

## RESEARCH ARTICLE



### OPEN ACCESS

**Received:** 08-07-2023

**Accepted:** 30-09-2023

**Published:** 27-10-2023

**Citation:** Sakthivel A, Akaash B, Akshaya AM, Raja UA, Raj MRA, Karthic V (2023) Replacement of Defected Battle Tankers in Military. Indian Journal of Science and Technology 16(40): 3522-3529. <https://doi.org/10.17485/IJST/v16i40.1696>

\* **Corresponding author.**

[asakthivel75@yahoo.com](mailto:asakthivel75@yahoo.com)

**Funding:** None

**Competing Interests:** None

**Copyright:** © 2023 Sakthivel et al. This is an open access article distributed under the terms of the [Creative Commons Attribution License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment (iSee)

**ISSN**

Print: 0974-6846

Electronic: 0974-5645

# Replacement of Defected Battle Tankers in Military

A Sakthivel<sup>1\*</sup>, B Akaash<sup>2</sup>, A M Akshaya<sup>2</sup>, U Arungiri Raja<sup>2</sup>, M R Akhil Raj<sup>2</sup>, V Karthic<sup>2</sup>

<sup>1</sup> Professor, Department of Computer Science and Engineering, KGISL Institute of Engineering and Technology, Coimbatore, India

<sup>2</sup> B.E Student, Department of Computer Science and Engineering, Sri Shakthi Institute of Engineering and Technology, Coimbatore, India

## Abstract

**Objective:** The main purpose is to replace the defected battle tankers in the military at the time of war with the nearby battle tanker for the purpose of security and at the same time to replace the repaired tankers with the nearby tanker and solving the problem occurred in the tanker through the specialist.

**Methods:** An application that stands as an intermediate between the system controller and the user for the purpose of Route Planning and Analysis and Communication and Coordination. **Findings:** It is found that due to lack of communication and coordination between the tanker networks in the military at the time of war the security of the place is been threatened. To overcome this an application is been found that is stands as an intermediate between the tanker networks for the effective communication and coordination between them and the overall controller of the system. **Novelty:** Shortest path algorithms have been used to find the path between the tanker networks for replacement in the application. Thus, the defected and repaired tanker are communicated and the problem have been solved through the application.

**Keywords:** Shortest path algorithm; Network; Communication; Route planning; Route analysis

## 1 Introduction

Modern tanks are ground-based electronic weapons with a wide range of weapons, the main armament being a large caliber tank gun mounted in a rotating turret, supplemented by other long-range weapons such as machine guns or anti-tank missiles, rocket, launcher. They are equipped with heavy transport equipment, transport magazines, fuel tanks and propulsion systems to protect the crew. The use of tracks instead of wheels enhances combat as the tank can navigate mud and adverse conditions such as mud and ice better than rival vehicles, making it easier in combat situations.

As they play a vital role in the military. The security measures for tankers at the time of war during the transportation of fuel or heavy-weighted defence vehicles are most important. These four main security measures involved in protecting the tankers at the time of war are Surveillance and Reconnaissance, Route Planning and Analysis<sup>(1-3)</sup>, Communication and coordination<sup>(4,5)</sup> and Contingency plans. In

modern military terminology, it is necessary to develop more new technologies and software applications that must be highly secured for use in the military to be more protective and to strengthen the armed forces.

Graph theory is a branch of mathematics that deals with the study of graphs, which are mathematical structures used to represent relationships between objects. In a graph, objects are represented as points (called vertices or nodes), and relationships between those objects are represented as lines (called edges or arcs) that connect the points. Graph theory has applications in various fields, including computer science, network analysis, social sciences, transportation planning, and others. It is used to solve problems related to connectivity, optimization and studying relationships in different systems. For example, it is the basis of algorithms used in finding the shortest path in GPS navigation systems, social network analysis, and designing efficient computer networks.

There are several common representations of graphs, including adjacency matrix, adjacency list, and edge list, to analyse tanker network parameters<sup>(6)</sup>, which contain the number of tankers (nodes) and their distances (edges). Graphs can be traversed and analysed using various algorithms. Some of the most common graph algorithms include depth-first search (DFS), breadth-first search (BFS), Dijkstra's algorithm<sup>(7,8)</sup> for finding the shortest path<sup>(9–11)</sup>, and Kruskal's algorithm for finding the minimum spanning tree. Graph theory provides a rich set of concepts and techniques for analysing and solving graph-related problems. Overall, graphs are a versatile and powerful data structure that plays a crucial role in solving a wide range of problems in various domains.

