Intelligent Automation & Soft Computing

Tech Science Press

DOI: 10.32604/iasc.2024.043854

ARTICLE

A Study on Optimizing the Double-Spine Type Flow Path Design for the Overhead Transportation System Using Tabu Search Algorithm

Nguyen Huu Loc Khuu[1,2,3], Thuy Duy Truong[1,2,3], Quoc Dien Le[1,2,3], Tran Thanh Cong Vu[1,2,3], Hoa Binh Tran[1,2,3] and Tuong Quan Vo[1,2,3,*]

[1]Department of Mechatronics, Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam

[2]Bach Khoa Research Center for Manufacturing Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam

[3]Vietnam National University Ho Chi Minh City, Ho Chi Minh City, Vietnam

*Corresponding Author: Tuong Quan Vo. Email: vtquan@hcmut.edu.vn

Received: 14 July 2023 Accepted: 05 February 2024 Published: 21 May 2024

ABSTRACT

Optimizing Flow Path Design (FPD) is a popular research area in transportation system design, but its application to Overhead Transportation Systems (OTSs) has been limited. This study focuses on optimizing a double-spine flow path design for OTSs with 10 stations by minimizing the total travel distance for both loaded and empty flows. We employ transportation methods, specifically the North-West Corner and Stepping-Stone methods, to determine empty vehicle travel flows. Additionally, the Tabu Search (TS) algorithm is applied to branch the 10 stations into two main layout branches. The results obtained from our proposed method demonstrate a reduction in the objective function value compared to the initial feasible solution. Furthermore, we explore how changes in the parameters of the TS algorithm affect the optimal result. We validate the feasibility of our approach by comparing it with relevant literature and conducting additional tests on layouts with 20 and 30 stations.

KEYWORDS

Overhead transportation systems; tabu search; double-spine layout; transportation method; empty travel; flow path design

1 Introduction

Vehicles or shuttles find common application across various industrial sectors for resolving internal transportation challenges. In the past, vehicle-based transportation was primarily associated with facility layouts. However, today, numerous research centers necessitate a zero-footprint approach to maintain clean rooms, and many facilities boast intricate machinery layouts that require an efficient materials handling system. Consequently, there is a growing need for OTSs to facilitate the movement of materials between stations. Numerous research efforts and systems pertain to OTSs, including Telelift's transport system, OTSs designed for semiconductor fabrication plants [1–5], multi-station, multi-container transport systems [6], and others. OTSs are designed to operate independently of fleet layout considerations [5], making them ideal for managing materials without disrupting floor activities. Furthermore, OTSs can alleviate staff shortages by automating material transport, allowing personnel to focus on their core tasks rather than manual material handling [7].

Although the transport system is known as OTSs, it can still be considered a familiar transportation system. Consequently, its design can be categorized as a type of Automated Guided Vehicle system (AGVs). Related research has identified several problems in designing transportation systems [8–10], including issues such as: (1) facility layout design (FLD); (2) estimating the number of vehicles required; (3) vehicle scheduling; (4) idle-vehicle positioning; (5) battery management; (6) vehicle routing; (7) deadlock resolution. Each of these issues plays a vital role in the design and control of the transportation system, with FLD being of strategic importance. FLD involves choosing the system's layout type, pick-up and drop-off points, and Flow Path Design (FPD) [10]. The performance of the transportation system is profoundly affected by FPD due to its significant influence on routing and dispatching rules. Conversely, a poorly designed flow path can lead to congestion and deadlock during deliveries [11].

The concept of Flow Path Design (FPD) was initially introduced in 1987 by Gaskin et al. [12]. This introduction sparked the development of numerous studies aimed at optimizing Flow Path Design (FPD). From related research, several basic features of how to optimize flow path design have emerged, including (1) layout; (2) flow path type; (3) optimization problems; (4) performance criteria; (5) input; and (6) methods to solve the objective function. The three most used layouts for designing flow paths are the conventional, single loop, and tandem layouts. Additionally, the rectilinear flow layout holds promise for future expansion of Overhead Transportation Systems (OTSs) [5], while the block layout format is currently under study [13–16]. Selecting the flow path type involves determining the direction for each path, which can be unidirectional, bidirectional, or multiple-lane flow path types. It is worth noting that bidirectional flow paths can reduce the total travel distance of vehicles; however, it is important to be aware that they may also lead to congestion and operational deadlock during transportation [8–10]. Most researchers construct objective functions to present the problem, aiming to optimize one or more criteria related to system operation. The most common method for modeling these objectives is mathematical programming. Performance criteria used for optimization include the total distance of both loaded and empty flows; the shortest path; the fixed cost such as path construction, space cost, and cost of control [17]; the total moving distance between the transfer points and workstations (tandem type); the total distance of congestion; and more. Optimization problems typically require two parameters: The facility layout and the from-to chart of material flows. Finally, methods for solving objective functions can be mathematical, simulation-based, or heuristic. The following tables (Tables 1 and 2) below provide summaries of studies employing various methods related to optimizing Facility Layout Design (FLD), in accordance with the basic features mentioned above.

Table 1: Differences between the related works about the layouts; the types of flow paths and the optimization problems

Authors	The layouts	The types of flow paths	The optimization problems
Suh et al. [4]	Rectilinear	Spine type	Mixed integer linear programming

(Continued)

Table 1 (continued)

Authors	The layouts	The types of flow paths	The optimization problems
Ting et al. [5]	Rectilinear	Spine type	Mixed integer linear programming
Gaskins et al. [12]	Conventional	Unidirectional	Zero-one integer linear programming
Nishi et al. [15]	Block	Unidirectional and bidirectional	Mixed integer linear programming
Hamzheei et al. [16]	Block	Bidirectional	Integer linear programming
Kim et al. [17]	Conventional	Unidirectional	Mixed integer linear programming
Guan et al. [18]	Conventional	Unidirectional	Zero-one integer linear programming
Kaspi et al. [19]	Conventional	Unidirectional	Zero-one integer linear programming
Kaspi et al. [20]	Conventional	Unidirectional	Mixed integer linear programming
Seo et al. [21]	Conventional	Unidirectional	Integer linear programming
Rubaszewski et al. [22]	Conventional	Unidirectional	Zero-one integer linear programming
Banerjee et al. [23]	Single loop	Unidirectional	Integer linear programming
Huang [24]	Tandem	Unidirectional and bidirectional	Integer linear programming
Akiyama et al. [25]	Block	Bidirectional	Mixed Integer linear programming
Rubaszewski et al. [26]	Conventional	Unidirectional	Integer linear programming
Xiao et al. [27]	Conventional	Unidirectional	Zero-one integer linear programming
Shanmugasundaram et al. [28]	Conventional	Unidirectional	Integer linear programming
Pourvaziri et al. [29]	Aisle structure	Bidirectional	Mixed integer linear programming

Table 2: Differences among related works pertain to performance criteria and solution techniques

Authors	The performance criteria	Solution technique
Suh et al. [4]	The total distance of congestion	The genetic algorithm (GA)
Ting et al. [5]	The total travel distance of the loaded vehicle	The haft-sum method and optimal allocation solution
Gaskins et al. [12]	The total travel distance of the loaded vehicle	Multipurpose Optimization Systems computer package
Nishi et al. [15]	The total of both fixed and travel costs	The cell-based local search heuristics
Hamzheei et al. [16]	The total length of the flow path	An ant colony-based algorithm
Kim et al. [17]	The total of both fixed and travel costs	The branch and bound procedure
Guan et al. [18]	The total travel distance of both loaded and empty vehicles	The discrete electromagnetism-like mechanism
Kaspi et al. [19]	The total travel distance of the loaded vehicles	The branch and bound procedure
Kaspi et al. [20]	The total travel distance of both loaded and empty vehicles	The branch and bound procedure
Seo et al. [21]	The total travel distance of the loaded vehicles	The Tabu search algorithm
Rubaszewski et al. [22]	The total travel distance of both loaded and empty vehicles	The genetic algorithm (GA)
Banerjee et al. [23]	The total travel distance of the loaded vehicles	The genetic algorithm (GA)
Huang [24]	The total moving distance between the transfer points and workstations	The solution algorithm of the minimal spanning tree [30]
Akiyama et al. [25]	The total of both fixed and travel costs	Sub-tour elimination method
Rubaszewski et al. [26]	The total travel distance of both loaded and empty vehicles	The heuristic methods such as Iterated Local Search (ILS); Muli-Start Local Search (MLS); Bee Algorithm (BA) and Ant Colony Optimization (ACO)
Xiao et al. [27]	The maximum completion time of all handling tasks (makespan)	A dual-population collaborative evolutionary genetic algorithm (CEGA)

(Continued)

Table 2 (continued)

Authors	The performance criteria	Solution technique
Shanmugasundaram et al. [28]	The total travel distance of both loaded and unloaded vehicles	The genetic algorithm (GA) and The branch and bound procedures
Pourvaziri et al. [29]	- The number, position, and width of the aisles - The position of the entrance and exit doors and how to connect them to the aisles	An improved branch and cut algorithm

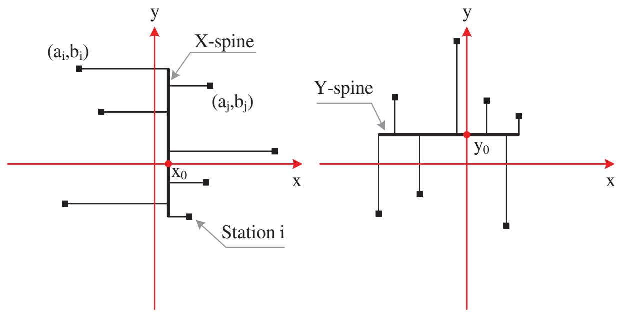
After conducting a literature review, it becomes evident that there is a wide range of methods available for optimizing Flow Path Design (FPD). Some commonly used methods to optimize FPD include exact mathematical analysis models, such as branch and bound, or simulation methods. However, it can be time-consuming when dealing with scaled-up problems with a large number of variables [15,18]. On the other hand, most related studies tend to focus solely on optimizing the loaded travel distance of vehicles, particularly for conventional types with layouts attached to the ground. However, there are certain critical issues that have not received as much research attention, namely, (1) optimizing the FPD for OTSs, and (2) minimizing the total travel distance of empty flows. Hence, this research aims to bridge this gap by proposing a metaheuristic method along with transportation methods to optimize the FPD for OTSs. The primary objective is to minimize the combined travel distance of both loaded and empty flows, ensuring efficient cargo delivery from pick-up stations to delivery stations. This system is designed to allow single-load transportation of vehicles along a designated flow path. For future expansion considerations, the double-spine layout type has been chosen. The proposed approach leverages the Tabu Search algorithm, the North-West Corner method, the Stepping-Stone Method, and the Rectilinear Minisum Location Problem as the appropriate tools to address these optimization problems.

