Maritime Technology and Research 2020; 2(4): 218-230 https://doi.org/10.33175/mtr.2020.240342

Maritime Technology and Research

https://so04.tci-thaijo.org/index.php/MTR

Research Article

Applying A* algorithm in routing for Inter-Terminal Transport System at Busan Port

Long Le Ngoc Bao[1], Duy Anh Nguyen[2,*] and Kim Hwan-Seong[1]

[1]Korea Maritime and Ocean University, Busan, South Korea
[2]Vietnam National University Ho Chi Minh City-Ho Chi Minh University of Technology, Vietnam

Article information

Received: March 11, 2020
Revised: May 4, 2020
Accepted: May 11, 2020

Keywords
Inter-Terminal Transport,
Monorail,
Busan Port,
Dispatching,
Heuristics,
Metaheuristics

Abstract

This paper proposes a heuristic approach named A* algorithm to find the shortest routes for the Inter-Terminal Transport Monorail System (ITT system) that are being applied in Busan Port (Korea). In every transport system, vehicle routing and task assignment are significant problems that strongly affect the overall system's efficiency, especially travel cost and time taken. At present, Busan Port is still using a roadway transportation mode, with manually driven trucks coming in and out of each terminal to load and unload containers, causing casual traffic congestion and wasting lots of time due to waiting at traffic lights. Expenses for fuel and drivers are also challenges for managers. From this, we try to create a new ITT system that could help handle the increasing demand at Busan Port, as well as be operated automatically in order to lower labour and operational costs. The proposed ITT system will use shuttles to carry containers along a monorail that links the internal terminals. A* algorithm is used to guide the shuttles in the shortest way automatically, from a known loading position to a designated unloading one. One of the most crucial problems in this new system is that there must be a method to guide these shuttles automatically and correctly, since the departure and destination points need to be recognized in a pre-defined layout to transfer containers. There are many terminals, resulting in many ways to reach the goal, and shuttles should pick the best way to optimize performance. In the first part of the paper, we will briefly describe briefly the ITT system that being considered in Korea. Next, we will explain why and how we implement A* algorithm in dispatching. Finally, we will give some comparisons between the performance of the new ITT system and the traditional transport system through simulation in MATLAB. The obtained results would prove the superior advantages of the new ITT system versus the traditional system (expressed through the complete time spent at each terminal), as well as emphasize the correctness of A* algorithm in the dispatching problem, as it helps substantially reduce the computational cost.

[*]Corresponding author: Vietnam National University Ho Chi Minh City-Ho Chi Minh University of Technology, Vietnam
E-mail address: duyanhnguyen@hcmut.edu.vn

1. Introduction

ITT system is a system that links the components of a container port (depots, container yards, terminals, etc.) with corresponding transport modes (Duinkerken et al., 2006; Rodrigue at al., 2017; Lee et al., 2012; Zhen et al., 2016) so that it can exploit the capability as much as possible. Major ports usually have many different types of terminal (sea terminal, barge terminal, railway terminal, etc.) with many different transport modalities (road, rail, sea, etc.) (Heilig & Voß, 2017). Due to the huge demand, containers need to be exchanged and moved between these terminals to distribute the workload properly. The key role of a general ITT system can be summarized as shown in **Figure 1** (Heilig & Voß, 2017).

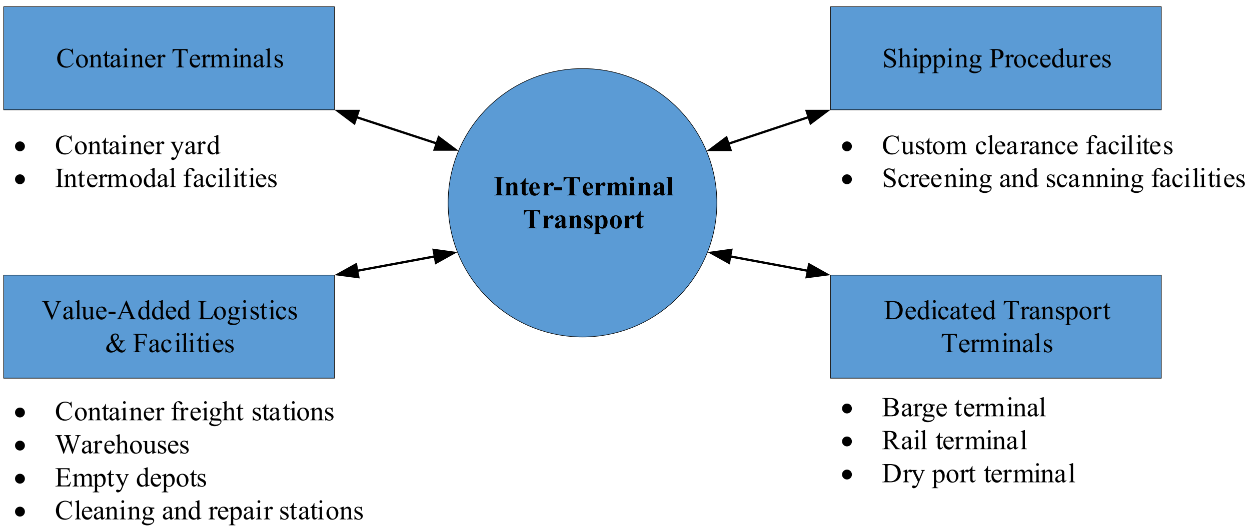

Figure 1 General ITT explanation.

In our situation, we are considering a monorail ITT system with shuttles moving across. The word *monorail* means that shuttles can only move in one direction at a time. In case they need to turn back, there are some special stations along the rail called *change stations* that could change the moving lane into opposite lanes. The change stations are presented in **Figure 2** (Kim, 2019). The overview layout of the monorail system is demonstrated by MATLAB R2018, as shown in **Figure 3**. MATLAB is a calculation software by MathWork that is commonly used by engineering students in technology universities, and is specialized in computing, analysing, and simulating data. In addition, MATLAB allows users to create a user-interface for visualising results. As a part of the project, managers need an interactive program that can simulate the performance of the monorail system with flexible inputs, as the project is currently at the design stage, where all parameters are adjustable.

