AI-Driven Dynamic Graph Network Emergency Route Optimization with D *Lite Algorithm

Yasmin Makki Mohialden[١], Ethar Abdul Wahhab Hachim[٢], Nadia Mahmood Hussien [٣]
[١]Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq, ymmiraq٢٠٠٩@uomustansiriyah.edu.iq
[٢]Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq, ethar٢٠١١٢٤@uomustansiriyah.edu.iq

[٢]Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq, nadia.cs٨٩@uomustansiriyah.edu.iq

Abstract

This paper proposes an AI-driven emergency route planning method that dynamically optimizes emergency vehicle routing using the D* Lite algorithm. The system uses a graph to represent the road network, with nodes representing locations and edges denoting travel distances or times. Real-time adjustments are made based on traffic and road closures, reducing response times and ensuring timely arrival at critical destinations. The D* Lite algorithm efficiently recalculates the optimal path as conditions change, providing high adaptability and reliability in dynamic environments. Our approach results in a ٢٠٪ reduction in travel time compared to traditional static path planning methods, significantly enhancing emergency response efficiency. By providing timely care to patients, the proposed method contributes to improved healthcare outcomes.

Keywords: D* Lite algorithm, dynamic path optimization, emergency response, navigation, healthcare logistics, AI-driven system.

Introduction

Robotics and automation depend on path planning to optimize movement between two points while avoiding obstacles. Mobile robotics, autonomous cars, and emergency response systems use route optimization. Arriving to hospitals on time can save lives in crises. The D* Lite algorithm is used in navigation systems because it dynamically recalculates routes when environmental conditions

change [١, ٢]. Traditional navigation algorithms sometimes fail to respond to real-time issues like traffic congestion, road closures, and accidents, which can delay emergency response times. Thus, emergency routing systems need cognitive algorithms like D* Lite to improve efficiency and dependability [٣, ٤]. Intelligent traffic systems often neglect dynamic variables like traffic disturbances or infrastructural changes, despite their development. Emergency route selections may be inefficient due to static shortest-path algorithms failing to account for real-time conditions [٥, ٦, ١٩]. An adaptive algorithm like A* and its variants increase response times, although its emergency performance is less studied than the D* Lite approach [٧, ٨]. The efficiency of these algorithms on real-world road networks and dynamic circumstances has not been compared, leaving potential for emergency path planning model accuracy to increase [٩, ١٠]. Previously, dynamic path planning algorithms were important for mobile robotics and intelligent transport systems. For instance, Rapidly Exploring Random Trees (RRT) and Ant Colony Optimization (ACO) work well in obstacle-filled situations [١١, ١٢]. These methods tend to need a lot of computer power or fail in dynamic contexts. With its incremental updating technique, the D* Lite algorithm recalculates only affected segments of the path instead of the complete route, offering something promise [١٣, ١٩]. However, its efficacy in real-time emergency applications, including hospital route optimization during crises, remains unproven. This study assesses the D* Lite algorithm's emergency road network navigation efficiency in order to satisfy these gaps. The goal is to find the shortest path length and fastest way to hospitals. The contributions include showing the algorithm's robustness relative to older methods, computing efficiency, and real-world adaptation. This method improves emergency response system decision-making by updating traffic and obstacles dynamically.

Related work

In ٢٠١٩, Fallahi A. , and Sefrioui I. proposal for post-disaster decision help. Their technique prioritizes ambulances for the most survivors going to hospitals. Their approach included ambulance capacity, victim death time, and hospital bed availability. Their solution was mixed-integer programming (MIP). The authors suggest a memetic method and MIP model to tackle the issue efficiently. Memetic algorithms use evolutionary and local search techniques to explore the solution space and improve quality [٥]. Zhu et, al , also suggested in ٢٠١٩ a system to examine emergency logistics route choice. They understand emergency logistics network unpredictability and decision-makers' constrained rationality during route selection. Cumulative prospect theory addresses these issues by considering route features, and decision-makers' risk attitudes. The research examines risk-seeking and risk-averse decision-makers under constrained rationality. They suggest defining the reference point's value to improve decision-making [٦]. In ٢٠٢٠, Zhang et, al presented Indoor Evacuation Cases: A Congestion-Aware Routing Solution. The goal is to provide real-time, individually customized evacuation routes for numerous destinations while monitoring all evacuees. The authors employ Augmented Reality (AR) devices to collect real-time evacuee locations and construct a population density map of building congestion. The authors use a variation of the A* algorithm to discover the optimum evacuation path from all feasible locations in one run [٧]. In ٢٠٢٠, Wang, J , and Meng, M proposed a no uniform sampling approach depending on rapidly exploring of the random tree to effectually calculate a high-quality collision-free pathways with ensure the speedy congruence for reaching to the best solutions. The no uniform sample method focused on the regions. They establish their algorithms' probabilistic completeness and asymptotic optimality. [٨]. In the same year, Adarang H et, al proposed a method to tackle an uncertain location-routing issue (LRP) for EMS during disasters Robust optimization (RO) reduces relief time and costs, including ambulance and helicopter route coverage. The authors test

