sensors

Article

Multi-Robot Task Scheduling with Ant Colony Optimization in Antarctic Environments

Seokyoung Kim and Heoncheol Lee *

Department of IT Convergence Engineering, Kumoh National Institute of Technology, Gumi 39177, Republic of Korea
* Correspondence: hclee@kumoh.ac.kr; Tel.: +82-54-478-7476

Abstract: This paper addresses the problem of multi-robot task scheduling in Antarctic environments. There are various algorithms for multi-robot task scheduling, but there is a risk in robot operation when applied in Antarctic environments. This paper proposes a practical multi-robot scheduling method using ant colony optimization in Antarctic environments. The proposed method was tested in both simulated and real Antarctic environments, and it was analyzed and compared with other existing algorithms. The improved performance of the proposed method was verified by finding more efficiently scheduled multiple paths with lower costs than the other algorithms.

Keywords: Antarctic environments; ant colony optimization; multi-robot task scheduling

check for updates

Citation: Kim, S.; Lee, H. Multi-Robot Task Scheduling with Ant Colony Optimization in Antarctic Environments. *Sensors* **2023**, *23*, 751. https://doi.org/10.3390/s23020751

Academic Editor: Aiguo Song

Received: 5 December 2022
Revised: 31 December 2022
Accepted: 3 January 2023
Published: 9 January 2023

1. Introduction

Due to the development of robot technology, robots are working instead of humans in many places. Robots have the advantage of being able to perform precise tasks that humans cannot do, and tasks that are dangerous for humans to do. Based on these advantages, robots are used in many places such as in homes, services, industry, medical applications, and military applications. Robot applications using a single object, such as a cleaning robot, a guide robot, and a process using a robot arm, have been commercialized. However, in the case of a single robot, the limitations are clear, such as low efficiency or unachievable tasks. To overcome this, multi-robots began to be introduced, which showed increased work efficiency and more diverse mission performance. In other words, using multi-robots enables efficient performance of tasks such as fast processing speed and large workload, but this requires advanced technology. This is because as the number of robots increases, the control structure and calculation time increase dramatically. For this reason, research fields for multi-robots such as task allocation, coverage, and scheduling have been created and are being studied steadily.

Among them, scheduling is for the efficient operation of the robot and aims to reduce the driving time of the robot. When there are multiple robots and multiple destinations, each robot is given an appropriate visit order to minimize the robot's travel distance and reduce driving time. This can be seen as a kind of traveling salesman problem (TSP) [1–5]. The TSP is to find the shortest possible route from a given set of cities, visiting every city exactly once and returning to the starting point. The TSP is NP-Hard, and there are a number of algorithms to solve this problem. Representatively, there are breadth-first search (BFS) and depth-first search (DFS) [6]. These are algorithms for traversing or searching tree or graph data structures, which guarantees the minimum distance, but has the disadvantage of consuming a lot of resources when the path is long and not guaranteeing a problem-solving time. Nearest neighbor [7] is a simple algorithm to repeat visiting the nearest city. This has the advantage of being easy to implement and fast to calculate, but it does not guarantee minimum distance. The genetic algorithm (GA) [8–11], one of the heuristic algorithms, guarantees distance and time according to the setting of the user parameter. If the user parameter is properly set, the distance can be calculated with a reasonable calculation time. Ant colony optimization (ACO) [12–20] is also a heuristic algorithm that was conceived in the way ants return home in search of food. Various approaches have been taken to solve the TSP using ACO, which also made it possible to obtain distances with reasonable computational time. When the TSP is applied to multi-robots, it is called the multiple traveling salesman problem (MTSP) [21–28]. The MTSP is an optimization problem in which multiple salesmen visit all destinations with minimal distance. Many approaches have been taken to solve the MTSP based on the above algorithms.

However, these TSP solutions will become more difficult to implement in extreme environments. The extreme environment examined here is Antarctica, a place with a large area of about 14,000,000 km^2, very low temperatures, and various adverse conditions including snow, ice, and crevasses. Antarctica is the southernmost continent of the Earth, and is an attractive unexplored region with enormous scientific value, fishery resources, and energy resources to cope with the Earth's climate change problems. To discover this value of Antarctica, 47 countries have joined the Antarctic Treaty and are fiercely competing for Antarctic research. Many scientists are trying to study Antarctica, but in extreme weather, crevasses make it difficult for humans to explore the polar regions. To overcome this, scientists have begun to research and introduce unmanned autonomous driving robots. The operation of robots in general environments such as roads and indoors is relatively free from the aforementioned constraints. The general environment does not make robot operation difficult because the floor is relatively flat and not slippery, and there are few fatal obstacles such as crevasses. However, in Antarctic environments, sloping areas such as hills and mountains, slippery floors caused by snow or ice, and crevasses should be considered. The Cold Regions Research and Engineering Lab (CRREL) in the United States has developed 'Cool Robot' [29] and 'Yeti' [30], autonomous vehicles that can be operated in polar environments, and they are used to collect research data in extreme areas such as Antarctica. However, they show disadvantages in environments such as snow and ice. In addition, the average speed of the robot is about 0.4 to 1.5 m/s, which is slow, and it is somewhat disadvantageous in time to explore the vast area of Antarctica. Therefore, the need for efficient exploration work using multi-robots rather than single robots has emerged. Figure 1 describes the concept of multi-robot scheduling.