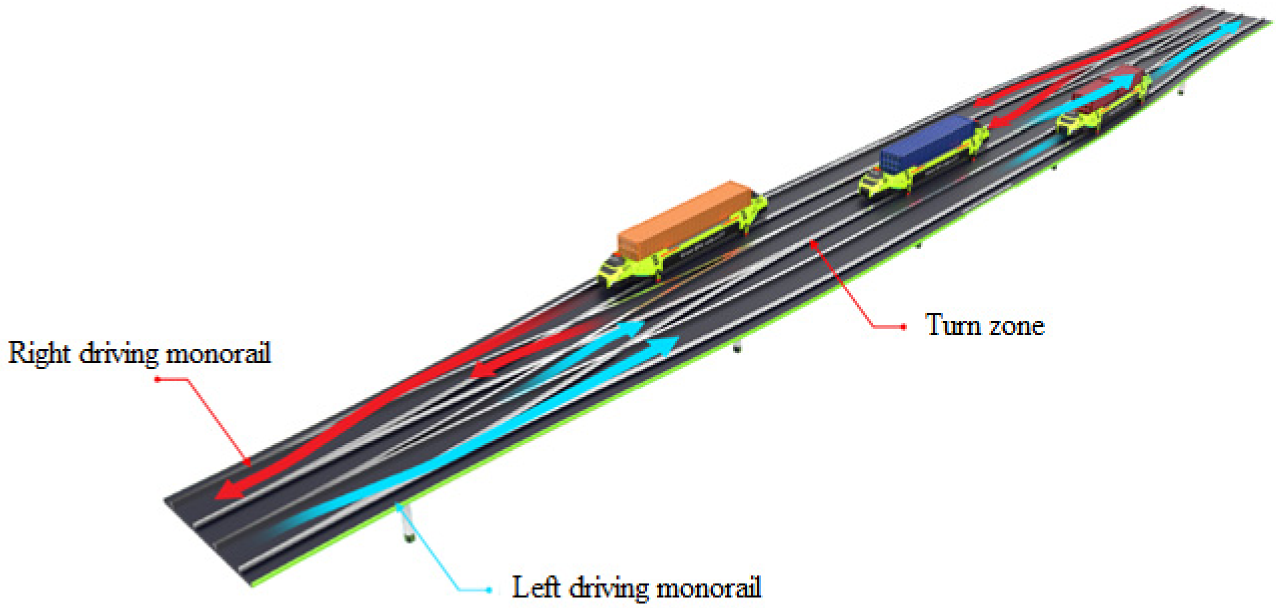

Figure 2 Shuttle operation at change stations.

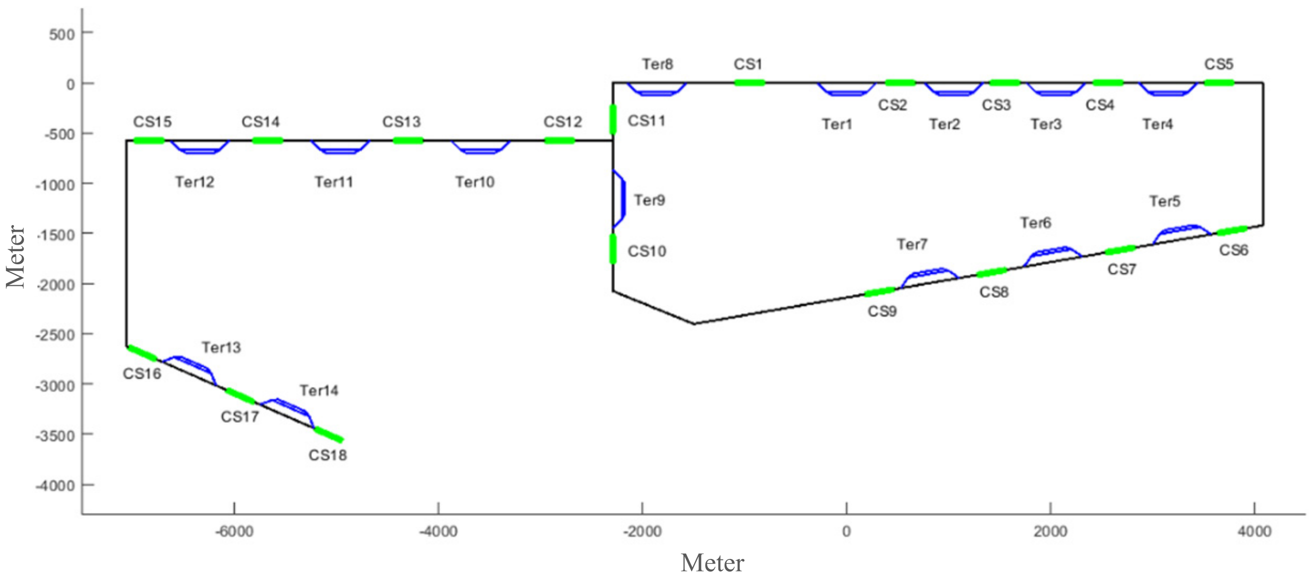

Figure 3 Layout for ITT monorail system.

From **Figure 3**, it is clear that there are 14 terminals in total, with many change stations lying between them to help direct the shuttles. Moreover, there is a closed loop on the layout, meaning that there will always exist 2 possible ways to go from 1 terminal to another. The difference between 2 choices may cost up to 10 km of distance travel, which means that some paths would waste time and energy. From this point, the dispatching problem plays a significant role in reducing travel costs. This has motivated us to solve the dispatching problem in an optimal way. In the next section, we will discuss A* algorithm, as well as some other common metaheuristic methods that will be implemented later, and give a brief evaluation of them.

2. Routing problem

2.1 Methodology

The routing problem is essential for every logistical operation. As shown in **Figure 3**, the layout has 14 terminals and 18 change stations. Each terminal will have a change station at the export gate and a change station at the import gate. Therefore, this leads to a simpler problem; that we will find a path from the change station of a departure terminal to the change station of a destination terminal instead. Moreover, we can use other change stations as checkpoints for the path because this would help in tracking the movement of shuttles through the change stations and measuring the traffic flow at these points laterally. Terminals and corresponding change stations are listed in **Table 1** as below:

Table 1 List of terminals and corresponding change stations.

