OmniaScience

Journal of Industrial Engineering and Management
JIEM, 2025 – 18(3): 577-593 – Online ISSN: 2013-0953 – Print ISSN: 2013-8423
https://doi.org/10.3926/jiem.8938

Multi-Objective Metaheuristics for Hub Location Problem: A Case Study of Rail Network Planning

Nam Pham-Van[1], Nha Nguyen-Dang[1], Ornurai Sangsawang[2], Sunarin Chanta[2*]

[1]*Vietnam-Korea University of Information and Communication Technology (Vietnam)*
[2]*King Mongkut's University of Technology North Bangkok (Thailand)*

nampv.21it@vku.udn.vn, nhand.21it@vku.udn.vn, ornurai.s@itm.kmutnb.ac.th
[*]*Corresponding author: sunarin.c@itm.kmutnb.ac.th*

Received: June 2025
Accepted: October 2025

Abstract:

Purpose: Hub location problems have been utilized in various applications including rail transportation network planning, where hub serves as a key transit point within the network. In this paper, we focus on determining the optimal location for a rail transportation hub, where cost and service are trade-off.

Design/methodology/approach: The problem is formulated as a multi-objective programming model with the objectives of minimizing total transportation costs and minimizing maximum travel time. A case study of rail transportation network hub planning in Thailand is presented. Given the complexity and large scale of the real-world case study, we develop and compare the Multi-Objective Tabu Search (MOTS) and Non-Dominated Sorting Genetic Algorithm II (NSGA-II) to solve the problem.

Findings: The proposed algorithms yield efficient performance in terms of computational time and solution quality. Performance comparison is further analyzed to see the difference in both algorithms.

Originality/value: The results offer valuable managerial insights for decision-makers in rail transportation hub network design.

Keywords: multi-objective optimization, hub location problem, rail transportation network, tabu search, genetic algorithm

To cite this article:

Pham-Van, N., Nguyen-Dang, N., Sangsawang, O., & Chanta, S. (2025). Multi-objective metaheuristics for hub location problem: A case study of rail network planning. *Journal of Industrial Engineering and Management*, 18(3), 577-593. https://doi.org/10.3926/jiem.8938

1. Introduction

Transportation hub network design is a complex process involving multiple factors such as location, passenger demand, connectivity, operational efficiency, and budget constraints. A hub functions as a central transit point in a transportation network model, facilitating the movement of passengers or goods by connecting various routes and modes of transport, thereby optimizing efficiency and reducing travel times. The location of a transportation hub

significantly influences passenger volume, as it affects accessibility, connectivity, and the convenience of travel within the network. Strategically located hubs can attract higher passenger traffic by offering efficient transfer points and reducing overall travel time. Conversely, poorly located hubs may deter usage due to inconvenient connections or extended travel durations. Therefore, careful consideration of hub placement is a key factor for optimizing passenger flow and enhancing the effectiveness of the transportation system.

Rail transport is recognized as an efficient mode of transportation, capable of moving substantial quantities of goods and passengers. Furthermore, it is generally more economical when evaluating transportation costs per unit compared to alternative modes of transport. However, rail transportation is subject to certain limitations, primarily due to the fixed locations of origin and destination stations. Consequently, passengers are required to travel from their residences to the train station and subsequently from the destination train station to their final destinations. In many instances, the total travel time associated with train transport may exceed that of a direct trip from origin to destination.

In this paper, we aim to determine the optimal location for a transportation hub in a railway network system. The objectives are to minimize total transportation cost and minimize maximum travel time. Transportation costs are comprised of three components: incoming transport to the train station, transit between train stations, and outgoing transport from the train station. Travel time is categorized into two parts: the time taken to travel from the origin to the destination and delay time. Several important factors are considered, including passenger demand, the locations of passenger origins and destinations, and hub capacity. The problem is formulated as a multi-objective programming model and applied to rail transportation hub network planning in Thailand. Given the complexity and large size of the real-world case, we develop metaheuristic algorithms, specifically the Multi-Objective Tabu Search (MOTS) and Non-Dominated Sorting Genetic Algorithm II (NSGA-II), to address the problem. The proposed algorithms demonstrate efficient performance with acceptable solving times and high-quality solutions.

The remainder of the paper is organized as follows. Section 2 reviews the literature on existing studies of the Hub Location Problem (HLP) and associated solution methods. Section 3 presents the proposed multi-objective HLP model. Sections 4 and 5 describe the developed metaheuristic algorithms, namely MOTS and NSGA-II, respectively. In Section 6, the two algorithms are evaluated on a real-world case study of rail transportation network design, along with computational results and performance comparisons. Finally, Section 7 provides the conclusions and directions of future research.

2. Literature Review

2.1. Hub Location Problems

A hub location network strategically designs transportation systems for goods and passengers by routing traffic through centralized hubs instead of direct connections between origins and destinations. This consolidation at hubs leverages economies of scale to reduce transportation costs. This model is applicable in logistics, distribution centers, international transport, public transit, postal services, and optimizing locations for airports and bus terminals to efficiently connect passengers to their final destinations (Wang, Liu & Yang, 2023; Attar, Irawan, Akbari & Zhong, 2024; Ogazon, Anaya-Arenas & Ruiz, 2025; Wandelt, Wang & Sun, 2025).

The foundational research on the p-hub median problem was first developed by O'Kelly (1987), who formulated it as a quadratic mathematical model for the uncapacitated single p-hub median problem (USApHMP). Campbell (1994) expanded this by introducing the uncapacitated multiple p-hub median problem (UMApHMP). Yaman (2011) extended the hub location problem by introducing the uncapacitated r-allocation p-hub median problem (UrApHMP), which simplifies to a single-allocation problem when $r = 1$ and becomes a multiple-allocation problem when $1 < r \leq p$. Peker, Kara, Campbell and Alumur (2016) contributed to this area by developing a clustering-based approach to identify potential hub sets. They focused on selecting optimal hub locations based on factors like flow volume and geographic data, particularly in major urban centers, while ensuring that hubs are not located too closely together. Several approaches have been explored in hub location research. Ernst and Krishnamoorthy (1996) developed an effective mathematical formulation for the single allocation p-hub median problem and applied the Simulated Annealing algorithm to solve the USApHMP. They later expanded their work to address the multiple-allocation version of the problem (Ernst, Hamacher, Jiang, Krishnamoorthy & Woeginger,

2009). The *p*-hub median problem is known for its computational complexity and is classified as *NP*-hard (Kara & Tansel, 2000). Since such problems require excessive computational effort to solve exactly, metaheuristic algorithms are often preferred as solution methods.

2.2. Solution Methods for Hub Location Problems

Various metaheuristic techniques have been employed, including Genetic Algorithms (GA) by Kratica, Stanimirovic, Tosic and Filipovic (2007) and Topcuoglu, Ergin, Ermis and Yilmaz (2005). Benaini, Berrajaa, Boukachour and Oudani (2019) introduced a parallel genetic algorithm to solve the USA*p*HMP. More recent advancements include the Simheuristic Approach developed by Jost, Grochala, Schumacher, Ammouriova & Juan (2023). Tabu search (TS) has also been investigated in hub location research, particularly through the development of the TABUHUB method by Skorin-Kapov and Skorin-Kapov (1994), which was later refined by Abyazi-Sani and Ghanbari (2016). Additionally, Guan, Lin and Feng (2018) implemented a Learning-based Tabu Search (LTS) to address the uncapacitated single allocation hub location problem.