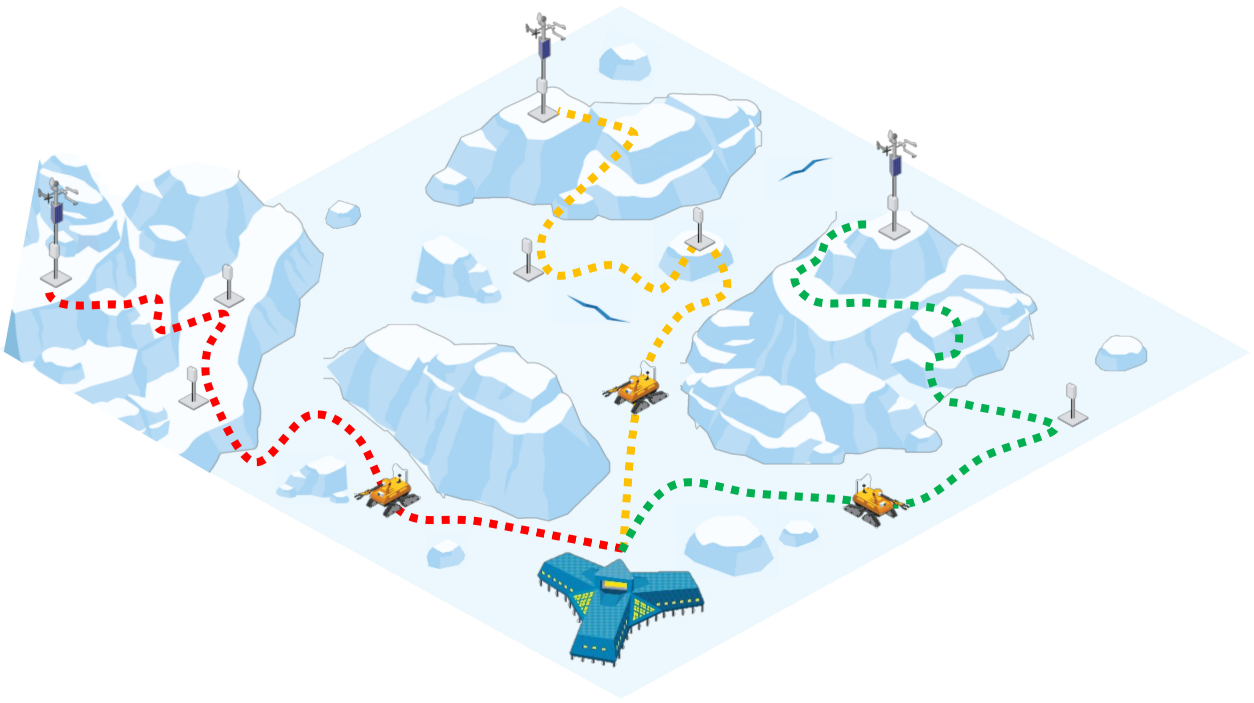


Figure 1. The concept of multi-robot scheduling in Antarctic environments.

In this paper, we propose a practical multi-robot scheduling method in Antarctic environments. This allows multi-robots to visit all nodes with the shortest distance. This can be seen as a kind of MTSP problem-solving, but considering the specificity of Antarctica, the process of returning to the starting point after visiting all nodes was omitted. In addition, stable driving can be realized by avoiding sharp slopes by reflecting Antarctic altitude information in scheduling. It can also produce better results with reasonable computational time. The contributions of this paper are as follows.

- To the best of our knowledge, this is the first approach which solves the multi-robot task scheduling problem in Antarctic environments.
- The performance of the multi-robot task scheduling result was tested and evaluated in both simulated and real Antarctic environments.
- The scheduled paths by the proposed method can improve the efficiency of operating multiple robots by considering the characteristics of robot movement in Antarctic environments.

The remainder of this paper is organized as follows. Section 2 describes constraints in the Antarctic environments and the necessity for scheduling including constraints, as well as the definition of MTSP and problems of applying existing algorithms to Antarctic environments. Section 3 describes the cost function and the structure of ACO used for multi-robot scheduling. Section 4 shows the experimental results and the comparison with other methods. Finally, Section 5 is the conclusion.

2. Problem Description

This paper addresses the problem of multi-robot scheduling in Antarctic environments. First, we define the specificity of the Antarctic environment and its problems. It addresses problems that can be caused by extremely low temperatures, snow and ice environments, and altitudes. A practical scheduling method for overcoming these problems is described later. This can be seen as a kind of MTSP, but considering the characteristics of Antarctic environments, it is assumed that the multi-robot does not return to the starting point after visiting all nodes.

2.1. Antarctic Environments

Antarctica is one of the coldest regions on Earth, covering an area of about 14,000,000 km^2, of which 98% is made up of snow and ice. In all regions, the temperature does not exceed 0 °C, and the lowest temperature is −89.2 °C, which is the coldest area. These conditions make it difficult to operate the robot. For example, when exposed to low temperatures, it causes damage to the battery and is bad for the chassis of the robot. Additionally, the snow and ice floor reduce the robot's ability to move. For stable driving in snow and on icy terrain, it will be necessary to avoid slopes in consideration of height. The crevasse, a deep crack on the glacial surface, is also one of the obstacles that must be avoided. For stable robot operation, these constraints should be avoided as much as possible. Therefore, scheduling in Antarctic environments needs to reflect elements of the terrain as well as distance in the cost.

2.2. Definition of MTSP

In this paper, the definition of MTSP is as follows. The multi-robot scheduling problem is defined as visiting a given a set of nodes $\mathbf{C} = \{c_1, c_2, c_3, \dots, c_N\}$, where $n = 1, \cdots, N$ for each robot r, with the shortest distance. Each robot has a number of visits $\mathbf{P} = \{p_1, p_2, p_3, \dots, p_R\}$, where $r = 1, \cdots, R$. It is defined as a single depot if there is one starting position and a multiple depot if there are multiple starting positions. In this paper, a single depot is assumed. Each of the R robots located in the single depot must visit one or more nodes and will not return to the starting position. Each robot has a tour $\mathbf{T}_r$, which is described as follows.

$$\mathbf{T}_r = \{\check{c}_i^r\}_{i=1,\cdots,p_R} \text{ where } \check{c}_i^r \in \mathbf{C} \tag{1}$$

where $\check{c}_i^r$ is the node visited by the robot r and p_r is the total number of nodes that the robot r will visit, calculated as follows.

$$\sum_{r=1}^{R} p_r = N \tag{2}$$

In the tour $\mathbf{T}_r$, when the distance between the nodes $\check{c}_i^r$ and $\check{c}_{i+1}^r$ is d_i^r, the total tour distance for the tour $\mathbf{T}_r$, is as follows.

$$D(\mathbf{T}_r) = \sum_{i=1}^{p_r-1} d_i^r \tag{3}$$

$\mathbf{T}_{r,min}$, the tour with the minimum total travel distance, is defined as follows.

$$\mathbf{T}_{r,min} = \underset{\mathbf{T}_r}{\operatorname{argmin}}(D(\mathbf{T}_r)) \tag{4}$$

Then, the goal is to obtain a set of $\mathbf{T}_{r,min}$ for R robots.

$$\mathbf{T}_{min} = \{\mathbf{T}_{1,min}, \mathbf{T}_{2,min}, \cdots, \mathbf{T}_{R,min}\} \tag{5}$$

To minimize the distance, it is required to set the number of tour nodes p_r for each robot r and obtain $\mathbf{T}_{min}$ through an appropriate algorithm.

2.3. The Problem of Applying the Existing Scheduling Algorithm to Antarctic Environments

Various algorithms have been studied to solve the multi-robot scheduling problem. Among them, the nearest neighbor algorithm is a simple algorithm, summarized as follows.

(1) Select a starting point for any city and register it as a visiting node.
(2) Move to the unvisited node with the lowest cost and register it as the visited node.
(3) Repeat Step 2 if there is a city that was not visited.

It is simple and effective. However, due to the greedy nature of the NN algorithm, it only seeks immediate benefits. Thus, it misses the opportunity to make long-term gains. This leads to the creation of a bad path. When scheduled based on the cost reflecting not only the distance but also the topographical elements, these characteristics will be revealed as disadvantages.

ACO is also one of the algorithms for solving the multi-robot scheduling problem. ACO is an algorithm that solves problems by exploring artificial ants. Ants have a rule that they prefer places with a low cost and high pheromones, which is summarized as follows.

(1) Explore ants.

An ant selects a node by the probability p, which is proportional to the amount of pheromones and the cost.