The organization of this manuscript is as follows. The proposed problem is described in Section 2. In Section 3, the numerical analysis will be discussed. Then Section 4 will assess the results. The conclusion of the research will be discussed in Section 5.

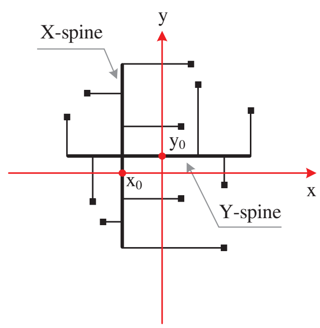
2 Problem Description

2.1 Optimize the Double-Spine Type Layout Design

The spine layout is commonly used for systems with future expansion needs, offering two main forms: Single-spine and double-spine. In the single-spine layout, there is one main branch (which can be either X-spine or Y-spine), with load ports or stations connecting to the spine as sub-branches. Similarly, the double-spine layout comprises two perpendicular single main branches (the X-spine and Y-spine). Each station will only connect to one of the two main spines (not both simultaneously). Figs. 1a and 1b below introduce the basic single-spine and double-spine layouts, respectively.



(a)



(b)

Figure 1: (a) The single-spine layout (X and Y types), (b) The double spine layout

The spine model is positioned in the layout referred to as the rectilinear layout. In this layout, the X-spine is positioned at coordinates of $x = x_0$, and the Y-spine is positioned at coordinates $y = y_0$. Each station is represented as a point with known coordinates, denoted as $P_i\,(a_i, b_i)$, as shown in Fig. 1a. The station positions in the coordinate system are illustrated in Fig. 2, which is employed in this research.

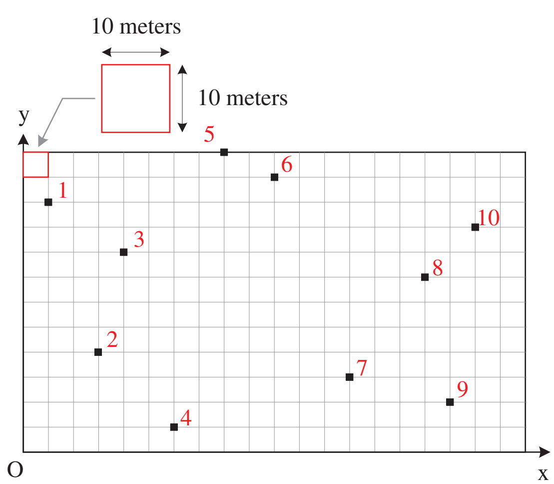


Figure 2: The 10 stations are arranged in a rectilinear layout

As mentioned earlier, each point is only connected to one main branch. Therefore, the stations are divided into two separate sets, respectively:

S_X: The set of points P_i associated with the X-spine.

S_Y: The set of points P_j associated with the Y-spine.

Based on Figs. 1a and 1b, we derive the general formula for the rectilinear distance between two points P_i and P_j as follows:

$d_{ij} = |a_i - x_0| + |b_i - b_j| + |x_0 - a_j|$: When P_i and P_j associated with the X-spine.

$d_{ij} = |b_i - y_0| + |a_i - a_j| + |y_0 - b_j|$: When P_i and P_j associated with the Y-spine.

$d_{ij} = |a_i - x_0| + |b_i - y_0| + |x_0 - a_j| + |b_j - y_0|$: When P_i associated with the X-spine and P_j associated with the Y-spine.

Denoting that the material flows from the P_i to P_j as f_{ij}. Because the empty travel flows (the travel that the vehicles do not deliver the loads) are considered, we denote e_{ij} to represent the empty travel from P_i to P_j. The transportation method can be applied to optimal the empty travel [31].

The integer programming model is employed to formulate the objective function for minimizing the total travel distance among the 10 stations. However, it is essential to divide the objective function into three parts: (1) the paths that include flows among the points in S_X; (2) the paths that encompass flows among the points in S_Y; and (3) the paths that involve flows among the points in both sets S_X and S_Y.

The objective function aims to minimize the total travel distance of both loaded and empty flows among the points in S_X:

$$\sum_{i=S_X}\sum_{j=S_X} f_{ij}d_{ij} + \sum_{i=S_X}\sum_{j=S_X} e_{ij}d_{ij} = \sum_{i=S_X}\sum_{j=S_X} \left(f_{ij} + e_{ij}\right)\left(|a_i - x_0| + |b_i - b_j| + |x_0 - a_j|\right) \tag{1}$$

The objective function aims to minimize the total travel distance of both loaded and empty flows among the points in S_Y:

$$\sum_{i=S_Y}\sum_{j=S_Y} f_{ij}d_{ij} + \sum_{i=S_Y}\sum_{j=S_Y} e_{ij}d_{ij} = \sum_{i=S_Y}\sum_{j=S_Y} \left(f_{ij} + e_{ij}\right)\left(|b_i - y_0| + |a_i - a_j| + |y_0 - b_j|\right) \tag{2}$$

The objective function aims to minimize the total travel distance of both loaded and empty flows among the points from two sets S_X and S_Y:

$$\begin{aligned}
&\sum_{i=S_X}\sum_{j=S_Y} f_{ij}d_{ij} + \sum_{i=S_Y}\sum_{j=S_X} f_{ij}d_{ij} + \sum_{i=S_X}\sum_{j=S_Y} e_{ij}d_{ij} + \sum_{i=S_Y}\sum_{j=S_X} e_{ij}d_{ij} \\
&= \sum_{i=S_X}\sum_{j=S_Y} \left(f_{ij} + e_{ij}\right)\left(|a_i - x_0| + |b_i - y_0| + |x_0 - a_j| + |b_j - y_0|\right) \\
&\quad + \sum_{i=S_Y}\sum_{j=S_X} \left(f_{ij} + e_{ij}\right)\left(|a_i - x_0| + |b_i - y_0| + |x_0 - a_j| + |b_j - y_0|\right)
\end{aligned} \tag{3}$$

According to Ting et al. [5] and Tompkins et al. [32], the optimization of x_0 and y_0, as well as the branching of stations into two sets S_X and S_Y can be solved separately. Consequently, three distinct subproblems need to be addressed: Optimizing the x_0 and y_0 values, and branching the stations.

Subproblem 1: Optimize x_0—This involves considering the flow between the points in S_X as Eq. (1) and the flow among the points from both sets S_X and S_Y as Eq. (3):

$$
\begin{aligned}
Z_{x_0} &= \sum_{i=S_X}\sum_{j=S_X}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+\left|x_0-a_j\right|\right)\\
&\quad+\sum_{i=S_X}\sum_{j=S_Y}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+\left|x_0-a_j\right|\right)\\
&\quad+\sum_{i=S_Y}\sum_{j=S_X}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+\left|x_0-a_j\right|\right)\\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+\left|x_0-a_j\right|\right)\\
&= \sum_{i=1}^{n}(w_i+z_i)\,|a_i-x_0| \qquad (4)
\end{aligned}
$$

Subproblem 2: Optimize y_0—This involves considering the flow between the points in S_Y as Eq. (2) and the flow among the points from both sets S_X and S_Y as Eq. (3):

$$
\begin{aligned}
Z_{y_0} &= \sum_{i=S_Y}\sum_{j=S_Y}\left(f_{ij}+e_{ij}\right)\left(|b_i-y_0|+\left|y_0-b_j\right|\right)\\
&\quad+\sum_{i=S_X}\sum_{j=S_Y}\left(f_{ij}+e_{ij}\right)\left(|b_i-y_0|+\left|y_0-b_j\right|\right)\\
&\quad+\sum_{i=S_Y}\sum_{j=S_X}\left(f_{ij}+e_{ij}\right)\left(|b_i-y_0|+\left|y_0-b_j\right|\right)\\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\left(f_{ij}+e_{ij}\right)\left(|b_i-y_0|+\left|y_0-b_j\right|\right)\\
&= \sum_{i=1}^{n}(w_i+z_i)\,|b_i-y_0| \qquad (5)
\end{aligned}
$$

Subproblem 3: Branching the stations–This subproblem involves considering all the flows in Eqs. (1)–(3):

$$
\begin{aligned}
Z_{x_0y_0} &= \sum_{i=S_X}\sum_{j=S_X}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+\left|b_i-b_j\right|+\left|x_0-a_j\right|\right)\\
&\quad+\sum_{i=S_Y}\sum_{j=S_Y}\left(f_{ij}+e_{ij}\right)\left(|b_i-y_0|+\left|a_i-a_j\right|+\left|y_0-b_j\right|\right)\\
&\quad+\sum_{i=S_X}\sum_{j=S_Y}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+|b_i-y_0|+\left|x_0-a_j\right|+\left|b_j-y_0\right|\right)\\
&\quad+\sum_{i=S_Y}\sum_{j=S_X}\left(f_{ij}+e_{ij}\right)\left(|a_i-x_0|+|b_i-y_0|+\left|x_0-a_j\right|+\left|b_j-y_0\right|\right) \qquad (6)
\end{aligned}
$$

As mentioned earlier, the objective functions represented by Eqs. (4) and (5) can be solved independently. The Rectilinear Minisum Location Problem [32] can be applied to optimize these two values, x_0 and y_0. It is important to note that w_i represents the total loaded travel flows to and from station i, and z_i represents the total empty travel flows to and from station i. Additionally, because $|a_i - a_j|$ and $|b_i - b_j|$ are constant, they can be removed from Eqs. (4) and (5) during the analysis.