Numerous studies have investigated multi-objective hub location problems. Ghaffarinasab (2020) introduced a bi-objective hub location problem aimed at minimizing total costs while maximizing service levels by reducing the longest path between origin and destination pairs. The author also developed a Tabu Search (TS) to efficiently solve this problem. Soylu and Katip (2019) proposed a multi-objective hub-airport location problem for an airline network design, using *p*-median hub problem to minimize total transportation cost and 2-stop routes. A variable neighborhood search (VNS) heuristic was proposed to solve the problem. Chobar, Adibi and Kazemi (2021) proposed a bi-objective mathematical model focused on minimizing transportation costs and pollution emissions. They utilized the Non-Dominated Sorting Genetic Algorithm (NSGA-II) and the Non-Dominated Ranking Genetic Algorithm (NRGA) to identify optimal locations for tourist hubs. Li, Han, Zhou and Gu (2023) developed a bi-objective mathematical model for food transportation within a hub location network during the COVID-19 pandemic, aiming to minimize both transportation time and cost. To solve the problem, they used Grey Wolf Optimizer (GWO).

Previous studies indicate that TS has been successfully applied to hub location problems due to its intensification capabilities and ability to generate effective allocation solutions (Skorin-Kapov & Skorin-Kapov, 1994; Abyazi-Sani & Ghanbari, 2016). Meanwhile, NSGA-II is widely used for solving multi-objective problems because it effectively preserves diversity points across the Pareto front (Chobar et al., 2021; Karimi-Mamaghan, Mohammadi, Pirayesh, Karimi-Mamaghan & Irani, 2020). Since our study focuses on a multi-objective rail hub location problem, the most suitable algorithm remains unclear. Moreover, no previous study has compared these two algorithms for railway hub location problems under a bi-objective model minimizing total cost and maximum travel time. Therefore, we develop both NSGA-II and TS and evaluate their performance on a real-world case study.

3. Mathematical Model

In this section, we introduce a mathematical model to determine the optimal location and allocation of hub in a network. The proposed multi-objective Hub Location Problem (HLP) model is formulated as follows. Let $G = (N, A)$ is a connected graph, where $N=\{1, 2, \ldots, n\}$ represents a set of nodes, n is the number of nodes, p is the number of hubs, and A represents a set of arcs. The indices $i, j, k, m \in N$, where i and j denote nodes, and k and m denote hubs. Parameters are defined as follows. There are two types of cost; C_{ij}^{km} indicates transportation cost from node i to node j via hubs k and m, where $C_{ij}^{km} = C_{ik} + \alpha C_{km} + C_{mj}$. The transportation cost is composited of the cost from node to hub, hub to hub, and hub to node, respectively, where α is the discount for transport via hub to hub. F_k indicates fixed cost of establishing hub k. W_{ij} indicates the amount of flow from node i to node j. The travel time is defined based on the travel time and delay time, where T_{ij}^{km} indicates traveling time from node i to node j. $T_{ij}^{km} = T_{ik} + \beta T_{km} + T_{mj} + S_k + S_m$, where T_{ik} indicates the traveling time between node i to hub k, S_k indicates delay time at hub k, and β indicates the discount factor for traveling time between a hub link (hub to hub). The hub capacity is also considered, where H_k indicates the maximum capacity of hub k. Decision variables are defined as follows; X_{ij}^{km} represents amount of flow that is routed from node i through hubs k and m, X_k represent hub location, where $X_k = 1$ if node k is selected to be hub, $X_k = 0$ otherwise. The mathematical model is detailed as the following.

$$\text{Minimize } Z_1 = \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{m=1}^{n}\sum_{j=1}^{n} W_{ij} C_{ij}^{km} X_{ij}^{km} + \sum_{k=1}^{n} F_k X_k \tag{1}$$

$$\text{Minimize } Z_2 = \max_{i,k,m,j} \{T_{ij}^{km} X_{ij}^{km}\} \tag{2}$$

Subject to:

$$\sum_{k=1}^{n}\sum_{m=1}^{n} X_{ij}^{km} = 1 \ \forall i, j \tag{3}$$

$$X_{ij}^{km} \le X_k \ \forall i,k,m,j \tag{4}$$

$$X_{ij}^{km} \le X_m \ \forall i,k,m,j \tag{5}$$

$$\sum_{i=1}^{n}\sum_{m=1}^{n}\sum_{j=1}^{n} W_{ij} X_{ij}^{km} + \sum_{i=1}^{n}\sum_{l=1}^{n}\sum_{j=1}^{n} W_{ij} X_{ij}^{lk} \le H_k X_k \ \forall i,k \tag{6}$$

$$\sum_{k=1}^{n} X_k = p \ \forall k \tag{7}$$

$$X_{ij}^{km} \ge 0 \ \forall i,k,m,j \tag{8}$$

$$X_k \in \{0,1\} \ \forall k \tag{9}$$

The first objective in Equation (1) is to minimize the total transportation cost, which is composed of two types of costs: the variable cost C_{ij}^{km} and the fixed cost F_k. The variable cost is associated with travelled distance and consists of three components, which are the collection cost, the distribution cost, and the transfer cost between hub arcs represented by the discount factor α. The fixed cost is the cost of establishing hubs. The second objective in Equation (2) is to minimize the maximum travel time of a passenger in the system. The travel time is composed of the traveling time from origin to destination including delay time at hubs. Note that the traveling time is associated with a distance traveling through hubs gets a discount factor β. Constraint (3) ensures that flow (from i to j) at each node is allocated to exactly one pair of hub locations (k and m). Constraints (4) - (5) allow assigning flow to a pair of hub locations after they have been located (k and m are selected as a hub). Constraint (6) restricts the amount of incoming and outgoing flow from nodes to a capacitated hub. Constraint (7) limits on the number of hubs that should be opened. Constraints (8) - (9) are the non-negativity and binary requirements, respectively. This problem is complex since the size of decision variables is of $O(N^4)$ and the second objective (Z_2) is not a linear function, so these make the whole problem hard to solve. Therefore, it is necessary to develop a metaheuristic algorithm for solving the problem.

4. Multi-Objective Tabu Search (MOTS)

We present an approach grounded in the Tabu Search (TS) algorithm, originally developed by Glover (1989) coupled with dominance principles, to address multi-objective optimization problems. Our MOTS method leverages the strengths of TS, particularly its ability to explore local neighborhoods effectively, enhancing efficiency when identifying solutions with superior objective values. Simultaneously, the Tabu Search mechanism mitigates the risk of premature convergence to local optima, enabling a more robust exploration of the solution space and facilitating the discovery of optimal Pareto frontiers (Gendreau & Potvin, 2010). The main step of the proposed MOTS consists of 4 parts.

4.1. Initial Solution

For solution representation, we use the permutation representation, as illustrated in Figures 1 and 2. In this structure, each index represents a node, while the corresponding value indicates the hub to which that node is allocated. This approach allows for clear and efficient mapping of nodes to hubs, facilitating the computation of the solution's objective functions.