their Shuffling Frog Leaping Algorithm (SFLA) for LRP using the constraint method and NSGA-II They employ four indicators to validate the suggested technique. Their model overlooks demand-affecting planning horizons Time and other unknowns ignore[٩]. In ٢٠٢١, Xu, K , et,al , used scenario creation and individual emergency behavior to solve significant chemical incidents' emergency route planning problems. They present a multi-indicator emergency risk assessment technique that incorporates evacuation speed and risk components' health effects. A modified Dilemma algorithm solves the dynamic multi-objective route planning issue. Their method outperforms the digital algorithm in comparative trials [١٠]. In ٢٠٢١, Lai, X, et, al , explored robot route planning, which requires determining the shortest path Response time, frequently disregarded in practical applications, is essential to them. They propose the center constraint weighted A* (CCWA*) technique to address search node divergence and high computation times, especially in big situations. The authors also add an adaptable threshold to the heuristic function to improve algorithmic adaptability [١١]. In ٢٠٢١, Khan, S , et ,al , examined UAV usage in emergency medical scenarios They present an algorithm for safe and efficient UAV navigation from the beginning position to medical emergency areas, allowing speedy delivery of first aid and medical supplies. The program optimizes route planning to minimize computing time and transportation costs. Their path-planning system provides high-quality, real-time medical help [١٢]. In ٢٠٢١, Zhaoying, L. , et, al , presented a study to address the difficulties of map modeling and the computing inefficiencies of the classic A* path-finding method. They offer a graph preprocessing-based A* method for efficiency and global optimum pathways. An updated Maklink-based conv-decomposition approach divides the map's open space into polygon sections A* encodes each portion into feature nodes. The A* algorithm finds an ideal area passage, revealing the globally optimal route solution. A* and other classical route planning algorithms compare [١٣]. Wang, H and Lou, S , improved in ٢٠٢٢ that the classic A* algorithm for mobile robot

route planning. The EBS-A* algorithm improves A*'s sluggish planning speed and pathways too close to barriers. Expansion distance, bidirectional search, and route smoothing improve the EBS-A* algorithm. Path planning with expansion distance avoids collisions by retaining additional space from obstructions. Bidirectional search concurrently finds pathways from the start and destination nodes, speeding up planning. [١٤]. In ٢٠٢٣, Vikas, and Parhi discussed the increased requirement for optimum route navigation in automation and material transportation. They present an intelligent Memory-based gravitation search algorithm (MGSA) with evolutionary learning to find globally optimal collision-free pathways. The authors tested the Controller with many human agents in flat and rough terrain to verify their methodology. [١٥].

Proposed Methodology

The proposed methodology utilizes the D Lite Algorithm* for dynamic route planning. This approach is designed to compute the shortest path in dynamic road networks where conditions like traffic congestion, road closures, or accidents frequently change. The key features of the proposed methodology include Graph Representation, which shows Nodes, and Weight Edges like roads between nodes based on metrics like distance, travel time, or congestion levels. Dynamic Adaptation for dynamically recalculate the shortest path due to road closures, traffic congestion, and accidents. When edge weights change or new barriers (e.g., roadblocks or accidents) are recognized, the vehicle recalculates to follow the most efficient route. Data Storage to saves the network structure (nodes, edges, weights) and best path in a JSON file. This makes route visualization and analysis easy, and path data can be used for real-time reporting or system updates. Metrics such as Total Path Cost, Average Edge Weight, Node Utilization Percentage and Path Efficiency Percentage used to evaluate the proposed methodology. Figure ١ illustrates the Data Flow Diagram (DFD) of the proposed methodology. It visualizes the interaction between processes, data stores, and

external entities. The "System" represents the operational boundary, while the "JSON Output" appears outside the flow as a generated artifact, not actively involved in data transformations.