2.2 The Tabu Search (TS) Algorithm and the Transportation Methods

The TS algorithm is a metaheuristic method that relies on an initial feasible solution. Once the initial feasible solution is obtained, a search algorithm is employed to identify the admissible neighborhood and replace the current solution with the best-improving neighborhood. Notably, one unique feature of TS is its ability to accept non-improving moves [33]. In this study, the TS algorithm will be characterized by two factors: Short-term memory and long-term memory. The TS algorithm is applied to handle the objective function represented by Eq. (6).

The function represented by Eq. (6) is converted into a binary integer programming form, as demonstrated in Eq. (7). In Eq. (7), d_{ij}^x and d_{ij}^y denote the distance between two stations intra X-spine and Y-spine, while d_{ij}^z and d_{ij}^w represent the distance between two stations inter X-spine and Y-spine. The formulas for d_{ij}^x, d_{ij}^y, d_{ij}^z, d_{ij}^w are provided in Eqs. (8) to (11).

$$Z = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(f_{ij} + e_{ij}\right) \left(d_{ij}^x + d_{ij}^y + d_{ij}^z + d_{ij}^w\right) \tag{7}$$

Two decision variables x_i and y_i, where $i \in \{1, 2, 3, \ldots, 10\}$, correspond to the branches that connect to X-spine and Y-spine, respectively. When $x_i = 1$, it signifies a branch from station i that links with the X-spine, and when $y_i = 1$, it indicates a branch from station i that connects to the Y-spine, and vice versa. To ensure that each station is exclusively associated with either the X-spine or Y-spine, a constraint is necessary, as depicted in Eq. (12).

Thus:

$$d_{ij}^x = \left(x_i + x_j - 1\right) \left(|a_i - x_0| + \left|b_i - b_j\right| + \left|x_0 - a_j\right|\right) \geq 0 \tag{8}$$

$$d_{ij}^y = \left(y_i + y_j - 1\right) \left(|b_i - y_0| + \left|a_i - a_j\right| + \left|y_0 - b_j\right|\right) \geq 0 \tag{9}$$

$$d_{ij}^z = \left(x_i + y_j - 1\right) \left(|a_i - x_0| + |b_i - y_0| + \left|x_0 - a_j\right| + \left|b_j - y_0\right|\right) \geq 0 \tag{10}$$

$$d_{ij}^w = \left(y_i + x_j - 1\right) \left(|a_i - x_0| + |b_i - y_0| + \left|x_0 - a_j\right| + \left|b_j - y_0\right|\right) \geq 0 \tag{11}$$

$$x_i + y_i = 1 \tag{12}$$

To execute the TS, an initial solution that represents the existence of associated branches with a binary string C must be determined. In this binary string, the bit 0 or 1 signifies the presence of branches from stations to either the X-spine or Y-spine. A value of 1 indicates that the station is connected to the X-spine, corresponding to $x_i = 1$ and $y_i = 0$. Conversely, a value of 0 denotes that the station is linked to the Y-spine, resulting in $x_i = 0$ and $y_i = 1$.

Assuming the initial possible solution as the current and the best solution to the problem, the next stage of TS will involve searching for neighbors by making changes bit-by-bit to the binary string C to achieve admissible move. In TS, "move" refers to transitioning from the current solution to the best-improved solution. For all admissible neighbors, the total travel distance is computed using Eq. (7) to identify the best-improving move, which is then executed a move.

Since this research considers empty travel flows, it is necessary to determine the empty travel matrix. According to Muckstadt and Maxwell [34], the net vehicle flow of each station, denoted as $NF(i)$, is defined as $NF(i) = \sum_j f_{ji} - \sum_j f_{ij}$. Here, $\sum_j f_{ji}$ represents the total travel distance of loaded flows to station i, and $\sum_j f_{ij}$ represents the total travel distance of loaded flows from station i. On the other hand, the supply constraint for each delivery station, denoted as S_m, and the demand constraint for each pick-up station, denoted as R_l, are defined in Eqs. (13) and (14) based on the net flow equation. In this study, the transportation method is utilized to optimize the empty travel matrix for this system.

$$S_m = \begin{cases} \sum_l f_{lm} - \sum_l f_{ml} & \sum_l f_{lm} - \sum_l f_{ml} > 0 \\ 0 & \sum_l f_{lm} - \sum_l f_{ml} \leq 0 \end{cases} \tag{13}$$

$$R_l = \begin{cases} \sum_m f_{lm} - \sum_m f_{ml} & \sum_m f_{lm} - \sum_m f_{ml} > 0 \\ 0 & \sum_m f_{lm} - \sum_m f_{ml} \leq 0 \end{cases} \tag{14}$$

The variable e_{ij} denotes the quantity of empty travel flows of the vehicle from station i to station j. According to Muckstadt et al. [34], if the net vehicle flow $NF(i)$ is non-negative, then $\sum_j e_{ij} = NF(i)$ and $\sum_j e_{ji} = 0$. Conversely, if the net vehicle flow $NF(i)$ is negative, then $\sum_j e_{ji} = |NF(i)|$ and $\sum_j e_{ij} = 0$. In these equations, $\sum_j e_{ij}$ is the total of empty travel flows from node i, and $\sum_j e_{ji}$ is the total of empty travel flows to node i.

The transportation method requires a distance matrix among the 10 stations and the net vehicle flow for each station. Using this distance matrix and the net vehicle flows, the matrix of empty travel flows can be calculated using transportation methods. Two methods are applied, namely the North-West Corner method and the Stepping-Stone Method [31]. The North-West Corner method can be summarized as follows:

Step 1: At the position $(1, 1)$—the North-West Corner, allocate as much as possible the flow value. It depends on the supply constraint—S_m, and the demand constraint—R_l. The assignment of supply and demand constraints of each station is $S_m^{(1)}$ to $S_m^{(n)}$, and $R_l^{(1)}$ to $R_l^{(n)}$.

Step 2: There are 3 cases.

Step 2.a: If $S_m^{(1)} < R_l^{(1)}$, the value at position $(1, 1)$ is set to $S_m^{(1)}$. Consequently, column 1 is disregarded. Meanwhile, $R_l^{(1)}$ becomes $\left(R_l^{(1)} - S_m^{(1)}\right)$, and position $(1, 2)$ is taken into consideration.

Step 2.b: If $S_m^{(1)} > R_l^{(1)}$, the value at position $(1, 1)$ is set to $R_l^{(1)}$. Consequently, the row 1 is disregarded. Meanwhile, $S_m^{(1)}$ becomes $\left(S_m^{(1)} - R_l^{(1)}\right)$, and position $(2, 1)$ is taken into consideration.

Step 2.c: If $S_m^{(1)} = R_l^{(1)}$, the value at position $(1, 1)$ is set to both $S_m^{(1)}$ and $R_l^{(1)}$. Consequently, both column 1 and row 1 are disregarded. Next, position $(2, 2)$ is taken into consideration.

Step 3: Continue this process until reaching the position (n, n), which is the South-East Corner.

The initial feasible matrix of empty travel flows can be obtained using the North-West Corner method. Subsequently, by applying the Stepping-Stone Method, the matrix of empty travel flows with the minimum total travel distance can be determined. The conditions for using this method are: (1)

the number of allocations in the initial feasible solution is $2n + 1$, where n is the number of stations; and (2) the allocations are independent. The Stepping-Stone Method can be summarized as follows:

Step 1: Define the initial feasible matrix of empty travel flows using the North-West Corner method.

Step 2: Examine the conditions mentioned above for the initial feasible matrix. Following that, evaluate all unoccupied positions and replace them with other occupied positions to achieve a more optimal solution. The replacement process can be handled as follows:

Step 2.1: Select an unoccupied position.

Step 2.2: Determine a closed loop through at least 3 occupied positions starting from this chosen position. The direction of the closed loop is not significant as it yields the same solution.

Step 2.3: At each corner of the closed loop, assign either a (+) or (–) sign, beginning with a (+) for the unoccupied position.

Step 2.4: Each corner of the closed loop corresponds to a cost (distance between the 2 stations). Calculate the net changes in cost for the closed loop using the assigned signs at each corner. The net change in cost is given by $c_1^* - c_2 + c_3 - c_4$, where c_1^* is the cost of the unoccupied position.

Step 2.5: Repeat steps 2.1 to 2.4 for all unoccupied positions.

Step 3: The matrix of empty travel is considered optimal if all net changes in cost are greater than or equal to zero, and the process concludes at this point. Otherwise, proceed to step 4.

