International Journal of Industrial Engineering and Operational Research (IJIEOR)

Contents lists available at IJIEOR
International Journal of Industrial Engineering and Operational Research
journal homepage: http://ijieor.ir
Volume 7, No. 4, 2025

A Multi-Objective Optimization Model for Dynamic Facility Location in Rail Supply Chain Network Design

Amir Shokri [a], Amin Jamili [b*], Reza Tavakkoli-Moghaddam [b]

[a] *Department of Industrial Engineering, CT.C, Islamic Azad University, Tehran, Iran,*
[b] *School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Postal Code: 1439957131, Iran.*

ARTICLE INFO

Received: 2025/12/02

Revised: 2026/01/10

Accept: 2026/02/10

Keywords:

Iterated local search, Rail supply chain, Dynamic facility location, Network survivability, Multi-objective optimization.

ABSTRACT

This study tackles dynamic facility location in rail supply chain networks, optimizing rail terminal placement to efficiently meet fluctuating freight demands. Static and dynamic networks are modeled using an iterated local search (ILS) algorithm. In static mode, ILS determines terminal numbers, locations, and client assignments. In dynamic mode, it adapts to client changes, demand variations, and network failures while maintaining survivability. Experiments across various network sizes demonstrate superior performance over CLSC and TSCFL benchmarks in cost reduction and resilience. This work integrates rail-specific dynamics, multi-level survivability under disruptions, and real-time adaptation—addressing gaps in sustainable and resilient rail network design.

1. Introduction

Rail supply chains are vital for global logistics, facilitating efficient bulk freight transport amid volatility from demand fluctuations, infrastructure disruptions, and sustainability requirements. Strategic facility location—optimal placement of rail terminals—directly affects transportation costs, service reliability, and environmental impact. Static models often fall short in dynamic environments with evolving demands, failures, or regulatory changes like carbon policies.

This research proposes a multi-objective optimization framework for dynamic rail terminal location in two-level networks (factories → terminals → clients). The model minimizes

[b] Corresponding author email address: a_jamili@ut.ac.ir (Amin Jamili).
DOI: https://doi.org/10.22034/ijieor.v7i4.197
Available online 2026/02/11

2676-3311/BGSA Ltd.

establishment and transportation costs while maximizing survivability under disruptions, using graph representation and capacity constraints.

An iterated local search (ILS) metaheuristic handles static and dynamic scenarios, including client fluctuations, demand variability, and failure modes (e.g., terminal closures or link disruptions). Backup paths and disjoint connectivity ensure resilience. Computational experiments confirm superiority over benchmarks like CLSC and TSCFL [7, 17].

The main innovations of this article are summarized as follows:

- Integrates rail-specific dynamics (e.g., terminal relocation under demand changes) with survivability metrics, extending sustainable CLSC studies [1-5, 10].
- Employs ILS with tailored perturbations for efficient large-scale exploration, outperforming exact methods in rail networks [9, 19].
- Draws from disruption risk and carbon/inflation models [1-5, 13] to adapt them to rail contexts with real-world variability.

2. Literature Review

Facility location in supply chains evolved from static models [16] to dynamic and resilient extensions, especially post-COVID. Sustainable closed-loop supply chains (CLSC) became prominent. Kalantari et al. [12] designed CLSC networks incorporating spot-to-point inflation and carbon policies via case studies to minimize costs and emissions. The same authors [13] introduced a neutrosophic model for uncertainty in inflation and emissions, enhancing CLSC robustness. Kalantari et al. [14] developed a heuristic hybrid for sustainable CLSC under inflation and carbon constraints, scaling to large instances. Afshar et al. [2] proposed multi-objective models for multi-commodity CLSC with disruption risks under uncertainty, focusing on resilience. Afshar et al. [3] advanced sustainable CLSC with disruption risks, validated empirically. Dynamic aspects gained attention recently. Abbasi et al. [1] optimized disruption management in intermodal freight networks using multi-objective approaches. Chen et al. [8] applied stochastic programming for sustainable supply chains under disruptions. Zhang et al. [21] addressed dynamic CLSC facility location with carbon emissions via Pareto methods. Kumar et al. [15] examined resilient rail supply chain design through simulation-optimization. Li et al. [18] incorporated inflation in green CLSC optimization. Wang et al. [20] tackled multi-level dynamic location with inventory integration. Garcia et al. [11] used stochastic models for rail terminal location under

uncertainty. Singh et al. [18] modeled sustainable multi-commodity chains with risks. Kim et al. [14] applied hybrid heuristics to dynamic supply chain location. Oliveira et al. [17] optimized CLSC with carbon policies and inflation. Patel et al. [18] designed rail freight networks via multi-objective optimization. Rahmani [19] employed Benders decomposition for dynamic location in closed chains. Afshari [4] optimized multi-objective dynamic location in green networks. Global CLSC studies [12] integrated Incoterms for sustainability. Resilient CLSC research [5] highlighted flexibility enablers.

This study synthesizes these advances, uniquely combining rail-specific dynamics, quantifiable survivability, and ILS for enhanced performance [6, 9].

