabi-Bahaabadi et al. (2021) [5] when recalibrating their ACO-taboo scheme; moreover, our solution maintained a constant minimum fleet size, whereas theirs required additional vehicles in some scenarios. These findings support the thesis of Kirkpatrick et al. (1983) [15] on SA's ability to escape local optima and converge with low dispersion when parameters are properly tuned.

Regarding the incorporation of stochasticity, previous studies by Laporte et al. (1992) [10] and Guevara et al. (2025) [11] addressed the modeling of stochastic travel times through generic simulations, without considering empirical data from the real environment. In contrast, the present research implements a stochastic model based on real data extracted from Google Maps, allowing a more precise capture of urban traffic variability. This approach responds to the criticism posed by Nguyen et al. (2016) [13], who acknowledged that their solutions for VRPSTTW, although effective in simulated environments, did not integrate real conditions such as traffic congestion or hourly variations.

Instead of using stochastic models based on theoretical assumptions, like the Monte Carlo simulations employed by Guevara et al. (2025) [11], this research implements a model grounded in the Box-Muller transformation, following the methodology described by Papacostas & Prevedouros (1993) [17]. This method allows generating normal distributions from empirical data, adjusting the mean and standard deviation for each defined scenario (off-peak/working day, peak/working day, off-peak/non-working day, peak/non-working day).

The results of the stochastic model show that, unlike the approach presented by Van Woensel et al. (2008) [12], where traffic congestion is modeled using queuing theory, this proposal integrates traffic variability directly into the travel time calculation. This not only allows capturing hourly fluctuations but also reflects operational differences between working and non-working days, as suggested by global studies from the Institute of Transportation Engineers (ITE).

In terms of efficiency, the algorithm solved the 100-customer instance in 0.507 minutes, surpassing the 36.4 s of Abdullahi et al.'s simheuristic [4] and approaching the performance of Iklassov et al. (2024) [1], whose reinforcement learning model infers routes in 0.4 s, albeit after extensive GPU training. Additionally, our average distance of 614.7 km improves by 3.2% over the best result published by Wei et al. (2024) [19] in the deterministic model for class RC101 using their LNS-MRSO hybrid, with a standard deviation of only 1.757 km versus the 3–5 km they report.

While previous literature addresses VRPSTTW in a fragmented manner (focusing on time windows [9, 14] or stochastic travel times [11,12]), this research integrates both aspects into a single algorithm, validated with real data and supported by statistical analyses of precision, robustness, and complexity $O(C \cdot n^2)$. Thus, the proposal positions itself as a computationally viable alternative for route optimization in urban contexts with high temporal uncertainty.

However, this empirical approach presents an operational limitation related to dependency on external services such as the Google Maps API. Although this source provides realistic and up-to-date data, its large-scale use can generate costs and quota restrictions, complicating model replication in production environments. Furthermore, although a formal verification of biases in travel time estimates was not conducted, previous studies have reported systematic inaccuracies [22-24]: in urban environments, travel times tend to be underestimated, while in rural areas with low data coverage, larger errors are observed. These limitations suggest the need to incorporate validation mechanisms in future research, especially when modeling contexts with high geographic or temporal variability.

6. Conclusion

This study presented an implementation based on simulated annealing to address the Vehicle Routing Problem with Stochastic Travel Times and Soft Time Windows (VRPSTTW), showing promising results in terms of operational efficiency and solution stability. The methodology was capable of solving urban instances with up to 100 customers, strictly complying with time constraints without incurring penalties. The applied factorial analysis allowed the identification of the cooling rate and the maximum number of iterations as key factors influencing the algorithm's behavior. In particular, 4,000 iterations stood out as the most influential parameter in stabilizing the objective function. The experimental runs showed an average energy of 10,690.112, low dispersion in the results, and constant use of the minimum number of required vehicles, evidencing both the consistency and efficiency of the proposed approach.