Step 4: If there is more than one net change in cost that is negative, select the closed loop with the most negative value. If multiple net changes in costs have the same negative value, choose the closed loop that allows for more flow transition at the minimum cost.

Step 5: Within the chosen closed loop, replace the flow of an unoccupied position with a value from an occupied position, maximizing this replacement as much as possible.

Step 6: Return to step 2 and continue the process until all net change values are greater than or equal to zero.

As mentioned earlier, we will utilize two fundamental aspects of the TS: Short-term memory and long-term memory. In this research, static memory with a fixed Tabu tenure T is employed for short-term memory. This implies that if station a has a value b (0 or 1) denoted as $[a, b]$ since the latest move, then after the next T iteration, the value $[a, b]$ will not be accepted. Short-term memory is thus utilized to prevent the solution from getting stuck in local optima and encourages exploration [33]. On the other hand, long-term memory involves counting the frequency of repetition of $[a, b]$, denoted as $F[a, b]$. This process of tracking the entire search is known as long-term memory. In the case of non-improving moves, the values of the objective function Z will increase proportionally with the repetition frequency $F[a, b]$ and a penalty factor ρ. This reduces the weight given to checking non-improving moves in subsequent iterations, promoting diversity in the TS algorithm [33]. The iterative process continues until the stopping conditions are met. As suggested by Seo et al. [21], these stopping conditions can include reaching the maximum number of iterations or failing to improve the solution beyond a certain threshold within a set number of iterations. These two criteria are also applied in this research. Additionally, an aspiration criterion will be used to accept neighbors in the Tabu list if the solution is better than the current best solution [33].

The TS algorithm applied for optimizing the FPD can be illustrated as follows:

Step 1: Find an initial possible solution and assign it to both the current solution and the current best solution, denoting $C_{best} = C_{cur} = C_{init}$. Calculate Z_{init} based on the value of C_{init}, then assign it to Z_{cur} and Z_{best}. Begin the process with an iteration count of 0, denoted as $n_{iteration} = 0$ and $n_{nonimpr} = 0$.

Step 2: Change bit-by-bit of C_{cur} to generate neighbor solutions. Evaluate all neighbors using short-term and long-term memory to determine admissible neighbors:

Step 2.1: Calculate $Z_{neighbor}$ for each neighbor.

Step 2.2: If the number of iterations since the latest move to the neighbor is less than the Tabu tenure value T, and the aspiration criterion is not met ($Z_{neighbor} > Z_{best}$), reject the neighbor and return to the beginning of step 2. Otherwise, include the neighbor in the set of admissible neighbors and proceed to step 2.3. This step utilizes short-term memory.

Step 2.3: If the $Z_{neighbor} - Z_{best} \geq 0$, set $Z_{neighbor} = Z_{neighbor} + \rho.F[a, b]$. This means that for non-improving moves, the value of the objective function Z will increase proportionally with the repetition frequency of $[a, b]$ and the penalty factor ρ. This step involves long-term memory.

Step 3: Among the set of admissible neighbors, select the neighbor with the minimum objective function Z as the current solution. Update $C_{cur} = C_{bestimproveneighbor}$ and $Z_{cur} = Z_{bestimproveneighbor}$. Also, update $F[a, b] = F[a, b] + 1$ accordingly. If $Z_{bestimproveneighbor} < Z_{best}$, assign $Z_{best} = Z_{bestimproveneighbor}$ and $C_{best} = C_{bestimproveneighbor}$.

Step 4: If $n_{nonimpr} \leq 0$ and $Z_{bestimproveneighbor} - Z_{cur} < 0$, the move is considered an improving moves, and $n_{nonimpr} = n_{nonimpr} - 1$. Else if $n_{nonimpr} \geq 0$ and the $Z_{bestimproveneighbor} - Z_{cur} > 0$, the move is a non-improving move, and $n_{nonimpr} = n_{nonimpr} + 1$. These actions account for consecutive improving and non-improving moves. If the move is neither consecutive improving nor consecutive non-improving, set $n_{nonimpr} = 0$.

Step 5–Checking the stop condition: If $n_{iteration} \geq N_{iteration}$ or $n_{nonimpr} \geq N_{nonimpr}$, terminate the search process. Otherwise, set $n_{iteration} = n_{iteration} + 1$ and return to step 2.

3 Calculation and Numerical Analysis for Optimizing the Double-Spine Type Flow Path Design

The from-to chart of material flows matrix is presented in Table 3.

Table 3: The from-to chart of material flows of the layout

Station	1	2	3	4	5	6	7	8	9	10	*To*
1		20	10	30		45	25		30	10	
2	10		20	15	10			5	10		
3	30	10		10	25	30	5			5	
4	25	10	5			10	15	25	30		
5	15	10		5		20		15	5	10	
6	20		15	40	30		15		10		
7		25	5	10	20			15	30	15	
8	15	20	25	30		40	5		20		
9	10	30	10		15	35		5		20	
10		5	30	25	10	5		40	15		
From											

The coordinates of the 10 stations, denoted as (a_i, b_i), are listed in Table 4. It is important to note that in Fig. 2, each unit of length corresponds to 10 meters, simplifying calculations by using this unit.

Table 4: The coordinates of each station in the layout

Stations	1	2	3	4	5	6	7	8	9	10
a_i	1	3	4	6	8	10	13	16	17	18
b_i	10	4	8	1	12	11	3	7	2	9

Utilizing the data from Table 3, the net vehicle flow matrix is computed based on the vehicle flows traveling to and from station i, as shown in Table 5.

Table 5 : The net flow matrix

Stations	1	2	3	4	5	6	7	8	9	10	Total
Total TO	125	130	120	165	110	185	65	105	150	60	1215
Total FROM	170	70	115	120	80	130	120	155	125	130	1215
Net Flow	−45	60	5	45	30	55	−55	−50	25	−70	0

Derived from Table 5, the values for S_m and R_l are determined for each station, as depicted in Table 6.

Table 6: The supply and demand constraints of each station

Stations	1	2	3	4	5	6	7	8	9	10	Total
S_m	0	60	5	45	30	55	0	0	25	0	220
R_l	45	0	0	0	0	0	55	50	0	70	220

The two objective functions presented in Eqs. (4) and (5) can be addressed using "The Rectilinear Minisum Location Problem". This method considers the total weight of both loaded and empty travel flows. The total weight is calculated as the sum of the flows to and from each station. Consequently, Table 7 has been generated, which illustrates the total weights of loaded vehicles and the total weight of empty vehicles at each station, based on the data from Tables 5 and 6.

Table 7: The total weight of loaded flows and the total weight of empty flows at each station

Stations	1	2	3	4	5	6	7	8	9	10
$\sum_j f_{ij}$	170	70	115	120	80	130	120	155	125	130
$\sum_j e_{ij}$	0	60	5	45	30	55	0	0	25	0
$\sum_j f_{ij} + \sum_j e_{ij}$	170	130	120	165	110	185	120	155	150	130

(Continued)

Table 7 (continued)

Stations	1	2	3	4	5	6	7	8	9	10
$\sum_j f_{ji}$	125	130	120	165	110	185	65	105	150	60
$\sum_j e_{ji}$	45	0	0	0	0	0	55	50	0	70
$\sum_j f_{ji} + \sum_j e_{ji}$	170	130	120	165	110	185	120	155	150	130
Total weight	340	260	240	330	220	370	240	310	300	260

According to Tompkins et al. [32], for functions like Eqs. (4) and (5) with piecewise linear structures, the optimal value of x_0 should be such that no more than half of the total weight lies to the left of x_0, and no more than half of the total weight lies to the right of x_0. Similarly, the optimal value of y_0 should be such that no more than half of the total weight is above y_0, and no more than half of the total weight is below y_0. Furthermore, each optimal solution corresponds to the coordinates of one station among the 10 stations. In other words, x_0 should be equal to one of the a_i values, and y_0 should be equal to one of the b_i values.

The following tables, Tables 8 and 9, provide the calculations for determining the optimal x_0 coordinate and the optimum y_0 coordinate, respectively.

Table 8: Solving the optimal $\boldsymbol{x_0}$ coordinate

Stations (i)	Coordinate (a_i)	Total weight (W_i)	$\sum_{i=1}^{n} W_i$
1	1	340	340
3	3	240	580
2	4	260	840
4	6	330	1170
5	8	220	$1390 < 1435$
6	10	370	$1760 > 1435$
7	13	240	240
8	16	310	2310
9	17	300	2610
10	18	260	2870

Table 9: Solving the optimal $\boldsymbol{y_0}$ coordinate

Stations (i)	Coordinate (b_i)	Total weight (W_i)	$\sum_{i=1}^{n} W_i$
4	1	330	330
9	2	300	630

(Continued)

Table 9 (continued)

Stations (i)	Coordinate (b_i)	Total weight (W_i)	$\sum_{i=1}^{n} W_i$
7	3	240	870
2	4	260	1130 < 1435
8	7	310	1440 > 1435
3	8	240	1680
10	9	260	1940
1	10	340	2280
6	11	370	2650
5	12	220	2870

Table 8 displays the sum of the weights as 2870 and half of this sum as 1435. Based on these values, the optimal x_0 coordinate is determined to be a_6, which corresponds to station 6. Similarly, Table 9 reveals that the optimal y_0 coordinate is b_8, representing station 8. This configuration results in the placement of the X-spine at station 6 and the Y-spine at station 8, as illustrated in Fig. 3.

