Multiobjective Optimization and Multicriteria Analysis of Rail Freight Transportation Logistics with Deterministic Queues

Willian Félix Souza e Silva
Graduate Program in Electrical Engineering
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
willianfelix.engenharia@gmail.com

Luís Henrique dos Santos
Graduate Program in Electrical Engineering
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
henriques.siul@gmail.com

Patrick Rodrigues Rocha
Graduate Program in Electrical Engineering
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
patrickrrocha@gmail.com

Michel Bessani
Dept. of Electrical Engineering
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
mbessani@ufmg.br

Lucas S. Batista
Dept. of Electrical Engineering
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
lusoba@ufmg.br

***Abstract*—The vehicle routing problem with queues is a critical challenge in freight railway logistics, where congestion and variable service times impact operational efficiency. This study presents a mathematical model to optimize the sequencing of railway flows, considering travel times, loading and unloading processes, and queue management. The problem is formulated as a multi-objective optimization model, aiming to maximize the transported volume while minimizing idle times and empty displacements. Due to the nonlinear nature of the problem, particularly the recursive behavior of queues, the Non-dominated Sorting Genetic Algorithm II (NSGA-II) metaheuristic was used to efficiently solve the problem. Although the exact evaluation of objective functions using the mathematical formulation was computationally intensive, it demonstrated that queues can be successfully modeled as an endogenous component of the system. This allowed for a structured analysis of queue propagation across routing decisions without resorting to full-scale simulation. A Pareto front was estimated using the NSGA-II, and a representative solution was selected. The use of the Analytic Hierarchy Process (AHP) and the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) in the decision-making stage provided a structured and effective selection process, consistent with the defined priorities.**

***Index Terms*—freight train scheduling, deterministic queuing, metaheuristics, resource allocation, Pareto optimization**

I. Introduction

The Vehicle Routing Problem (VRP) emerged with the Truck Dispatching Problem [1], in which the authors modeled the allocation of trucks to supply gas stations while minimizing the distance traveled. A few years later, in [2], other researchers generalized this problem into a linear optimization model applied to logistics and transportation, involving a fleet of vehicles with different capacities serving dispersed customers. From these formulations, the VRP established itself as one of the main topics in Operations Research [3].

Since then, various optimization strategies have been applied to the VRP. In [4], the authors demonstrated the effectiveness of NSGA-II (Non-dominated Sorting Genetic Algorithm II) [5] in solving the Capacitated VRP, optimizing multiple objectives such as response time and infection rate in pandemic scenarios. Similarly, In [6] the authors explored genetic algorithms for routing with multiple depots, highlighting their flexibility in handling complex constraints. In [7], a hybrid approach combining deep learning and NSGA-II was proposed for routing with time windows. Finally, in [8], the authors analyzed representations for genetic algorithms applied to the VRP, highlighting their impact on solution quality.

A variant of interest is the Vehicle Routing Problem with queues, which considers waiting times at loading and unloading points [9]. This problem is critical in freight rail transportation, where congestion and variable service times affect operational efficiency. In the context of a fixed fleet of trains operating between stations, optimized planning is essential to avoid bottlenecks and ensure smooth transportation, resulting in an operation optimized for the planned schedule.

Previous studies have addressed rail transport optimization through resource allocation and cost minimization [10], [11]. However, few works incorporate station queues into operational results, and those that do — such as [9], [12] — rely on stochastic models or treat queues as uncertain.

To address this gap, we propose a deterministic model that incorporates queue dynamics endogenously—avoiding hour-by-hour simulation and enabling analysis in aggregated inter-

This study was supported by Brazilian agencies Coordination for the Improvement of Higher Education Personnel (CAPES) – Finance Code 001, the Research Support Foundation of the State of Minas Gerais (FAPEMIG), and the National Council for Scientific and Technological Development (CNPq). Lucas S. Batista is a FAPEMIG-CNPQ scholarship holder (APQ-06716-24).

vals. It optimizes the sequencing of rail flows under constraints on flow frequencies, travel times, and loading/unloading operations. The main challenge is the recursive nature of queue calculations, which significantly affects system efficiency. The objective is to maximize transported volume while minimizing empty trips and excessive waiting times.

The approach formulates a multi-objective model and proposes a method to handle queue recursion. The NSGA-II metaheuristic estimates a Pareto front, from which a solution is selected using Analytic Hierarchy Process (AHP) [13] and Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) [14], ensuring alignment with user priorities.

The approach successfully estimated diverse Pareto fronts and demonstrated that queues can be handled endogenously within a mathematical model. The selected solution achieved a transported volume of over 291,000 tons with moderate operational inefficiencies, confirming the model's practical applicability.

The remainder of this paper is organized as follows. Section II presents the problem description and underlying assumptions. Section III details the solution methodology, including the mathematical formulation and the genetic algorithm framework. Section IV reports and discusses the main results, including the performance of the estimated Pareto fronts and the decision-making process. Finally, Section V concludes the study and outlines directions for future research.