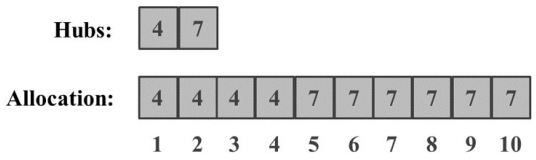

Figure 1. Solution representation

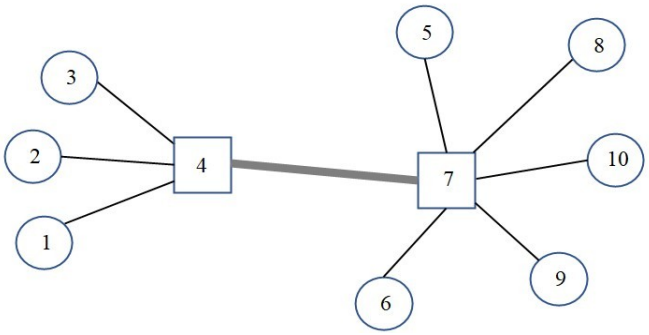

Figure 2. The hubs and nodes in the network have represented in Figure 1

The initialization phase is crucial as it significantly impacts both the convergence speed and the likelihood of finding the optimal solution. To ensure a robust start, we prioritize nodes with higher potential to become hubs based on their total weight (inbound and outbound). After calculating the total weight for each node, we rank them accordingly and select the top n nodes as initial hubs for the first solution. This method enhances the efficiency and effectiveness of the algorithm's search process. The steps of selecting the initial hub based on total weights are described in pseudocode of Algorithm1.

Algorithm 1 Select Potential Hubs

Require: Number of nodes n, weight matrix $weights$
Ensure: Dictionary of potential hubs
Initialize an empty dictionary $potential\_hub$
Create a list of nodes from 0 to $n-1$
for each node i in nodes **do**
  Compute the sum of weights for the current node i:
  $sum\_weights \leftarrow \sum_{j=0}^{n-1} weights[j,i] + \sum_{j=0}^{n-1} weights[i,j]$
  Set $potential\_hub[i] \leftarrow sum\_weights$
end for
return $potential\_hub$

Once the initial hubs are selected, the next step is to focus on node allocation. We use the nearest distance method, where each node is assigned to the hub that is closest in distance. This ensures that each node is efficiently connected to a hub, minimizing the overall travel distance. After completing this process, we obtain a fully initialized solution, consisting of both the selected hubs and the corresponding node allocations. The steps of allocating nodes to the p potential hubs are described in pseudocode of Algorithm 2.

Algorithm 2 Allocate nodes to p potential hubs

Require: Number of nodes n, number of hubs p, weight matrix *weights*, distance matrix *distances*
Ensure: Selected hubs *hubs* and allocation for each node *allocation*
Compute potential hubs using the **Algorithms 1:** Select Potential Hubs
Sort potential hubs based on their weights
Select the top p hubs from the sorted list
Initialize allocation list
for each node i in range 0 to $n-1$ **do**
  Assign the closest hub to node i:
  $allocation[i] \leftarrow \text{argmin}_{h \in hubs}\, distances[i][h]$
end for
return *hubs*, *allocation*

4.2. Generating Neighborhood

At each iteration, a new solution is generated based on the current solution's neighborhood. To generate a neighborhood, we explore a modification to the hub set. We retain one hub from the current solution and replace the remaining hubs with nodes. This strategy broadens the search space, allowing for a more comprehensive exploration of potential optimal solutions. For allocation, we continue to apply the nearest distance method, as it remains efficient even with large-scale inputs. The steps of generating a neighborhood are described in pseudocode of Algorithm 3.

Algorithm 3 Generate Neighborhood

Require: List of hubs *hubs*, number of nodes n, distance matrix *distances*
Ensure: List of neighborhood solutions *neighborhood*
Initialize an empty list *neighborhoods*
Create a set of hubs *hub_set*
Identify non-hub nodes as the difference between all nodes and *hub_set*
for each hub in hubs **do**
  for each non-hub in non_hubs **do**
    Create a copy of current hubs *new_hubs*
    Remove the current hub from *new_hubs*
    Add the non-hub to *new_hubs*
    Compute new assignments based on *new_hubs*:
    $new\_assignments[i] \leftarrow \text{argmin}_{h \in new\_hubs}\, distances[i][h]$ for each node i
    Append the tuple $(new\_hubs, new\_assignments)$ to *neighborhoods*
  end for
end for
return *neighborhoods*

4.3. Identifying Pareto Frontier

To be able to identify the Pareto Frontier, solutions are classified into two groups; dominated solutions and non-dominated solutions. First, we check the dominance of the current solution whether the current solution dominates any neighboring solution—if it does, that neighbor is discarded. If not, we retain the neighbor. The steps of checking the dominance are described in pseudocode of Algorithm 4.

Algorithm 4 Dominates Function

Require: Two solutions $sol1 = (cost1, time1)$ and $sol2 = (cost2, time2)$
Ensure: Boolean result indicating if $sol1$ dominates $sol2$
Extract $cost1$ and $time1$ from $sol1$
Extract $cost2$ and $time2$ from $sol2$
return $(cost1 \leq cost2$ **and** $time1 < time2)$ **or** $(cost1 < cost2$ **and** $time1 \leq time2)$

After the non-dominated neighboring solutions are successfully generated, we further refine the set of Pareto Frontier by classifying the remaining non-dominated neighbors. Any solution dominated by another in this subset is also removed. This process ensures that the final set consists of superior solutions that are non-dominated by both the current solution and other neighboring solutions. The steps of removing dominated solutions are described in pseudocode of Algorithm 5.

Algorithm 5 Remove Dominated Solutions

Require: A list of solutions *solutions*
Ensure: A list of non-dominated solutions
Initialize an empty list *non_dominated*
for each solution *candidate* in *solutions* **do**
  Set *dominated* ← False
  Initialize an empty list *to_remove*
  for each solution *other* in *non_dominated* **do**
    if *candidate* dominates *other* **then**
      Add *other* to *to_remove*
    else if *other* dominates *candidate* **then**
      Set *dominated* ← True
      break the loop
    end if
  end for
  if *dominated* = False **then**
    Add *candidate* to *non_dominated*
  end if
  for each solution *sol* in *to_remove* **do**
    Remove *sol* from *non_dominated*
  end for
end for
return *non_dominated*

4.4. Main Algorithm of MOTS

In the main algorithm of our proposed MOTS, we have input data, which are number of nodes, number of hubs, distance matrix, flow matrix or weights, hub capacity, and discount factors; α, β. The initial solution is firstly generated and set as a Pareto Frontier. It is important to note that the initial solution is critical as it directly affects the performance of the algorithm. A poorly generated initial solution may lead to suboptimal outcomes and extend the time required to reach optimality, thereby increasing computational costs and complexity. Next, we improve the current solution by generating a neighborhood solution, where the number of neighborhood solutions at each iteration is $2(n\text{-}p)$. Then, we check for dominance between the current solution and its neighbors. Once new neighboring solutions are generated, we also check for dominance among all neighboring solutions. The tabu tenure list and the Pareto Frontier set is updated at each iteration. The tabu tenure is set at 15 based on Silva and Cunha (2017) and preliminary tests (5, 10, 15) to maintain intensification. The process is continued until the algorithm reaches the maximum iterations at 50 iterations or show no improvement for 10 consecutive iterations. The steps of the main algorithm of MOTS are demonstrated in Figure 3.

5. Non-Dominated Sorting Genetic Algorithm II (NSGA-II)

NSGA-II was introduced by Deb, Pratap, Agarwal and Meyarivan (2002). It gained prominence due to its superior performance in solving multi-objective optimization problems. Building on the foundation of NSGA, NSGA-II inherits and preserves beneficial mechanisms such as retaining the best solutions. Additionally, it optimizes the complexity of non-dominated sorting, simplifies the calculation of crowding distance, and enhances efficiency in maintaining population diversity. As a result, NSGA-II can tackle larger problems and deliver better outcomes. The main step of NSGA-II consists of 6 parts.