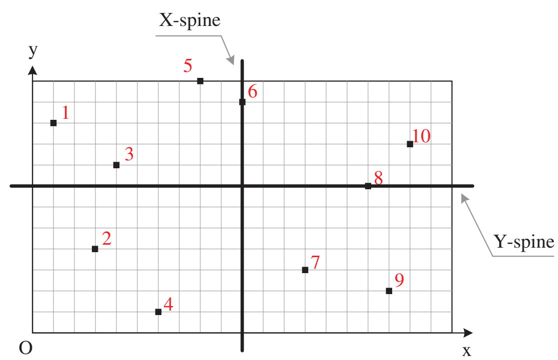


Figure 3: The two optimal spines in the rectilinear layout

Once the coordinates of the two spines, x_0 and y_0, have been determined, the next step is to branch the 8 stations to these two spines in a way that each station is connected to only one spine. To achieve this, an initial feasible solution is selected for the TS algorithm. The initial feasible solution is presented as $C_{init} = \overline{x_{10}x_9x_8x_7x_6x_5x_4x_3x_2x_1}$. The TS algorithm parameters include a Tabu tenure $T = 5$, a penalty factor $\rho = 50$, a maximum iteration count $N_{iteration} = 100$, and a maximum number of consecutive non-improving moves $N_{nonimpr} = 10$. These parameters are determined through a trial-and-error process. A MATLAB program is used to find the optimal solution of the objective function Eq. (7) using the TS algorithm. Assure that, the initial feasible solution is set to $C_{init} = 1101111111$. With the C_{init} established, the distance matrix among the 10 stations is calculated, as shown in Table 10.

Using the transportation method, the matrix of empty travel flows is optimized based on the matrix of loaded travel flows and the distance matrix among the 10 stations. As a result, Table 11 illustrates the empty travel flows among the 10 stations for the initial feasible solution C_{init}.

Table 10: The distance matrix among 10 stations based on $\boldsymbol{C}_{init}$

Station	1	2	3	4	5	6	7	8	9	10
1	0	22	17	22	13	10	19	18	24	18
2	22	0	17	14	17	1	11	16	16	20
3	17	17	0	17	12	9	14	13	19	15
4	22	14	17	0	17	14	9	16	12	20
5	13	17	12	17	0	3	14	13	19	13
6	10	14	9	14	3	0	11	10	16	10
7	19	11	14	9	14	11	0	13	11	17
8	18	16	13	16	13	10	13	0	18	16
9	24	16	19	12	19	16	11	18	0	22
10	18	20	15	20	13	10	17	16	22	0

Table 11: The matrix of empty travel among 10 stations of $\boldsymbol{C}_{init}$

Station	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	35	0	25
3	0	0	0	0	0	0	0	0	0	5
4	0	0	0	0	0	0	45	0	0	0
5	30	0	0	0	0	0	0	0	0	0
6	15	0	0	0	0	0	0	0	0	40
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	10	15	0	0
10	0	0	0	0	0	0	0	0	0	0

After 33 iterations of the TS algorithm, the optimal solution has been obtained with $C_{best} = 0000110000$ and $Z_{best} = 19360$. The empty travel flows for the optimal solution are shown in Table 12. When compared to the initial feasible solution, the optimal solution, which minimizes the objective function Eq. (7), represents as an improvement $\Delta_Z = 1590$. The flow path layout for $C_{best} = 0000110000$ is depicted in Fig. 4.

Table 12: The matrix of empty travel of the vehicles based on C_{best}

Station	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	45	0	0	0	0	0	15	0	0	0
3	0	0	0	0	0	0	5	0	0	0
4	0	0	0	0	0	0	35	10	0	0
5	0	0	0	0	0	0	0	30	0	0
6	0	0	0	0	0	0	0	10	0	45
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	25
10	0	0	0	0	0	0	0	0	0	0

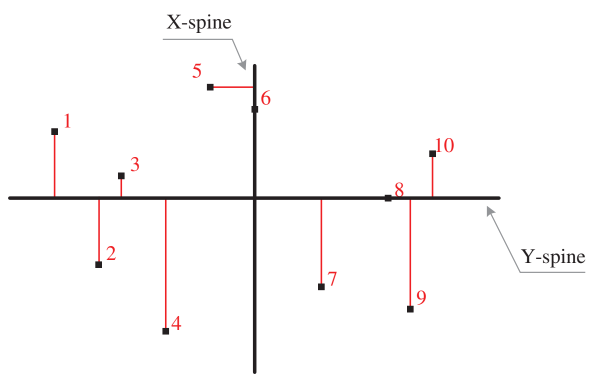


Figure 4: The optimal flow path after 33 iteration times

4 Evaluate the Numerical Analysis Results

The results obtained from the proposed method indicate a reduction in the objective function value when compared to the initial feasible solution. This reduction amounts to $\Delta_Z = 1590$, which corresponds to a distance reduction of $S_{impr} = 15900(m)$. Fig. 5 below illustrates the graph depicting the objective function values across 100 iteration times. At the 33^{th} iteration time, the value of the best objective function remains unchanged. Therefore, the optimal solution, $C_{best} = 0000110000$, is achieved at the 33^{th} iteration time, with an objective function value of $Z_{best} = 19360$.

The number of iterations required to obtain the optimal solution can vary depending on the Tabu tenure and penalty factor used in the TS algorithm. In Fig. 6, it is shown that with $T = 4$, the optimal solution is reached after 35 iteration times. Conversely, with $T = 12$, it takes 111 iterations to reach the optimal solution. Similarly, with $T = 13$, it takes 225 iterations, and with $T = 14$, it still takes after 225 iterations to reach the optimal solution. In Fig. 7, the impact of the penalty factor on the number of iterations is demonstrated. When $\rho = 100$, the optimal solution is achieved after 29 iterations. With $\rho = 200$, it takes 19 iterations to reach the optimal solution. When $\rho = 300$, it takes 23 iterations, and with $\rho = 400$, it still takes 23 iterations to reach the optimal solution. Since the number of iterations directly affects the running time of the search process, it is important to carefully consider the choice of Tabu tenure and penalty factor when applying the TS algorithm. Finding an appropriate balance between these parameters is crucial for efficient optimization.

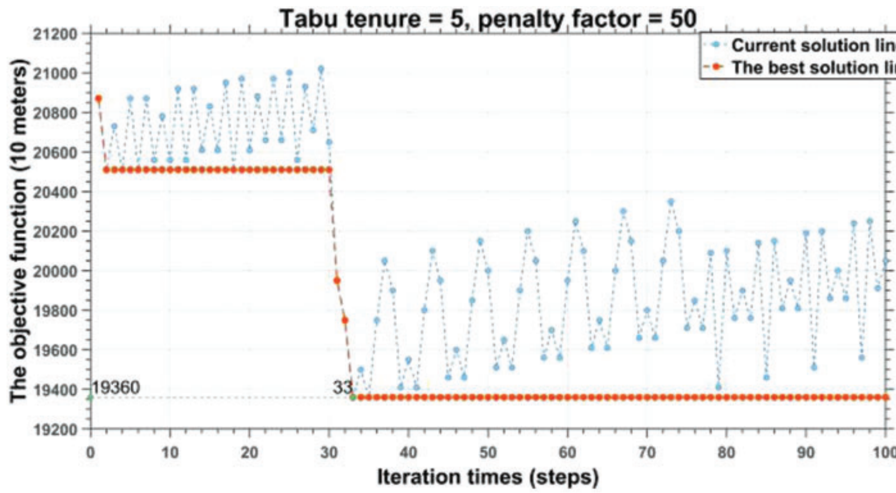


Figure 5: The optimal solution by using the TS algorithm

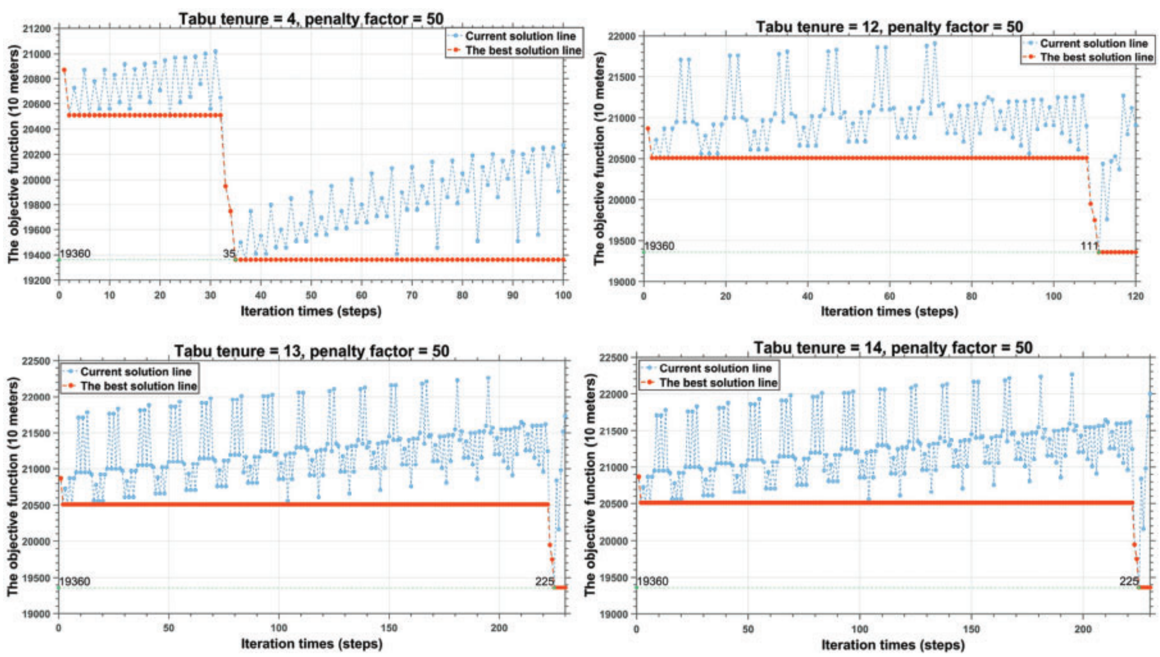


Figure 6: The optimal solutions when changing the Tabu tenure

To assess the feasibility of the proposed metaheuristic method, it was applied to the layout with 10 stations, whose research was conducted by Ting et al. [5], without considering the empty travel flows. The algorithm was configured with the following parameters: Tabu tenure $T = 5$, penalty factor $\rho = 5000$, maximum iteration $N_{iteration} = 100$, and maximum consecutive non-improving move $N_{nonimpr} = 100$. The initial feasible solution was set as $C_{init} = 1101111111$ and the initial objective function value was $Z_{init} = 3825367$. After 10 iteration times, the result is $C_{best} = 0000100000$ and $Z_{best} = 3039339$. The result means that all 10 stations only connect with the Y-spine, and the Y-spine is located at station 8. This result is consistent with the findings of the research conducted by Ting et al. [5], demonstrating the appropriateness of the proposed method for optimizing spine flow path design. The result is shown in Fig. 8.