(2) When the ants finish their search, they leave pheromones in the path of the ant that has the lowest cost.
(3) Repeat as iteration.
(4) After that, the ant that moved to the lowest cost becomes a solution.

In ACO, pheromones as well as cost are additionally considered. Moreover, probabilistic node exploration allows ants to explore various paths, which gives them an opportunity to choose better nodes in the long term. This eventually makes it possible to find a better path.

ACO can easily control the cost function, so it is easy to evaluate factors other than distance. It is also immediate and intuitive because it reflects the cost each time when visiting the nodes one by one.

3. Proposed Method

3.1. Overview

Figure 2 is a flowchart of the proposed algorithm. This algorithm is based on ACO, but two main features are added. First, for multi-robot scheduling, the number of nodes each robot will visit is set. This is determined by the user or automatically divided by the number of robots. Then, paths for each robot, an MTSP solution, is generated for multi-robots using ACO. After paths for multi-robots are created using the proposed method, each robot moves according to its path. In this paper, a new cost function for ACO is proposed to properly reflect the characteristics of Antarctic environments.

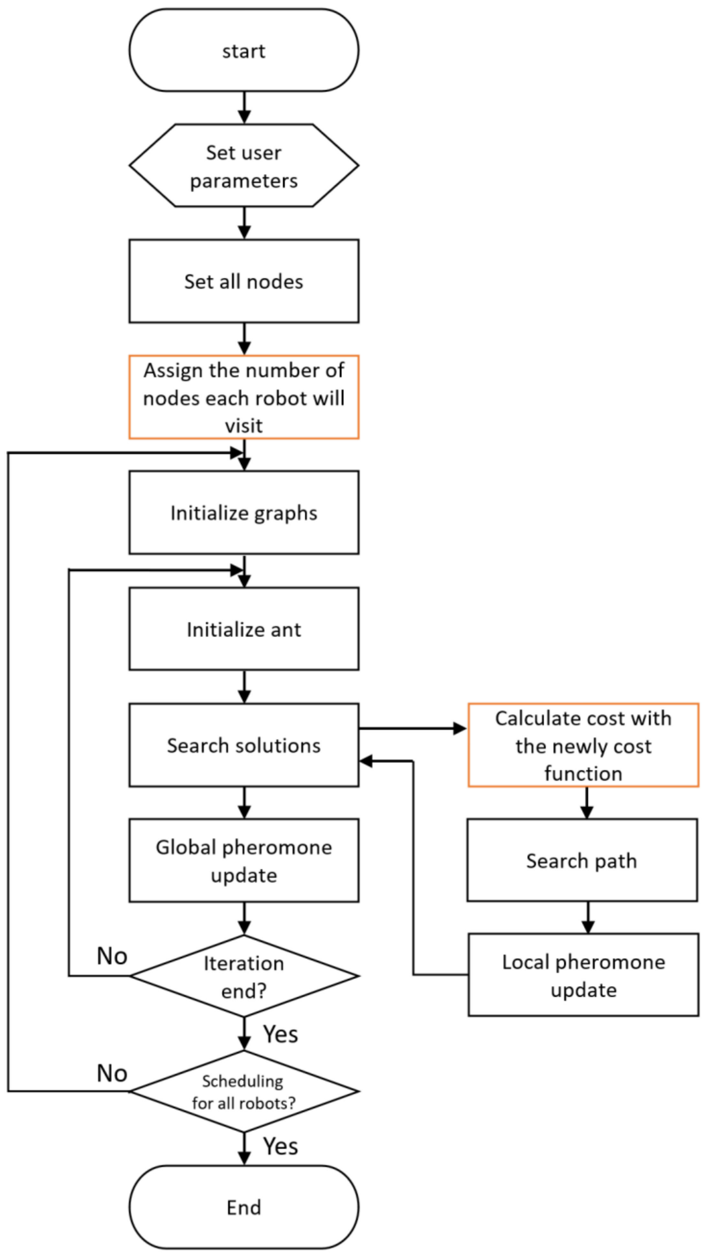


Figure 2. The flowchart of the proposed method.

3.2. Cost Function

Only the distance between nodes was considered for cost function used in the existing ACO. This is difficult to reflect Antarctic environments. The proposed cost function contains elevation information. In the existing cost function, the distance between nodes p and q is calculated as the Euclidean distance. However, this is a straight distance, which becomes inaccurate if altitude information is added. It is also difficult to measure the exact distance between nodes p and q including altitude information. This was overcome by obtaining an approximate distance value by sampling between nodes. A method of obtaining the distance between nodes p and q including the altitude value is as follows.

The node p and the node q are sampled k times, and the distance obtained by dividing $d_{p,q}$ by k is d_k, where $d_{p,q}$ is the distance between nodes p and q. The altitude value $H_{p,q}$ sampled between node p and node q is as follows.

$$H_{p,q} = (h_1, h_2, h_{3,} \cdots, h_{k-1}, h_k) \tag{6}$$

where h_k is the height of the kth sampled point. The difference l_i of the sampled height value is as follows.

$$l_i = h_{i+1} - h_i \tag{7}$$

The elevation distance $d'_{p,q}$ for reflecting the altitude information between nodes p and q is defined as follows.

$$d'_{p,q} = \sum_{i=1}^{k-1} \sqrt{{d_k}^2 + {l_i}^2} \tag{8}$$

The value $\Theta_{p,q}$ for reflecting the altitude information between node p and node q is as follows.

$$\Theta_{p,q} = \sum_{i=1}^{k-1} \theta_i \tag{9}$$

where θ_i is the angle between the straight line between h_i and h_{i+1} and the straight line parallel to the x-axis. Finally, the proposed cost function $c_{p,q}$ is as follows.

$$c_{p,q} = Ad'_{p,q} + B\Theta_{p,q} \tag{10}$$

where A and B are weights, which can be arbitrarily determined by the user. $d'_{p,q}$ is the distance of the city time, and $\Theta_{p,q}$ is the altitude value of the city time. Figure 3 shows that the cost function is calculated based on the altitude information included at the edge between nodes, where the curve is height information and the straight red line is a straight line connecting points sampled at regular intervals; the sum of the lengths of the straight red line is $d'_{p,q}$, and the sum of the angles is $\Theta_{p,q}$.