II. Problem Description

Suppose a logistics company manages a fixed set of N_t trains that transport cargo between N stations, referred to as nodes. These nodes form a fully connected graph. Additionally, it is assumed that the inventories at the stations are sufficiently large to accommodate any routing configuration and that the transported product is always the same, ensuring homogeneous priority among trains. Since the demand for transportation is assumed to be sufficiently high to fully utilize the available fleet, the company must optimize the sequence of trips to improve the operational efficiency of the transportation service.

The complexity of the problem arises from two main factors. First, the transportation times between stations vary depending on the direction of travel and the load status of the train (loaded or empty). Second, the loading and unloading operations require time and impact the availability of stations to receive incoming trains. As a consequence, queues form, affecting the system's efficiency.

To illustrate the problem dynamics and introduce the nomenclature, the following didactic example is presented.

Example 1 Figure 1a) illustrates a schematic of a railway network with three nodes (A, B, and C). The company has three trains (x, y, and z) available to fulfill the transportation demands. Each transportation demand is referred to as a flow. The first table, in Figure 1b) shows the sequence of flows executed by each train. In the table header, column t specifies the train, and row d indicates the count of decision steps. For example, the second decision for train z is to operate the flow BA. To facilitate implementation, a number is associated with each flow, as shown in the rightmost table in Figure 1b); thus, the second decision for train z can also be referred to as flow 3. In this table, the header OD stands for Origin-Destination.

Time counting is the fundamental principle for computing queues and planning the sequence of flows. Each segment has four time parameters, corresponding to the travel time depending on the direction (outbound or return) and the train's load status (empty or loaded). Additionally, each node has at least three more time parameters: loading time, unloading time, and queue time (with a different queue time for each train). See these variables indicated for node B and segment BC in Figure 1a). The total time required to complete each flow starts counting from the moment the train arrives at the flow's origin.

For example, suppose train y arrives to operates the flow corresponding to its first assigned origin-destination pair, AC. The time spent to complete the entire flow is:

$$Q_{Ay} + L_A + TL_{AC} + Q_{Cy} + U_C + TE_{CB}, \qquad (1)$$

where Q_{Ay} denotes the queue time faced by train y at node A, L_A is the loading time at node A, TL_{AC} is the loaded transit time between nodes A and C, Q_{Cy} is the queue time incurred by train y at node C, U_C is the unloading time at node C, and finally, TE_{CB} is the empty transit time between node C and node B (which is the origin of train y's second decision).

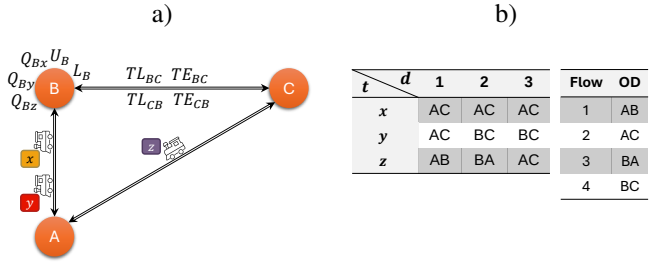


Figure 1: Example involving three nodes, A, B and C and three trains x, y and z. In panel (a), the initial condition is illustrated, where all three trains are in route to node A. Panel (b) presents tables detailing the sequence of transportation demands assigned to each train.

Building on the foundational understanding of the transportation logistics underlying this study, the following multi-objective problem is defined.

Problem 1 Determine the sequence of flows to be executed by each train operated, with the objective of maximizing the transported cargo volume while minimizing efficiency loss indicator, particularly time lost due to empty movements and queuing. The planning must comply with the following constraints:

1) *a minimum quantity of each flow has been contractually agreed upon and must be fulfilled; and*
2) *a maximum quantity of each flow must not be exceeded.*

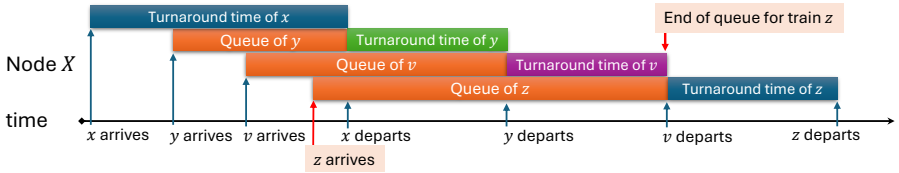


Figure 2: Gantt chart illustrating a representative scenario for queue calculation. From the perspective of train z, the queue time at node X is determined by the difference between the departure time of the last train at the node, namely train v, and the arrival time of train z.