The following graph algorithms are used to implement most of the graph applications: Breadth First Search (BFS), Depth First Search (DFS), Dijkstra's Algorithm, Prim's Algorithm, Kruskal's algorithm, and Floyd's algorithm<sup>(12)</sup>.

Prim's algorithm is a greedy algorithm that is used to find the minimum spanning tree from a graph. Prim's algorithm finds the subset of edges that includes every vertex of the graph such that the sum of the weights of the edges can be minimised. Kruskal's algorithm is used to find the minimum spanning tree for a connected weighted graph. The main target of the algorithm is to find the subset of edges using which it can traverse every vertex of the graph. It follows the greedy approach that finds an optimum solution at every stage instead of focusing on a global optimum.

Dijkstra's algorithm is a single-source shortest path algorithm. Here, single-source means that only one source is given, and it is used to find the shortest path from the source to all the nodes. Floyd's Algorithm is for solving all pairs of shortest-path problems. The problem is to find the shortest distances between every pair of vertices in a given edge-weighted directed Graph. It is an algorithm for finding the shortest path between all the pairs of vertices in a weighted graph. This algorithm follows the dynamic programming approach to find the shortest path<sup>(12)</sup>.

Breadth-first search is a graph traversal algorithm that starts at the root node and explores all the neighbouring nodes. Then, it selects the nearest node and explores all the unexplored nodes. While using BFS for traversal, any node in the graph can be considered the root node. Depth-first search is a recursive algorithm to search all the vertices of a tree data structure or a graph. The depth-first search (DFS) algorithm starts with the initial node of graph G and goes deeper until it finds the goal node or the node with no children.

The main aim is to develop a software application that acts as an intermediary between the controller and the military tankers. In the first step, the controller is an overall controller of all the attributes, such as tanker, specialist, and location of the tanker. When a problem occurs in the tanker, the head Commander of that tanker sends a notification to the controller about the problem. The Controller immediately finds the type of problem that occurred and rectifies it with the help of the other attributes mentioned. If the tanker is damaged, then it must be completely replaced with the nearby tanker, which is known by calculating the shortest path<sup>(9–11)</sup> from the source tanker. Otherwise, if the tanker is in repairable condition, the controller identifies the type of repair that occurred, calls the specialist for that type of repair, and rectifies the problem. This is an overview of how an application works. In this model, all important security measures like Surveillance and Reconnaissance, Route Planning and Analysis<sup>(1–3)</sup>, Communication and coordination<sup>(4,5)</sup>, and Contingency plans have been considered.

## 2 Methodology

The model consists of the basics of graph for tanker network where each tanker network attribute is compared with graph attributes and the prototype model for the tanker network have been determined.

### 2.1 Basics of Graph For Tanker Network

The tankers in the tanker network are known as nodes in the tanker network. The connections between tankers in the network are known as edges. A weight can be assigned to an edge, representing the cost or distance between two tankers in the network. The degree of a tanker is the number of connections that connect to it in a tanker network. In a directed graph, the in-degree of a tanker is the number of connections that point to it, and the out-degree is the number of connections that start from it. A path