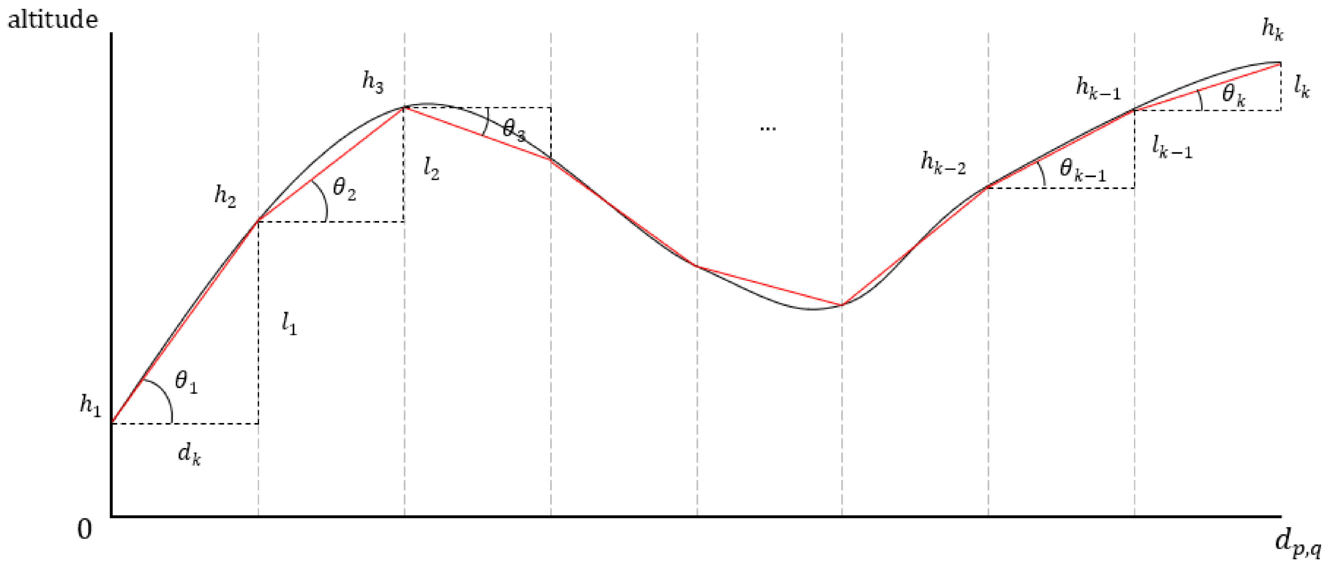


Figure 3. Example of calculating the proposed cost function between nodes p and q.

3.3. Ant Colony Optimization

There are various types of ACOs; among them, Ant Colony System (ACS) [13] was used. ACS is an improved algorithm by adding several processes to the existing Ant System (AS) [14–16]. Based on this, multi-robot scheduling using cost function adapted to Antarctic environments was implemented.

First, it is the random-proportional rule that ant k visits from node p to node q.

$$\omega_{pq}^{k} = \begin{cases} \frac{{\tau_{pq}}^{\alpha}{\eta_{pq}}^{\beta}}{\sum_{l \notin V_k} {\tau_{pg}}^{\alpha}{\eta_{pg}}^{\beta}}, & if\ g \notin V_k \\ 0, & otherwise \end{cases} \tag{11}$$

where τ is the pheromones and η is the importance of the edge, which is the inverse of the cost. The cost is the value obtained using Equation (10). V_k is the set of nodes visited by ant k. α is a parameter that determines the importance of pheromones, and β is a parameter that determines the importance of edge cost. Here, the state transition rule of ACS is applied as follows.

$$s = \begin{cases} argmax_{q \notin V_k}\{{\tau_{pq}}^{\alpha}{\eta_{pq}}^{\beta}\}, & if\ z < z_0 \\ S, & otherwise \end{cases} \tag{12}$$

where S is a random variable determined by Equation (11). It is a rule about ants choosing a pheromone-rich path. If the random number z $(0 \leq z \leq 1)$ is less than z_0, ants choose the path where the pheromone level is high, and if not, it follows the random-proportional rule of the AS. This prevents falling into the local optimal solution. Random-proportional rules and the state transition rules are applied, and local parent update is performed according to Equation (13) whenever visiting a node.

$$\tau_{pq} = (1 - \varphi)\tau_{pq} + \varphi\tau_0 \tag{13}$$

where $0 \leq \varphi \leq 1$ is a pheromone decay parameter. τ_0 is a initial value of pheromone; it is usually $\tau_0 = 1/nc_{nn}$. c is the number of cities, and c_{nn} is the cost calculated by the nearest neighbor. This allows all ants to be affected by pheromones in real time and avoid local optimums.

When all ants generate a tour, global pheromone update is performed through Equations (14) and (15).

$$\tau_{pq} = (1 - \rho)\tau_{pq} + \Delta\tau_{pq}^{best} \tag{14}$$

$$\Delta\tau_{pq}^{best} = \begin{cases} 1/c_{best}, & if\ best\ ant\ travels\ on\ node\ p,q \\ 0, & otherwise \end{cases} \tag{15}$$

where c_{best} is the best ant's tour. This increases the probability of exploring a better path in the next iteration by accumulating pheromones along the tour of the best solution.

The proposed method appropriately divided the number of nodes so that multi-robots can perform TSP. Although the user may determine the number of nodes to be visited by the robot, it is basically implemented by dividing the number of nodes by the number of robots. Algorithm 1 is a pseudo-code for the proposed method.

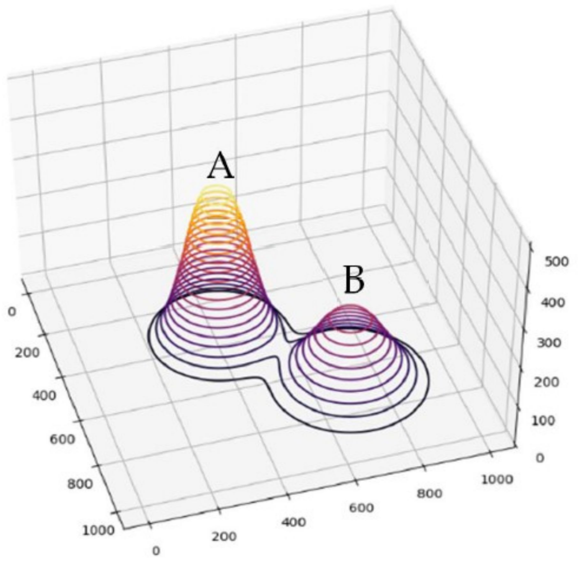
Algorithm 1 Multi-robot scheduling algorithm in Antarctic Environments

```
1:  Initialize the multi robot's tour $T$
2:  Set number of nodes that the robot will visit $N$ and iteration $I$
3:  for $i \leftarrow N$ do
4:    for $j \leftarrow I$ do:
5:      for each ant do
6:        Build a solution according to the number of nodes $i$
7:        Update local pheromone
8:      end for
9:      Update global pheromone
10:   end for
11:   Append best ant's tour to $T$
12: end for
13: return Multi robot's tour $T$
```