Additionally, when controlled variations were introduced into the optimal parameters, the algorithm maintained solution quality within acceptable ranges, demonstrating adequate robustness against perturbations. In all executions, the model complied with the imposed operational constraints, even under conditions of high temporal uncertainty. In general terms, the proposal represents a computationally viable and effective alternative for route optimization in dynamic logistics contexts, such as last-mile distribution systems or fleet management in emergency situations. As future

research directions, it is recommended to explore the algorithm's behavior on larger-scale instances (200 to 1,000 customers), as well as to integrate hybrid approaches that incorporate machine learning-based predictive models to anticipate stochastic scenarios and improve the model's adaptability to dynamic data.

In the current approach, stochastic scenarios are manually defined based on four representative combinations of peak hour and workday, which imposes an important limitation in terms of coverage and realism. By integrating machine learning models, such as XGBoost or recurrent neural networks, it would be possible to train travel time predictors based on multiple contextual variables (hour, day, weather, traffic history, etc.), which would allow generalization to millions of possible scenarios without explicitly defining them. This would enable better anticipation of congestion and greater adaptability of the algorithm to changing conditions, evolving the model from a static system to a truly dynamic and intelligent one

7. Declarations

7.1. Author Contributions

Conceptualization, M.J.C. and K.M.O.H.; methodology, K.M.O.H.; software, K.M.O.H.; validation, K.M.O.H. and M.J.C.; formal analysis, M.J.C.; investigation, M.J.C.; resources, M.J.C. and G.A.F.F.; data curation, M.M.O.H. and M.J.C.; writing—original draft preparation, M.J.C.; writing—review and editing, M.J.C.; visualization, M.J.C. and K.M.O.H.; supervision, M.J.C.; project administration, M.J.C.; funding acquisition, M.J.C. All authors have read and agreed to the published version of the manuscript.

7.2. Data Availability Statement

The data presented in this study are available within the article.

7.3. Funding

This research is being funded by the Universidad Nacional de Piura, Peru.

7.4. Institutional Review Board Statement

Not applicable.

7.5. Informed Consent Statement

Not applicable.

7.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

8. References

[1] Iklassov, Z., Sobirov, I., Solozabal, R., & Takac, M. (2024). Reinforcement Learning for Solving Stochastic Vehicle Routing Problem with Time Windows. arXiv Preprint, arXiv:2402.09765. doi:10.48550/arXiv.2402.09765.

[2] Jiang, Z., Chen, W., Zheng, X., & Gao, F. (2024). Research on vehicle path planning of automated guided vehicle with simultaneous pickup and delivery with mixed time windows. IET Collaborative Intelligent Manufacturing, 6(2), 12105. doi:10.1049/cim2.12105.

[3] Zacharia, P., & Stavrinidis, S. (2024). The Vehicle Routing Problem with Simultaneous Pick-Up and Delivery under Fuzziness Considering Fuel Consumption. Vehicles, 6(1), 231–241. doi:10.3390/vehicles6010009.

[4] Abdullahi, H., Reyes-Rubiano, L., Ouelhadj, D., Faulin, J., & Juan, A. A. (2025). A reliability-extended simheuristic for the sustainable vehicle routing problem with stochastic travel times and demands. Journal of Heuristics, 31(2), 1-39. doi:10.1007/s10732-025-09555-4.

[5] Rajabi-Bahaabadi, M., Shariat-Mohaymany, A., Babaei, M., & Vigo, D. (2021). Reliable vehicle routing problem in stochastic networks with correlated travel times. Operational Research, 21(1), 299–330. doi:10.1007/s12351-019-00452-w.

[6] Muñoz-Villamizar, A., Faulin, J., Reyes-Rubiano, L., Henriquez-Machado, R., & Solano-Charris, E. (2024). Integration of Google Maps API with mathematical modeling for solving the Real-Time VRP. Transportation Research Procedia, 78, 32–39. doi:10.1016/j.trpro.2024.02.005.