Terminal	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Departure CS	2	3	4	5	7	8	9	1	11	12	13	14	16	17
Destination CS	1	2	3	4	6	7	8	11	10	13	14	15	17	18

Considering each change station as a specific node in a predefined graph, the routing problem now turns into finding the shortest way that links 2 nodes. Because every node could only be connected with its adjacent nodes, this graph can be considered as a weighted graph, where the unconnected nodes can be marked "out of reach". In the following subsections, we will discuss the common approaches that can be used to solve this problem: *A* algorithm*, *GRASP*, and *ACO*.

2.2 A* algorithm

A* algorithm is one of the most common *heuristic* approaches for solving path-finding problems (Yao et al., 2009; Shi et al., 2008; Edmond, 2005). This algorithm is widely applied for mobile robots, as it can be implemented in real-time operations. In our context, the implementation of A* algorithm can be explained via the flowchart in **Figure 4**.

The general formula for cost function f of a node n_0 is described as below (Yao et al., 2009):

$$f(n_0) = g(n_0) + h(n_0) \tag{1}$$

where:

- $g(n_0)$ is chosen as the Euclidean distance from node n_0 to current node n with weight $w(n,n_0)$:

$$g(n_0) = w(n,n_0) \times DistanceMatrix(n,n_0) \tag{2}$$

$$w(n,n_0) = \begin{cases} 1, n_0 \in Adjacent\_set(n) \\ \infty, otherwise \end{cases} \tag{3}$$

- $h(n_0)$ is the heuristic function that is usually chosen as the Euclidean distance from node n_0 to destination node n_2:

$$h(n_0) = DistanceMatrix(n_0,n_2) \tag{4}$$

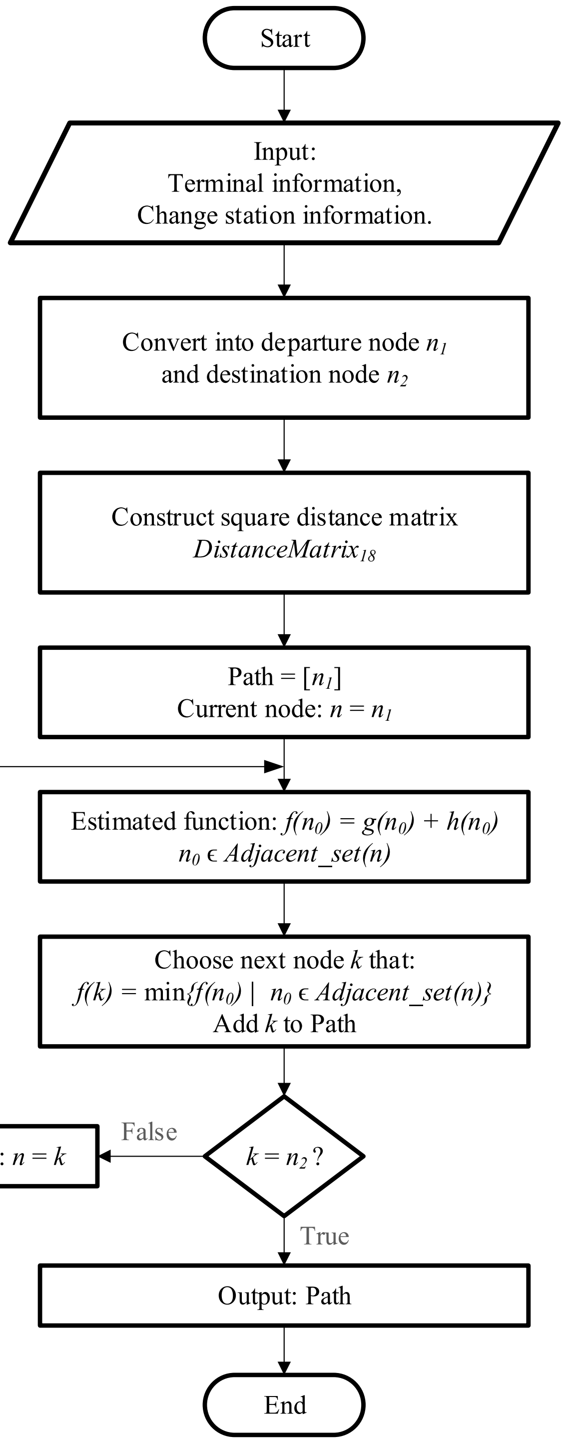

Figure 4 Flowchart of A* algorithm implementation.