III. Solution Methodology

A. Mathematical Modeling

The company aims to maximize the number of flows executed within a 720-hour planning horizon (one month). To that end, we introduce the decision variable x, which determines the assignment of flows to each train. This variable is represented as a matrix x that is of size $N_t \times N_d$, and each $x_{td} \in \{1, 2, \dots N_\phi\} \forall t,d$ where N_ϕ denotes the total number of available flows and N_d represents the maximum number of decision steps, computed as:

$$N_d = H / \min(\text{base trips}), \tag{2}$$

with 'base trips' referring to flow durations under ideal, queue-free conditions—i.e., the shortest possible travel time for each route.

The objective function is defined as the total number of flows completed within the planning horizon. To assess whether a given flow occurs within this period, we introduce the binary variable δ, which indicates whether a train's departure falls within the time window H:

$$\delta_{td} = \begin{cases} 1, & \text{if } 0 \le a_{td} \le H, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where a_{td} denotes the time at which train t arrives at the origin of its d-th scheduled flow. This arrival time is computed recursively from the beginning of the planning window:

$$\begin{aligned} a_{td} = a_{t,d-1} &+ Q_{O_{x(t,d-1)}t} + L_{O_{x(t,d-1)}} + \\ &+ TL_{O_{x(t,d-1)}D_{x(t,d-1)}} + Q_{D_{x(t,d-1)}t} + U_{D_{x(t,d-1)}} + \\ &+ TE_{D_{x(t,d-1)}O_{x(t,d)}}, \end{aligned} \tag{4}$$

where the right-hand side corresponds to the generalized form of equation (1), excluding the initial arrival time $a_{t,d-1}$. The notation $O_{x(t,d)}$ and $D_{x(t,d)}$ denote, respectively, the origin and destination associated with the d-th flow assigned to train t.

All of these variables are assumed to be given as problem inputs, with the exception of queue time, which must be dynamically computed. In a simplified formulation, the queue time experienced by train t at any given node is defined as the interval between its arrival and the departure time of the last train occupying that node. Figure 2 illustrates this with a Gantt chart involving four trains. When train x arrives at node X (representing an arbitrary node in the network), it encounters no prior occupancy and thus proceeds immediately to its turnaround operation (which encompasses either loading or unloading). In contrast, when a train arrives at a node already occupied — such as train z — it incurs a waiting time. From the perspective of train z, the queue time is computed as the difference between the completion time of the turnaround operation of train v (the last train occupying node X) and the arrival time of train z.

It is therefore evident that two key variables must be computed. The first is the arrival time of train t at a given node X, defined as:

$$AT_{Xtd} = \begin{cases} a_{td}, & \text{if } X = O_{x(t,d)}, \\ a_{td} + Q_{O_{x(t,d)}} + & \\ + L_{O_{x(t,d)}} + TL_{O_{x(t,d)}X}, & \text{if } X = D_{x(t,d)}, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

That is, if node X corresponds to the origin of the d-th decision of train t, the arrival time is directly given by a_{td}, as defined in equation (4). If X corresponds to the destination, the arrival time is computed by summing the time of arrival at the origin with the total duration required for executing the corresponding flow—comprising the queue time at the origin, loading time, and the loaded transit time from the origin to node X.

The departure times of all other trains positioned at the same node X, denoted by $\bar{t} \neq t$, are computed as:

$$DT_{X\bar{t}d} = AT_{X\bar{t}d} + Q_{X_{x(\bar{t},d)}} + T_{X\bar{t}d}, \tag{6}$$

where T_{Xtd} denotes the turnaround time of train t at node X. This variable represents either the loading or unloading duration, depending on whether node X is the origin or the destination of the corresponding flow. Mathematically, it is defined as:

$$T_{Xtd} = \begin{cases} L_X, & \text{if } X = O_{x(t,d)}, \\ U_X, & \text{if } X = D_{x(t,d)}, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Accordingly, the departure time of any train $\bar{t}$ (excluding train t) from node X is given by its arrival time at the node, incremented by its respective queue time and turnaround duration. This behavior is exemplified by trains y, z, and v in Fig. 2.

Using the previously defined variables, the queue time experienced by train t at node X can be computed by

evaluating multiple candidate contributions from all other trains $\bar{t} \neq t$, as follows:

$$q_{Xtd} = \begin{cases} DT_{X\bar{t}d} - \\ -AT_{Xtd} \\ 0, \end{cases} \quad \text{if} \quad \begin{array}{l} T_{X\bar{t}d} > 0 \ \text{ and} \\ (AT_{X\bar{t}d} < AT_{Xtd} < DT_{X\bar{t}d} \ \text{ or} \\ (AT_{X\bar{t}d} = AT_{Xtd} \ \text{ and } \ \bar{t} < t)) \\ \text{otherwise.} \end{array} \tag{8}$$

A non-zero queue time from the perspective of train t is therefore contingent on specific conditions: (i) train $\bar{t}$ must have a non-zero turnaround time at node X; (ii) it must arrive at X prior to train t and depart after t's arrival; and (iii) in the event of simultaneous arrivals, priority is given to the train with the lower index, which is assumed to have precedence and thus incurs no queue. As illustrated in Fig. 2, this scenario occurs for trains x, y, and v relative to train z.