**Table 1.** Comparison among the Different Types of the Shortest Path Algorithm

Algorithm		Prims Algorithm	Kruskal's Algorithm	Dijkstra's Algorithm	Floyd's Algorithm	BFS	DFS
Notation		If (!selected[j] && G[i][j]) { if (min > G[i][j]) {min = G[i][j]; x = i; y = j; } }	if find-Set(u) $\neq$ find-Set(v): A = A $\cup$ {(u, v)} union(u, v) return A	dist[u] = dist[v] + Graph.Edges[v,u].distance[i][k] + dist[u]	distance[i][j] = min(distance[i][j], distance[i][k] + distance[k][j])	If(!nodes[a]) {nodes[a] = true; que.add(a);}	if (!visited[n]) DFS(n); } visited[vertex] = true
Space Complexity		$O( V +E)$	$O( V +E)$	$O( V +E)$	$O( V ^2)$	$O(V)$	$O(V)$
Time Complexity	Best	$O( V + E \log V )$	$O( E \log V )$	$O( V + E \log V )$	$O( V ^3)$	$O(1)$	$O(1)$
	Average	$O( E \log V )$	$O( E \log E )$	$O(( V + E )\log V )$	$O( V ^3)$	$O(n)$	$O(V+E)$
	Worst	$O( E \log V )$	$O( E \log E )$	$O(( V + E )\log V )$	$O( V ^3)$	$O(n^2)$	$O(V+E)$
Algorithm Features	Input	Prim's algorithm prefer list data structures.	Kruskal's algorithm prefer heap data structures	It takes a single source vertex as an input.	It takes all pairs as input.	Queue data structure is used for the BFS traversal	Stack data structure is used for the DFS traversal.
	Process	Prim's algorithm runs faster in dense graphs.	Kruskal's algorithm runs faster in sparse graphs.	It takes a single source vertex as an input.	It takes all pair shortest path and computes shortest path between all pair of nodes.	BFS is slower than DFS.	DFS is faster than BFS.
	Output	It generates the minimum spanning tree starting from the root vertex.	It generates the minimum spanning tree starting from the least weighted edge.	It takes all pair shortest path and computes shortest path between all pair of nodes.	It can be implemented in distributed systems, so output can be found.	It is not memory efficient as it requires more memory than DFS.	It is memory efficient as it requires less memory than BFS.

is a sequence of tanks that are connected by connections. A cycle is a path that starts and ends at the same tanker. A network is said to be connected if there is a path between any two tankers. A disconnected network is a network that is not connected. A network is said to be planar if it can be drawn on a plane without any A network is said to be bipartite if its tankers can be divided into two disjoint sets such that no two tankers in the same set are connected by a connection.

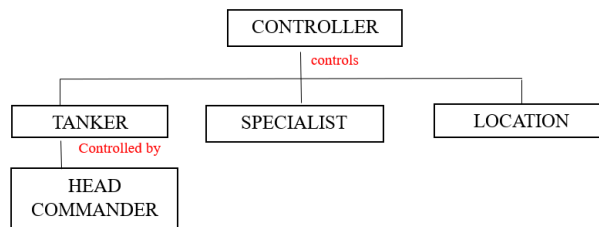
## 2.2 Prototype Model

The prototype model contains the following attributes: controller, specialist, tanker, and location, as shown in Figure 1.

**Controller:** The controller acts as an overall controller for the system. The controller controls the following entities: the tanker, the specialist, and the location of the tanker. **Tanker:** The tanker is controlled by the head commander, who is the head of the tanker. Each action that takes place in the tanker entity is controlled by the controller. **Specialist:** The specialist is controlled by the controller entity. There will be several specialists who perform the work specified by the controller when the tanker is in trouble. **Location:** Location is also controlled by the controller entity. It specifies the location of the tanker.

## 2.3 Suggested System

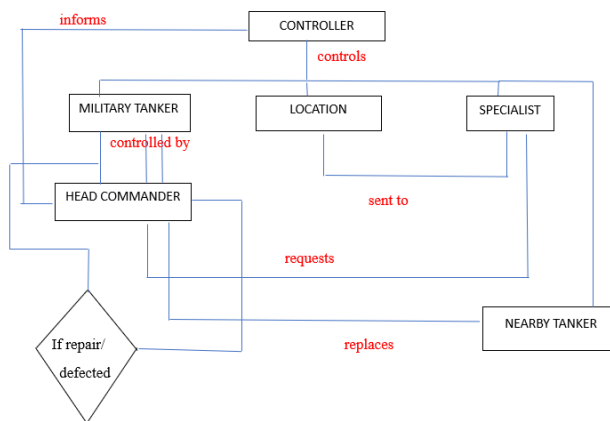
Graph data structures are used in various real-time applications. It is used to solve various applications around us every day. It uses various algorithms to solve real-time problems. The architecture diagram describes a high-level overview of major system



**Fig 1.** Prototype Model for Suggested Methodology

components and important working relationships. It represents the flow of execution and involves the following five major steps: The architecture diagram is defined by the flow of the process—how the application works and solves the problem using graph data structures. The Controller acts as a system head and specifies that it controls Military tankers, specialists, and the location of the tanker.