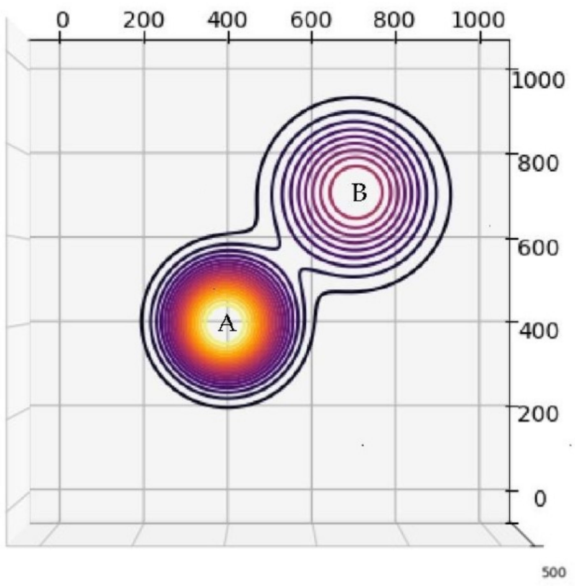
4. Results

4.1. Results in Simulation Environments

Simulations were performed to compare the proposed method with NN and GA. They show a specific performance difference by comparing the elevation distance. The elevation distance uses the value according to Equation (8) according to the x, y, and z coordinates of the node and the edge. The simulations were performed in Python 3.9.7 and the results were visualized using matplotlib. The environment within the simulation is a virtual 3D space, which is $1000 \times 1000 \times 500$ pixels. In order to realize the height in the virtual space, virtual hills A and B according to the normal distribution were generated using the probability density function. The normal distribution function value of hill A is as follows: $\sigma = 75$, $\mu = 20$. The height is 400. The normal distribution function value of hill B is as follows: $\sigma = 100$, $\mu = 15$. The height is 200. The location of the nodes was randomly set, and simulations were performed on 20, 30, and 40 nodes. Figure 4 shows a virtual space, where (a) is the space viewed from the side, and (b) is the space viewed vertically.



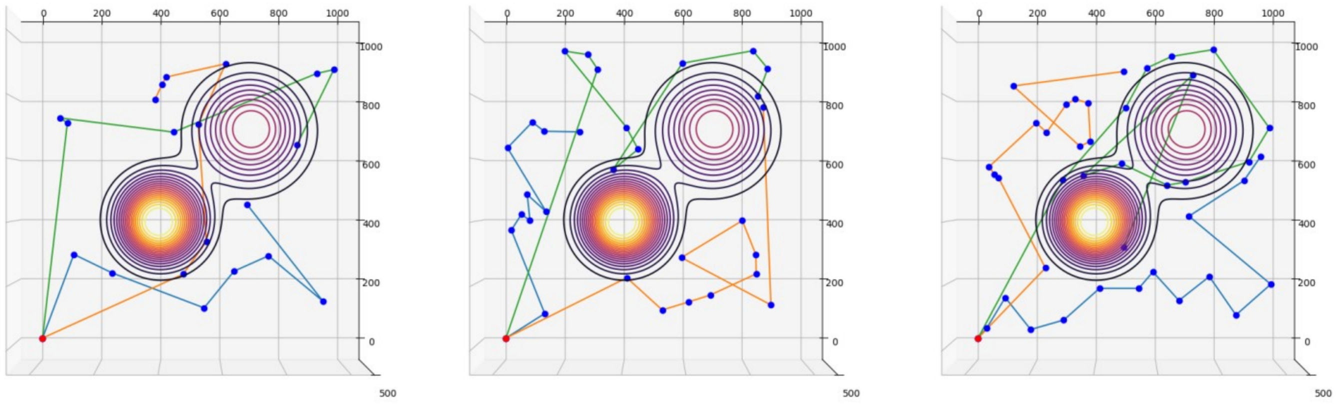
(**a**)



(**b**)

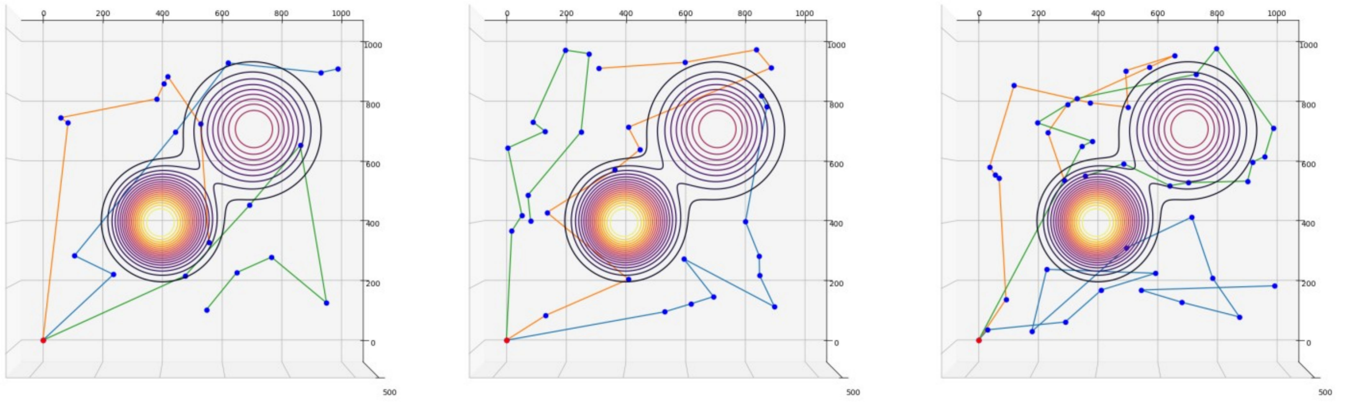
Figure 4. (**a**) The side view of the simulation environment. (**b**) The vertical view of the simulation environment. A and B are the hills of the simulation environment.

For the elevation distance comparison with the proposed method, the nearest neighbor algorithm and the genetic algorithm were performed. The cost functions of the proposed method, NN, and GA were defined according to Equation (10), and the parameter values were as follows: $A = 10$, $B = 15$. The parameter values of GA were as follows: $mutation\ rate = 0.05$, $population = 50$, $generation = 300$, selection operator was tournament, crossover operator was two-point crossover and elitism was applied. and elitism was applied. The parameter values of ACO in the proposed method were as follows: $ants = 40$, $iteration = 20$, $\alpha = 2$, $\beta = b$, $\varphi = 0.1$, $\rho = 0.05$, $z_0 = 0.5$, $ants$ is the number of ants, $iteration$ is the number of iterations. Figures 5–7 are the results of NN, GA, and the proposed method in the simulation environment, and Table 1 is a comparison table of the elevation distance. Figure 8 is a comparison chart of the elevation distance.



(**a**) (**b**) (**c**)

Figure 5. Simulation results of the nearest neighbor in virtual space for: (**a**) 20 nodes, (**b**) 30 nodes, (**c**) 40 nodes. The red dot is the starting point and the blue dots are the nodes to visit. Each color line is the path of each robot.



(**a**) (**b**) (**c**)

Figure 6. Simulation results of the genetic algorithm in virtual space for: (**a**) 20 nodes, (**b**) 30 nodes, (**c**) 40 nodes.

Table 1. The elevation distance comparison results of the simulation.