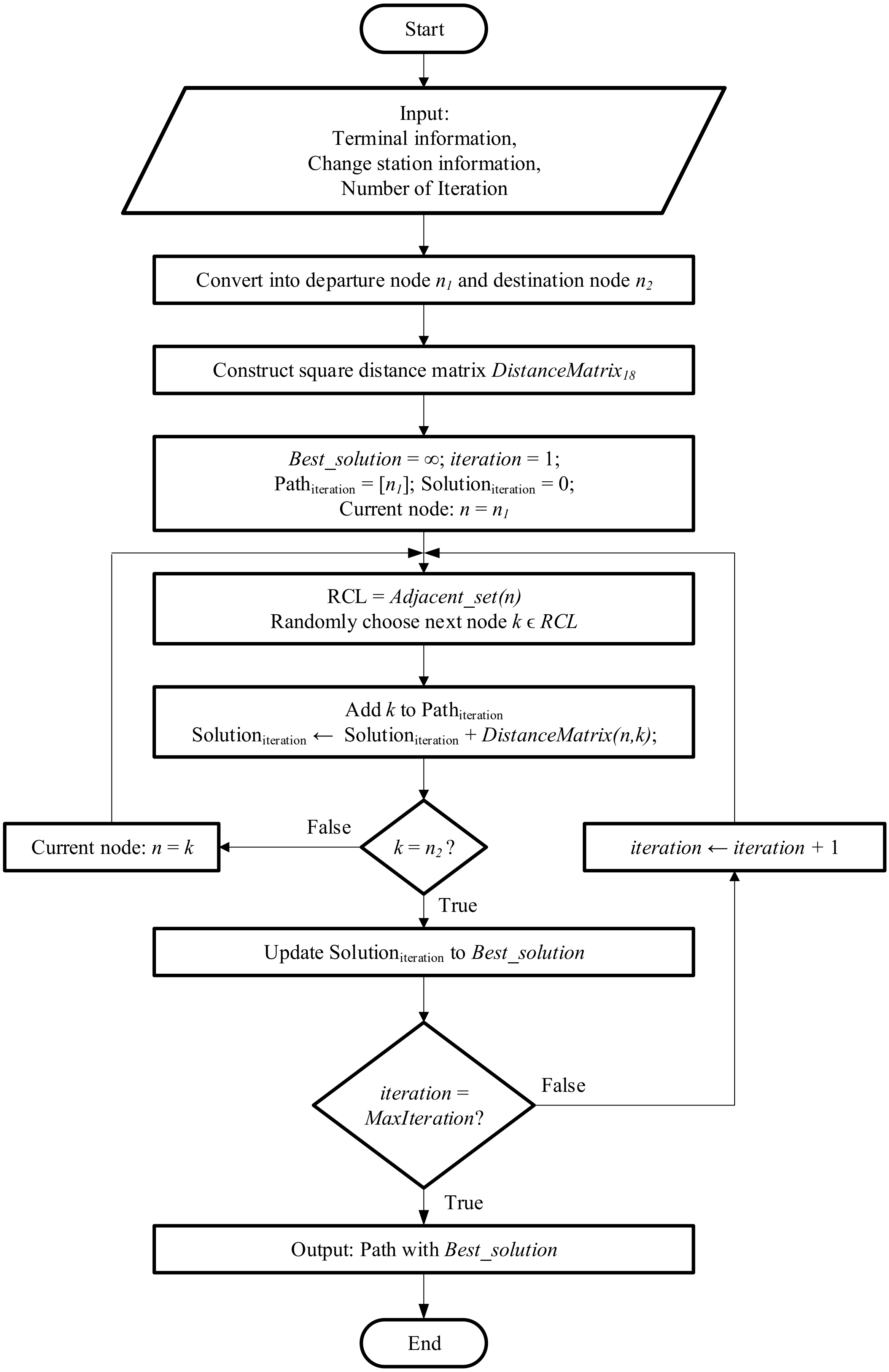

Figure 5 Flowchart of GRASP implementation.

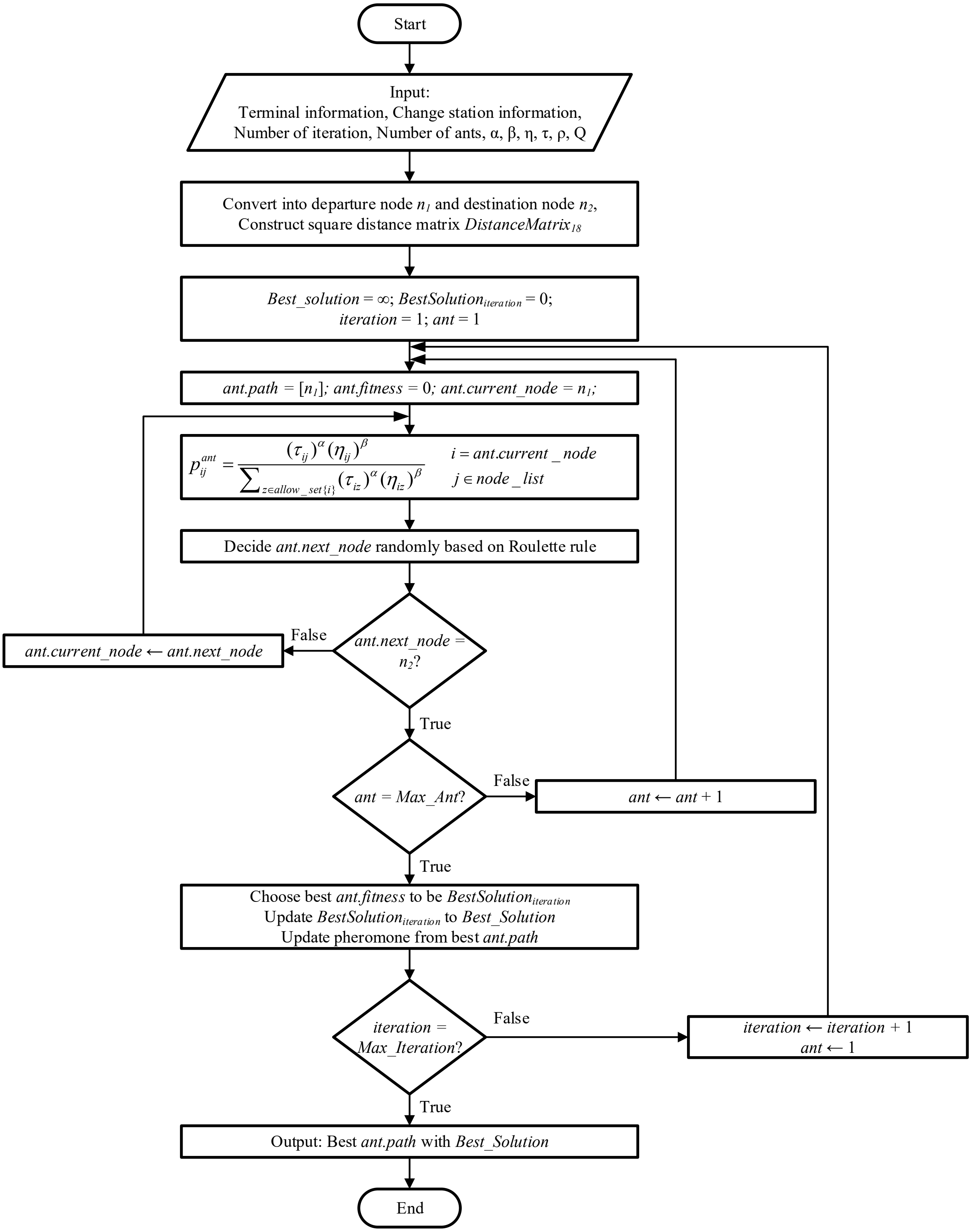

Figure 6 Flowchart of ACO implementation.