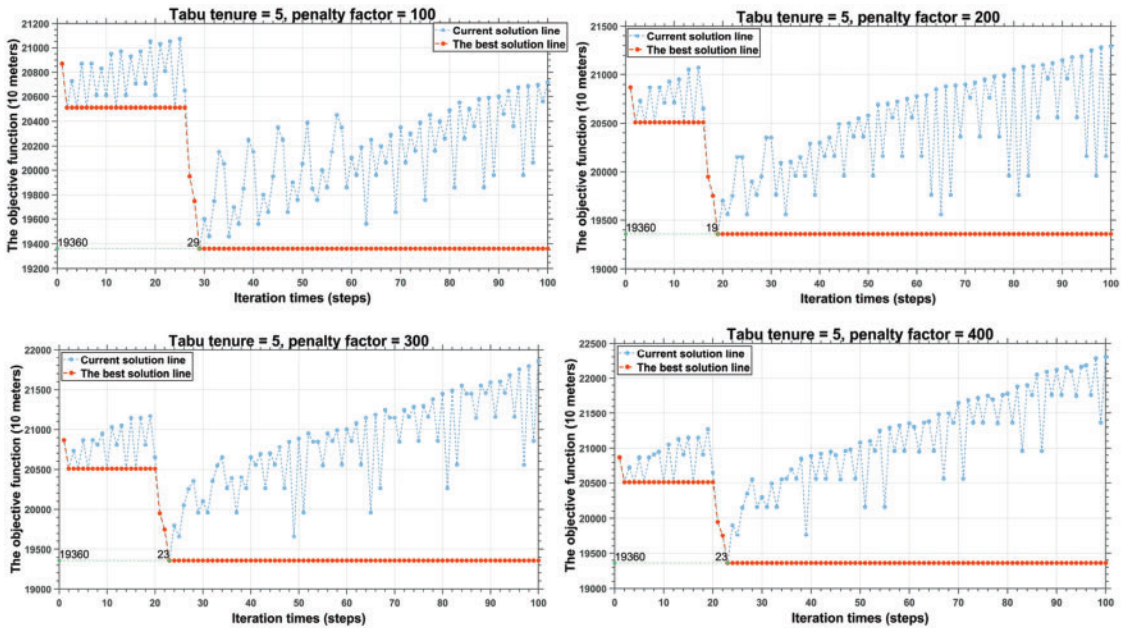


Figure 7: The optimal solutions when changing the penalty factor

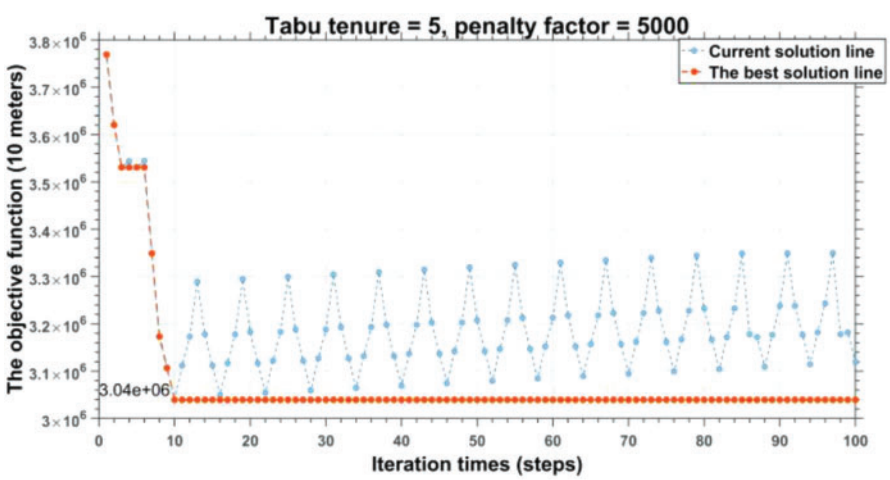


Figure 8: The optimal solution of the example in research [5]

In addition, two examples of layouts with 20 stations and 30 stations were generated using our proposed method. First, with the 20-station layout, the algorithm was configured with the following parameters: Tabu tenure $T = 5$, penalty factor $\rho = 1000$, the maximum iteration $N_{iteration} = 100$, and the maximum of the consecutive non-improving move $N_{nonimpr} = 100$. The optimal solution, $C_{best} =$

01111011110001110101, with an objective function value of $Z_{best} = 645908$, was achieved after 74 iterations, starting with $C_{init} = 11111111111111110111$ and $Z_{init} = 656150$, as shown in Fig. 9. In this layout, the X-spine and Y-spine are located at station 7 and station 4, respectively.

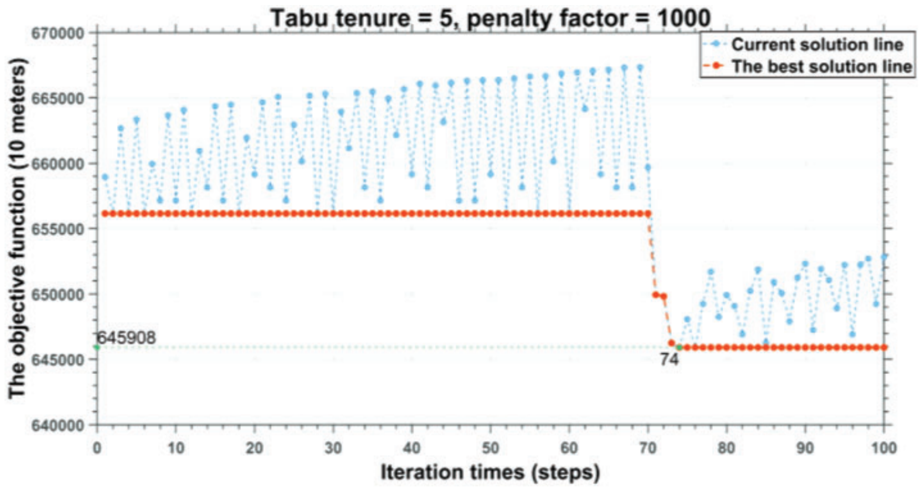


Figure 9: The optimal solution of the example with 20 stations

On the other hand, with the 30 stations layout, the algorithm was configured with the following parameters: Tabu tenure $T = 5$, penalty factor $\rho = 5000$, the maximum iteration $N_{iteration} = 100$, and the maximum of the consecutive non-improving move $N_{nonimpr} = 100$. The optimal solution, $C_{best} = 111111110111111101011111011101$, with an objective function value of $Z_{best} = 2058714$, was achieved after 34 iterations, starting with $C_{best} = 111111111111111111111111011111$ and $Z_{init} = 2068350$, as demonstrated in Fig. 10. In this example, the X-spine and Y-spine are located at station 3 and station 6, respectively. The from-to chart of material flows and the coordinates table of the two examples are shown in Appendix A (Tables 13 to 16).

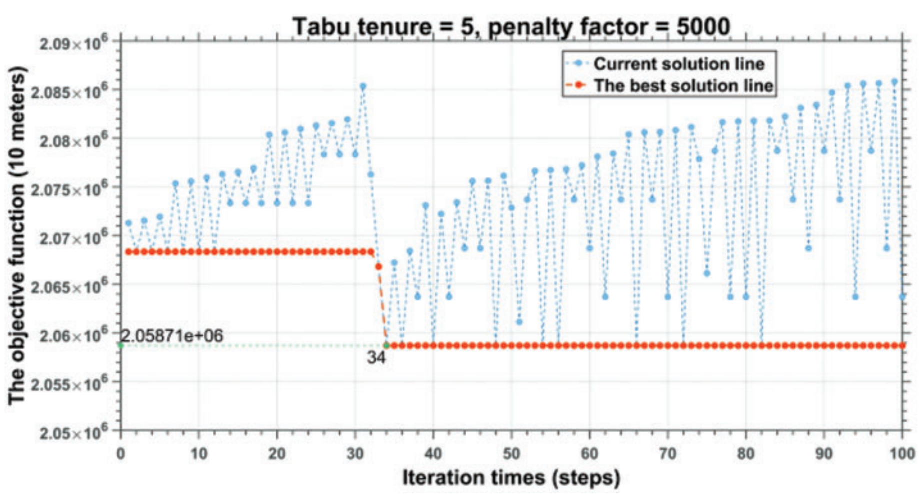


Figure 10: The optimal solution of the example with 30 stations

5 Conclusions

The primary objective of this research is to explore the design of OTSs, considering various critical factors outlined in the Introduction section. Among these factors, optimizing the FPD is a strategic factor that should be considered carefully. Furthermore, since this is an OTSs, the spine layout is chosen to ensure future scalability. Hence, this study places a specific emphasis on the optimization of a double-spine flow path design to minimize the total travel distance for both loaded and empty flows. Additionally, this research takes empty travel flows into account, a factor frequently overlooked in previous work. To address this, the transportation methods, including the North-West Corner and the Stepping-Stone Method, are applied to optimize empty travel flows. The Rectilinear Minisum Location Problem is also utilized to determine the locations of the two main branches, X-spine and Y-spine. Finally, the problem of branching the 10 stations into the two main branches is resolved using the TS algorithm. Key elements of this method involve specific coefficients like Tabu tenure, penalty factor, and stop criteria, which significantly impact its effectiveness.