The military tanker contains a head Commander who informs the controller each time a problem occurs. When a problem occurs, the head Commander notifies the controller, and the controller makes decisions according to the problem in the tanker. If the tanker is damaged, then it must be completely replaced with the nearby tanker, which is known by calculating the shortest path from the source tanker. If the tanker is in repairable condition, the controller identifies the type of repair that occurred, calls the specialist for that type of repair, and rectifies the problem. MODULE DIVISION: The different modules in the suggested methodology are: creating a dataset, connecting to a dataset, Controller, Military tanker, Specialist, Location, applying algorithms, and testing code, as shown in Figure 2.



**Fig 2.** Work Flow Diagram for Suggested Methodology

The manual dataset is created by MySQL. At first, a database is created with the name military. And next, the tables have been created for the following entities: controller, tanker, status, specialist, and user. The table controller is created with the following attributes: tanker ID, tanker name; head commander ID, head Commander name; location number; location name; specialist ID, specialist name. The table tanker is created with the following attributes: tanker ID, tanker name, head commander ID, and head commander name. The table status is created with the following attributes: tanker ID, tanker name, and status. The table specialist is created with the following attributes: specialist ID, specialist name, and specialist type. The user table is created with the following attributes: username and password. And then, each table is inserted with manual data in it.

The MySQL database is connected using JDBC connectivity. It is a Java API that provides a standard interface for connecting Java applications to databases and executing SQL queries. It includes some key components for establishing connections, such as Connection, Statement, ResultSet, and Driver Manager.

The Controller controls the overall tanker network. It acts as an authority of access for all the tankers, the specialists, and the location of the tankers. It has a workflow for controlling and updating the application. Each time a problem occurs in the tanker, it is updated in the database by the controller, and it controls all the functionalities that must be done when a problem

occurs and when no problem occurs in the tanker.

In a military tanker, the head Commander acts as the tanker's head. The tanker is controlled by the controller System. The head Commander in the tanker sends the problem state to the controller, and the required action has been taken by the controller and updated in the database. The military tanker does the following functionalities, such as updating the problem state to the controller if there is any problem and informing the controller of every piece of information.

The specialist is controlled by the controller module. Whenever a problem occurs in the tanker, the head commander of the tanker intimates the problem to the controller. If the problem is of the repaired type, the controller finds the specialist required for the specified problem that occurred in the tanker. Finally, the controller sends the location of the tanker to the specialist. The specialist will move to the repaired tanker location and rectify the problem.

The location module is controlled by the controller module. Whenever a problem occurs in the tanker, the head commander sends the problem type to the controller. If the problem that occurred is of type defected, then the controller finds the shortest path between the tankers with the help of the location of the source tanker using Dijkstra's algorithm, as shown as follows:

The distance is calculated as:

**$\text{dist}[u] = \text{dist}[v] + \text{Graph.Edges}[v,u].\text{dist}[u]$**  (**u,v is vertices**) and its corresponding java implementation as follows:

```
int[] dijkstra(int graph[][], int s)
{
    int distance[] = new int[totalVertex];
    Boolean spSet[] = new Boolean[totalVertex];
    for (int j = 0; j < totalVertex; j++)
    {
        distance[j] = Integer.MAX_VALUE;
        spSet[j] = false;
    }
    distance[s] = 0;
    for (int cnt = 0; cnt < totalVertex - 1; cnt++)
    {
        int ux = minimumDistance(distance, spSet);
        spSet[ux] = true;
        for (int vx = 0; vx < totalVertex; vx++)
            if (!spSet[vx] && graph[ux][vx] != -1 && distance[ux] != Integer.MAX_VALUE &&
                distance[ux] + graph[ux][vx] < distance[vx])
            {
                distance[vx] = distance[ux] + graph[ux][vx];
            }
    }
    int[] res=printSolution(distance, totalVertex);
    return res;
}
```

### 3 Result and Discussion

#### 3.1 Experimental Environment and Result Discussuion

The RAM requirement is 4GB. The storage requirement is 10 to 20 GB, and the processor requirement is 500 MHz. The software required is Java. The operating system requirement is Windows 11, and the tool that is required is the JDBC API.

The dataset has been manually created for all the prototypes, and then the Java application has been created for various test cases, such as defected and repaired cases, and has been tested with the manual data. Dijkstra's algorithm has been implemented to find the shortest path in the application. The data has been visualised using a line chart for all climatic conditions and surfaces at which a tanker moves per hour.

The graphical representation of tankers in the war field that are consider as an tanker network is shown in the Figure 5.