2.3 GRASP

Greedy Randomized Adaptive Search Procedure (GRASP) is a basic *metaheuristic* that is based on Greedy Algorithm (Feo & Resende, 1995; Abdesslem et al., 2013). GRASP is commonly used in path-finding problems, especially in linking nodes in a weighted graph. In concept, this algorithm looks for many local solutions similar to how Greedy Algorithm acts, and then picks out the best one as a final solution. According to our problem, the behaviour of GRASP can be summarized as illustrated in **Figure 5**.

In general, the Restricted Candidate List (RCL) is often formed by a set of unvisited nodes that satisfies the equation below:

$$RCL = \{n \mid f(n) \in [f_{\min}, f_{\min} + \alpha(f_{\max} - f_{\min})]\} \tag{5}$$

where f_{max} and f_{min} are maximum and minimum values of cost function f for the set of unvisited nodes in the graph, and $\alpha \in [0, 1]$ is a factor that modifies the range of RCL.

In computation, this α can determine how strict the RCL is. If $\alpha = 1$, then GRASP turns into a fully random search, while $\alpha = 0$ makes GRASP exactly as the basic Greedy Algorithm.

However, in our case, the process can be made easier by deciding how many desired elements are to be put in the RCL. Since our graph is a closed-loop form, we can simply choose the RCL as the 2 nearest nodes of the current node n. To say this in another way, these are 2 nodes that lay on 2 sides of the current node. The procedure for setting the RCL can be summarized as below:

- For every node n, make a list of unvisited nodes S
- Calculate the distance from n to all nodes in S
- Sort the distance in an ascending direction
- Pick two first nodes in the sorted array to form the RCL

2.4 ACO

Ant Colony Optimization (ACO) is a nature-based *metaheuristic* that mimics how ants find food (Dorigo & Stützle, 2004; Stützle & Dorigo, 2002; Dorigo et al., 1996; Long et al., 2018). In nature, ants tend to look for food in the shortest way and then leave trails using *pheromones*. Other ants are attracted to the pheromones and follow the trail, marking the trail as the best way. The general principle of ACO is demonstrated in **Figure 6**. For the formula in **Figure 6**, τ is the pheromone value representing the ant's traces; η is a heuristic value representing the ant's prediction to the goal, which is often chosen similarly to A* algorithm; α is the pheromone exponent, and β is the heuristic exponent. The value of α and β shows the extent of how current information and predictive information affect the probability of finding the next nodes. To simplify, α and β are usually chosen as $\alpha = \beta = 1$. The pheromones are updated on each edge *i-j* after each iteration is formulated, as below (Dorigo & Stützle, 2004; Dorigo et al., 1996):

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \frac{Q}{L^{ant}} \tag{6}$$

where ρ is the evaporation factor, L^{ant} is the distance the best *ant* travelled through edge *i-j*, and Q is a constant.

In fact, Eq. (6) can be separated into 2 components: a local update Q/L^{ant} for the best *ant* with local update factor Q, and a global update $(1 - \rho)\tau$ for all ants in the whole iteration.

2.5 Evaluation

In this subsection, we will make comparisons between the 3 approaches mentioned above. With a pair of departure node and destination node, we will evaluate the solution found, as well as the time taken. First of all, we will reproduce the layout as a weighted graph, where change stations are represented by nodes, as shown in **Figure 7**. Note that, to make the path more realistic and precise, we add some linking nodes. All nodes displayed are numbered for convenience in computation.

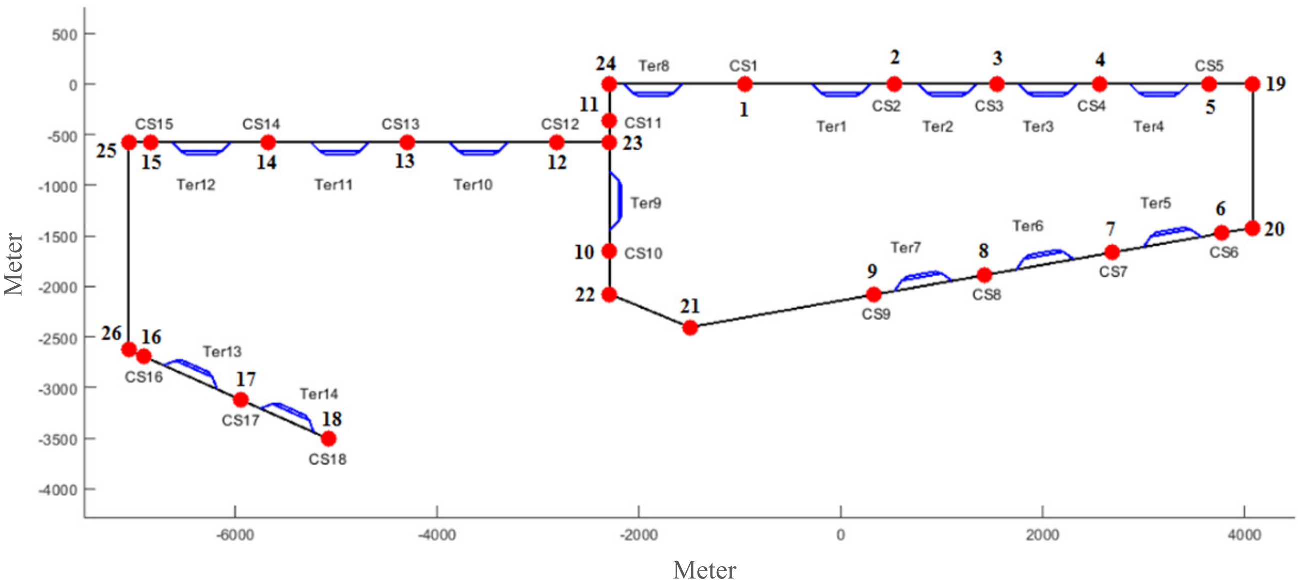

Figure 7 Weighted graph with numbered nodes.