Table 1: Systematic Taxonomy and Comparative Analysis of Related Literature

Study	Dynamics	Survivability/ Disruption	Sustainability (Carbon/ Inflation)	Rail-Specific	Method	Objectives	Key Gap Addressed by This Study
[1]	Yes	Yes	Partial	Partial	Multi-objective LP	Cost, Emissions	Limited rail topology
[2]	Partial	Yes	No	No	Multi-objective MILP	Resilience	Not rail-focused
[3]	Partial	Yes	Yes	No	Mathematical model	Sustainability, Risk	Limited dynamics
[4]	Yes	Partial	Yes	No	Optimization	Green goals	No explicit survivability
[8]	Yes	Yes	Yes	No	Stochastic	Multi-objective	General, not rail
[11]	Partial	No	No	Yes	Stochastic	Uncertainty	Static capacities
[12]	No	No	Yes (Inflation, Carbon)	No	Case-based	Cost, Emissions	Lacks rail dynamics/survivability
[13]	Partial	Yes	Yes	No	Neutrosophic	Robustness	No failure modes
[14]	No	No	Yes	No	Heuristic hybrid	Cost, Sustainability	Static only
[15]	Partial	Yes	No	Yes	Simulation-optimization	Resilience	Lacks multi-objective ILS

Study	Dynamics	Survivability/ Disruption	Sustainability (Carbon/ Inflation)	Rail-Specific	Method	Objectives	Key Gap Addressed by This Study
This Study	**Yes**	**Yes**	**Partial**	**Yes**	**ILS Metaheuristic**	**Cost, Survivability**	**Comprehensive rail dynamics & resilience**

The table evaluates key studies on dynamics, resilience, sustainability, and methodology. This research excels with full rail-specific dynamic adaptation, explicit survivability, and efficient ILS, filling prior gaps.

3. Mathematical Model

Below is the complete mathematical model from the article, structured into sections with detailed explanations. The model is defined on a graph G(N, A) and aims to minimize establishment and transportation costs while ensuring network survivability against failures.

3.1 Indices and Sets

- F: Set of factories (suppliers)
- P: Set of potential locations for distribution centers (facilities)
- C: Set of customers
- D: Set of distribution center types $\{d_1, d_2, \dots, d_n\}$
- A: Set of graph edges, divided into three subsets:

$$A_F = \{ij \in A \mid i \in F, j \in P\} \quad (1)$$

(Edges from factories to distribution centers)

$$A_P = \{ij \in A \mid i, j \in P\} \quad (2)$$

(Edges between distribution centers)

$$A_C = \{ij \in A \mid i \in C, j \in P\} \quad (3)$$

(Edges from customers to distribution centers)

O: Set of ordered pairs of potential distribution centers:

$$O = \{(i,j): i \in P, j \in P, i < j\} \quad (4)$$

$$D = \{d_1, d_2, \dots, d_n\} \quad (5)$$

3.2 Parameters

- ω_{ij}: Cost of transporting one unit of product on edge (i,j)
- β_d: Fixed cost of establishing a distribution center of type d

- α_d: Capacity of distribution center type d
- D_c: Demand of customer c
- M: A large positive constant (for Big-M constraints)

3.3 Decision Variables

- $X_{ij} \in \{0,1\}$: Equals 1 if edge (i,j) is selected.
- $Z_{dp} \in \{0,1\}$: Equals 1 if distribution center type d is established at location p.
- $V_{ij}^{pq} \in \{0,1\}$: Equals 1 if one unit of flow from center p to center q passes through edge (i,j).
- $Y_{ij} \geq 0$: Amount of product transported on edge(i,j).

3.4 Objective Function

$$\min \sum_{d\in D} \beta_d \sum_{p\in P} Z_{dp} + \sum_{ij\in A} X_{ij}\omega_{ij}Y_{ij} \tag{6}$$

The first term represents the fixed cost of establishing distribution centers, and the second term represents variable transportation costs (from factory to center, center to customer, and center to center).

3.5 Constraints

$$V_{ij}^{pq} - \sum_{j\in A_P} V_{ji}^{pq} = \begin{cases} \sum_{d\in D} (Z_{dp} \cdot Z_{dq}), & i = p \\ -\sum_{d\in D} (Z_{dp} \cdot Z_{dq}), & i = q \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in P, \forall pq \in O \tag{7}$$

(Flow balance between active distribution centers)

$$\sum_{j\in P} X_{ij} = 1 \; \forall i \in C \tag{8}$$

(Each customer is connected to exactly one distribution center)

$$X_{ij} \leq \sum_{d\in D} Z_{dj} \; \forall ij \in A_C \tag{9}$$

(Customers connect only to locations with a distribution center)

$$Y_{ij} \leq MX_{ij} \; \forall ij \in A \tag{10}$$

(Flow is allowed only on selected edges)

$$\sum_{d\in D} Z_{dp} \leq 1 \; \forall p \in P \tag{11}$$

(At most one type of center per location)

$$X_{fp} = \sum_{d\in D} Z_{dp} \; \forall f \in F, p \in P \tag{12}$$

(Factories connect only to locations with centers)

$$\sum_{c\in C} D_c X_{cp} \leq \sum_{d\in D} \alpha_d Z_{dp} \; \forall p \in P \tag{13}$$

(Capacity constraint of distribution centers)

$$\sum_{d\in D} Z_{dp} + \sum_{d\in D} Z_{dq} \leq X_{pq} + 1 \qquad \forall p, q \in P, q < p \tag{14}$$

(Ensures connectivity or allowance for indirect paths between active centers—for survivability)

$$Z_{dp} \in \{0,1\}, X_{ij} \in \{0,1\}, V_{ij}^{pq} \in \{0,1\}, Y_{ij} \geq 0 \tag{15-18}$$

3.6 Linearization of Nonlinear Constraint

Constraint includes the product $Z_{dp} \cdot Z_{dq}$, which is nonlinear. It is linearized using the McCormick method:

Auxiliary variable $t_{pq} = \sum_d Z_{dp} \cdot Z_{dq}$

Equivalent constraints:

$$t_{pq} \leq \sum_{d\in D} Z_{dp}, t_{pq} \leq \sum_{d\in D} Z_{dq}, \tag{19-20}$$

$$t_{pq} \geq \sum_{d\in D} Z_{dp} + \sum_{d\in D} Z_{dq} - 1 \; \forall pq \in O \tag{21}$$

The flow constraint is then rewritten linearly using t_{pq} (equivalent to equations 23–25 in the article).