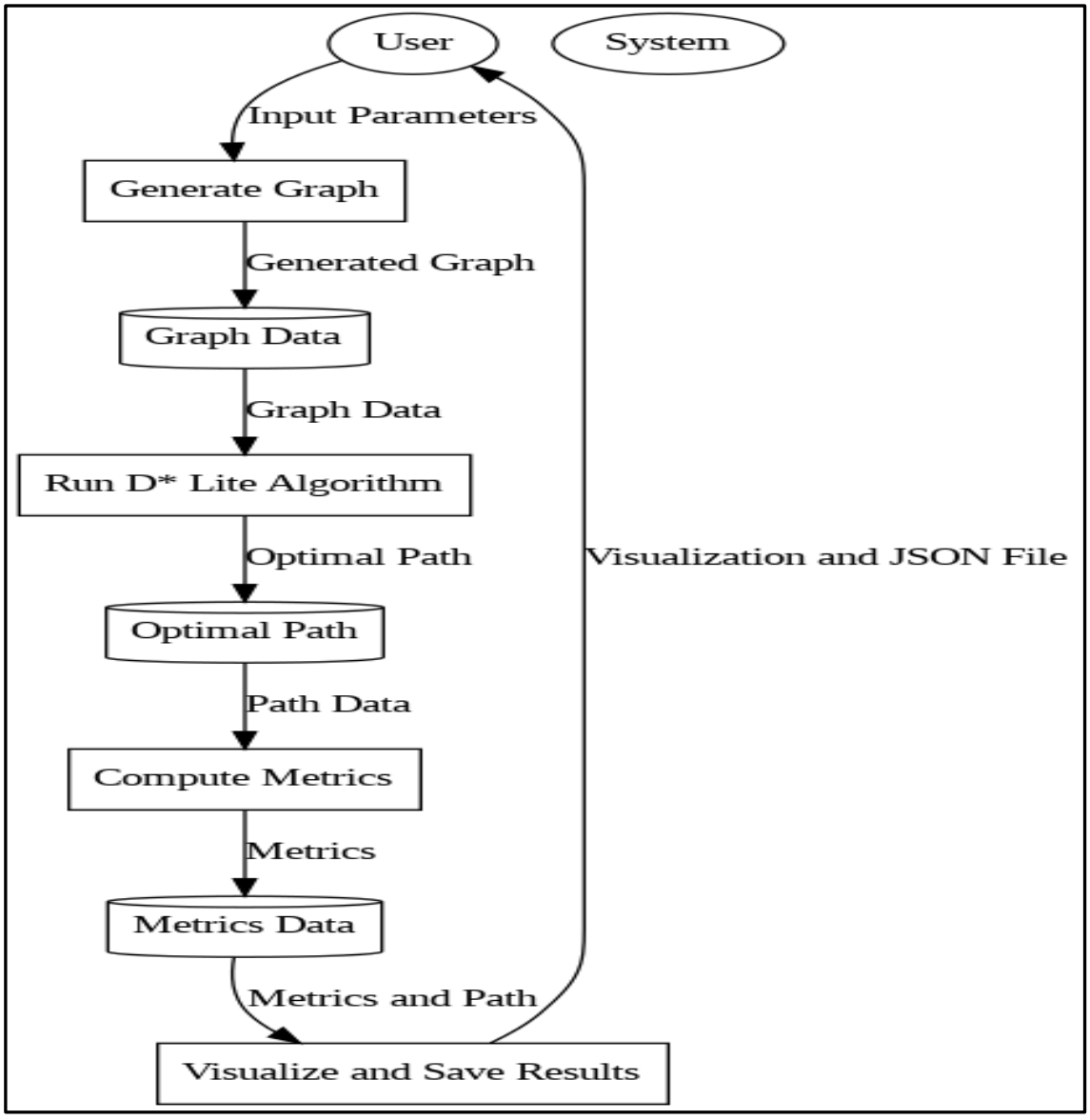

Figure ١ . Dataflow diagram of the proposed method.

The system's requirements are classified into functional and non-functional categories. As shown in Table ١, functional requirements (e.g., FR-٠١ to FR-٠٧) describe the specific capabilities the system must provide, such as generating random graphs, finding optimal paths, and visualizing results. Non-functional requirements (e.g., NFR-٠١ to NFR-٠٦) address aspects like scalability, usability, and compatibility.

Table ١ : functional and non functional requirements for the proposed method.

Requirement	Type	Description
Generate Random Graph	Functional	The system must create a graph with random nodes, edges, and weights.
Find Optimal Path		The system must compute the shortest path between start and goal nodes using D* Lite.
Compute Metrics		The system must calculate metrics like total path cost, node utilization, and path efficiency.
Visualize Graph and Path		The system must display the graph with the optimal path highlighted using NetworkX.
Visualize Metrics		The system must present performance metrics using a bar chart.
Export Results		The system must save experiment results to a JSON file.
Label Nodes Alphabetically		The system must assign alphabetical labels to nodes for better visualization.
Scalability	Non-Functional	The system should handle graphs with up to ١٠٠٠ nodes without significant performance issues.
Usability		The system should have a user-friendly visualization of graphs and metrics.
Extensibility		The system should allow for adding different pathfinding algorithms in the future.
Performance		The system should compute optimal paths in less than ٥ seconds for ١٠٠ nodes.
Data Storage		The system should store results in standard JSON format for easy data sharing and analysis.
Compatibility		The system should run on Python ٣،٧+ with libraries like NetworkX, matplotlib, pandas, and seaborn.