Then, we will analyse and evaluate how each approach works. As described in the previous subsection, we will test the searches among 18 change stations or, in other words, 18 nodes in the weighted graph in **Figure 7**. For each trial, we measure the time taken for all cases (18×18= 324 cases in total) and then calculate the average time. The time record after 10 trials and the final average time for each approach are listed in **Table 2**.

Table 2 Time measure when applying 3 approaches.

Time taken	Applying A* algorithm	Applying GRASP	Applying ACO
1st time	0.1297×10^{-3} s	0.0710 s	0.5325 s
2nd time	0.1227×10^{-3} s	0.0702 s	0.5340 s
3rd time	0.1184×10^{-3} s	0.0693 s	0.5365 s
4th time	0.1269×10^{-3} s	0.0706 s	0.5291 s
5th time	0.1105×10^{-3} s	0.0703 s	0.5353 s
6th time	0.1212×10^{-3} s	0.0690 s	0.5310 s
7th time	0.1044×10^{-3} s	0.0692 s	0.5259 s
8th time	0.1055×10^{-3} s	0.0700 s	0.5373 s
9th time	0.0698×10^{-3} s	0.0693 s	0.5298 s
10th time	0.1216×10^{-3} s	0.0696 s	0.5323 s
Average time for an approach	0.1131×10^{-3} s	0.0699 s	0.5324 s

From **Table 2**, it is clear that there is a huge difference in the time taken between A* algorithm and other approaches. The main reason for this is that A* algorithm is a heuristic that ensures a local optimum, while GRASP and ACO are metaheuristics that attempt to expand the search range for global optima. However, in our situation, it is unnecessary for a global solution, because this takes much longer. In consequence, we manage to use A* algorithm for further computation in this paper.

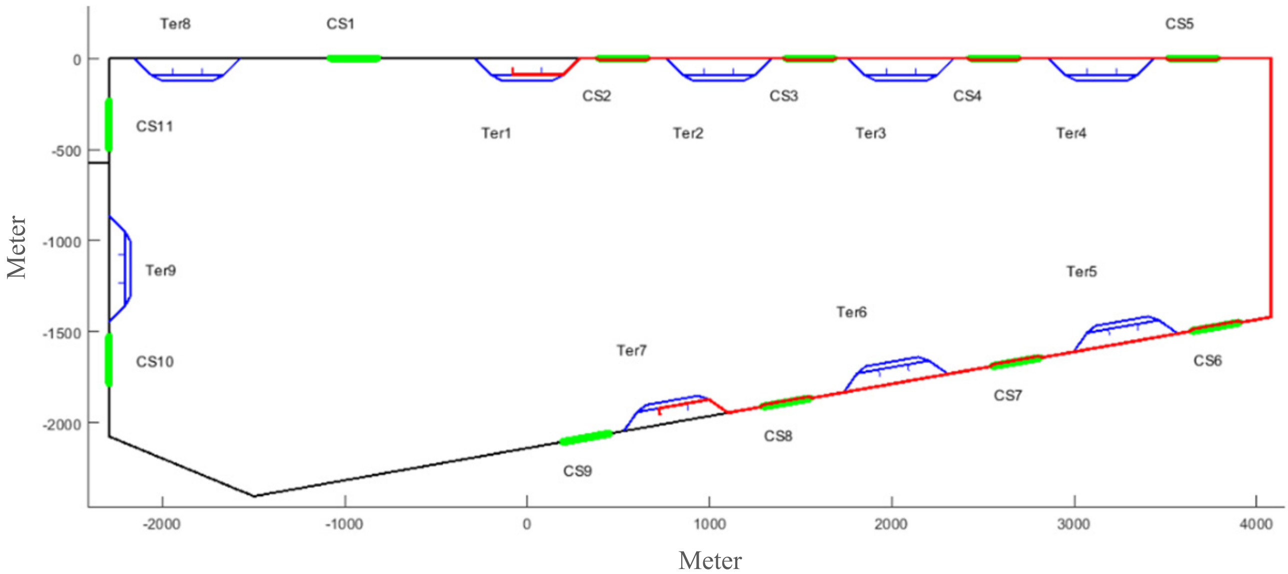

Figure 8 Example of using A* algorithm to dispatch from a loading position to an unloading position.

After successfully build a path across the change stations, we can easily extend it to corresponding terminals and loading-unloading positions in the layout. **Figure 8** shows an example using A* algorithm of a route for a shuttle from loading position 1 of Terminal 1 to unloading position 2 of Terminal 7, with the solved path highlighted. The total length of the path found is 9.1626×10^{-3} m.

3. Appliance to the ITT Monorail System

After showing the possibility of A* algorithm in routing shuttles, in this section, we will implement it in a realistic system, which is under construction in Busan Port. As mentioned above, the ITT Monorail System will connect 14 terminals of Busan Port by a monorail, in which shuttles move along to carry containers. To compare this with traditional transport modality, this system is supposed to improve the efficiency of overall operation by significantly reducing the time of travel between terminals and stations, as well as traffic congestion when using road trucks.

3.1 Assumptions

Before simulation progress, we make some assumptions based on realistic working conditions as below:

- The simulation program aims to reveal the effectiveness of the new ITT Monorail System in comparison with a traditional transportation system.
- Time specification, such as time at change stations, handling time, moving-in time, and moving-out time are provided based on experiments.
- Each terminal may have, at most, 4 loaders that handle the loading-unloading operation.
- Each terminal has a number of shuttles parking at loading positions at the beginning. After finishing the assigned tasks, shuttles rest at the current positions where they last reached.

- Each loader only handles one container at a time.
- Each shuttle only carries one container at a time. If a shuttle is available, it will look for a requested container until all requests have been completed.