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

[8] Pratiwi, A. B., Pratama, A., Sa'diyah, I., & Suprajitno, H. (2018). Vehicle routing problem with time windows using natural inspired algorithms. Journal of Physics: Conference Series, 974, 012025.:10.1088/1742-6596/974/1/012025.

[9] Gibbons, E., & Ombuki-Berman, B. (2024). A Memetic Algorithm for Large-Scale Real-World Vehicle Routing Problems with Simultaneous Pickup and Delivery with Time Windows. MIC 2024. Lecture Notes in Computer Science, vol 14753, Springer, Cham, Switzerland. doi:10.1007/978-3-031-62912-9_8.

[10] Laporte, G., Louveaux, F., & Mercure, H. (1992). Vehicle routing problem with stochastic travel times. Transportation Science, 26(3), 161–170. doi:10.1287/trsc.26.3.161.

[11] Guevara, W., Mena, C. del C., Pérez, L., Montoya, C., Caro, M., Bejarano, C., Bolívar, S., & Delgado, C. (2025). An algorithm for the stochastic delivery-and-pick-up vehicle routing problem with time windows as applied to surgical medical supplies. Ingenieria y Universidad, 29, 1–22. doi:10.11144/javeriana.iued29.asdp.

[12] Van Woensel, T., Kerbache, L., Peremans, H., & Vandaele, N. (2008). Vehicle routing with dynamic travel times: A queueing approach. European Journal of Operational Research, 186(3), 990–1007. doi:10.1016/j.ejor.2007.03.012.

[13] Nguyen, V. A., Jiang, J., Ng, K. M., & Teo, K. M. (2016). Satisficing measure approach for vehicle routing problem with time windows under uncertainty. European Journal of Operational Research, 248(2), 404–414. doi:10.1016/j.ejor.2015.07.041.

[14] Solomon, M. M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems With Time Window Constraints. Operations Research, 35(2), 254–265. doi:10.1287/opre.35.2.254.

[15] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. Science, 220(4598), 671–680. doi:10.1126/science.220.4598.671.

[16] Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. Journal of Optimization Theory and Applications, 45(1), 41–51. doi:10.1007/BF00940812.

[17] Papacostas, C. S., & Prevedouros, P. D. (1993). Transportation Engineering and Planning (3rd Ed.). Prentice Hall, New Jersey, United States.

[18] Mazmanyan, L., & Trietsch, D. (2014). Stochastic travelling salesperson and shortest route models with safety time. International Journal of Planning and Scheduling, 2(1), 53. doi:10.1504/ijps.2014.066707.

[19] Wei, X., Xiao, Z., & Wang, Y. (2024). Solving the Vehicle Routing Problem with Time Windows Using Modified Rat Swarm Optimization Algorithm Based on Large Neighborhood Search. Mathematics, 12(11), 1702. doi:10.3390/math12111702.

[20] Rodrigue, J.-P. (2024). The Geography of Transport Systems. Milton Park, United Kingdom. doi:10.4324/9781003343196.

[21] Helmer, J., Gough, P., Jim, & Peng. (2010). Institute of Transportation Engineers. ITE Journal, 33–35.

[22] Alkaissi, Z. A. (2024). Evaluating the Performance of Right Turn Lanes at Signalized Intersection Using Traffic Simulation Model. Civil Engineering Journal, 10(7), 2233–2243. doi:10.28991/CEJ-2024-010-07-010.

[23] Wagner, B., Human, S., & Winkler, T. (2021). Bias in Geographic Information Systems: The Case of Google Maps. Proceedings of the 54th Hawaii International Conference on System Sciences, 147-158. doi:10.24251/hicss.2021.103.

[24] Alomari, A. H., Al-Omari, B. H., & Al-Hamdan, A. B. (2020). Validating trip travel time provided by smartphone navigation applications in Jordan. Jordan Journal of Civil Engineering, 14(4), 500–510.