Based on the optimal results presented in Section 4, it is evident that the suggested methods are suitable for optimizing the spine flow path design. Furthermore, to validate our methods, comparisons and examples have been conducted. The results demonstrate that the optimal solution's value can be reduced, and the iteration count in the search process can be influenced by adjusting the coefficients used in the TS algorithm. In this study, the coefficients, including Tabu tenure and penalty factor, were determined through a trial-and-error method. However, future research could explore methods for optimizing these coefficients using techniques such as Neural Networks, Genetic Algorithms, and others. Additionally, practical experiments should be conducted in the near future to assess the feasibility and effectiveness of the proposed solution in real-world scenarios.

Acknowledgement: This research is funded by Ho Chi Minh City University of Technology (HCMUT), VNU-HCM under Grant Number B2021-20-04. We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

Funding Statement: This research is funded by Ho Chi Minh City University of Technology (HCMUT), VNU-HCM under Grant Number B2021-20-04.

Author Contributions: Mr. Nguyen Huu Loc Khuu is the main author who wrote the manuscript. He is also the main author who did the survey about this research and proposed the method to develop this research. Mr. Thuy Duy Truong and Mr. Quoc Dien Le are the authors who contributed to the exploration of all the research and also proposed the ideas for the research problems. Mr. Tran Thanh Cong Vu, and Mr. Hoa Binh Tran designed, fabricated the mechanical part of this proposed issue and preparing for the experiments in the near future of this system. Assoc. Prof. Tuong Quan Vo is the supervisor of this research topic. He proposes the ideas to the authors to reach the necessary problems about this system. And he is also correct about the scientific and technological issues of this manuscript.

Availability of Data and Materials: The whole data sets proposed in this manuscript are not publicly available, but they are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] B. I. Kim, S. Oh, J. Shin, M. Jung, J. Chae and S. Lee, "Effectiveness of vehicle reassignment in a large-scale overhead hoist transport system," *Int. J. Produc. Res.*, vol. 45, no. 4, pp. 789–802, Feb. 2007. doi: 10.1080/00207540600675819.

[2] A. Zhakov *et al.*, "Application of ANN for fault detection in overhead transport systems for semiconductor fab," *IEEE Trans. Semicond. Manuf.*, vol. 33, no. 3, pp. 337–345, Aug. 2020. doi: 10.1109/TSM.66.

[3] C. H. Duo, "Modelling and performance evaluation of an overhead hoist transport system in a 300 mm fabrication plant," *Int. J. Adv. Manuf. Technol.*, vol. 20, pp. 153–161, 2002. doi: 10.1007/s001700200137.

[4] Y. J. Suh and J. Y. Choi, "Efficient Fab facility layout with spine structure using genetic algorithm under various material-handling considerations," *Int. J. Prod. Res.*, vol. 60, no. 9, pp. 2816–2829, 2022. doi: 10.1080/00207543.2021.1904159.

[5] J. H. Ting and J. M. A. Tanchoco, "Optimal bidirectional spine layout for overhead material handling systems," *IEEE Trans. Semiconduct. Manuf.*, vol. 14, no. 1, pp. 57–64, 2001. doi: 10.1109/66.909655.

[6] N. H. L. Khuu, V. A. Pham, T. T. C. Vu, V. T. B. Dao, T. D. Nguyen and V. Quan Tuong, "A study on design and control of the multi-station multi-container transportation system," *Appl. Sci.*, vol. 12, no. 5, pp. 2686–2713, 2022. doi: 10.3390/app12052686.

[7] W. A. Chen, R. D. de Koster, and Y. Gong, "Performance evaluation of automated medicine delivery systems," *Transp. Res. Part E: Logist. Transp. Rev.*, vol. 147, pp. 102242, Mar. 2021. doi: 10.1016/j.tre.2021.102242.

[8] I. F. A. Vis, "Survey of research in the design and control of automated guided vehicle systems," *Eur. J. Oper. Res.*, vol. 170, no. 3, pp. 677–709, May 2006. doi: 10.1016/j.ejor.2004.09.020.

[9] T. A. Le and M. B. M. de Koster, "A review of design and control of automated guided vehicle systems," *Eur J. Oper Res*, vol. 171, no. 1, pp. 1–23, May 2006. doi: 10.1016/j.ejor.2005.01.036.

[10] P. R. Gutta, V. S. Chinthala, R. V. Manchoju, V. C. MVN, and R. Purohit, "A review on facility layout design of an automated guided vehicle in flexible manufacturing system," *Mater. Today: Proc.*, vol. 5, no. 2, pp. 3981–3986, 2018.

[11] E. B. Hoff and B. R. Sarker, "An overview of path design and dispatching methods for automated guided vehicles," *Integr. Manuf. Syst.*, vol. 9, no. 5, pp. 296–307, 1998. doi: 10.1108/09576069810230400.

[12] R. J. Gaskins and J. M. A. Tanchoco, "Flow path design for automated guided vehicle systems," *Int. J. Prod. Res.*, vol. 25, no. 5, pp. 667–676, 1987. doi: 10.1080/00207548708919869.

[13] R. F. Zanjirani, G. Laporte, and M. Sharifyazdi, "A practical exact algorithm for the shortest loop design problem in a block layout," *Int. J. Prod. Res.*, vol. 43, no. 9, pp. 1879–1887, May 2005. doi: 10.1080/00207540500031980.

[14] K. Eshghi and M. Kazemi, "Ant colony algorithm for the shortest loop design problem," *Comput. Ind. Eng.*, vol. 50, no. 4, pp. 358–366, Aug. 2006. doi: 10.1016/j.cie.2005.05.003.

[15] T. Nishi, S. Akiyama, T. Higashi, and K. Kumagai, "Cell-based local search heuristics for guide path design of automated guided vehicle systems with dynamic multicommodity flow," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 2, pp. 966–980, Apr. 2020. doi: 10.1109/TASE.8856.

[16] M. Hamzheei, R. Z. Farahani, and H. B. Rashidi, "An ant colony-based algorithm for finding the shortest bidirectional path for automated guided vehicles in a block layout," *Int. J. Adv. Manuf. Technol.*, vol. 64, pp. 399–409, Jan. 2013. doi: 10.1007/s00170-012-3999-1.

[17] K. H. Kim and J. M. A. Tanchoco, "Economical design of material flow paths," *Int. J. Prod. Res.*, vol. 31, no. 6, pp. 1387–1407, 1993. doi: 10.1080/00207549308956797.

[18] X. Guan, X. Dai, and J. Li, "Revised electromagnetism-like mechanism for flow path design of unidirectional AGV systems," *Int. J. Prod. Res.*, vol. 49, no. 2, pp. 401–429, Jan. 2011. doi: 10.1080/00207540903490155.

[19] M. Kaspi and J. M. A. Tanchoco, "Optimal flow path design of unidirectional AGV systems," *Int. J. Prod. Res.*, vol. 28, no. 6, pp. 1023–1030, 1990. doi: 10.1080/00207549008942772.