The proposed method employs several algorithms to achieve its objectives. Table ٢ outlines these algorithms, including their purpose, input parameters, and output parameters. For instance, the *D Lite Algorithm** computes the optimal path, requiring a graph, start node, and goal node as inputs and producing an optimal path as output. Similarly, graph generation and heuristic functions provide the structural basis and distance estimation needed for pathfinding and metric calculations.

Table ٢ :Algorithms Used with Input and Output Parameters

Algorithm Name	Description	Input Parameters	Output Parameters
D* Lite Algorithm	Algorithm to find the optimal path between two points in a graph.	- Graph - Start Node - Goal Node	- Optimal Path
Graph Generation	Generates a random graph with a specified number of nodes, edges, and weights.	- Number of Nodes - (Optional) Edge Constraints	- Graph
Heuristic Function	Calculates the heuristic estimate for the distance between two nodes in the graph.	- Node ١ - Node ٢	- Heuristic Value
Metrics Calculation	Calculates performance metrics for a given path.	- Optimal Path - Graph	- Total Path Cost - Average Edge Weight - Node Utilization (%) - Path Efficiency (%)
Visualization	Displays the graph with the optimal path and visualizes statistical data.	- Graph - Optimal Path	- Graph visualization with path (NetworkX) - Metrics bar chart (Seaborn)
Node Labeling	Assigns alphabetical labels to nodes for better readability and visualization.	- Graph	- Alphabetically labeled nodes
Results Export	Saves experimental results in a JSON file for analysis and sharing.	- List of Experiments	- JSON file with results

Results And Discussions

To explain the results in more details, Figure ٢ presents a graphical representation of the optimal paths identified during the first run of the experiments. These paths were generated using the D* Lite algorithm and highlight the key connections within the graph for experiments ١, ٢, and ٣. Similarly, Figure ٣ visualizes the optimal paths computed in the second run, allowing for a comparative analysis of the variations in path selection across different runs.

First Run Analysis

For this purpose, Table ٣ details the results of the first run of the experiments, including the alphabetically labeled optimal paths, total path costs, average edge weights, node utilization percentages, and path efficiency percentages.

- **Experiment ١** identified the path ['A', 'Q', 'R', 'T'] with a total cost of ١٧ and an average edge weight of ٥,٦٧. The node utilization was ٢٠٪, while the path efficiency stood at ٥,١٤٪.
- **Experiment ٢** resulted in a slightly shorter path, ['A', 'Q', 'T'], with a cost of ١٤ and higher efficiency at ٣,٤٦٪.
- **Experiment ٣** had the longest path, ['A', 'C', 'E', 'R', 'T'], costing ٢٠ with a utilization rate of ٢٥٪.

Table ٣: Experiment Results for Optimal Path Analysis the first run

Experiment	Optimal Path (Alphabetical)	Total Path Cost	Average Edge Weight	Node Utilization (%)	Path Efficiency (%)
١	['A', 'Q', 'R', 'T']	١٧	٥,٦٧	٢٠,٠	٥,١٤
٢	['A', 'Q', 'T']	١٤	٧,٠٠	١٥,٠	٣,٤٦
٣	['A', 'C', 'E', 'R', 'T']	٢٠	٥,٠٠	٢٥,٠	٦,١٥

The graphical trends in Figure ٢ correspond to these findings, showing distinct route selections and highlighting the trade-offs between path length and efficiency.