The dataset values for tanker network parameters such as distance, environment, and surface have been calculated for every 500 metres of the speed at which the tanker moves in the only climatic conditions and surfaces. The distance attribute describes the distance in km, and all other attributes are climatic conditions and surfaces that have been given in km per hour.

mysql> desc controller;							mysql> desc specialist;						
Field	Type	Null	Key	Default	Extra		Field	Type	Null	Key	Default	Extra	
tankerID	int	YES		NULL			spec_ID	int	NO		NULL		
tankerName	varchar(50)	YES		NULL			spec_name	varchar(50)	NO		NULL		
headCommanderID	int	YES		NULL			spec_type	varchar(100)	NO		NULL		
headCommanderName	varchar(50)	YES		NULL									
locationNo	int	YES		NULL									
locationName	varchar(40)	YES		NULL									
specialistID	int	YES		NULL									
specialistName	varchar(25)	YES		NULL									
8 rows in set (0.00 sec)							3 rows in set (0.00 sec)						

mysql> desc status;							mysql> desc tanker;						
Field	Type	Null	Key	Default	Extra		Field	Type	Null	Key	Default	Extra	
tankerID	int	NO		NULL			tankerID	int	YES		NULL		
tankerName	varchar(40)	NO		NULL			tankerName	varchar(50)	YES		NULL		
status	varchar(40)	NO		NULL			headCommanderID	int	YES		NULL		
							headCommanderName	varchar(50)	YES		NULL		
3 rows in set (0.00 sec)							4 rows in set (0.00 sec)						

Fig 3. Default Values for Tanker Network Parameters

mysql> select * from controller;							mysql> select * from specialist;						
tankerID	tankerName	headCommanderID	headCommanderName	locationNo	locationName	specialistID	specialistName	spec_ID	spec_name	spec_type			
101	ARjuna 1	1	AAA	1	L001	201	SP01	201	SP01	Structural Repairs			
102	ARjuna 2	2	BBB	2	L002	202	SP02	202	SP02	Track Repairs			
103	ARjuna 3	3	CCC	3	L003	203	SP03	203	SP03	Engine and Powertrain Repairs			
104	ARjuna 4	4	DDD	4	L004	204	SP04	204	SP04	Weapon System Repairs			
105	ARjuna 5	5	EEE	5	L005	205	SP05	205	SP05	Electronics and Communication Repairs			
106	ARjuna 6	6	FFF	6	L006	206	SP06	206	SP06	Auxiliary Equipment Repairs			
6 rows in set (0.00 sec)							6 rows in set (0.00 sec)						

mysql> select * from status;							mysql> select * from tanker;						
tankerID	tankerName	status					tankerID	tankerName	headCommanderID	headCommanderName			
101	ARjuna 1	good					101	ARjuna 1	1	AAA			
102	ARjuna 2	good					102	ARjuna 2	2	BBB			
103	ARjuna 3	good					103	ARjuna 3	3	CCC			
104	ARjuna 4	good					104	ARjuna 4	4	DDD			
105	ARjuna 5	good					105	ARjuna 5	5	EEE			
106	ARjuna 6	good					106	ARjuna 6	6	FFF			
6 rows in set (0.00 sec)							6 rows in set (0.00 sec)						