3.2 Implementation

We have built a user-interface program in MATLAB to simulate and to analyse the operation of the system. The interface allows user to flexibly configure the input parameters, such as number of containers, number of loaders and shuttles at each terminal, shuttle velocity, truck velocity, handling time, etc. The appearance of the interface after setting up will appear as per **Figure 9**. Note that all parameters are proposed by a project manager and can be modified in later stages.

After filling in the necessary information, the user can press the "Process" button to start the calculation. The status bar will automatically update when completed. The simulation result will then be animated, as displayed in **Figure 10**. In addition, the efficiency of each terminal is also presented, as shown in **Figure 11**. The measured data is evaluated in a duration of 1 h.

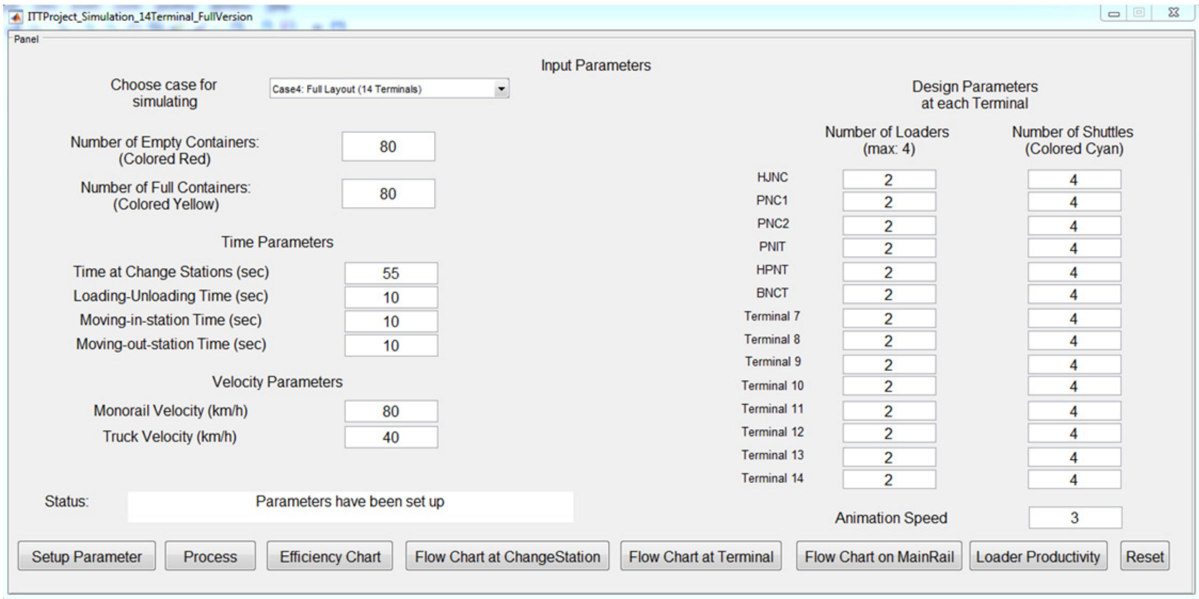

Figure 9 Interface of program after setting up parameters.

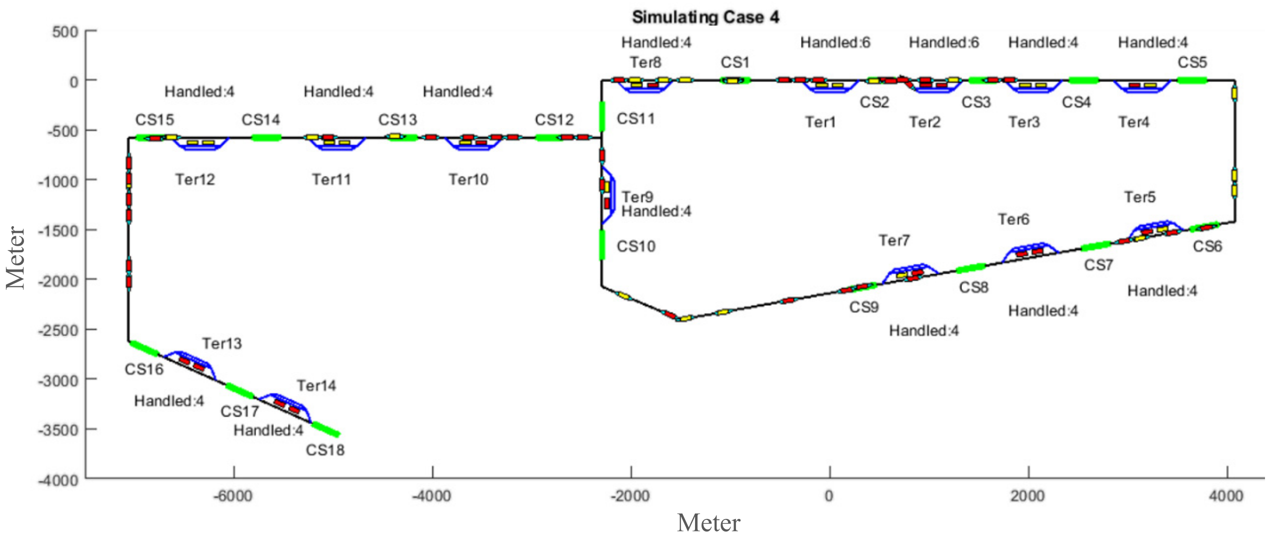

Figure 10 Animated movement of shuttles.