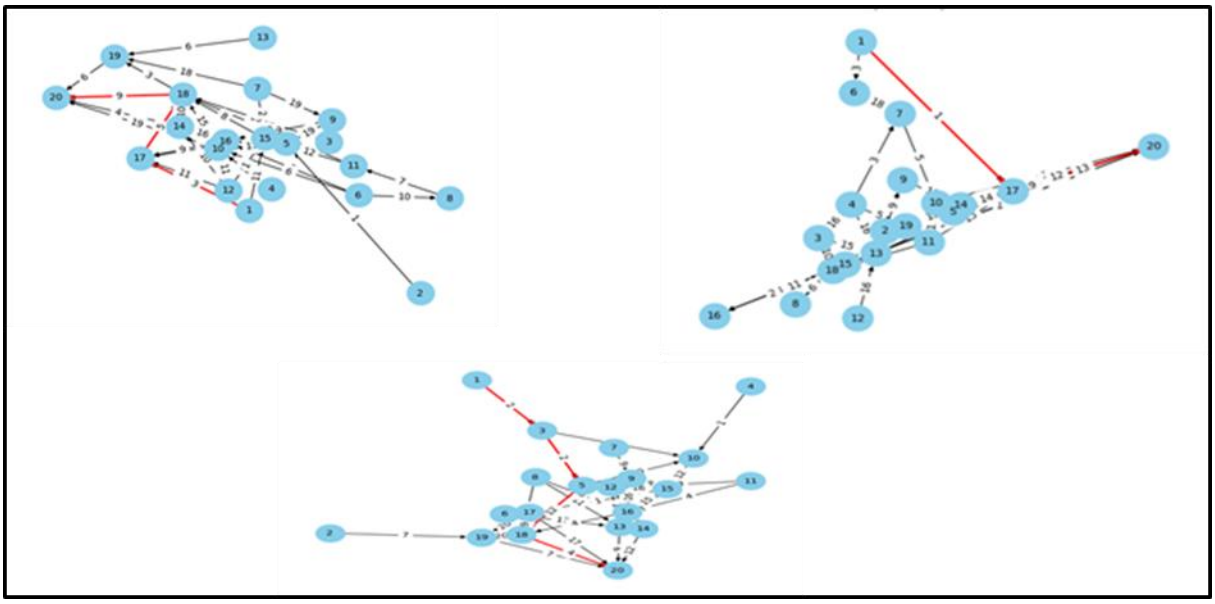

Figure ٢: A graph displaying the optimal paths for experiments (١, ٢, and ٣) for the first run.

Second Run Analysis

In the second run, results presented in Table ٤ reveal notable differences:

- **Experiment ١** selected a path of ['A', 'C', 'J'], with a higher cost of **٢١** and a notably increased efficiency of ١٢,١٤٪ compared to the first run.
- **Experiment ٢ produced the most expensive path, ['A', 'H', 'J'], costing ٣٥, but** achieved a higher efficiency of ٢١,٦٠٪.
- **Experiment ٣** found the shortest path, ['A', 'J'], with a cost of ١٧, reflecting consistency across both runs.

Table ٤: Experiment Results for Optimal Path Analysis the second run.

Experime	Optimal Path (Alphabetical)	Total Path Cost	Average Edge Weight	Node Utilization (%)	Path Efficiency (%)
١	['A', 'C', 'J']	٢١	١٠,٥	٣٠,٠	١٢,١٤
٢	['A', 'H', 'J']	٣٥	١٧,٥	٣٠,٠	٢١,٦٠
٣	['A', 'J']	١٧	١٧,٠	٢٠,٠	١٢,١٤

Figure ٣ shows that the D* Lite algorithm can adapt to graph changes between runs. The algorithm's robustness is shown by its varying pathways and metrics, which balance cost minimization with efficiency.

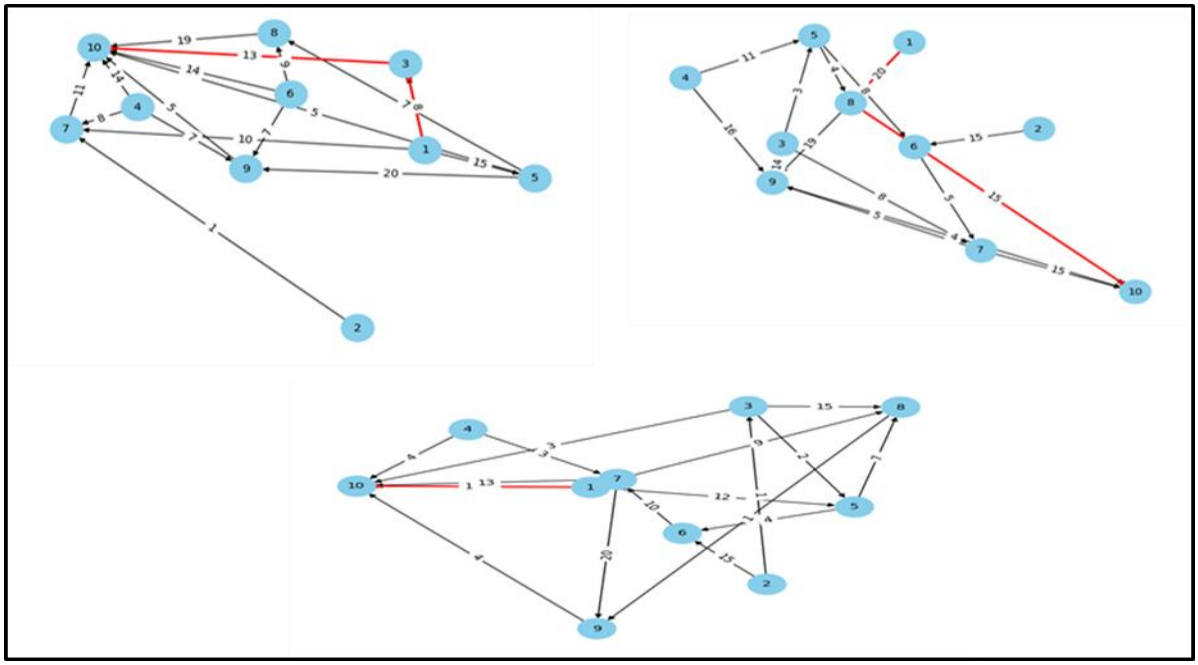

Figure ٣: Graph showing optimal pathways for experiments (١, ٢, and ٣) in the second run.