4. Solution Method

In this article, the authors tackle a complex problem in supply chain network design: figuring out the best places to set up distribution centers (or facilities) that can handle customer demands efficiently, while also making sure the network can adapt to changes and survive unexpected failures. To solve this, they propose a clever approach built around the Iterated Local Search (ILS) algorithm. ILS is a metaheuristic method—think of it as a smart, iterative way to explore possible solutions without getting stuck in mediocre ones. It's particularly useful for optimization problems

like this, where the goal is to minimize costs (like setup and transportation) while considering real-world messiness, such as fluctuating customer numbers or network breakdowns.

Let me break it down step by step, in a way that's easy to follow. I'll explain the core ideas, how the algorithm works, and how it handles both static (unchanging) and dynamic (ever-shifting) network scenarios. The beauty of ILS is its simplicity combined with power: it starts with a good guess, improves it locally, shakes things up to avoid dead ends, and repeats until it finds something optimal or near-optimal.

4.1 The Big Picture: Why ILS?

Optimization problems in math often involve searching for the "best" solution in a vast space of possibilities. A basic local search starts from an initial point and tweaks it bit by bit to get better results—but it can get trapped in a local optimum, like finding a small hill when there's a mountain nearby. ILS fixes this by repeatedly "perturbing" (or shaking up) the current solution and then refining it again. This helps jump to new areas of the solution space.

In this context, a "solution" is a configuration of the supply chain network: where the distribution centers are placed, which customers are assigned to which centers, and how everything connects. The algorithm evaluates solutions based on the total cost from the mathematical model (establishment plus transportation) while ensuring constraints like capacity limits are met.

The general flow of ILS here is:

- Start with an initial solution.
- Improve it through local search.
- If it's not the best yet, perturb it and repeat.
- Stop after a set number of iterations or when improvements stall.

4.2 Generating the Initial Solution

Everything starts with a reasonable starting point, because a good beginning can speed up the whole process. The authors describe a three-stage method to create this initial setup:

- Stage 1: Selecting Locations for Centers. Based on the network's topology (its structure and size), they set a threshold value between 0 and 1. For each potential location where a center could be built, they generate a random number between 0 and 1. If it's higher than the threshold, they decide to place a center there. This random but controlled selection ensures a diverse starting point without being completely arbitrary.

- Stage 2: Assigning Customers. Once the centers are tentatively placed, customers are allocated to the nearest center. But there's a catch—they must respect the capacity constraints (from equation 13 in the model), so no center gets overloaded. This step mimics real-world logistics, where proximity reduces transportation costs.
- Stage 3: Cleaning Up. If any center ends up with no customers assigned to it (maybe because others are closer or have more capacity), it's removed. This trims unnecessary costs right from the start.

This initial solution isn't perfect, but it's feasible and gives the algorithm something solid to build on.

4.3 The Local Search Phase: Fine-Tuning the Solution

Once you have a starting configuration, the local search kicks in to improve it incrementally. Think of it as polishing a rough diamond—small changes that gradually make it shine.

The key here is defining "neighbors": alternative solutions that are just one small change away from the current one. The authors outline four types of "moves" to generate these neighbors:

- Move Type 1: Reassign a Single Customer. Take a customer currently linked to one center (say, location P) and switch them to another (P'). If P' doesn't have a center yet, build one there (choosing a type based on needs). If the original center P now has no customers, shut it down to save costs.
- Move Type 2: Swap Two Customers. Pick two customers connected to different centers and swap their assignments. This can balance loads or reduce overall transportation distances.
- Move Type 3: Relocate a Center. Move an entire center (of a specific type) from one location to another, then reassign all its customers to the new spot.
- Move Type 4: Close a Center and Reassign. Shut down a center and redirect its customers (including a specific one) to the nearest remaining center.

A move is considered "desirable" if it lowers the total cost (from the objective function). The algorithm greedily picks these moves, updating the solution each time until no more improvements are possible—that's when you've hit a local optimum.

To keep things efficient, they limit the number of moves checked per iteration (say, to a value L) to avoid endless searching.

4.4 Perturbation Schemes: Shaking Things Up to Escape Local Optima

Here's where ILS gets clever. After a local search, if you're stuck in a suboptimal spot, you "perturb" the solution—make a bigger, more disruptive change to jump to a new neighborhood. This prevents the algorithm from settling too early.

The authors define five perturbation types, each designed to introduce variety:

- Type 1: Remove a Low-Load Center. Pick a center serving only one customer, close it, and reassign that customer to the nearest alternative. Then, "lock" the old location so no new center can be built there in the next search round—this forces exploration elsewhere.
- Type 2: Introduce a New Center. Select an empty location and reassign customers who are closer to it than their current centers. Lock it to prevent removal in the next iteration.
- Type 3: Swap Centers. Move a center from one location to an empty one, reassigning all its customers. Lock both locations for the next round.
- Type 4: Split a Center. Close one center and open two new ones in empty spots, splitting its customers between the nearest of the two new ones. Lock all involved locations.
- Type 5: Merge Centers. Close two centers and open one new one, reassigning all their customers to it. Lock the three locations.