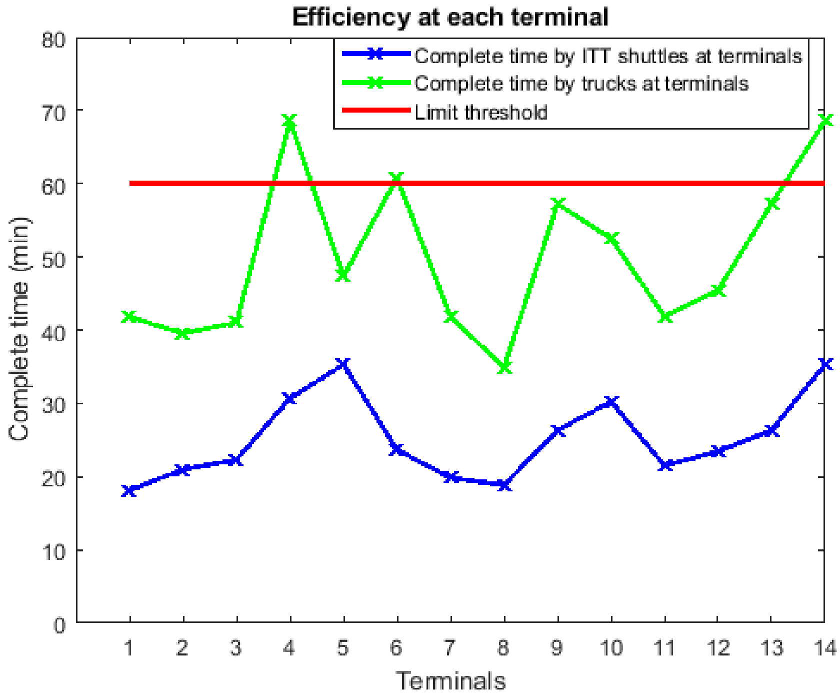

Figure 11 Time taken to complete all requested tasks in deadline of 60 min.

From these figures, it is concluded that the new ITT Monorail System has increased in overall performance so far, compared to a traditional transport system. In reality, all assigned tasks are planned to be accomplished in 60 min, which means vehicle speed and loader productivity also have to reach a limitation to be qualified.

Furthermore, in our simulation, the number of containers is not large in comparison with real demands. Usually, Busan Port has to transfer about 30.000 - 40.000 containers a day, which is approximately 20 times greater than in the simulation. However, all equipment used in the project is also calculated and is proportional with the transport demand, such as number of shuttles and loaders at each terminal. In case the demand rises, the facility also has to be equivalent, to ensure the capability of the system, so the overall efficiency (**Figure 11**) seems not to change much.

4. Conclusions

The new ITT Monorail System has solved the problem of transport deadlines, especially at a major container port such as Busan Port. The workload is extreme and is not allowed to be delayed, so the system requires a proper method to ensure sustainability in operation. With support from A* algorithm, the shuttles are dispatched in an optimal way to get to the task position as fast as possible, which would lower the travel cost dramatically.

However, in this paper, the assignment problem has still not been considered yet. At the moment, the shuttles will be assigned to the nearest available tasks, using the same concepts as A* algorithm, while there might be some tasks with long delays. As a consequence, it would expand the working duration, and also the operational cost. It is supposed that the assignment problem is a combinatorial optimization problem and needs special methods in order to seek the best solution. Currently, the authors are trying to deal with this problem. A suggestion is to consider the assignment problem as a path-finding problem, where the possible orders are likened to nodes in a graph; then, we could apply heuristic and metaheuristic methods to sort the tasks given in an appropriate sequence.

In future, we will make research and solve the assignment problem, in order to fulfil the study about the system's operation and figure out the global optima that could further increase the system's performance.

References

Abdesslem, L., Meryem, A., & Salim, C. (2013). A GRASP algorithm based on new randomized heuristic for vehicle routing problem. *Journal of Computing and Information Technology*, *21*(1), 35-46. doi:10.2498/cit.1002085

Dorigo, M., & Stützle, T. (2004). *The ant colony optimization metaheuristic* (pp. 25-64). MIT Press eBook, USA.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems Man and Cybernetics Part B: Cybernetics, 26*(1), 29-41. doi:10.1109/3477.484436

Duinkerken, M. B., Dekker, R., Kurstjens, S. T. G. L., Ottjes, J. A., & Dellaert, N. P. (2006). Comparing transportation systems for inter-terminal transport at the Maasvlakte container terminals. *OR Spectrum, 28*(4), 469-493. doi:10.1007/s00291-006-0056-1

Edmond, C. (2005). *A graph search heuristic for shortest distance paths*. Center for Applied Scientific Computing, Lawrence Livermore National Laboratory.

Feo, T. A., & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization, 6*, 109-133. doi:10.1007/BF01096763

Heilig, L., & Voss, S. (2017). Inter-terminal transportation: An annotated bibliography and research agenda. *Flexible Services and Manufacturing Journal, 29*(1), 35-63. doi:10.1007/s10696-016-9237-7

Kim, H. S. (2019). *Busan Port New Port ITT Infrastructure Project* (*in Korea*). Project Document, Korea Maritime and Ocean University.

Lee, D. H., Jin, J. G., & Chen, J. H. (2012). Terminal and yard allocation problem for a container transshipment hub with multiple terminals. *Transportation Research Part E: Logistics and Transportation Review, 48*(2), 516-528. doi:10.1016/j.tre.2011.09.004

Long, L. N. B., Duy-Anh, N., & Hanh, L. D. (2018). Application of combinatorial optimization in logistics. In: Proceedings of the 2018 4th International Conference on Green Technology and Sustainable Development. HCMC University of Technology and Education, Vietnam.

Rodrigue, J. P., Comtois, C., & Slack, B. (2017). *Transportation Modes: The Geography of Transport Systems.* 4th eds. London: Taylor & Francis.

Shi, J., Li, J., Cao, H., & Wang, X. (2008). Study on near-optimal path finding strategies in a road network. *Journal of Algorithms & Computational Technology, 2*(3), 319-333. doi:10.1260/174830108785302878

Stützle, T., & Dorigo, M. (2002). A short convergence proof for a class of ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation, 6*(4), 358-365. doi:10.1109/TEVC.2002.802444

Yao, J., Zhang, B., & Zhou, Q. (2009). *The optimization of A* algorithm in the practical path finding application* (pp. 514-518). In: Proceedings of the 2009 WRI World Congress on Software Engineering. Xiamen, China.

Zhen, L., Wang, S., & Wang, K. (2016). Terminal allocation problem in a transshipment hub considering bunker consumption. *Naval Research Logistics, 63*(7), 529-548. doi:10.1002/nav.21717