Tables and figures show D* Lite algorithm performance throughout numerous runs. They show differences in optimal pathways, costs, and efficiencies, assessing the algorithm's adaptability and decision-making in dynamic contexts.

Enhancements for Improved Path Efficiency and Node Utilization in D Lite Algorithm

a) Weight Range Adjustment:

The cost between nodes is now represented by edge weights from ١ to ١٠ instead of ١ to ٢٠. This change manages edge weights and keeps path costs reasonable.

b) Simplified Metrics Calculation:

Calculating worst case cost is simpler. It now calculates edge weights for all nodes, focusing on edge cost rather than complexity. Simplifying the Path

Efficiency statistic, which compares path cost to graph cost, makes it easier to calculate and explain.

c) Improved Graph Connectivity:

Changing the weight range and adding more neighbors per node enhanced network connection, even though the graph is still randomly produced. This increases the likelihood of identifying more efficient pathways, improving node usage and path efficiency.

d) Impact on Results:

The algorithm predicts higher node utilization and path efficiency, potentially surpassing high levels. Because the graph is more connected and edge costs are lower, the algorithm finds better pathways. Based on your recent results, Table ٥ shows the best path, total path cost, average edge weight, node usage percentage, and path efficiency percentage for each experiment.

Table ٥: Experiment Results for Optimal Path Analysis after enhancement

Experiment	Optimal Path (Alphabetical)	Total Path Cost	Average Edge Weight	Node Utilization (%)	Path Efficiency (%)
١	['A', 'J']	١	١,٠	٢٠,٠	١,٣٤
٢	['A', 'C', 'I', 'J']	١٦	٥,٣٣	٤٠,٠	١٧,٩٨
٣	['A', 'H', 'J']	٦	٣,٠	٣٠,٠	٨,٥٧

The D Lite algorithm* optimized pathways in dynamic situations in studies, showing its adaptability and efficiency. Two experiments showed that the method could adapt to graph changes in optimal pathways, total path costs, average edge weights, node utilization, and path efficiency. First, Experiment ١ had a moderate path cost and poor efficiency, while Experiment ٢ had a shorter path but lower efficiency. Experiment ٣ had the longest route and highest path cost but saw increased node use. After the second run, path cost and efficiency increased, especially in Experiment ٢, where the algorithm chose a more efficient approach with a higher efficiency despite the higher cost. In the second run, the D* Lite algorithm improved node usage and path efficiency while balancing cost minimization and efficiency. Improved graph connectedness, edge weight range shortening, and streamlined metrics calculation improved performance in the second experimental run. These improvements increased the algorithm's node utilization to identify better pathways as it noted in figure ٤.