Part	Node 20	Node 30	Node 40
Nearest Neighbor	5476.98	6255.67	7943.39
Genetic Algorithm	5729.22	6348.65	8335.68
Proposed Method	5584.36	5784.93	6484.76

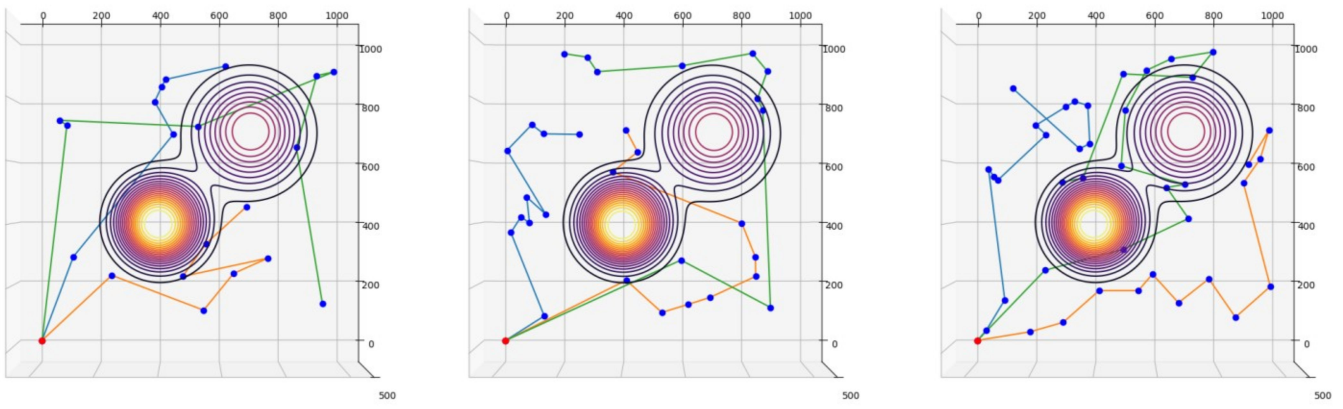

(**a**) (**b**) (**c**)

Figure 7. Simulation results of the proposed method in virtual space for: (**a**) 20 nodes, (**b**) 30 nodes, (**c**) 40 nodes.

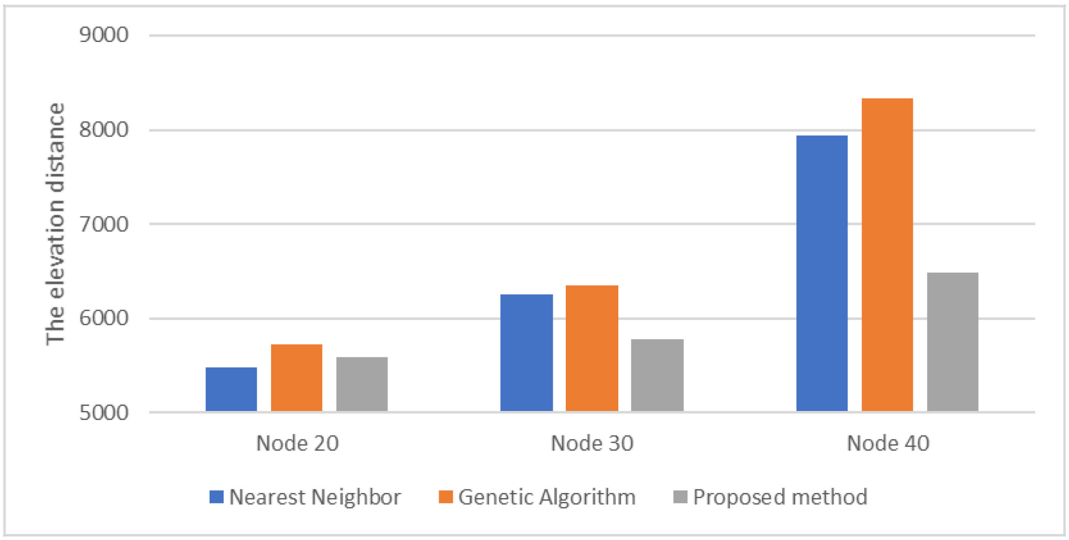


Figure 8. The elevation distance comparison in simulation environments.

Regarding the results in simulation environments, as shown in Table 1, NN and the proposed method showed similar results for 20 nodes. However, as the number of nodes increased, the proposed method showed a shorter elevation distance. GA showed low performance in all of the results. NN is shorter in computational time, but real time does not need to be guaranteed, so the proposed method is reasonable by generating shorter and more stable paths with less than 10 s.

4.2. Results in Real Antarctic Environments

Simulations in Antarctic environments were performed to compare the proposed method with NN and GA. As described above, specific performance differences are presented by comparing the elevation distance.

In the simulation, the Antarctic environment was located at 74°37.4′ S, 164°13.7′ E, and nodes were randomly set nearby. The latitude and longitude values of arbitrary nodes were extracted from Google Earth. The distance between nodes was obtained using the Haversine Formula. The altitude information obtained the altitude values of nodes and edges using the Google Maps API. The altitude values for the edges were sampled 500 times at the same interval. For performance comparison with the proposed method, the nearest neighbor algorithm and the genetic algorithm were performed. The cost functions of the proposed method, NN, and GA were defined according to Equation (10), and the parameter values were as follows: $A = 3$, $B = 2$. The parameter values of GA were as follows: $Selection = tournament$, $Crossover = two - point\ crossover$, $mutation\ rate = 0.05$, $population = 50$, $generation = 300$, and elitism is applied. The parameter values of ACO in the proposed method were as follows: $ants = 40$, $iteration = 20$, $\alpha = 2$, $\beta = b$, $\varphi = 0.1$, $\rho = 0.05$, $z_0 = 0.5$, $ants$ is the number of ants, $iteration$ is the number of iterations. Figures 9–11 are the simulation results of NN, GA, and the proposed method in the Antarctic environments, respectively, and Table 2 is a comparison table of the elevation distance. Figure 12 is a comparison chart of the elevation distance.

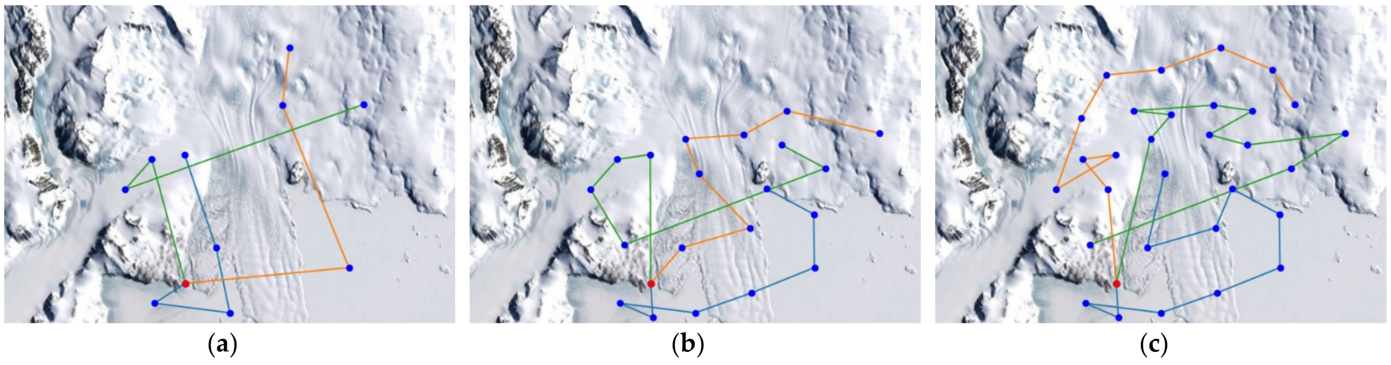


Figure 9. Simulation results of the nearest neighbor in Antarctic environments for: (**a**) 10 nodes, (**b**) 20 nodes, (**c**) 30 nodes. The red dot is the starting point and the blue dots are the nodes to visit. Each color line is the path of each robot.