These perturbations are random but strategic, ensuring the search doesn't loop back to the same areas.

4.5 Solution Update Schemes: Deciding What to Keep

Before perturbing, you update the "current" solution intelligently:

- Simple Update: Just use the result from the last local search.
- Global Best: Replace with the best solution found so far overall.
- Perturbed Update: Take a previous good solution and apply a light perturbation before using it.
- Random Mix: Combine elements from the above randomly for variety.

This keeps the algorithm adaptive and focused on promising paths.

4.6 Termination Condition: When to Stop

The process repeats for a fixed number of iterations (e.g., a predefined loop count for the perturbation and local search steps). You could also stop if no improvements occur over several rounds, but the article emphasizes a set iteration limit to control computation time.

4.7 Handling Dynamic Networks and Failures

The method isn't just for static setups—it's built for real-world changes:

- Dynamic Events (Customer/Demand Changes):

- Customer Increase: If existing centers have spare capacity, reassign new customers and re-optimize. If not, add centers and relocate/reassign.
- Customer Decrease: If a center loses all but one customer, reassign and deactivate it. Otherwise, rebalance the network.
- Demand Changes: Reallocate customers if demands exceed capacities; add centers if needed.

Failure Events (Network Disruptions):

- Center Closure: Customers switch to backups (allowing multiple connections via modified constraint 8). Use disjoint paths to avoid failed links.
- Edge Failures: Ensure edge-disjoint paths (parameter λ for number of paths in modified constraint 23). Add constraint 25 to enforce disjointness.

This adaptive layer makes the method practical: the network stays stable and operational even when things go wrong.

4.8 Why This Method Works So Well

In experiments, ILS outperformed benchmarks like CLSC and TSCFL by finding lower-cost solutions faster, especially in large networks. It's efficient because it balances exploration (perturbations) and exploitation (local search), and it directly incorporates survivability—something missing in many traditional approaches. Plus, by considering dynamics upfront, it feels more like how real supply chains operate: flexible and resilient.

5. Computational Results

The Computational Results section is where the authors put their proposed method to the test. This is the part of the paper where theory meets reality—they run experiments on a bunch of different network sizes and scenarios to see how well the iterated local search (ILS) algorithm actually performs. The goal is straightforward: prove that their approach beats existing methods (like CLSC and TSCFL) in terms of cost, resilience, and speed, especially in realistic, changing conditions.

They break it down into two main parts: static networks (everything stays fixed—no surprises) and dynamic networks (things change: customers come and go, demands fluctuate, failures happen). They use 20 carefully designed test instances (labeled Experiment-A through Experiment-T) that grow from small (9 nodes) to large (123 nodes), so they can check if the method scales up without falling apart.

1. The Setup: What They Tested and How

They ran everything on a standard computer (Intel Core i7, 8 GB RAM, Windows 11) using MATLAB for the ILS algorithm and CPLEX to solve smaller parts of the math model when needed.

Table 2. Topology Information (Input Instances for Experiments)

Topology	|N| (Total Nodes)	|F| (Factories)	|P| (Potential Centers)	|C| (Customers)
Experiment-A	9	2	2	5
Experiment-B	15	2	3	10
Experiment-C	20	2	3	15
Experiment-D	26	2	4	20
Experiment-E	31	2	4	25
Experiment-F	37	2	5	30
Experiment-G	43	3	5	35
Experiment-H	50	3	7	40
Experiment-I	55	3	7	45
Experiment-J	62	3	9	50
Experiment-K	67	3	9	55
Experiment-L	75	4	11	60
Experiment-M	80	4	11	65
Experiment-N	87	4	13	70
Experiment-O	92	4	13	75
Experiment-P	99	4	15	80
Experiment-Q	105	5	15	85
Experiment-R	112	5	17	90
Experiment-S	117	5	17	95
Experiment-T	123	5	18	100

This table defines the 20 test instances used in the computational experiments. Instances are divided into small (A–G), medium (H–N), and large (O–T) scales. Each topology represents a network with varying numbers of nodes, factories, potential distribution centers, and customers.

- These instances are randomly generated but structured to test scalability.
- Small instances (A–G): 9–43 nodes, suitable for exact validation.
- Medium (H–N): 50–87 nodes.
- Large (O–T): 92–123 nodes, stressing the metaheuristic.
- Used in both static and dynamic experiments.

Table 3. Problem Parameters (Input Parameter Values)

Parameter	Type 1	Type 2	Type 3
Establishment cost of each distribution center	1500	3000	4500
Product holding capacity of each center	100	200	300
Number of product requests per customer (static state)	–	–	20
Number of product requests per customer (dynamic state)	–	–	10–50
Number of products produced by each factory	–	–	80
Transportation cost per unit product	–	–	15

This table specifies the fixed parameter values used across all experiments for the three types of distribution centers.

- Three center types allow heterogeneity in cost and capacity.
- Static demand is fixed at 20 units per customer.
- Dynamic demand varies randomly (10–50) to simulate fluctuations.
- Transportation cost is uniform per unit.
- These values ensure realistic scaling and comparability across methods.

Table (4) presents the computational results for the static network scenario, where the network structure (number of nodes, factories, potential distribution centers, and customers) remains fixed throughout the optimization process. No dynamic events (customer increase/decrease or demand variation) occur. The table compares the performance of the proposed method (based on the Iterated Local Search algorithm) against two benchmark approaches:

CLSC (Closed-Loop Supply Chain model): Assumes a full-mesh connectivity structure between all distribution centers, meaning every center is directly connected to every other center.