Although several potential queues may be computed for train t at node X, only the most constraining one is relevant for scheduling. Therefore, the actual queue time experienced by train t is defined as the maximum of all computed queue candidates:

$$Q_{X_{x(t,d)}} = \max(q_{Xtd}), \tag{9}$$

which captures the most restrictive interference from other trains at node X.

As a problem constraint, the initial position of each train at the start of the simulation must be taken into account. This condition restricts the set of flows that can be assigned to each train as its first decision, since trains are already in transit to predefined locations before routing decisions are determined. For instance, if a train is initially heading toward node A, then the first flow assigned to it must involve node A either as its origin or destination. Mathematically:

$$x_{t0} = \sum_{i \in P_t} i \cdot y_{ti}, \quad \forall t \quad \text{and} \quad \sum_{i \in P_t} y_{ti} = 1, \quad \forall t \tag{10}$$

where P_t is the set of flows compatible with the initial position of train t, and y is a binary auxiliary variable.

With all supporting variables formally defined, the first objective function — representing the total volume transported across the railway network — is expressed as:

$$z_1(x) = \sum_{t=1}^{N_t} \sum_{d=1}^{N_d} \delta_{td} w_{x(t,d)}, \tag{11}$$

where $w_{x(t,d)}$ denotes the cargo volume associated with the d-th flow assigned to train t, and δ_{td} ensures that only flows initiated within the planning horizon are considered.

The second objective seeks to minimize non-productive time — defined as the periods in which trains are not actively engaged in revenue-generating tasks. Under the assumption that transportation demand exceeds fleet capacity, such idle time results from queue delays and empty transits between mismatched destination-origin pairs across successive flows. For instance, the company would favor the sequence $(AB,\ BA)$ over $(AB,\ CA)$, as the former eliminates the need for an empty return movement. This inefficiency is captured mathematically by:

$$z_2(x) = \frac{1}{\max\left(\frac{CE_{OD}}{TE_{OD}}\right)} \sum_{t=1}^{N_t} \sum_{d=1}^{N_d} \delta_{td} CE_{x(t,d)} + \sum_{X=1}^{N} \sum_{t=1}^{N_t} \sum_{d=1}^{N_d} Q_{Xtd}, \tag{12}$$

where $CE_{x(t,d)}$ represents the cost of an empty transit associated with the flow, and $\max\left(CE_{OD}/TE_{OD}\right)$ is the highest observed cost-to-duration ratio among all flows, serving as a normalization factor to align units with queue time (in hours).

The company imposes lower and upper bounds — denoted by $\underline{\Phi}$ and $\overline{\Phi}$, respectively — on the frequency with which each flow ϕ must be executed, in order to ensure a more balanced distribution of flows. These constraints are formally defined as:

$$\sum_{t=1}^{N_t} \sum_{d=1}^{N_d} \delta_{td} \gamma_{td\phi} \geq \underline{\Phi}_{\phi} \quad \text{and} \quad \sum_{t=1}^{N_t} \sum_{d=1}^{N_d} \delta_{td} \gamma_{td\phi} \leq \overline{\Phi}_{\phi}, \tag{13}$$

with the indicator variable:

$$\gamma_{td\phi} = \begin{cases} 1, & \text{if } x_{td} = \phi, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

The proposed model enables the assessment of the operational efficiency of a given flow allocation, with respect to both the total volume transported and the associated resource inefficiencies — namely, time-related losses. Based on this, a formal multi-objective optimization problem can be established as follows:

$$\max z_1(x), \quad \min z_2(x),$$

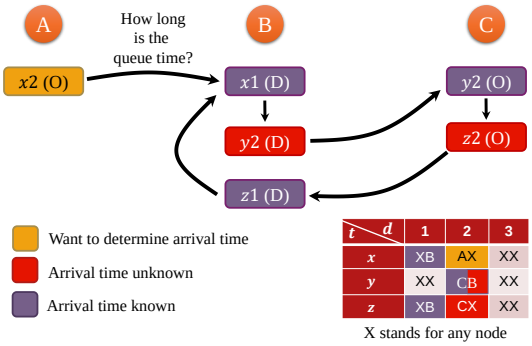
where $z_1(x)$ is defined in (11) and $z_2(x)$ in (12), both of which are dependent on auxiliary expressions and variables introduced in equations (2) through (9) and feasible solutions must satisfy (10) and (13), where $x_{td} \in \{1, 2, \ldots, N_\phi\}, \quad \forall\{t,d\}$.

B. Problem with Queue Recursiveness

During the implementation of the proposed formulation, a computational challenge was encountered, even though the model presented in Subsection III-A accurately represented what needed to be computed. The difficulty arises from the recursive nature of the queues. This occurs because, initially, the algorithm does not have the arrival time of each train-decision pair, as described in (4), and computing this arrival time depends on the queue times experienced by the same train in the previous decision. However, the queue calculated for the previous decision depends on all other train-decisions (flows) whose origin or destination is related to the origin or destination of the train-decision in question, as described in (8). Yet, some of these train-decisions may also have unknown arrival times, requiring their computation, which in turn may depend on other trains, creating a recursive dependency.