Fig 4. Experimental Values for Tanker Network Parameters

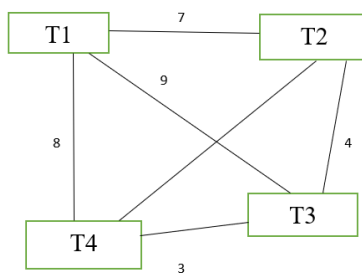


Fig 5. Tanker Network

DISTANCE (m)	PLAIN (m/hr)	FOREST (m/hr)	MUD (m/hr)	DESERT (m/hr)	RIVER (m/hr)	MOUNTAIN (m/hr)	ICE MOUNTAIN (m/hr)	RAIN (m/hr)	SNOW (m/hr)	DRY (m/hr)
1000	0.04	0.01	0.005	0.04	0.008	0.02	0.01	0.04	0.02	0.04
1500	0.0266	0.0066	0.0033	0.0266	0.0053	0.0133	0.0066	0.0266	0.0133	0.0266
2000	0.02	0.005	0.0025	0.02	0.004	0.01	0.005	0.02	0.01	0.02
2500	0.016	0.004	0.002	0.016	0.0032	0.008	0.004	0.016	0.008	0.016
3000	0.0133	0.0033	0.0016	0.0133	0.00266	0.0066	0.0033	0.0133	0.0066	0.0133
3500	0.0114	0.0028	0.0014	0.0114	0.00228	0.0057	0.0028	0.0114	0.0057	0.0114
4000	0.01	0.0025	0.0012	0.01	0.002	0.005	0.0025	0.01	0.005	0.01
4500	0.0088	0.0022	0.0011	0.0088	0.00177	0.00444	0.0022	0.0088	0.00444	0.0088
5000	0.008	0.002	0.001	0.008	0.0016	0.004	0.002	0.008	0.004	0.008
5500	0.0072	0.0018	0.0009	0.0072	0.00145	0.00363	0.0018	0.0072	0.00363	0.0072
6000	0.0066	0.0016	0.00083	0.0066	0.0013	0.00333	0.0016	0.0066	0.00333	0.0066
6500	0.0061	0.0015	0.00076	0.0061	0.00123	0.00307	0.0015	0.0061	0.00307	0.0061
7000	0.0057	0.0014	0.00071	0.0057	0.00114	0.002857	0.0014	0.0057	0.002857	0.0057
7500	0.0053	0.0013	0.00066	0.0053	0.00106	0.00266	0.0013	0.0053	0.00266	0.0053
8000	0.005	0.0012	0.000625	0.005	0.001	0.0025	0.0012	0.005	0.0025	0.005

Fig 6. Dataset Values for Tanker Network Parameters (Distance, Environment And Surface)

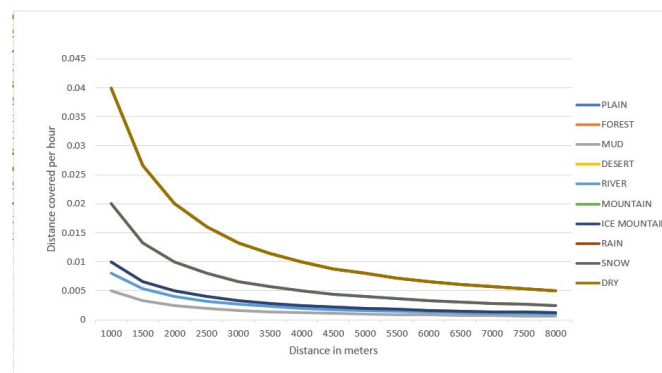


Fig 7. Graphical Representation for Tanker Network Parameters (Distance, Environment And Surface)

The distance has been taken in metres (x-axis), and the speed of the tanker has been taken in kilometres per hour (y-axis), which is shown in Figure 7. The attributes that have been considered are plain, forest, mud, desert, river, mountain, ice mountain, rain, snow, and dry. It has been visualised for the given attribute with a distance in metres. And finally, on plain and desert surfaces with dry climatic conditions, the tanker moves at 40 to 60 km/hr.

For various environments, Dijkstra's algorithm has been tested, and the best-case time complexity is  $O(|V|+|E|\log|V|)$  for plain and desert surfaces with dry climatic conditions. The average case time complexity is  $O((|V|+|E|)\log|V|)$  for snow climatic conditions and forest and mountain surfaces, and time complexity is  $O((|V|+|E|)\log|V|)$  for river and mud surfaces. Where  $E$  denotes Edges and  $V$  denotes Vertices.

In the military, contemporary communication technologies like Secure Radio Communication, Satellite Communication, and Secure Digital Messaging are utilized. Despite this, there is typically no direct communication between the military's central command, responsible for overseeing all wartime activities, and personnel in military tankers. Conversely, in a Java application, direct communication is established on both ends, and the application is tailored to accommodate various constraints, including climatic conditions and the tanker's speed under different environmental circumstances. These factors have been figured in Figure 5 and analysed. The application employs Dijkstra's algorithm, a graph algorithm, to determine the best-case time complexity across various test scenarios. This approach has yielded optimal outcomes for diverse environmental conditions.