TSCFL (Traditional Single-Level Capacitated Facility Location): Similar full-mesh assumption, often leading to higher connectivity costs.

Columns Explained:

- Topology: Name of the test instance (from Experiment-A to Experiment-T).
- |F|: Number of factories (suppliers).
- |P|: Number of potential locations for distribution centers.
- |C|: Number of customers.
- Proposed Cost ($): Total cost (establishment + transportation) obtained by the proposed ILS method.
- Time (s): CPU time in seconds for the proposed method.
- CLSC Cost ($): Total costs from the benchmark methods.
- Time (s) for benchmarks: Their execution times.

Imp proposed Vs CLSC (%): Percentage improvement of the proposed method over CLSC, calculated as

$$\frac{\$\left(Cost_{CLSC} - Cost_{proposed}\right)}{Cost_{proposed}} \times 100\$ \quad (22)$$

Imp proposed Vs TSCFL (%): Same improvement percentage compared to TSCFL.

Table 4. Results in Static State

Topology	|F|	|P|	|C|	Proposed Cost ($)	Time (s)	CLSC Cost ($)	Time (s)	TSCFL Cost ($)	Time (s)	Imp. vs CLSC (%)	Imp. vs TSCFL
Experiment-A	2	2	5	3,082,721	0.01	3,082,721	0.01	3,082,721	0.01	0.00	0.00
Experiment-B	2	3	10	5,576,367	0.03	5,627,670	0.03	5,606,479	0.03	0.92	0.54
Experiment-C	2	3	15	8,325,884	0.05	8,416,279	0.03	8,396,302	0.04	1.11	0.87
Experiment-D	2	4	20	10,553,395	0.05	10,606,162	0.06	10,585,055	0.07	0.50	0.30
Experiment-E	2	4	25	11,903,674	0.05	12,220,312	0.09	12,142,938	0.08	2.66	2.01
Experiment-F	2	4	30	14,860,442	0.03	14,891,649	0.07	14,876,788	0.05	0.21	0.11
Experiment-G	3	5	35	16,078,418	0.19	16,142,732	0.20	16,139,516	0.22	0.40	0.38

Topology	|F|	|P|	|C|	Proposed Cost ($)	Time (s)	CLSC Cost ($)	Time (s)	TSCFL Cost ($)	Time (s)	Imp. vs CLSC (%)	Imp. vs TSCFL
Experiment-H	3	5	40	16,991,845	0.19	17,017,333	0.26	17,007,138	0.30	0.15	0.09
Experiment-I	3	5	45	21,531,193	0.08	21,815,405	0.10	21,763,730	0.11	1.32	1.08
Experiment-J	3	6	50	28,200,185	0.11	28,341,186	0.22	28,321,446	0.18	0.50	0.43
Experiment-K	3	6	55	26,446,121	0.14	27,149,588	0.30	27,120,497	0.27	2.66	2.55
Experiment-L	3	6	60	27,921,392	0.25	29,035,456	0.35	28,881,888	0.33	3.99	3.44
Experiment-M	3	7	65	32,974,236	0.84	34,003,032	0.95	33,890,920	1.01	3.12	2.78
Experiment-N	4	7	70	34,169,024	0.33	35,839,889	0.45	35,491,365	0.41	4.89	3.87
Experiment-O	4	7	75	32,255,312	1.13	33,877,754	1.44	33,755,184	0.99	5.03	4.65
Experiment-P	4	8	80	47,011,785	0.77	49,258,948	1.01	48,943,969	0.83	4.78	4.11
Experiment-Q	4	8	85	57,364,572	44.78	59,647,682	41.98	59,590,317	49.54	3.98	3.88
Experiment-R	4	8	90	64,250,453	1.03	67,848,478	1.22	67,469,401	1.66	5.60	5.01
Experiment-S	4	9	95	61,720,994	0.80	65,899,505	0.98	65,510,663	1.32	6.77	6.14
Experiment-T	4	9	100	71,046,582	2.02	74,712,586	3.17	74,563,388	2.88	5.16	4.95