To elucidate the recursive nature of the queue computation problem, consider the illustrative scenario shown in Fig. 3. Suppose the objective is to determine the arrival time of train x for its second decision at node A—its origin for this leg—denoted as $x2$(O). To compute this, it is necessary

to first determine the departure time of train x from the destination of its previous decision ($d = 1$), which is node B, since the travel time between nodes is assumed to be known a priori. As depicted in the figure, the arrival time



t \ d	1	2	3
x	XB	AX	XX
y	XX	CB	XX
z	XB	CX	XX

Figure 3: Illustrative example of queue-time recursion.

of train x at node B is already available. To compute its departure, however, one must account for the turnaround time (known input) and the queue time (which must be computed), as per equation (6). Calculating the queue time requires knowing when train $x1$ began processing at node B, which, in turn, depends on analyzing its position relative to other trains—such as $y2$—present at the same node. Since the arrival time of $y2$ is not yet known, it too must be computed.

This leads to a chain of dependencies: the departure of $y2$ from its origin (node C) depends on the arrival of $z2$ at node C, which itself depends on the departure of $z1$ from its previous destination, node B. Critically, the departure of $z1$ at B is influenced by the queue state involving $x1$ and $y2$, thus completing a closed loop of interdependent calculations. Within an algorithmic implementation, this results in an unresolved recursive cycle.

To break this cycle, the proposed solution abandons the strategy of evaluating all flows in a single batch. Instead, the approach incrementally computes arrival times for train-decision pairs whose inputs are already available. When a recursive dependency is detected—as in the scenario described—the algorithm identifies the train with the earliest known arrival time with unknown end time and assigns a zero queue time to it. This assumption enables forward propagation: once a train's queue time is resolved, its departure can be computed, which in turn unlocks dependent arrival times downstream. In the given example, $x1$(D) is identified as the earliest among the known arrivals, and thus it is assumed to face no queue, allowing for the computation of $x2$'s arrival time.

The guiding principle of this strategy is that even in the absence of exact timing information, the partial ordering of arrivals provides sufficient information to justify deterministic approximations. Once a train-decision pair's time is resolved, the value is cached. If a new computation depends on unresolved variables, it is flagged and deferred until further information becomes available — allowing for progressive resolution of the full dependency graph.

C. Genetic Algorithm Framework

To address the bi-objective optimization problem, a Genetic Algorithm (GA) was implemented in Python. The overall effectiveness of the GA is contingent upon several critical components, including solution representation, individual initialization, mutation and crossover operators, selection strategy, and parameter tuning. The NSGA-II was adopted, given its proven efficiency in handling multi-objective problems. The subsequent sections provide a detailed discussion of some components of the algorithm.

1) Initialization of Individuals: The initialization procedure was carefully designed to ensure that all individuals in the population are feasible, i.e., they satisfy the problem's feasibility requirements, as the frequency of each flow is pre-adjusted to remain within the defined limits, and are consistent with the initial positions of the trains, which are provided as input data. Each individual is encoded as a matrix of dimensions $N_t \times N_d$, where N_t is the number of trains and N_d denotes the number of decision steps assigned to each train.

To illustrate, consider a simplified case with 3 trains and 4 decision steps. The minimum required number of assignments for each flow is 1, 2, 1, and 1, respectively, while the corresponding upper bounds are 5, 3, 2, and 4.

- First, a vector is created containing the minimum required occurrences of each flow, concatenated. In this case: [1, 2, 2, 3, 4].
- Next, flows are randomly assigned to the first decision of each train, respecting its next locations. An example assignment for each train could be: [3, 1, 2], respectively.
- A vector is then created with all the remaining flow assignments needed to meet the upper bounds, subtracting what has already been used (both the minimums and the first decisions). Example: [1, 1, 1, 4, 4, 4].
- The total number of decisions in the individual is the product of N_t and N_d — 12 in this case. Since 8 positions are already filled (minimum values + first decisions), 4 additional values are randomly selected from the previous vector of possible assignments. For example: [1, 4, 1, 4].
- These vectors are then concatenated and randomly shuffled. For instance: [1, 3, 2, 1, 1, 2, 4, 4, 1].
- Finally, the first decision values are inserted into their correct positions, and the rest of the individual is completed with values from the concatenated vector. Fig. 4 presents the resulting individual.

t \ d	1	2	3	4
x	3	1	3	2
y	1	1	1	2
z	2	4	4	1

Figure 4: Example of individual.

2) Mutation Operator: The mutation operator performs up to N_t swaps between positions in the decision matrix. Each occurs with a defined probability, introducing variation while preserving feasibility. An illustrative example of the mutation operator is provided in Fig. 5.