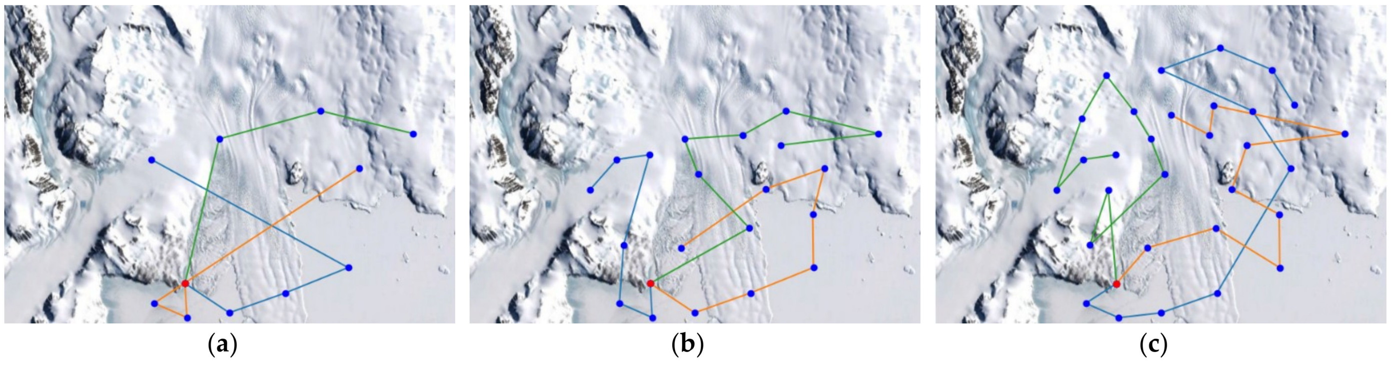


Figure 10. Simulation results of the genetic algorithm in Antarctic environments for: (**a**) 10 nodes, (**b**) 20 nodes, (**c**) 30 nodes.

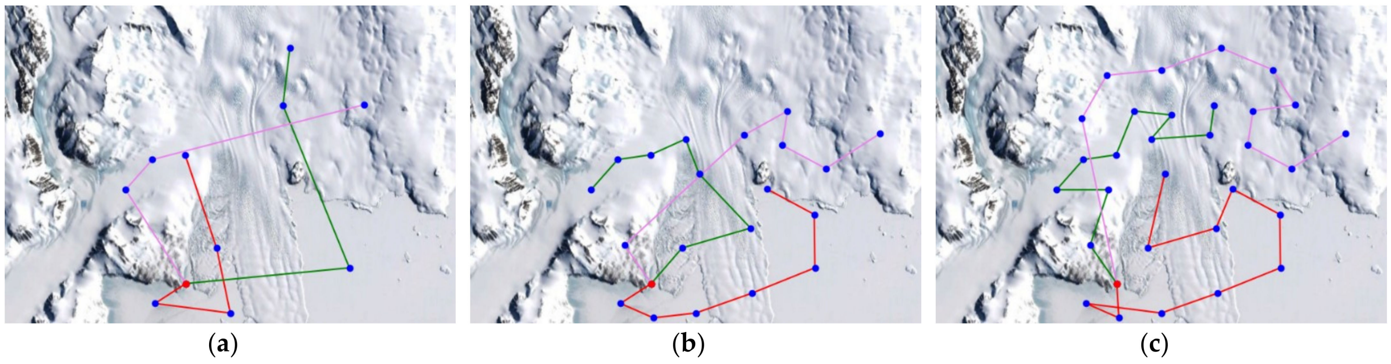


Figure 11. Simulation results of the proposed method in Antarctic environments for: (**a**) 10 nodes, (**b**) 20 nodes, (**c**) 30 nodes.

Table 2. The elevation distance comparison results of real Antarctic environments.

Part	Node 10	Node 20	Node 30
Nearest Neighbor	82.01 km	99.89 km	145.67 km
Genetic Algorithm	79.41 km	93.93 km	141.36 km
Proposed Method	78.17 km	82.95 km	122.78 km

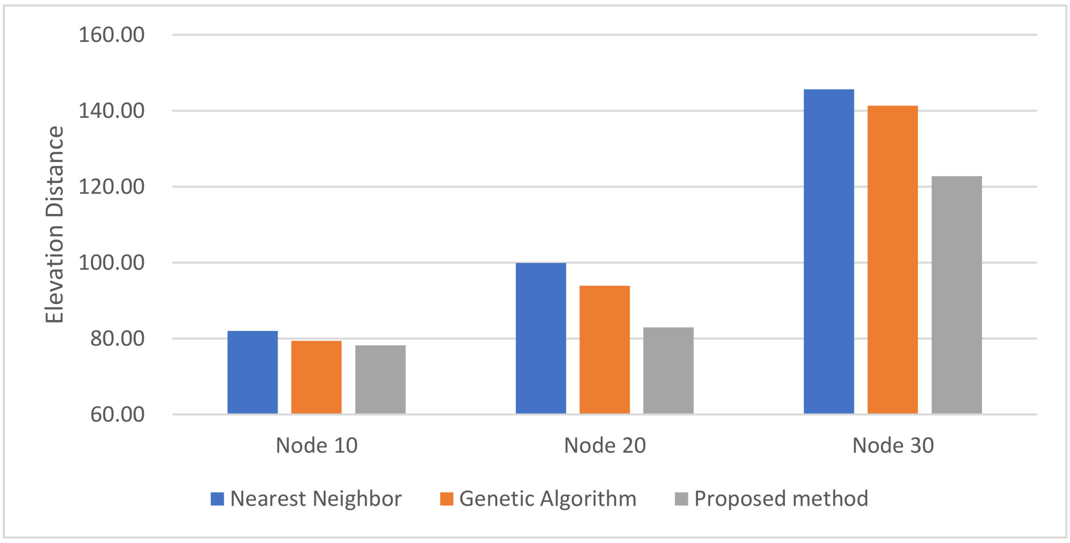


Figure 12. The elevation distance comparison in Antarctic environments.

Regarding the results in Antarctic environments. As shown in Table 2, for all cases, the ACO generated a shorter path, especially when the number of nodes was more than 20, showing better performance. Although NN had a shorter computational time, the proposed method was less than 5 s, which can be considered reasonable if it generates shorter and more stable paths.