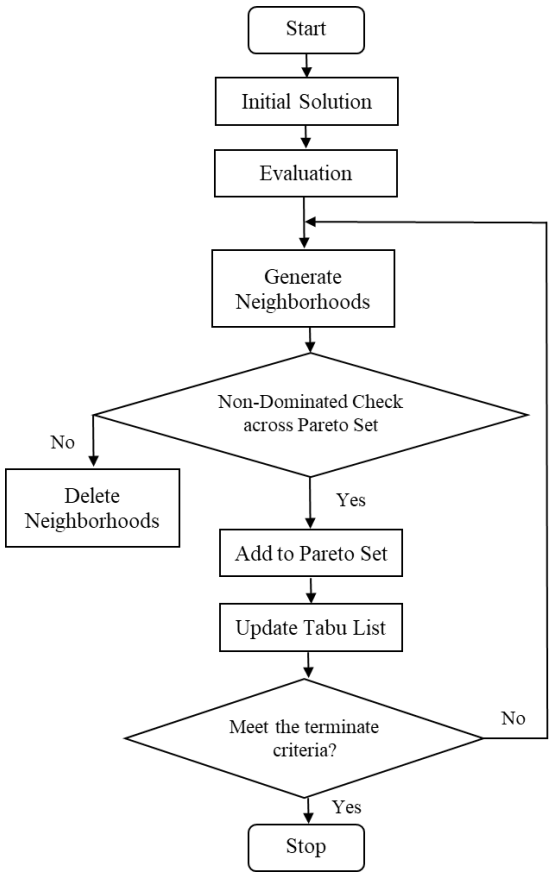

Figure 3. Flowchart of Multi-Objective Tabu Search

5.1. Initial Population

We use a permutation representation for the hub location problem. A solution is represented as an array, where the array's length corresponds to the number of nodes in the problem. Each element in the array represents a node, and from the set of nodes, a certain number of nodes is selected to serve as hubs, with the number of hubs determined by the specific problem requirements. The value of each element in the array indicates the hub to which the corresponding node will be allocated.

Population initialization is an important step in NSGA-II, significantly influencing the quality of subsequent generations. The better the quality of the initial population, the easier it is to reach the global optimum. The first population must ensure diversity, essential for effective crossover and mutation processes. A population that contains many high-quality solutions will make exploring the solution space more efficient.

First and foremost, the population initialization step must ensure that all solutions generated are feasible. For the hub location problem, solutions must satisfy constraints such as hub capacity, number of nodes, and the number of hubs. During the initialization process, solutions are generated with a predetermined number of hubs and nodes. These solutions are then checked for constraints to ensure they are feasible. Additionally, to increase the likelihood of good solutions in the population, a fixed number of solutions will be created based on elite nodes selected as hubs, while the rest of the solutions will be generated by randomly selecting hubs. The allocation of each node will be based on the nearest hub. Figure 4 illustrates the initial population.

After the new population is initialized, we evaluate the fitness and check the feasibility of each solution within the population. The fitness of the solution is based on the objective functions, which are minimizing total transportation cost and maximum travel time. The feasibility is based on the validation of all constraints of the problem. The population size is set to 200 based on Demir, Ergin and Kiraz (2016) and preliminary tests (50, 100, 200), as smaller sizes reduce diversity and increase the risk of local optima.

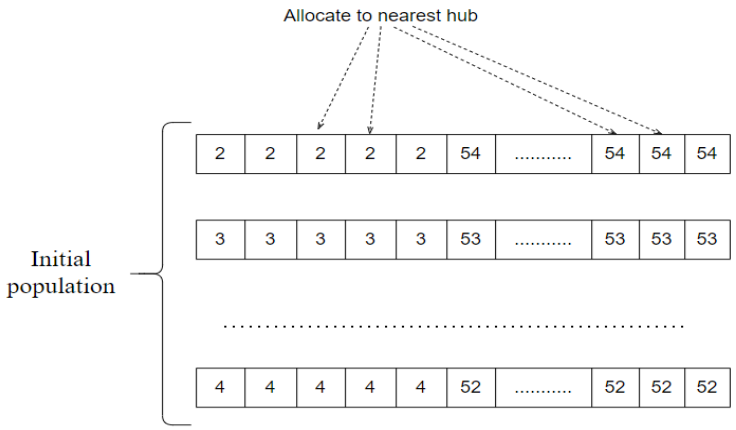

Figure 4. Initial population

5.2. Non-Dominated Sorting

In this step, we classify solutions based on dominance. Each solution has to be compared with other solutions, whether it dominates other solutions or it is dominated by any solution. We set up 2 vectors for non-dominated sorting; dominance count and dominated list. The dominance count records the number of solutions that dominate a given solution. A lower count means the solution is closer to the Pareto Front. The dominated list is a set of solutions that are dominated by a given solution. The steps of non-dominated sorting are as follows.

Step 1: Compare each solution pair. If solution A is at least as good as solution B in all objectives and better in at least one, then A dominates B. We update B's dominance count and add B to A's dominated list.

Step 2: Solutions with a dominance count of 0 are ranked as Rank 1 (Pareto Front).

Step 3: For each solution, reduce the dominance count of the solution in its dominated list. Solutions with a new dominance count of 0 are moved to Rank 1. Solutions with a high number of dominance counts are put in Rank 2, and Rank 3, respectively. Keep doing this until all solutions are ranked, ensuring that every solution is placed in the correct tier based on its dominant relationship with others.

The steps of non-dominated sorting are described in pseudocode of Algorithm 6.

5.3. Crowding Distance Calculation

The calculation of crowding distance plays a critical role in ensuring population diversity. It helps prevent the algorithm from prematurely converging to local optima and getting stuck there. For each rank, the first step in calculating crowding distance is identifying the boundary solutions with the smallest and largest values for each objective function using negative infinity and positive infinity, respectively. The solution with the smallest value for an objective function is assigned a crowding distance of negative infinity. This assignment ensures that the algorithm recognizes it as a boundary solution, which is crucial for preserving the extremities of the Pareto Front. Similarly, the solution with the largest value for an objective function is assigned a crowding distance of positive infinity.

By assigning these extreme values, the algorithm effectively sets boundaries for the solution space, ensuring that the solutions at the edges are retained in the next generation. For solutions that are not at the boundaries, the crowding distance is calculated based on the distance between adjacent solutions in the objective space. This calculation helps quantify how isolated a solution is compared to its neighbors. For example: to calculate the crowding distance for solution i, we consider its immediate neighbor's solutions $i-1$ and $i+1$.

For each objective function, we calculate the difference in objective values between solutions $i-1$ and $i+1$. The crowding distance for solution i is then determined by summing these differences across all objective functions. This sum gives a measure of how much the solution is "crowded" by its neighbors. The steps of calculating crowding distance are described in pseudocode of Algorithm 7.