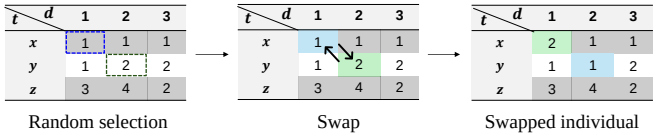


Figure 5: Mutation operator.

3) Crossover Operator: The crossover operator is implemented by combining genetic material from two parent individuals. Up to N_t trains are randomly selected for recombination. For each selected train, a random crossover point is chosen. For the corresponding row (train) in the decision matrix, offspring 1 inherits all decisions from parent 1 up to the crossover point, and the remaining values from parent 2. Offspring 2 follows the complementary pattern, inheriting the first segment from parent 2 and the remainder from parent 1. This mechanism promotes diversity while preserving partial structures from both parents. Fig. 6 provides an illustration of this operator.

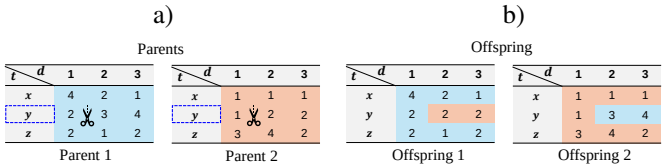


Figure 6: Illustrative example of the crossover mechanism.

4) Repair method: The repair method has two stages: (i) the first corrects initial train position violations by swapping or replacing the first decision; (ii) the second adjusts flow frequencies to stay within defined bounds via random reassignments. Afterward, step (i) is repeated to ensure consistency.

5) Penalty and filter: In addition to the repair method — which ensures compatibility with initial train positions and enforces flow frequency bounds — a complementary strategy was adopted to address remaining constraint violations. Specifically, individuals that include trips scheduled outside the valid time horizon are considered infeasible and penalized in their objective function values. This approach aims to promote convergence towards feasible solutions while preserving population diversity and enhancing the exploration of the solution space. Specifically, each constraint violation results in a penalty of 50% of the individual's fitness per violated bound. At the end of the simulation, the final population—and consequently the Pareto front—is filtered to retain only feasible individuals.

D. Solution Quality Metrics

To assess the quality of the obtained Pareto front, well-established performance indicators such as the spread metric (Δ) and the hypervolume indicator (HV, also known as the s-metric) are employed. These metrics capture solution diversity and convergence toward the Pareto-optimal front [15], [16].

The Δ indicator quantifies the distributional uniformity of solutions along the Pareto front, serving as a measure of solution diversity [16]. The hypervolume (HV) indicator, in turn, assesses the quality of a solution set A relative to a predefined reference point that is dominated by all elements of A [15]. Specifically, it calculates the hypervolume (or area in two-dimensional cases) of the objective space that is simultaneously dominated by A and bounded by the reference point. A larger hypervolume indicates a superior solution set in terms of both convergence toward and coverage of the true Pareto-optimal front.

E. Implementation Details

The algorithm was implemented in Python 3.10, using NumPy and Matplotlib libraries. All experiments were executed on a machine equipped with an Intel Core i7-8565U × 8 processor and 16 GB RAM, running Ubuntu 22.04. No GPU acceleration was employed.

IV. Results and Discussion

This section reports the results obtained from the estimation of Pareto fronts and analyzes their performance using the quality indicators Δ and hypervolume. Finally, a multi-criteria decision-making process is conducted using the Analytic Hierarchy Process (AHP) and the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS).

A. Hyperparameter Configuration of the Genetic Algorithm

Four hyperparameters are required for implementing the genetic algorithm: population size, number of generations, crossover probability, and mutation probability. Recognizing the significant impact that their values have on the overall performance and solution quality of evolutionary algorithms, a series of parameter tuning experiments were conducted. Initially, the values 200, 50, 0.9, and 0.3 were adopted, respectively, based on empirical knowledge. The following value ranges were systematically explored:

- Population size: 100, 200, 300 and 400 individuals;
- Number of generations: 30, 40, 50 and 60;
- Crossover probability: 0.7, 0.8, 0.9 and 0.95;
- Mutation probability: 0.1, 0.2, 0.3 and 0.4.

To guide the selection of suitable parameter values, a visual inspection of the Pareto fronts was conducted, supported by quantitative assessment using the Δ and HV metrics. Based on this analysis, a population size of 300 individuals was adopted. With this parameter fixed, subsequent experiments confirmed that 50 generations remained an appropriate choice. The initial crossover probability of 0.9 was also retained. However, the most significant improvement was observed when setting the mutation probability to 0.2.

Fig. 7 illustrates five estimated Pareto fronts obtained under the selected configuration. Collectively, the five estimated fronts cover a transported volume ranging from 249,000 to 294,000 tons, and an efficiency loss indicator range from 34 to 123 units, approximately.