5. Conclusions

This paper addresses the problem of practical multi-robot task scheduling in Antarctic environments. We analyzed the difficulties of robot operation in Antarctica and present a solution. For stable robot operation, ACO with a novel cost function including altitude information is proposed. The proposed method creates a path that avoids steep slopes, enabling stable robot operation in Antarctic environments consisting of snow and ice. Furthermore, the comparison of the results with the nearest neighbor algorithm shows that the proposed method generates shorter paths, enabling efficient scheduling. However, as mentioned earlier, there are a number of constraints in the Antarctic environment, such as altitude, snow, ice, crevasses, wind speed, and limited communications. In this paper, only a few constraints are considered. Various factors must be considered for more efficient robot operation. In the future, scheduling will be carried out considering various constraints while including altitude information. Stable and efficient robot operation will be possible by further reflecting the factors of Antarctic environments.

Author Contributions: Conceptualization, H.L.; Methodology, S.K.; Writing—original draft, S.K.; Writing—review & editing, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the project titled 'Research on Co-Operative Mobile Robot System Technology for Polar Region Development and Exploration', funded by the Korean Ministry of Trade, Industry, and Energy (1525011633); in part by the Government-wide R&D Fund for Infections Disease Research (GFID), funded by the Ministry of the Interior and Safety, Republic of Korea (grant number: 20014854); and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (no. 2021R1F1A1064358).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Flood, M.M. The traveling-salesman problem. *Oper. Res.* **1956**, *4*, 61–75. [CrossRef]
2. Croes, G.A. A Method for Solving Traveling-Salesman Problems. *Oper. Res.* **1958**, *6*, 791–812. [CrossRef]
3. Beardwood, J.; Halton, J.H.; Hammersley, J.M. The shortest path through many points. In *Mathematical Proceedings of the Cambridge Philosophical Society*; Cambridge University Press: Cambridge, UK, 1959; pp. 299–327.
4. Matai, R.; Singh, S.P.; Mittal, M.L. Traveling salesman problem: An overview of applications, formulations, and solution approaches. *Travel. Salesm. Probl. Theory Appl.* **2010**, *1*. [CrossRef]
5. Chandra, A.; Naro, A. A Comparative Study of Metaheuristics Methods for Solving Traveling Salesman Problem. *Int. J. Inf. Sci. Technol.* **2022**, *6*, 1–7.
6. Sathya, N.; Muthukumaravel, A. A survey of travelling salesman problem using heuristic search techniques. *Int. J. Innov. Res. Comput. Commun. Eng.* **2016**, *4*, 847–851.
7. Johnson, D.S.; McGeoch, L.A. The traveling salesman problem: A case study in local optimization. *Local Search Com-Binatorial Optim.* **1997**, *1*, 215–310.
8. Golberg, D.E. Genetic algorithms in search, optimization, and machine learning. *Addion Wesley* **1989**, *102*, 36.
9. Mitchell, M. *An Introduction to Genetic Algorithms*; MIT Press: Cambridge, MA, USA, 1998.
10. Al Rahedi, N.T.; Atoum, J. Solving TSP problem using New Operator in Genetic Algorithms. *Am. J. Appl. Sci.* **2009**, *6*, 1586–1590.
11. Dong, X.; Cai, Y. A novel genetic algorithm for large scale colored balanced traveling salesman problem. *Futur. Gener. Comput. Syst.* **2019**, *95*, 727–742. [CrossRef]
12. Gambardella, L.M.; Dorigo, M. Solving symmetric and asymmetric TSPs by ant colonies. In Proceedings of the IEEE International Conference on Evolutionary Computation, Nagoya, Japan, 20–22 May 1996; pp. 622–627.
13. Dorigo, M.; Gambardella, L. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Trans. Evol. Comput.* **1997**, *1*, 53–66. [CrossRef]
14. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **1996**, *26*, 29–41. [CrossRef]
15. Dorigo, M.; Gambardella, L.M. Ant colonies for the travelling salesman problem. *Biosystems* **1997**, *43*, 73–81. [CrossRef]
16. Dorigo, M.; Di Caro, G. Ant colony optimization: A new meta-heuristic. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999; pp. 1470–1477.
17. Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. *IEEE Comput. Intell. Mag.* **2006**, *1*, 28–39. [CrossRef]
18. Peker, M.; Şen, B.; Kumru, P.Y. An efficient solving of the traveling salesman problem: The ant colony system having parameters optimized by the Taguchi method. *Turk. J. Electr. Eng. Comput. Sci.* **2013**, *21*, 2015–2036. [CrossRef]
19. Gülcü, Ş.; Mahi, M.; Baykan, Ö.K.; Kodaz, H. A parallel cooperative hybrid method based on ant colony optimization and 3-Opt algorithm for solving traveling salesman problem. *Soft Comput.* **2018**, *22*, 1669–1685. [CrossRef]
20. Wang, Y.; Han, Z. Ant colony optimization for traveling salesman problem based on parameters optimization. *Appl. Soft Comput.* **2021**, *107*, 107439. [CrossRef]
21. Bellmore, M.; Hong, S. Transformation of Multisalesman Problem to the Standard Traveling Salesman Problem. *J. ACM* **1974**, *21*, 500–504. [CrossRef]
22. Rao, M.R. Technical Note—A Note on the Multiple Traveling Salesmen Problem. *Oper. Res.* **1980**, *28*, 628–632. [CrossRef]
23. Kara, I.; Bektas, T. Integer linear programming formulations of multiple salesman problems and its variations. *Eur. J. Oper. Res.* **2006**, *174*, 1449–1458. [CrossRef]
24. Venkatesh, P.; Singh, A. Two metaheuristic approaches for the multiple traveling salesperson problem. *Appl. Soft Comput.* **2015**, *26*, 74–89. [CrossRef]
25. Jiang, C.; Wan, Z.; Peng, Z. A new efficient hybrid algorithm for large scale multiple traveling salesman problems. *Expert Syst. Appl.* **2019**, *139*, 112867. [CrossRef]
26. Bektas, T. The multiple traveling salesman problem: An overview of formulations and solution procedures. *Omega* **2006**, *34*, 209–219. [CrossRef]

27. Cheikhrouhou, O.; Khoufi, I. A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy. *Comput. Sci. Rev.* **2021**, *40*, 100369. [CrossRef]
28. Zheng, J.; Hong, Y.; Xu, W.; Li, W.; Chen, Y. An effective iterated two-stage heuristic algorithm for the multiple Traveling Salesmen Problem. *Comput. Oper. Res.* **2022**, *143*, 105772. [CrossRef]
29. Ray, L.E.; Lever, J.H.; Streeter, A.D.; Price, A.D. Design and power management of a solar-powered "cool robot" for polar instrument networks. *J. Field Robot.* **2007**, *24*, 581–599. [CrossRef]
30. Lever, J.H.; Delaney, A.J.; E Ray, L.; Trautmann, E.; Barna, L.A.; Burzynski, A.M. Autonomous GPR Surveys using the Polar Rover *Yeti*. *J. Field Robot.* **2012**, *30*, 194–215. [CrossRef]