Algorithm 6 Non-Dominated Sorting

Require: Cost values ($cost\_value$), Maximum time values (max_time_value), Population size (pop_size)
Ensure: Sorted fronts ($fronts$)

$fronts \leftarrow [[]]$
$domination\_count \leftarrow [0, 0, \ldots, 0]$
$dominated\_solutions \leftarrow$ [[] for each solution in population]
for each solution p in population **do**
  for each solution q in population **do**
    if $(cost\_value[p] < cost\_value[q] \land max\_time\_value[p] \leq max\_time\_value[q]) \lor (cost\_value[p] \leq cost\_value[q] \land max\_time\_value[p] < max\_time\_value[q])$ **then**
      $dominated\_solutions[p]$.append(q)
    else if $(cost\_value[q] < cost\_value[p] \land max\_time\_value[q] \leq max\_time\_value[p]) \lor (cost\_value[q] \leq cost\_value[p] \land max\_time\_value[q] < max\_time\_value[p])$ **then**
      $domination\_count[p] \leftarrow domination\_count[p] + 1$
    end if
  end for
  if $domination\_count[p] == 0$ **then**
    $fronts[0]$.append(p)
  end if
end for
$i \leftarrow 0$
while $fronts[i]$ is not empty **do**
  $next\_front \leftarrow []$
  for each solution p in $fronts[i]$ **do**
    for each solution q in $dominated\_solutions[p]$ **do**
      $domination\_count[q] \leftarrow domination\_count[q] - 1$
      if $domination\_count[q] == 0$ **then**
        $next\_front$.append(q)
      end if
    end for
  end for
  $i \leftarrow i + 1$
  $fronts$.append($next\_front$)
end while
$fronts$.pop() ▷ Remove the last empty front
return $fronts$

Algorithm 7 Calculate Crowding Distance

Require: Front of solutions $front$, Cost values $cost\_value$, Maximum time values max_time_value
Ensure: Crowding distances $distance$

$distance \leftarrow [0, 0, \ldots, 0]$ ▷ Initialize distance array
$sorted\_by\_cost \leftarrow$ sort($front$, by cost values)
$sorted\_by\_time \leftarrow$ sort($front$, by time values)
$distance[sorted\_by\_cost[0]] \leftarrow \infty$
$distance[sorted\_by\_cost[-1]] \leftarrow \infty$
$distance[sorted\_by\_time[0]] \leftarrow \infty$
$distance[sorted\_by\_time[-1]] \leftarrow \infty$
for each solution i in 1 to $|front| - 2$ **do**
  $distance[sorted\_by\_cost[i]] \leftarrow distance[sorted\_by\_cost[i]] + \frac{cost\_value[sorted\_by\_cost[i+1]] - cost\_value[sorted\_by\_cost[i-1]]}{\max(cost\_value) - \min(cost\_value)}$
  $distance[sorted\_by\_time[i]] \leftarrow distance[sorted\_by\_time[i]] + \frac{max\_time\_value[sorted\_by\_time[i+1]] - max\_time\_value[sorted\_by\_time[i-1]]}{\max(max\_time\_value) - \min(max\_time\_value)}$
end for
return $distance$

5.4. Parent Selection

To select parents for the crossover and mutation processes, the goal is to choose the best solutions to undergo these operations. We select 4 solutions from the population and pair them into two groups. Each pair is then compared based on their rank in the non-dominated sorting and their crowding distance values. The two solutions that are selected will become the parents. This process is repeated until the number of parents (equal to the number of offspring that will be created) equals the number of solutions in the population.

5.5. Crossover and Mutation

To increase solution diversity, crossover and mutation are applied to the parents selected in the previous step. For each pair of parents, random crossover points are chosen, and crossover is performed to produce two offspring. For each offspring, a random position is selected for mutation, after which all generated offspring are added to the offspring set. The crossover rate (P_c) and mutation rate (P_m) are set at 0.9 and 0.3 based on Demir et al. (2016) and preliminary tests (P_c:0.7, 0.8, 0.9; P_m:0.1, 0.2, 0.3), respectively, to maximize diversity.

5.6. Next Generation

The next step in the NSGA-II algorithm involves integrating the newly generated offspring with the current population to form a combined set of solutions. This process allows us to consider both the best solutions from the previous generation (200 solutions) and the new offspring set (100 solutions). By combining these two sets, we create a more diverse and comprehensive pool of potential solutions that can be evaluated and ranked to identify the most optimal solutions for the next generation.

We evaluate the solutions in this combined set once again using the two methods: non-dominated sorting and crowding distance, just as before. We then initialize a new population by adding solutions based on priority—first by rank (from lowest to highest) and then by crowding distance (from highest to lowest). Only 200 solutions are selected to be the new population. The termination criterion is set at 50 maximum iterations or no improvement for 10 consecutive iterations. The steps of the procedure of the NSGA-II are demonstrated as a flow chart in Figure 5.

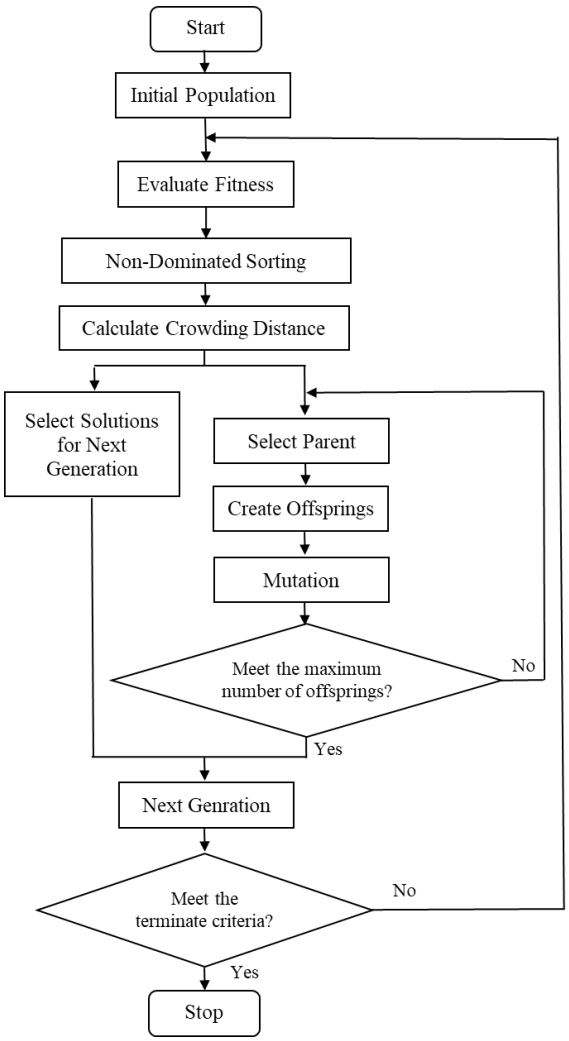

Figure 5. Flowchart of NSGA-II

6. Computational Results

6.1. A Case Study of Rail Network Planning

The main rail network in Thailand is constructed from the north to the south. In this study, we use a north route line, which starts from Bangkok to Chiang Mai. The total number of stations is more than 100, however, only 54 stations are considered to be a candidate of rail hub. In this problem, we aim to determine the optimal location of rail hub in this network. The proposed metaheuristics algorithms, MOTS and NSGA-II are implemented to find the optimal solutions. The number of passengers, the distance between stations and nodes are given, the number of nodes (n) is fixed at 54, the number of hubs (p) is varied from 2 to 10. The experiments are conducted and run on a computer with Intel Core i5-12450H 2.0 GHz CPU, 8GB RAM, and 512GB memory. The results are shown in Table 1, where the minimum solution of each case is reported. According to total transportation costs (minimizing) and maximum travel time (minimizing), both algorithms provide competitive solutions. Both MOTS and NSGA-II are similarly capable of finding the lowest total cost, while MOTS is more likely to find the lowest maximum travel time. However, NSGA-II tends to perform faster than MOTS, especially when number of hubs is increased.