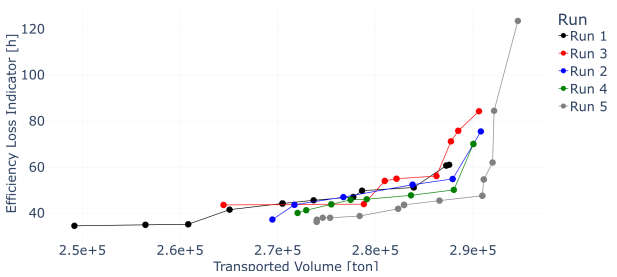


Figure 7: Pareto front estimates. Five runs were performed using the same set of hyperparameters, after their tuning.

B. Pareto Front Selection

Among the five available approximations of the Pareto front, one was selected for further analysis based on the highest number of non-dominated individuals, Run 1. For the hypervolume calculation, the reference point was defined by taking the minimum transported volume observed across all five fronts (reduced by 10%) and the maximum efficiency loss indicator (increased by 10%). This combined evaluation reinforces the selection of the Pareto front from Run 1, which also exhibited the best performance across the assessed metrics.

The selected Pareto front encompasses a solution space with transported volumes ranging from 274×10^3 to 294×10^3 tons and efficiency loss indicator values between 36 and 123 hours. Although 49 individuals in the final population contributed to the non-dominated set, they mapped to only 13 unique points in the objective space—indicating that many distinct solutions resulted in identical objective values, a common sign of convergence.

The quality metrics associated with this front were as follows: a Δ value of 0.857, a normalized hypervolume of 0.635. The diversity metric (Δ) is within the range generally regarded as satisfactory. Since the selected front does not contain solutions with lower transported volumes, its overall coverage is limited in that direction.

C. Decision-Making Process

Given the 13 non-dominated solutions comprising the estimated Pareto front, it became necessary to select a single representative solution to serve as the final recommendation. For this purpose, two well-established multi-criteria decision-making (MCDM) methods — AHP and TOPSIS — were employed.

In addition to the two primary objectives, namely transported volume and efficiency loss indicator, two supplementary decision criteria were incorporated: own-flow and environmental impact.

The own-flow metric counts trips along company-operated routes, favoring internal infrastructure over third-party assets.

The environmental impact metric was defined as the total accumulated travel time (both loaded and empty) across all valid flows. While this definition does not capture all aspects of environmental performance, it serves as a practical approximation for evaluating sustainability-related concerns in the absence of more detailed emissions data.

Once the four criteria were established, pairwise comparisons were performed to derive relative weights, as shown in Table I. These weights reflect the following scale: **1** (equal importance), **3** (moderate importance), **5** (strong importance), **7** (very strong importance), **9** (extreme importance), and **2, 4, 6, 8** as intermediate values.

Accordingly, transported volume was deemed the most critical criterion, with undisputed priority. Efficiency loss indicator was ranked as the second most important, albeit with only a moderate margin over the remaining criteria. Environmental impact was assigned the lowest relative importance among the four, reflecting its secondary role in the present decision context.

Table I: Pairwise comparison matrix of decision criteria (AHP).

Criteria	Transported volume	Efficiency loss	Own cargo	Environmental impact
Transp. vol.	1	5	7	8
Effic. loss	1/5	1	3	2
Own cargo	1/7	1/3	1	2
Env. Imp.	1/8	1/2	1/2	1
Sum	1.4679	6.8333	11.5	13

Subsequently, each element in the pairwise comparison matrix was normalized by dividing it by the total of its respective column. The average of the normalized values across each row was then computed to obtain the relative weight of each decision criterion, as presented in Table II.

Table II: Normalized decision matrix and derived priority weights.

Criteria	Transported volume	Efficiency loss	Own cargo	Environmental impact	Weights
Transp. vol.	0.6812	0.7317	0.6087	0.6154	0.6593
Effic. loss	0.1362	0.1463	0.2609	0.1538	0.1743
Own cargo	0.0973	0.0488	0.0870	0.1538	0.0967
Env. Imp.	0.0852	0.0732	0.0434	0.0769	0.0697

A consistency ratio of the matrix of 0.0465 was obtained. As this value is below the commonly accepted threshold of 0.1, the pairwise comparison matrix is considered consistent.

Subsequently, the weights derived from AHP were applied within the TOPSIS framework to automatically identify the most suitable solution from the selected Pareto front. TOPSIS was chosen for its straightforward implementation, interpretability of results, and natural alignment with the priority weights established via AHP. The technique identifies the alternative that is simultaneously closest to the ideal solution and farthest from the nadir (non-ideal) solution in the normalized objective space. The selected point is shown in Fig. 8. Note that it is consistent with the assigned weights, as it prioritizes transported volume over the other criteria, while

still remaining close to an intermediate value—an inherent characteristic of the TOPSIS method.