## 4 Conclusion

The conclusion of the application is that it acts as an intermediary between the controller of the system and the armed forces. As per the requirement of the application to provide security measures like Surveillance and Reconnaissance, Route Planning and Analysis, Communication and coordination, and Contingency plans, all have been successfully implemented in the application. The implementation of various graph algorithms to find the shortest path between the tankers will be given rise to various experimental analyses on time complexity, space complexity, and various algorithmic features of each algorithm used for finding the shortest path. The shortest path analyzation for the replacement of the tankers has been implemented using Dijkstra's algorithm. The application has been analysed with various test cases, and an optimal output has been known for each test case. The future outcome of the project includes adding encryption and decryption to the application for higher security measures and data safety. Increasing data safety measures, including new technologies for the mode of communication between the system and the user, Analysing the present model with real-time datasets for better and more effective results and using tank authentication to provide data privacy for the System administrator.

## References

- 1) Pak J, Kim J, Park Y, Son HI. Field Evaluation of Path-Planning Algorithms for Autonomous Mobile Robot in Smart Farms. *IEEE Access*. 2022;10:60253–60266. Available from: <https://doi.org/10.1109/ACCESS.2022.3181131>.
- 2) Guo H, Hou X, Cao Z, Zhang J. GP3: Gaussian Process Path Planning for Reliable Shortest Path in Transportation Networks. *IEEE Transactions on Intelligent Transportation Systems*. 2022;23(8):11575–11590. Available from: <https://doi.org/10.1109/TITS.2021.3105415>.
- 3) Jung S, Pramanik S. An efficient path computation model for hierarchically structured topographical road maps. *IEEE Transactions on Knowledge and Data Engineering*. 2002;14(5):1029–1046. Available from: <https://doi.org/10.1109/TKDE.2002.1033772>.
- 4) Gismalla MSM, Azmi AI, Salim MRB, Abdullah MFL, Iqbal F, Mabrouk WA, et al. Survey on Device to Device (D2D) Communication for 5GB/6G Networks: Concept, Applications, Challenges, and Future Directions. *IEEE Access*. 2022;10:30792–30821. Available from: <https://doi.org/10.1109/ACCESS.2022.3160215>.
- 5) Rouzic EL, Renais O, Meuric J, Marcot T, Betoule C, Thouenon G, et al. Operator view on optical transport network automation in a multi-vendor context. *Journal of Optical Communications and Networking*. 2022;14(6):11–22. Available from: <https://doi.org/10.1364/JOCN.450008>.
- 6) Jia M, Gabrys B, Musial K. A Network Science Perspective of Graph Convolutional Networks: A Survey. *IEEE Access*. 2023;11:39083–39122. Available from: <https://doi.org/10.48550/arXiv.2301.04824>.
- 7) Zhu Z, Li L, Wu W, Jiao Y. Application of improved Dijkstra algorithm in intelligent ship path planning. In: 2021 33rd Chinese Control and Decision Conference (CCDC). IEEE. 2021;p. 4926–4931. Available from: <https://doi.org/10.1109/CCDC52312.2021.9602021>.
- 8) Zúñiga BC, Mendoza AM. PST-Prim Heuristic for the Open Vehicle Routing Problem. *IEEE Latin America Transactions*. 2019;17(02):253–259. Available from: <https://doi.org/10.1109/TLA.2019.8863171>.
- 9) Ardizzoni S, Consolini L, Laurini M, Locatelli M. Solution Algorithms for the Bounded Acceleration Shortest Path Problem. *IEEE Transactions on Automatic Control*. 2023;68(3):1910–1917. Available from: <https://doi.org/10.1109/TAC.2022.3172169>.
- 10) Ameen AT, Fouad AM, Matta EM, Elkhishen RA, Alalem SA, Safan AM, et al. Comparison of Parallel and Serial Execution of Shortest Path Algorithms. In: 2023 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE). IEEE. 2023;p. 232–237. Available from: <https://doi.org/10.1109/ICCIKE58312.2023.10131705>.
- 11) Lee HT, Choi HM, Lee JS, Yang H, Cho IS. Generation of Ship's Passage Plan Using Data-Driven Shortest Path Algorithms. *IEEE Access*. 2022;10:126217–126231. Available from: <https://doi.org/10.1109/ACCESS.2022.3225571>.
- 12) chuan Wei D. Implementation of Route Selection Function Based on Improved Floyd Algorithm. In: 2010 WASE International Conference on Information Engineering. IEEE. 2010;p. 223–227. Available from: <https://doi.org/10.1109/ICIE.2010.149>.