p	Algorithm	Hub Set	Total Cost (baht)	Max Travel Time (min)	Run Time (s)
2	MOTS	5,40	28,707,690	594.17	23.36
	NSGA-II	4,40	28,741,786	600.14	252.46
3	MOTS	1,21,40	21,128,862	531.41	51.98
	NSGA-II	1,21,40	21,128,862	531.41	255.24
4	MOTS	4,24,46,52	21,188,402	420.83	210.22
	NSGA-II	1,22,40, 54	17,423,540	494.80	259.57
5	MOTS	1,13,31,46,52	16,118,298	371.67	144.89
	NSGA-II	1,12,22,40,54	14,765,891	488.37	261.35
6	MOTS	3,12,22,37,47,52	13,701,317	359.66	74.05
	NSGA-II	1,12,22,34,47,54	13,132,487	414.96	261.43
7	MOTS	1,12,22,27,37,47,52	12,159,936	363.38	117.19
	NSGA-II	1,12,22,30,40,47,54	11,844,744	414.96	256.07
8	MOTS	1,6,12,22,27,37,47,52	11,229,865	349.30	113.25
	NSGA-II	1,6,12,22,32,40,47,54	10,822,743	401.19	261.24
9	MOTS	1,6,12,22,27,34,40,47,52	10,215,346	349.30	217.51
	NSGA-II	1,34,6,40,12,47,22,54,27	9,939,782	401.19	264.09
10	MOTS	1,5,12,22,27,34,40,47,51,54	9,624,971	345.14	649.78
	NSGA-II	1,6,12,22,27,30,34,40,47,54	9,480,293	398.27	267.59

Table 1. Hub set solutions of MOTS and NSGA-II

The two main objectives that we considering are total transportation cost and maximum travel time. These objectives often conflict, as solutions with lower transportation costs may lead to longer travel times due to more complex routing. Conversely, achieving shorter overall travel time may result in higher costs. The Pareto Front obtained of both algorithms are graphically presented in Figures 6-8, which illustrate the trade-off between the two objectives. The non-dominated solutions across the three cases show notable differences. For the case with 3 hubs, NSGA-II tends to outperform MOTS by producing lower objective values. In contrast, for the cases with 5 and 7 hubs, MOTS performs slightly better than NSGA-II. To draw more robust conclusions, the next section presents a statistical hypothesis test to further analyze these differences.

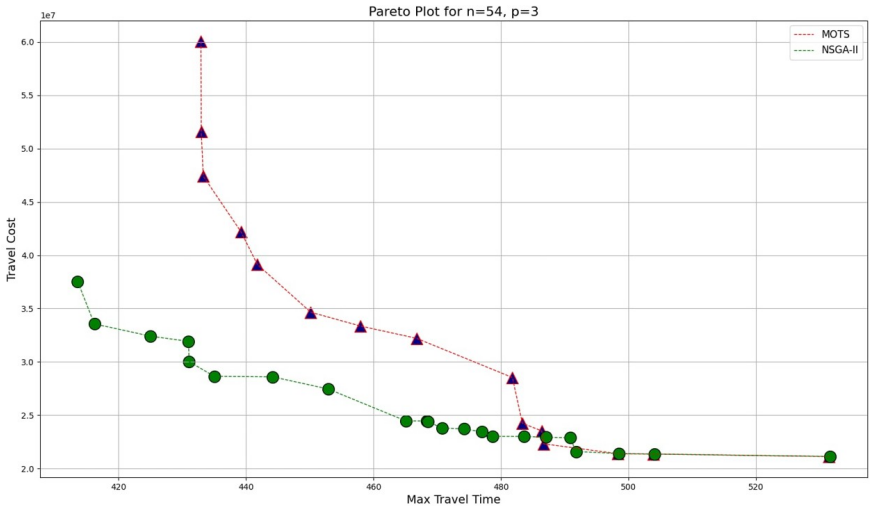

Figure 6. Pareto Fronts of the NSGA-II and MOTS for 54 nodes and 3 hubs

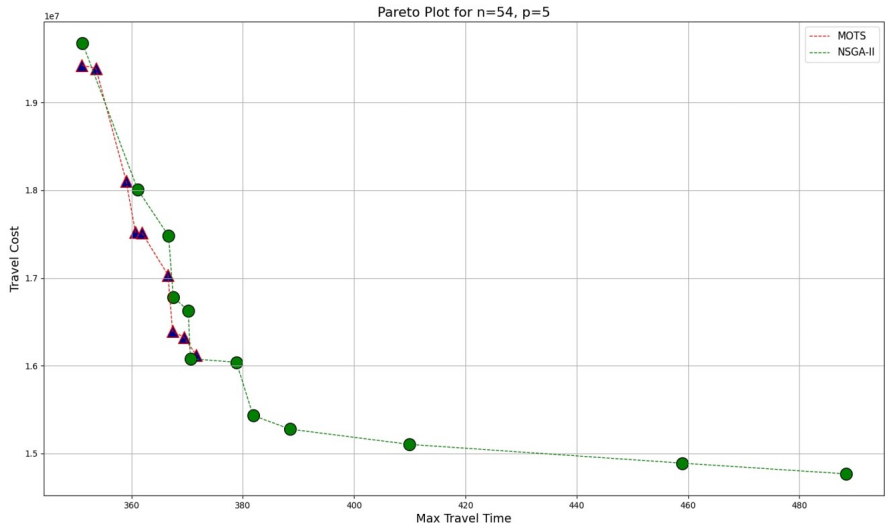

Figure 7. Pareto Fronts of the NSGA-II and MOTS for 54 nodes and 5 hubs

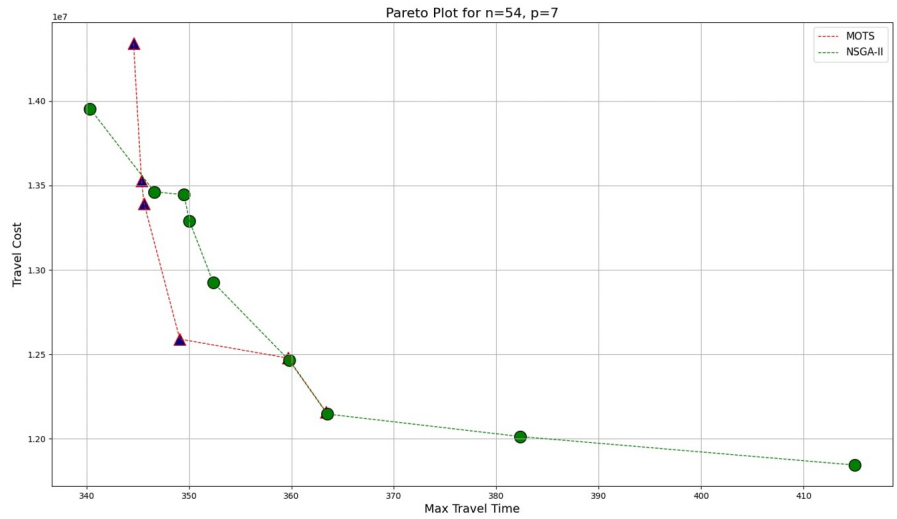

Figure 8. Pareto Fronts of the NSGA-II and MOTS for 54 nodes and 7 hubs

6.2. Performance Comparison

This section we perform the statistical comparison between 2 algorithms. The statistical results show in Table 2. Although NSGA-II obtains the lower mean of the first objective (total transportation cost) compare to MOTS, the statistics show not significantly different at the significance level of 0.05 with *p*-value 0.805. For the second objective (maximum travel time), MOTS obtains the lower mean compare to NSGA-II, however the statistics show not significantly different at the significance level of 0.05 with *p*-value 0.205. Therefore, the performance of algorithm in term of quality of solution of both algorithms are the same. For performance of computational time, the statistics show not significantly different at the significance level of 0.05 with *p*-value 0.212. However, the standard deviation of NSGA-II is very low, indicating that its time performance is more consistent than MOTS. In contrast, the run times of MOTS tend to increase as the number of hubs increases.