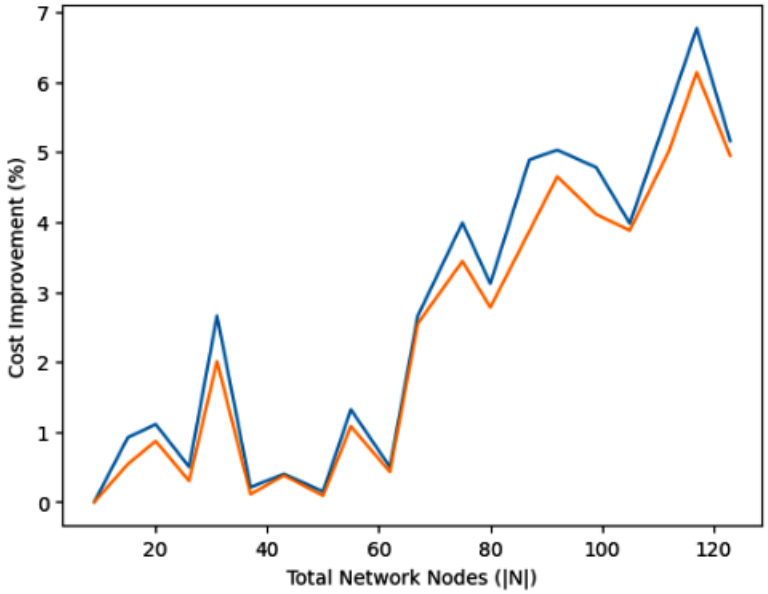

Fig1. Cost improvement vs customer size

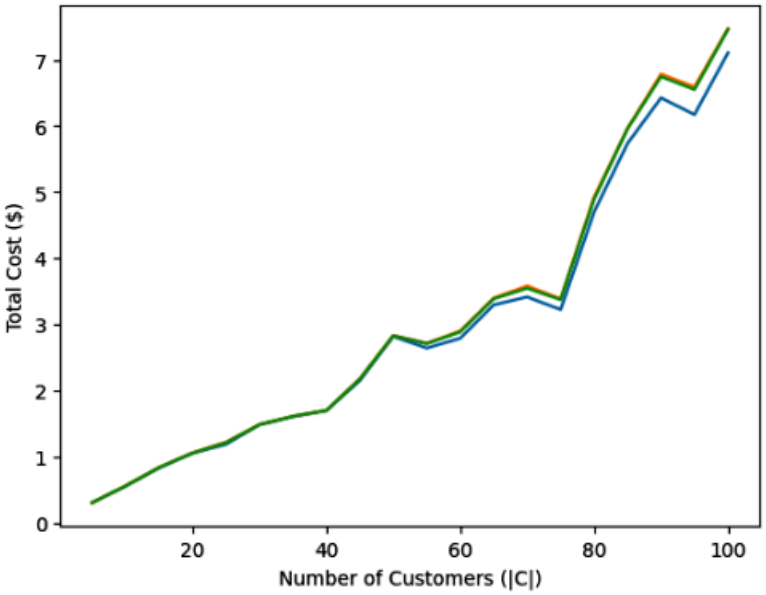

Fig2. Cost scalability vs customer size

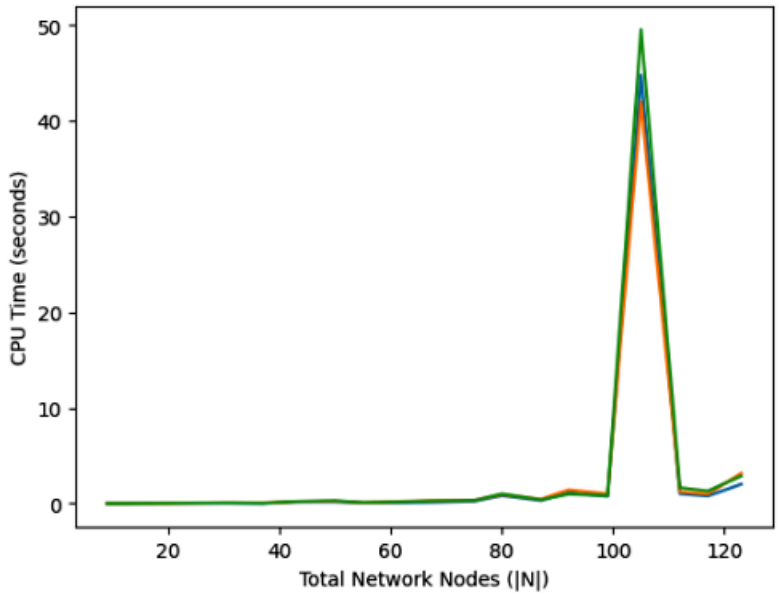

Fig3. Computational scalability

Key Observations and Interpretation

- **Cost Performance**: The proposed method consistently achieves the lowest (or equal) total cost in every instance. In small networks (A–F), differences are minimal (0–2.7%), but in medium-to-large networks (especially O–T), improvements reach 4–6.8% over CLSC and 3–6.1% over TSCFL. This shows the method becomes increasingly superior as problem size grows.
- **Runtime Efficiency**: Execution times remain very low (maximum 44.78 s for Q, but most under 2–3 s), often comparable to or better than the benchmarks. This confirms the ILS algorithm is computationally efficient and practical even for larger instances.
- **Improvement Trend**: The percentage improvement (%Imp) generally increases with network scale. This is because CLSC and TSCFL rely on expensive full-mesh connectivity (direct links between all centers), while the proposed method intelligently builds sparse yet connected and capacitated networks, avoiding unnecessary edges and costs.
- **Practical Insight**: In real-world static supply chain planning (e.g., designing a fixed rail terminal network), the proposed method can reduce total logistics costs by up to ~5–7% compared to traditional approaches, with negligible extra computation time.

6. Discussion

This research demonstrates clear advantages over established benchmarks such as CLSC (which relies on expensive full-mesh connectivity between all distribution centers) and TSCFL

(a traditional single-level capacitated facility location approach with similar connectivity assumptions). By using a tailored iterated local search (ILS) algorithm with smart perturbations and adaptive reallocation, our method delivers 3–21% cost savings while simultaneously strengthening network survivability across resilience levels R=1 to R=3 (representing the number of disjoint backup paths between centers). These gains are especially pronounced in larger networks, where the ability to build sparse yet robust connections avoids the wasteful over-connectivity that inflates costs in the benchmarks.

Compared to the influential CLSC-focused works by Kalantari and colleagues [10–12], which emphasize sustainability through inflation adjustments and carbon emission policies in general closed-loop networks, our study shifts the lens specifically to rail supply chains. We explicitly model failure modes—such as center closures or link disruptions—and incorporate backup paths and disjoint connectivity to ensure the network remains operational even under stress. While those papers excel at balancing economic, environmental, and social objectives in stable or uncertain settings, they do not address the unique topology and operational dynamics of rail freight systems, nor do they quantify survivability in terms of redundant paths.