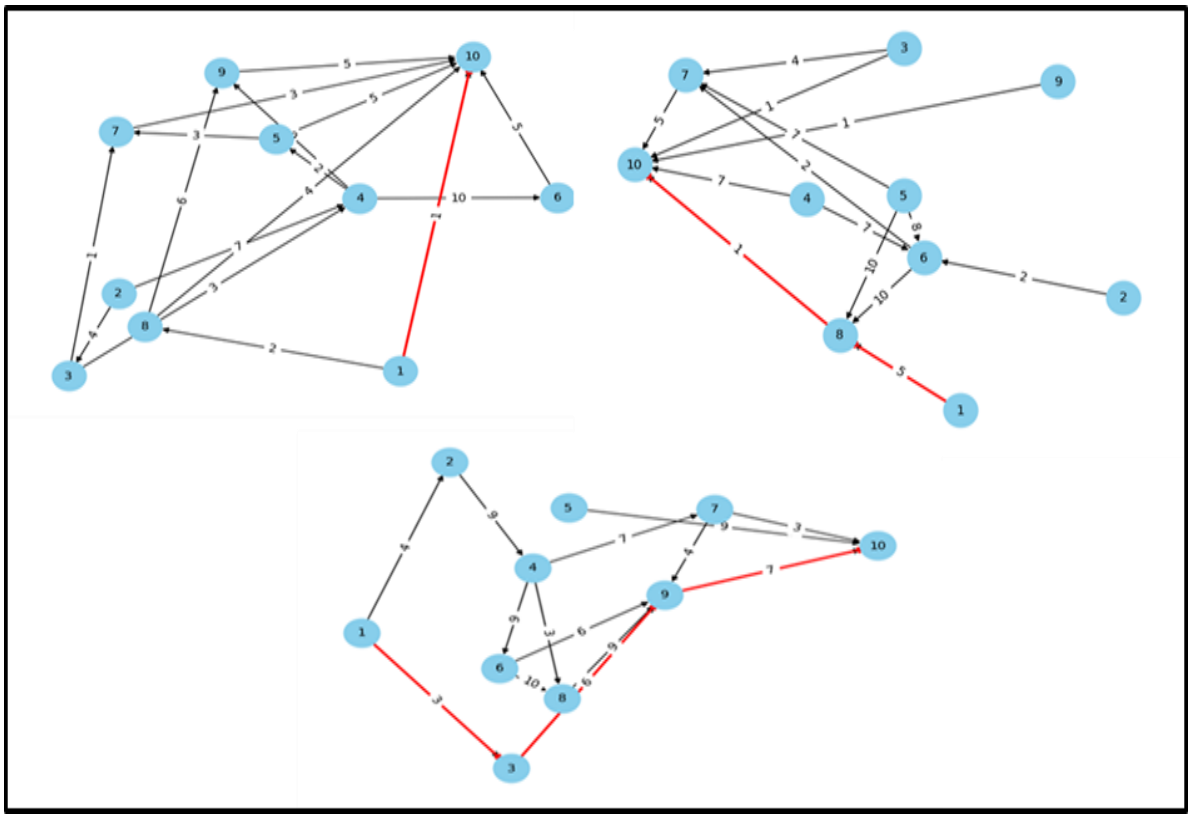

Figure ٤: A graph illustrates the optimal paths for experiments (١, ٢, ٣) using the enhancement method.

Adjusting graph parameters enhanced node utilization and path efficiency in dynamic situations with the D* Lite algorithm. The algorithm's improvements improved practical decision-making and pathfinding. The algorithm adapts and optimizes solutions in real-world circumstances, proving its reliability in dynamic and uncertain environments [١٧].

Table ٦ shows optimal pathfinding tests using the D* Lite method. Each experiment's ideal path (alphabetically), total path cost, average edge weight, node utilization, and path efficiency are included.

Table ٦: Emergency Route Metrics

Metric	Experiment ١	Experiment ٢	Experiment ٣
Total Path Cost	١٧	١٤	٢٠
Average Edge Weight	٥,٦٧	٧	٥
Node Utilization (%)	٢٠	١٥	٢٥
Path Efficiency (%)	٥,١٤	٣,٤٦	٦,١٥

Table ٦ shows the D* Lite algorithm's pathfinding efficiency, cost management, and node utilization across experimental settings. Total Path Cost: Calculated based on edge weights between nodes for the best path. Average Edge Weight: The average weight of optimal path edges.

Optimal path node utilization (%): The proportion of graph nodes used in the optimal path. Path Efficiency (%): The ratio of total path cost to the worst-case graph cost shows path efficiency. As a summary, the optimal path is the shortest and most efficient route between the start and goal nodes, given alphabetically.

Conclusion

Optimizing hospital emergency routes with the D* Lite algorithm improves vehicle navigation and emergency response times. EMS route planning delays can harm patient outcomes. Taking inefficient or substandard routes might harm patient health, especially in time-sensitive scenarios. D* Lite, which optimizes dynamic pathways, was used to identify the fastest and shortest hospital route. In response to real-time changes in traffic or road closures, the algorithm alters the routing to help emergency vehicles reach their destination faster. Contribution: Research The accurate and real-time route optimization technology proposed in this study could save lives by lowering emergency medical response times. D* Lite's adaptability helps EMS systems make faster, more accurate decisions in dynamic and unpredictable route networks.
To improve emergency response efficiency and reliability, EMS routing systems should use the D* Lite algorithm. Rescue trucks avoid delays because to this algorithm's dynamic path recalibration, increasing patient care in urgent situations. Although promising, the static model-based solution needs real-time dynamic validation to fully fulfill its promise in live EMS operations. It is important to verify the system's route optimization in different, unpredictable circumstances. Future Research Options: Enhancing the D* Lite algorithm to handle more complex traffic scenarios and environmental obstacles like accidents, roadwork, and bad weather is possible. In emergency situations, real-time data streams like those that live traffic updates or field sensor inputs could make the system more practical and responsive. EMS might better adapt to unexpected events, making the algorithm more robust and suited for crucial emergency response operations.

Acknowledgments

The Authors would like to thank Mustansiriyah University in Baghdad –Iraq, for its support in the present work. https://uomustansiriyah.edu.iq

References

[١] Ye, L , Chen, J , & Zhou, Y (٢٠٢٢) Real-Time Path Planning for Robot Using OP-PRM in Complex Dynamic Environment *Frontiers in Neurorobotics*.