[20] M. Kaspi, U. Kesselman, and J. M. A. Tanchoco, "Optimal solution for the flow path design problem of a balanced unidirectional AGV system," *Int. J. Prod. Res.*, vol. 40, no. 2, pp. 389–401, 2002. doi: 10.1080/00207540110079761.
[21] Y. Seo, C. Lee, and C. Moon, "Tabu search algorithm for flexible flow path design of unidirectional automated-guided vehicle systems," *OR Spectrum*, vol. 29, pp. 471–487, Jul. 2007.
[22] J. Rubaszewski, A. Yalaoui, L. Amodeo, and S. Fuchs, "Efficient genetic algorithm for unidirectional flow path design," *IFAC Proc.*, vol. 45, no. 6, pp. 883–888, 2012.
[23] P. Banerjee and Y. Zhou, "Facilities layout design optimization with single loop material flow path configuration," *The Int. J. Prod. Res.*, vol. 33, no. 1, pp. 183–203, 1995. doi: 10.1080/00207549508930143.
[24] C. Huang, "Design of material transportation system for tandem automated guided vehicle systems," *Int. J. Prod. Res.*, vol. 35, no. 4, pp. 943–953, 1997. doi: 10.1080/002075497195461.
[25] S. Akiyama, T. Nishi, T. Higashi, K. Kumagai, and M. Hashizume, "A multi-commodity flow model for guide path layout design of AGV systems," in *2017 IEEE Int. Conf. Indust. Eng. Eng. Manag. (IEEM)*, 2017, pp. 1251–1255.
[26] J. Rubaszewski, A. Yalaoui, and L. Amodeo, "Solving unidirectional flow path design problems using metaheuristics," in *Metaheuristic for Production Systems*, Cham: Springer, Operations Research/Computer Science Interfaces Series, 2016, vol. 60, pp. 25–56. doi: 10.1007/978-3-319-23350-5.
[27] H. Xiao, X. Wu, Y. Zeng, and J. Zhai, "A CEGA-based optimization approach for integrated designing of a unidirectional guide-path network and scheduling of AGVs," *Math. Problems Eng.*, vol. 2020, no. 2, pp. 1–16, 2020.
[28] N. Shanmugasundaram, K. Sushita, S. P. Kumar, and E. N. Ganesh, "Genetic algorithm-based road network design for optimising the vehicle travel distance," *Int. J. Veh. Inf. Commun. Syst.*, vol. 4, no. 4, pp. 344–354, 2019.
[29] H. Pourvaziri, H. Pierreval, and H. Marian, "Integrating facility layout design and aisle structure in manufacturing systems: Formulation and exact solution," *Eur J. Oper. Res.*, vol. 290, no. 2, pp. 499–513, 2021. doi: 10.1016/j.ejor.2020.08.012.
[30] F. S. Hiller and G. J. Lieberman, "The minimum spanning tree problem," in *Introduction to Operations Research*, 7th ed. New York: Mc Graw-Hill, 1990, pp. 415–420.
[31] C. F. Palmer and A. E. Innes, "The transportation method," in *Operational Research by Example*, UK: Macmillan Education, 1980, pp. 121–142.
[32] J. A. Tompkins, J. A. White, Y. A. Bozer, and J. M. A. Tanchoco, "Rectilinear-distance facility location problems," in *Facilities Planning*, 4th ed. Hoboken, New Jersey: John Wiley & Sons, 2010, pp. 520–521.
[33] M. Laguna, R. Martí, P. Pardalos, and M. Resende, "Tabu search," in *Handbook of Heuristics*, Berlin, Germany, Springer, 2018, pp. 741–758.
[34] W. L. Maxwell and J. A. Muckstadt, "Design of automatic guided vehicle systems," *AIIE Trans.*, vol. 14, no. 2, pp. 114–124, 1982. doi: 10.1080/05695558208975046.

Appendix A.

The following tables show the from-to chart and the coordinates of the layout with 20 stations and 30 stations.

Table 13: The from-to chart of material flows of the layout with 20 stations

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	21	36	66	4	23	11	33	61	33	60	11	4	24	24	67	56	57	57	93
2	73	0	75	63	18	62	21	62	85	95	1	76	95	27	0	81	70	72	47	96
3	91	83	0	17	55	61	6	55	96	22	34	89	44	29	5	54	8	93	4	26

(Continued)

Table 13 (continued)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
4	63	10	5	0	90	6	81	54	99	40	58	61	72	88	84	31	26	31	9	86
5	1	1	4	99	0	26	84	76	57	40	27	39	10	48	73	36	82	67	75	81
6	20	72	26	0	50	0	13	67	23	27	83	95	76	68	44	68	78	43	18	5
7	14	63	5	29	28	40	0	8	24	30	96	82	48	87	82	49	36	68	15	26
8	43	61	24	6	21	57	3	0	92	79	10	21	15	62	35	71	57	25	36	79
9	25	48	40	36	85	99	1	1	0	18	61	80	19	71	76	29	26	57	6	94
10	41	80	79	82	43	75	37	0	40	0	84	4	6	15	32	18	31	20	41	22
11	61	60	7	3	89	67	54	26	16	68	0	99	83	82	11	14	73	4	77	4
12	84	60	11	4	17	96	42	78	49	24	41	0	9	49	63	19	79	86	32	64
13	23	17	79	17	9	14	5	13	88	17	86	25	0	99	17	87	21	14	79	57
14	11	16	33	96	6	78	58	77	99	16	73	64	91	0	49	74	13	79	31	96
15	10	11	8	80	98	83	40	79	53	51	1	69	14	28	0	92	93	87	73	81
16	90	25	88	69	97	71	81	36	47	62	28	20	73	16	81	0	4	79	47	66
17	15	49	32	84	20	60	36	46	92	23	95	41	47	37	30	33	0	65	21	93
18	89	69	1	67	67	26	83	44	90	9	71	58	42	61	70	1	25	0	92	96
19	17	0	43	29	20	98	84	47	98	1	14	0	89	33	91	74	45	55	0	34
20	28	96	48	48	61	3	94	80	38	75	65	1	25	37	77	42	3	36	37	0

Table 14: The coordinates of each station in the layout 20 stations

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a_i	11	8	37	2	35	36	25	3	4	22	26	34	31	25	15	29	21	27	33	6
b_i	4	27	3	20	16	29	1	6	35	30	10	28	13	25	11	2	5	26	22	33

Table 15: The from-to chart of material flows of the layout with 30 stations

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	0	7	13	86	82	34	58	96	84	14	70	91	57	36	14	68	69	96	93	63	85	35	68	33	50	71	67	25	71	54
2	6	0	15	15	54	69	4	65	71	39	37	95	62	9	49	47	80	50	30	26	91	0	74	76	15	94	39	70	57	51
3	15	33	0	50	59	79	67	83	67	34	3	12	90	87	46	42	53	57	95	19	70	5	9	93	90	8	98	0	52	99
4	12	19	50	0	28	73	47	37	27	8	17	80	70	69	14	82	24	97	0	15	66	13	30	96	57	22	16	99	39	59
5	6	38	59	88	0	38	46	4	16	28	32	6	86	88	64	10	9	57	12	90	29	40	44	77	77	9	11	83	94	6
6	84	67	83	65	72	0	28	90	94	85	40	3	65	99	74	45	33	41	25	79	36	2	55	82	67	71	99	46	31	40
7	42	37	15	2	7	59	0	5	5	62	98	38	36	54	59	62	52	11	38	50	89	90	65	99	39	99	6	10	54	93
8	98	96	86	57	10	59	75	0	79	76	75	80	32	36	58	78	79	63	11	38	26	37	36	43	45	9	25	9	59	31
9	89	8	36	20	61	63	16	97	0	84	45	44	77	99	48	37	65	14	88	6	12	20	13	78	13	86	47	93	79	78
10	84	89	92	67	51	97	77	65	21	0	21	44	70	56	26	40	88	16	65	73	58	61	43	7	51	28	33	75	86	62
11	9	29	33	85	85	98	4	57	53	10	0	60	31	80	37	19	10	67	81	12	96	3	47	71	26	39	29	83	96	42
12	84	1	79	51	30	93	14	31	98	33	92	0	84	91	29	26	71	53	16	55	11	67	6	50	99	10	56	94	18	83
13	92	39	33	65	69	59	45	85	46	59	47	18	0	27	4	42	62	32	45	81	70	70	90	17	97	97	46	13	15	34
14	5	79	35	44	24	93	78	90	44	93	86	29	43	0	20	35	56	95	66	7	22	86	51	84	73	66	22	58	85	88
15	59	54	27	99	59	77	32	32	64	93	46	3	38	35	0	75	27	77	15	53	65	63	27	30	29	66	40	22	25	38

(Continued)

Table 15 (continued)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
16	18	17	4	59	51	6	1	13	57	86	6	35	83	32	16	0	30	19	56	33	38	44	8	55	17	43	70	64	86	29
17	38	57	64	82	8	19	31	53	72	85	57	5	12	25	11	37	0	94	81	84	13	51	62	44	84	28	9	58	51	32
18	22	37	36	19	84	33	75	54	72	55	0	20	90	48	16	34	5	0	22	15	20	10	8	41	82	90	37	85	28	31
19	58	84	86	68	71	51	65	8	41	23	26	7	22	53	9	96	20	62	0	79	3	95	20	75	33	66	48	97	70	1
20	70	55	74	28	28	22	76	98	29	31	57	54	62	37	36	32	58	76	86	0	7	19	66	8	35	56	72	27	0	6
21	43	76	44	65	49	62	81	3	77	19	41	96	72	62	78	87	59	98	46	86	0	43	16	92	96	14	18	39	11	77
22	97	94	80	18	35	24	34	59	3	22	93	66	94	91	39	76	70	48	5	38	44	0	57	16	26	81	43	41	50	14
23	70	22	90	59	82	68	98	42	21	74	29	73	35	60	50	66	86	32	12	80	16	91	0	54	93	16	59	32	7	77
24	18	78	15	12	42	65	38	16	18	24	0	91	84	9	1	73	2	78	92	10	87	53	71	0	7	80	1	38	96	27
25	54	9	66	54	42	77	8	68	27	67	67	14	80	19	91	71	16	81	33	85	43	87	10	71	0	6	62	7	70	42
26	35	5	99	71	14	98	56	75	14	12	10	3	32	16	16	18	62	43	27	14	32	13	64	51	75	0	48	31	11	26
27	81	96	52	35	5	15	78	70	14	94	8	34	10	64	84	58	32	11	99	98	36	36	14	12	48	12	0	92	29	28
28	61	66	70	99	33	17	40	18	0	57	81	48	83	16	0	64	77	24	0	14	6	25	50	80	50	81	76	0	10	5
29	12	97	10	37	62	54	62	13	99	18	94	94	11	53	44	63	70	39	77	28	87	72	42	87	40	48	13	79	0	53
30	44	80	89	72	75	14	90	44	18	71	5	19	13	83	92	26	42	71	63	47	69	73	5	23	77	35	77	3	82	0

Table 16: The coordinates of each station in the layout 30 stations

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
a_i	19	52	27	22	23	39	1	37	7	15	59	56	32	53	18	16	20	12	29	11	31	46	13	28	35	5	45	24	38	8
b_i	2	36	56	34	53	31	30	5	38	1	44	33	3	26	39	42	51	9	37	20	10	29	32	40	8	18	58	19	60	22