Metric	Algorithm	Mean	SD	t	p
Total transportation cost	MOTS	16,008,298.56	6,438,273.64	0.251	0.805
	NSGA-II	15,253,347.56	6,311,854.05		
Maximum travel time	MOTS	409.42	91.19	-1.321	0.205
	NSGA-II	460.58	72.05		
Run time	MOTS	178.02	188.66	-1.301	0.212
	NSGA-II	259.89	4.67		

Table 2. Statistical comparison of algorithms

Next, we analyze the Pareto Front generated by the MOTS and NSGA-II algorithms, comparing the diversity of solutions and the effectiveness in achieving optimal solutions. The deviation of solutions from both algorithms in each objective at different numbers of hubs are showed in Figures 9-10. There is a high diversity of solutions in the case of fewer number of hubs (*p*=2, 3, and 4) in both algorithms as indicated by the wide range of the boxplots. Although, MOTS has variety of solutions in the total cost objective (*p*=3), however, NSGA-II has more diversity of solutions in overall cases. So, in the case of solving multi-objective problems, the more number of optimal solutions means more alternatives to decision makers. Therefore, NSGA-II is able to provide more alternatives for decision makers.

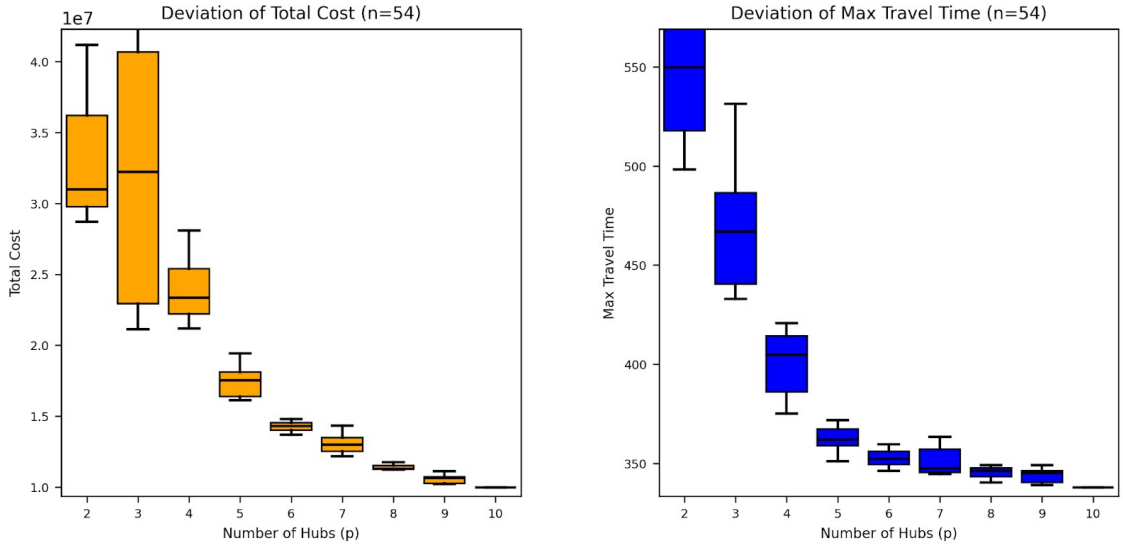

Figure 9. Deviation of solutions from MOTS at different numbers of hubs

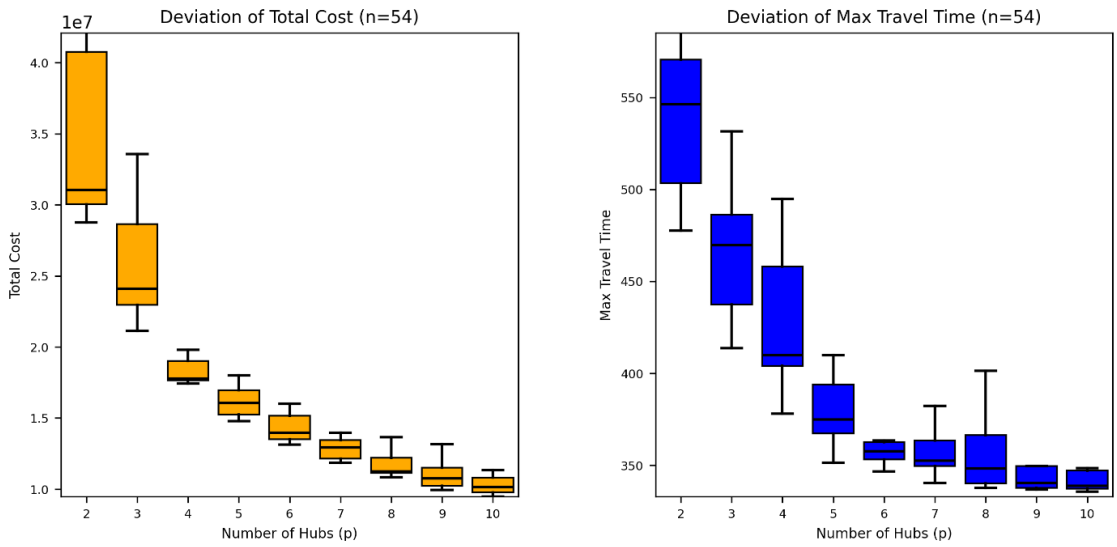

Figure 10. Deviation of solutions from NSGA-II at different numbers of hubs

7. Conclusion

This research presents a railway transportation network planning approach that addresses the selection of the most suitable location for a railway transportation hub, considering both cost and service. The problem is formulated as a multi-objective model with the objectives of minimizing total transportation cost and maximum travel time, and applied to a real-world case study in Thailand. Due to the size and complexity of the problem, we develop two metaheuristics algorithms; MOTS and NSGA-II, to solve the problem. Computational results indicate that both algorithms perform efficiently in terms of solution quality and computational time. MOTS tends to perform better by yielding lower non-dominated solutions for the first objective (cost), while NSGA-II performs better by yielding lower non-dominated solutions for the second objective (service). The findings support the effectiveness of MOTS (Ghaffarinasab, 2020; Ying & Junping, 2014) and NSGA-II (Demir et al., 2016; Karimi-Mamaghan et al., 2020) for solving multi-objective hub location problems, consistent with previous studies. However, at the 0.05 significance level, there is no statistically significant difference between MOTS and NSGA-II. Regarding computational time, both algorithms perform equally well with no significant difference, although NSGA-II tends to achieve higher solution diversity, as also reported in prior studies (Karimi-Mamaghan et al., 2020).

The proposed multi-objective model extends the classical single-objective hub location problem model that focuses only on minimizing transportation cost. Instead of providing a single optimal solution, it highlights the trade-off between cost and travel time, thus offering decision-makers more alternatives for railway hub network design. This contributes to both the construction phase of new rail networks and the improvement phase for redesigning or expanding existing hubs, with the potential to enhance efficiency, service quality, and passenger satisfaction. The decision-makers can balance cost efficiency and service performance, making the findings valuable for both policymakers and railway operators.