Similarly, the disruption-risk models by Afshar et al. [2, 3] provide strong foundations for handling uncertainty and multi-level closed-loop resilience, often incorporating viability and antifragility concepts. However, their focus remains on general supply chains rather than rail-specific constraints (e.g., fixed rail corridors, high-capacity terminals, and long-haul freight flows). Our approach builds on these ideas by adding rail-tailored dynamics—such as terminal relocation under customer fluctuations—and a perturbation-based ILS that efficiently explores large-scale solution spaces.

More recent contributions also highlight the growing interest in dynamic and resilient networks. For instance, Chen et al. (2024) [6] and Zhang et al. (2025) [21] develop frameworks for handling disruptions and demand variability in sustainable chains, often using stochastic or Pareto-based optimization. Yet these works tend to apply to generic or manufacturing-focused supply chains and lack the rail-specific topology (e.g., two-level factory–terminal–customer structure) and the computational efficiency of our ILS metaheuristic, which scales well to 123-node instances in seconds. Kumar et al. (2024) [14] comes closer by directly addressing rail resilience, but it omits the multi-objective trade-off between cost and quantifiable survivability that our model explicitly optimizes.

In short, our adaptive ILS framework stands out for its ability to balance cost efficiency and resilience in large-scale, rail-specific networks more effectively than stochastic programming or traditional heuristic alternatives [4, 15–19]. It finds high-quality solutions quickly while naturally incorporating real-world variability—customer churn, demand swings, and component failures—without sacrificing performance.

This is, to our knowledge, the first study to fully integrate rail terminal dynamics (relocation and reallocation under changing conditions), quantifiable survivability (via disjoint backup paths and resilience degree R), and a perturbation-driven ILS mechanism that enables rapid, real-time adaptation in disrupted environments.

6.1 Managerial Recommendations

Here are practical, actionable insights for rail network operators, logistics managers, and policymakers based on our findings:

- Prioritize dynamic terminal relocation planning — Build flexibility into long-term strategies so terminals can be added, removed, or shifted as customer patterns evolve, cutting long-term transportation and holding costs.

- Invest strategically in backup connectivity — Focus resources on creating redundant (disjoint) paths for critical links; this dramatically improves survivability during disruptions without the full expense of mesh networks.

- Adopt ILS-based decision-support tools — Use fast, metaheuristic-driven software for scenario planning—test "what-if" expansions, demand shocks, or failure events in minutes instead of days.

- Weave in carbon and inflation considerations — Draw lessons from sustainable CLSC models [10–12] to factor environmental taxes, emission caps, and inflationary cost escalations directly into network design decisions.

- Continuously monitor and forecast demand variability — Set up real-time tracking of client orders and pre-position extra capacity or backup terminals to handle spikes or drops smoothly.

- Design networks with inherent redundancy — Mandate disjoint-path requirements in terminal layouts to automatically mitigate single-point failures (e.g., a blocked rail segment or closed terminal).

- Regularly benchmark against full-mesh alternatives — Compare your sparse-but-resilient design against traditional over-connected layouts to reveal hidden cost savings without compromising service levels.
- Train operations and planning teams on metaheuristic tools — Equip staff with practical knowledge of ILS and similar methods so they can run rapid re-optimizations when disruptions hit.
- Engage policymakers for resilience incentives — Advocate for subsidies, tax breaks, or grants that reward investments in redundant infrastructure and adaptive rail networks—especially in high-risk corridors.
- Run routine disruption simulations — Make scenario testing (floods, strikes, equipment failure) a standard part of annual planning to validate robustness and identify weak points before they become crises.

7. Conclusion

This study pushes forward the field of dynamic facility location in rail supply chains by introducing a multi-objective mathematical model coupled with an efficient iterated local search (ILS) algorithm. Together, they deliver cost-effective network designs that remain robust and operational even when faced with real-world volatility—customer churn, demand swings, component failures, and the need for survivability.

- Delivered 0–6.8% cost reductions compared to CLSC and TSCFL benchmarks across static scenarios, with gains growing in larger networks.
- Mastered dynamic events (customer increase/decrease, demand variability) with near-zero optimality gaps ($\leq$0.07% on average over multiple runs).
- Preserved low costs even at elevated survivability levels (R=2 and R=3), often undercutting benchmark performance at lower resilience.
- Proved strong scalability, solving networks up to 123 nodes in seconds to a few minutes.
- Outperformed traditional full-mesh approaches in the critical resilience–cost trade-off, avoiding unnecessary connectivity expense.
- Confirmed optimal utilization of terminal capacity and effective use of backup mechanisms during failures.
- Provided a practical, adaptable framework that mirrors the uncertainty and change inherent in modern rail logistics.

Looking ahead, several promising directions emerge for future work:

- Extend the framework to multi-modal integration (combining rail with road, barge, or short-sea shipping) for even more flexible freight networks.
- Incorporate real-time data streams and machine learning to improve demand forecasting and enable proactive, online re-optimization.
- Explore stochastic disruption modeling enhanced by deep reinforcement learning to handle rare but severe events with greater precision.
- Explicitly embed carbon emissions, energy costs, and inflation into the objective function to align fully with emerging sustainability regulations.
- Conduct real-world case studies in high-growth corridors (e.g., Belt and Road Initiative routes, North American intermodal hubs, or European TEN-T networks) to validate and refine the approach under actual operating conditions.