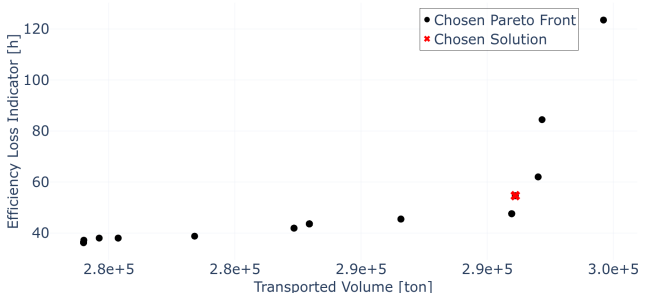


Figure 8: Pareto front estimation and selected solution.

The selected solution resulted in a transported volume of 291,114 tons, efficiency loss indicator of 54.64 hours, 53 own-flow operations, and 3,708 hours of environmental impact.

V. Conclusion

This study addressed freight train scheduling with queues at loading/unloading stations — often neglected or treated exogenously. The core contribution lies in demonstrating that it is possible to endogenously model queue dynamics within a deterministic mathematical framework. By embedding queue-time calculations directly into the formulation, the proposed approach enables the evaluation of operational efficiency in a way that reflects the causal propagation of congestion throughout the network.

Although the recursive nature of queues introduces significant computational challenges, the model allowed for consistent evaluation of solution quality without relying on simulation-based methods. This representation not only maintains the analytical structure of the problem, but also supports integration into optimization frameworks. To explore the solution space, a multi-objective genetic algorithm was applied, enabling the estimation of Pareto fronts that balance transported volume with system inefficiencies. Ensuring feasibility required careful individual construction and a repair mechanism, while time-window violations were addressed via penalization.

The results confirmed the viability of the proposed model: a representative solution was selected from a high-quality Pareto front, and multi-criteria decision-making methods supported this choice based on defined operational priorities. As future work, we suggest extending the model to incorporate uncertainty in service and travel times, exploring hybrid analytical-simulation techniques, and refining the solution process with adaptive or learning-based heuristics. The approach introduced in this study opens new possibilities for integrating queue behavior into network-level optimization models with greater fidelity and realism.

The simplifications adopted in this study — such as assuming unlimited station inventories, consistently high demand to fully utilize the fleet, and approximating environmental impact via total accumulated travel time — were necessary to enable a tractable deterministic model and preserve its analytical structure. As future work, we suggest incorporating more realistic constraints on inventory and demand, adopting direct environmental metrics (e.g., fuel consumption or estimated CO_2 emissions), and validating results through event-based simulation or comparison with real operational data. These extensions could enhance the model's fidelity without compromising its structural advantages, thereby broadening its practical applicability.

References

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[2] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations research*, vol. 12, no. 4, pp. 568–581, 1964.

[3] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Computers & industrial engineering*, vol. 99, pp. 300–313, 2016.

[4] M. Altinoz and O. T. Altinoz, "Multiobjective problem modeling of the capacitated vehicle routing problem with urgency in a pandemic period," *Neural Computing and Applications*, vol. 35, no. 5, pp. 3865–3882, 2023.

[5] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[6] A. Rybickova, J. Brodsky, A. Karaskova, and D. Mockova, "A genetic algorithm for the multi-depot vehicle routing problem," *Applied Mechanics and Materials*, vol. 803, pp. 69–75, 2015.

[7] R. Wu, R. Wang, J. Hao, Q. Wu, P. Wang, and D. Niyato, "Multiobjective vehicle routing optimization with time windows: A hybrid approach using deep reinforcement learning and NSGA-II," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[8] R. A. R. Santos, S. J. de Araujo Lima, and S. A. de Araujo, "Análise de representações cromossômicas para algoritmos genéticos na solução do problema de roteamento de veículos," in *Anais do XXXVI Encontro Nacional de Engenharia de Produção*, João Pessoa, PB, Brasil, 2016.

[9] G. Poonthalir and R. Nadarajan, "Green vehicle routing problem with queues," *Expert Systems with Applications*, vol. 138, p. 112823, 2019.

[10] J.-F. Cordeau, P. Toth, and D. Vigo, "A survey of optimization models for train routing and scheduling," *Transportation science*, vol. 32, no. 4, pp. 380–404, 1998.

[11] T. G. Crainic and G. Laporte, "Planning models for freight transportation," *European journal of operational research*, vol. 97, no. 3, pp. 409–438, 1997.

[12] A. Gómez, R. Mariño, R. Akhavan-Tabatabaei, A. L. Medaglia, and J. E. Mendoza, "On modeling stochastic travel and service times in vehicle routing," *Transportation Science*, vol. 50, no. 2, pp. 627–641, 2015.

[13] T. L. Saaty, "A scaling method for priorities in hierarchical structures," *Journal of mathematical psychology*, vol. 15, no. 3, pp. 234–281, 1977.

[14] C. Hwang and K. Yoon, *Multiple Attribute Decision Making: Methods and Applications*. Berlin, Germany: Springer-Verlag, 1981.

[15] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.