For future research, there are several directions. First, demand in this study is assumed to be deterministic; incorporating demand uncertainty or time-dependent passenger flows could make the model more realistic. Second, adding a third objective such as coverage or profit could further enrich the analysis. Finally, testing other advanced optimization methods or extending the framework to multimodal transportation systems could enhance both the robustness and applicability of the proposed approach.

Declaration of Conflicting Interests

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

This research budget was allocated by National Science, Research and Innovation Fund (NSRF), and King Mongkut's University of Technology North Bangkok (Project no. KMUTNB-FF-67-B-52).

References

Abyazi-Sani, R., & Ghanbari, R. (2016). An efficient tabu search for solving the uncapacitated single allocation hub location problem. *Computers & Industrial Engineering,* 93, 99-109. https://doi.org/10.1016/j.cie.2015.12.028

Attar, A., Irawan, C.A., Akbari, A.A., & Zhong, S. (2024). Multi-disruption resilient hub location-allocation network design for less-than-truckload logistics. *Transportation Research Part A: Policy and Practice,* 190, 104260. https://doi.org/10.1016/j.tra.2024.104260

Benaini, A., Berrajaa, A., Boukachour, J., & Oudani, M. (2019). Solving the Uncapacitated Single Allocation p-Hub Median Problem on GPU. *Studies in Computational Intelligence,* 774, 27-42. https://doi.org/10.1007/978-3-319-95104-1_2

Campbell, J.F. (1994). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research,* 72(2), 387-405. https://doi.org/10.1016/0377-2217(94)90318-2

Chobar, A.P., Adibi, M.A., & Kazemi, A. (2021). A novel multi-objective model for hub location problem considering dynamic demand and environmental issues. *Journal of Industrial Engineering and Management Studies,* 8(1), 1-31.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation,* 6(2), 182-197. https://doi.org/10.1109/4235.996017

Demir, I., Ergin, F.C., & Kiraz, B. (2016). A new model for the multi-objective multiple allocation hub network design and routing problem. *IEEE Access,* 7, 90678-90689. https://doi.org/10.1109/ACCESS.2019.2927418

Ernst, A.T., Hamacher, H.W., Jiang, H., Krishnamoorthy, M., & Woeginger, G. (2009). Uncapacitated single and multiple allocation p-hub center problems. *Computers & Operations Research,* 36(7), 2230-2241. https://doi.org/10.1016/j.cor.2008.08.021

Ernst, A.T., & Krishnamoorthy, M. (1996). Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science,* 4(3), 139-154. https://doi.org/10.1016/S0966-8349(96)00011-3

Gendreau, M., & Potvin, J. (2010). Handbook of Metaheuristics (3rd ed.). Springer. https://doi.org/10.1007/978-1-4419-1665-5

Ghaffarinasab, N. (2020). A tabu search heuristic for the bi-objective star hub location problem. *International Journal of Management Science and Engineering Management,* 15(3), 213-225. https://doi.org/10.1080/17509653.2019.1709992

Glover, F. (1989). Tabu Search-Part I. *ORSA Journal on Computing,* 1(3), 190-206. https://doi.org/10.1287/ijoc.1.3.190

Guan, J., Lin, G., & Feng, H. (2018). A learning-based probabilistic tabu search for the uncapacitated single allocation hub location problem. *Computers and Operations Research,* 98, 1-12. https://doi.org/10.1016/j.cor.2018.04.020

Jost, N., Grochala, A., Schumacher, C., Ammouriova, M., & Juan, A.A. (2023). Solving the Multi-Allocation p-Hub Median Problem with Stochastic Travel Times: A Simheuristic Approach. *Proceedings of 2023 the Winter Simulation Conference (WSC)* (1854-1863). San Antonio, TX, USA. https://doi.org/10.1109/WSC60868.2023.10407621

Kara, B.Y., & Tansel, B.C. (2000). On the single assignment p-hub center problem. *European Journal of Operational Research,* 125(3), 648-655. https://doi.org/10.1016/S0377-2217(99)00274-X

Karimi-Mamaghan M., Mohammadi, M. Pirayesh, A., Karimi-Mamaghan, A.M. & Irani, H. (2020). Hub-and-spoke network design under congestion: A learning based metaheuristic. *Transportation Research Part E: Logistics and Transportation Review,* 142, 102069. https://doi.org/10.1016/j.tre.2020.102069

Kratica, J., Stanimirovic, Z., Tosic, D., & Filipovic, E. (2007). Two genetic algorithms for solving the uncapacitated single allocation p-hub median problem. *European Journal of Operational Research,* 182(1), 15-28. https://doi.org/10.1016/j.ejor.2006.06.056

Li, C., Han, P., Zhou, M., & Gu, M. (2023). Design of multimodal hub-and-spoke transportation network for emergency relief under COVID-19 pandemic: A meta-heuristic approach. *Applied Soft Computing,* 133, 109925. https://doi.org/10.1016/j.asoc.2022.109925

Ogazon, E., Anaya-Arenas, A.M., & Ruiz, A. (2025). Designing hub-based regional transportation networks with service level constraints. *Transportation Research Part E,* 195, 103991. https://doi.org/10.1016/j.tre.2025.103991

O'Kelly, M.E. (1987). A quadratic integer programming for the location of interacting hub facilities. *European Journal of Operational Research,* 32(3), 393-404. https://doi.org/10.1016/S0377-2217(87)80007-3

Peker, M., Kara, B.Y., Campbell, J.F., & Alumur, S.A. (2016). Spatial analysis of single allocation hub location problems. *Networks and Spatial Economics,* 16, 1075-1101. https://doi.org/10.1007/s11067-015-9311-9

Silva, M.R., & Cunha, C.B. (2017). A tabu search heuristic for the uncapacitated single allocation p-hub maximal covering problem. *European Journal of Operational Research,* 262, 954-965. https://doi.org/10.1016/j.ejor.2017.03.066

Skorin-Kapov, D., & Skorin-Kapov, J. (1994). On tabu search for the location of interacting hub facilities. *European Journal of Operational Research,* 73(3), 502-509. https://doi.org/10.1016/0377-2217(94)90245-3

Soylu, B. & Katip, H. (2019). A multiobjective hub-airport location problem for an airline network. *European Journal of Operational Research,* 277, 412-425. https://doi.org/10.1016/j.ejor.2019.02.056

Topcuoglu, H., Ergin, F.C., Ermis, M., & Yilmaz, G. (2005). Solving the uncapacitated hub location problem using genetic algorithms. *Computers & Operations Research,* 32(4), 967-984. https://doi.org/10.1016/j.cor.2003.09.008

Wandelt, S., Wang, S., & Sun, X. (2025). A literature review on hub location-routing models and their solution techniques. *Computers and Operations Research,* 173, 106861. https://doi.org/10.1016/j.cor.2024.106861

Wang, C., Liu, Y., & Yang, G. (2023). Adaptive distributionally robust hub location and routing problem with a third-party logistics strategy. *Socio-Economic Planning Sciences,* 87, Part A, 101563. https://doi.org/10.1016/j.seps.2023.101563

Yaman, H. (2011). Allocation strategies in hub networks. *European Journal of Operational Research,* 211(3), 442-451. https://doi.org/10.1016/j.ejor.2011.01.014

Ying, Y. & Junping, X. (2014). Multi-objective hub location problem in hub-and-spoke network. *Computer Modelling & New Technologies,* 18(7), 309-316. Available at: http://www.cmnt.lv/upload-files/ns_39art47_CMNT1807-61_YING_Lu.pdf