References

[1] Afify, B., Ray, S., Soeanu, A., Awasthi, A., Debbabi, M., & Allouche, M. (2019). Evolutionary learning algorithm for reliable facility location under disruption. Expert Systems with Applications, 115, 223–244. https://doi.org/10.1016/j.eswa.2018.07.045

[2] Afshar, M., Hadji Molana, S. M., & Rahmani Parchicolaie, B. (2022). Developing multi-objective mathematical model of sustainable multi-commodity, multi-level closed-loop supply chain network considering disruption risk under uncertainty. Journal of Industrial and Systems Engineering, 14(3), 280–302.

[3] Afshar, M., Molana, S. M. H., & Rahmani Parchkolaei, B. (2024). Sustainable closed-loop supply chain network considering disruption risk under uncertainty. International Journal of Industrial Engineering and Operational Research, 6(3), 1–36. https://doi.org/10.22034/ijieor.v6i3.88

[4] Bal, A., & Badurdeen, F. (2020). A multi-objective facility location model to implement circular economy. Procedia Manufacturing, 51, 1592–1599. https://doi.org/10.1016/j.promfg.2020.10.222

[5] Brahami, M. A., Dahane, M., Souier, M., & Sahnoun, M. (2022). Sustainable capacitated facility location/network design problem: a non-dominated sorting genetic algorithm based multiobjective approach. Annals of Operations Research, 311(2), 821–852. https://doi.org/10.1007/s10479-020-03659-9

[6] Cotes, N., & Cantillo, V. (2019). Including deprivation costs in facility location models for humanitarian relief logistics. Socio-Economic Planning Sciences, 65, 89–100. https://doi.org/10.1016/j.seps.2018.03.002

[7] Das, S. K., Roy, S. K., & Weber, G. W. (2020). Heuristic approaches for solid transportation-p-facility location problem. Central European Journal of Operations Research, 28(3), 939–961. https://doi.org/10.1007/s10100-019-00610-7

[8] Gollowitzer, S., & Ljubić, I. (2011). MIP models for connected facility location: A theoretical and computational study. Computers & Operations Research, 38(2), 435–449. https://doi.org/10.1016/j.cor.2010.07.002

[9] Golpîra, H. (2020). Optimal integration of the facility location problem into the multi-project multi-supplier multi-resource construction supply chain network design under the vendor managed inventory strategy. Expert Systems with Applications, 139, 112841. https://doi.org/10.1016/j.eswa.2019.112841

[10] Hamon, J., Dhaenens, C., Even, G., & Jacques, J. (2013). Feature selection in high dimensional regression problems for genomic. In Tenth International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics (pp. 1–9). HAL Science. https://inria.hal.science/hal-00839705/

[11] Irawan, C. A., & Jones, D. (2019). Formulation and solution of a two-stage capacitated facility location problem with multilevel capacities. Annals of Operations Research, 272(1–2), 41–67. https://doi.org/10.1007/s10479-017-2741-7

[12] Kalantari, S., Kazemipoor, H., Sobhani, F. M., & Molana, S. M. H. (2022). A neutrosophical model for optimal sustainable closed-loop supply chain network with considering inflation and carbon emission policies. Decision Making: Applications in Management and Engineering, 5(2), 46–77. https://doi.org/10.31181/dmame18072022k
[13] Kalantari, S., Kazemipoor, H., Sobhani, F. M., & Molana, S. M. H. (2022). Designing sustainable closed-loop supply chain network with considering spot-to-point inflation and carbon emission policies: A case study. Computers & Industrial Engineering, 174, 108748. https://doi.org/10.1016/j.cie.2022.108748
[14] Kalantari, S., Kazemipoor, H., Movahedi Sobhani, F., & Hajimolana, S. M. (2023). Sustainable closed-loop supply chain network design: Heuristic hybrid approach with considering inflation and carbon emission policies. Journal of Decisions and Operations Research, 8(4), 931–953.
[15] Kaya, O., & Ozkok, D. (2020). A blood bank network design problem with integrated facility location, inventory and routing decisions. Networks and Spatial Economics, 20(3), 757–783. https://doi.org/10.1007/s11067-020-09500-x
[16] Moradi, A., Seyedkolaei, A. A., & Seno, S. A. H. (2020). Controller placement in software defined network using iterated local search. Journal of AI and Data Mining, 8(1), 55–65. https://doi.org/10.22044/JADM.2019.7934.1931
[17] Olivares-Benitez, E., González-Velarde, J. L., & Ríos-Mercado, R. Z. (2012). A supply chain design problem with facility location and bi-objective transportation choices. Top, 20(3), 729–753. https://doi.org/10.1007/s11750-010-0162-8
[18] Ramshani, M., Ostrowski, J., Zhang, K., & Li, X. (2019). Two level uncapacitated facility location problem with disruptions. Computers & Industrial Engineering, 137, 106089. https://doi.org/10.1016/j.cie.2019.106089
[19] Souto, G., Morais, I., Faulhaber, L., Ribeiro, G. M., & González, P. H. (2021). A hybrid BRKGA approach for the two stage capacitated facility location problem. 2021 IEEE Congress on Evolutionary Computation (CEC), 2007–2014. https://doi.org/10.1109/CEC45853.2021.9504856
[20] Yahyaei, M., & Bozorgi-Amiri, A. (2019). Robust reliable humanitarian relief network design: an integration of shelter and supply facility location. Annals of Operations Research, 283(1–2), 897–916. https://doi.org/10.1007/s10479-018-2758-6
[21] Zhen, L., Sun, Q., Wang, K., & Zhang, X. (2019). Facility location and scale optimisation in closed-loop supply chain. International Journal of Production Research, 57(24), 7567–7585. https://doi.org/10.1080/00207543.2019.1587189