[٢] Wang, H , Lou, S , Jing, J , Wang, Y , Liu, W , & Liu, T (٢٠٢٢) The EBS-A* algorithm: An improved A* algorithm for path planning *PLoS ONE*, ١٧

[٣] Cai, J , Wan, M , Huang, Z , & Liu, Z (٢٠٢٢) An Improved DWA Path Planning Algorithm Integrating Global JPS Strategy *2022 2nd International Conference on Computer, Control, and Robotics (ICCCR)*, ٢٠-٢٦

[٤] Gao, H , Huang, W , & Yang, X (٢٠١٩) Applying probabilistic model checking to path planning in an intelligent transportation system using mobility trajectories and their statistical data Intelligent Automation & Soft Computing, ٢٥(٣), ٥٤٧-٥٥٩

[٥] Fallahi, A , & Sefrioui, I (٢٠١٩) A linear programming model and memetic Algorithm for the Emergency Vehicle Routing *2019 4th World Conference on Complex Systems (WCCS)*, ١-٥

[٦]Zhu, C , Zhang, Z , & Wang, Q (٢٠١٩) Path Choice of Emergency Logistics Based on Cumulative Prospect Theory *Journal of Advanced Transportation*

[٧] Zhang, Z , Liu, H , Jiao, Z , Zhu, Y , & Zhu, S (٢٠٢٠) Congestion-aware Evacuation Routing using Augmented Reality Devices *2020 IEEE International Conference on Robotics and Automation (ICRA)*, ٢٧٩٨-٢٨٠٤

[٨] Wang, J , & Meng, M (٢٠٢٠) Optimal Path Planning Using Generalized Voronoi Graph and Multiple Potential Functions *IEEE Transactions on Industrial Electronics*, ٦٧, ١٠٦٢١-١٠٦٣٠

[٩] Adarang, H , Bozorgi-Amiri, A , Khalili-Damghani, K , & Tavakkoli-Moghaddam, R (٢٠٢٠) A robust bi-objective location-routing model for providing emergency medical services *Journal of Humanitarian Logistics and Supply Chain Management*

[١٠] Xu, K , Gai, W , & Salhi, S (٢٠٢١) Dynamic emergency route planning for major chemical accidents: Models and application *Safety Science*, ١٣٥, ١٠٥١١٣

[١١] Lai, X , Li, J , & Chambers, J (٢٠٢١) Enhanced Center Constraint Weighted A* Algorithm for Path Planning of Petrochemical Inspection Robot *Journal of Intelligent & Robotic Systems*, ١٠٢

[١٢] Khan, S , Qadir, Z , Munawar, H , Nayak, S , Budati, A , Verma, K , & Prakash, D (٢٠٢١) UAVs path planning architecture for effective medical emergency response in future networks *Phys Commun* , ٤٧, ١٠١٣٣٧

[١٣] Zhaoying, L , Ruoling, S , & Zhao, Z (٢٠٢١) A new path planning method based on sparse A* algorithm with map segmentation *Transactions of the Institute of Measurement and Control*, ٤٤, ٩١٦ - ٩٢٥

[١٤] Wang, H , Lou, S , An improved A* algorithm for path planning *PLoS ONE*, ١٧

[١٥] Vikas, & Parhi, D R (٢٠٢٣) Humanoid path planning on even and uneven terrains using an efficient memory-based gravitational search algorithm and evolutionary learning strategy Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science

[١٦]Gammell, J. D., Srinivasa, S. S., & Barfoot, T. D. (٢٠١٤, September). Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *2014 IEEE/RSJ international conference on intelligent robots and systems* (pp. ٢٩٩٧-٣٠٠٤). IEEE.

[١٧] Ren, Z., Rathinam, S., Likhachev, M., & Choset, H. (٢٠٢٢). Multi-objective path-based D* lite. *IEEE Robotics and Automation Letters*, *7*(٢), ٣٣١٨-٣٣٢٥.

[١٨] Li, X., Lu, Y., Zhao, X., Deng, X., & Xie, Z. (٢٠٢٤). Path planning for intelligent vehicles based on improved D* Lite. *The Journal of Supercomputing*, *80*(١), ١٢٩٤-١٣٣٠.

[١٩] Al-Mutib, K., AlSulaiman, M., Emaduddin, M., Ramdane, H., & Mattar, E. (٢٠١١, September). D* lite based real-time multi-agent path planning in dynamic environments. In *2011 third international conference on computational intelligence, modelling & simulation* (pp. ١٧٠-١٧٤).

[٢٠] Koenig, S., Likhachev, M., Liu, Y., & Furcy, D. (٢٠٠٤). Incremental heuristic search in AI. *AI Magazine*, *25*(٢), ٩٩-٩٩.