International Journal of Research in Engineering and Innovation (IJREI)
journal home page: http://www.ijrei.com
ISSN (Online): 2456-6934

RESEARCH PAPER

Shortest route optimization for vehicular movement using Dijkstra's and A* pathfinding algorithms

Binod Kumar Singh[1], Sonjoli Kaushik[2]

[1]*Research scholar, Department of Computer Application, Maharaja Agrasen Himalayan Garhwal University, Uttrakhand, India*
[2]*Professor, Department of Computer Application, Maharaja Agrasen Himalayan Garhwal University, Uttrakhand, India*

Article Information

Received: 23 June 2025
Revised: 05 Aug 2025
Accepted: 29 August 2025
Available online: 08 Sep 2025

Keywords:

Shortest Path, Dijkstra's Algorithm, A* Algorithm, Vehicular Routing, Transportation Networks, Optimization, Intelligent Transport Systems.

Abstract

Urban mobility has become one of the central challenges in modern transportation planning. Congestion, road closures, and inefficient routing increase travel time, fuel consumption, and emissions. To address these issues, computational models that can determine shortest and most efficient routes are essential. This study employs shortest route planning algorithms, specifically Dijkstra's algorithm and the A* (A-star) algorithm, to suggest optimal routes for vehicular movement. Dijkstra's algorithm guarantees the shortest path by exhaustively exploring all nodes, while A* enhances efficiency by incorporating heuristic estimations of the distance to the destination. The study compares both algorithms on a simulated road network and evaluates performance in terms of path optimality, computational efficiency, and applicability in dynamic environments. Results demonstrate that while Dijkstra's algorithm is reliable for static networks, A* performs better in large and complex real-world road systems due to its heuristic-driven efficiency. These findings highlight the potential of shortest path algorithms to improve transportation systems and pave the way for intelligent traffic management applications.

1. Introduction

The rapid growth of urbanization and motorization has resulted in severe traffic congestion, increased travel delays, and higher fuel costs. Efficient routing of vehicles is critical to improving road mobility, reducing emissions, and enhancing commuter satisfaction. The challenge of finding the shortest or most efficient path within a road network has been studied extensively in computer science, operations research, and transportation engineering. Shortest path algorithms form the foundation of modern navigation systems such as Google Maps, GPS-based car navigation, and Intelligent Transportation Systems (ITS). Among these, Dijkstra's algorithm and the A* algorithm have been widely adopted due to their robustness and efficiency. This paper explores the application of these algorithms in vehicular routing. By simulating their performance on a transportation network, the study highlights their comparative strengths, weaknesses, and potential integration into real-world traffic management systems. The rapid growth of urbanization and motorization has fundamentally transformed modern cities, leading to a surge in the number of vehicles on roads and the increasing complexity of transportation networks. While road infrastructure continues to expand, the rate of urban growth often surpasses these developments, resulting in severe traffic congestion, longer travel delays, and escalating fuel costs. The pressing need for sustainable and efficient mobility solutions has drawn considerable attention from researchers and urban planners alike. Efficient vehicle routing, therefore, emerges as a critical component in enhancing road mobility, reducing travel times, minimizing emissions, and ultimately improving commuter satisfaction. The problem of

Corresponding author: Binod Kumar Singh
Email Address: pinkipainting@gmail.com
https://doi.org/10.36037/IJREI.2025.9503

determining the shortest or most efficient path within a transportation network is not a new one; it has been widely studied in computer science, operations research, and transportation engineering for several decades. In particular, shortest path algorithms play a pivotal role in enabling optimal routing solutions. These algorithms form the foundation of modern navigation and decision-support systems such as Google Maps, Waze, and GPS-based in-vehicle navigation systems. Additionally, Intelligent Transportation Systems (ITS) rely heavily on such algorithms to provide real-time traffic management and route guidance, thereby enhancing overall urban mobility. Among the various methods developed, Dijkstra's algorithm and the A* algorithm stand out as two of the most widely adopted shortest path planning approaches. Dijkstra's algorithm, introduced in 1959, is recognized for its ability to compute the shortest path between a source and destination node with guaranteed accuracy. Its robustness and generality have made it one of the cornerstones of graph theory applications. However, its exhaustive search nature may lead to inefficiencies in very large or dynamic networks. In contrast, the A* algorithm, developed in the 1960s, integrates heuristics with cost functions to improve computational efficiency. By intelligently estimating the remaining distance to the target, A* is able to prioritize promising paths and significantly reduce computation time. This makes it particularly suitable for real-time applications where speed and adaptability are critical. The relevance of these algorithms extends beyond theoretical constructs, as they have real-world implications in diverse contexts. In transportation networks, they are crucial for managing congestion and guiding vehicles through urban roadways. In logistics and supply chain management, they contribute to efficient delivery scheduling and fuel optimization. Furthermore, with the emergence of autonomous vehicles, the demand for intelligent, adaptive, and computationally efficient routing algorithms is greater than ever before. Algorithms like Dijkstra's and A* thus play an instrumental role in shaping the future of intelligent mobility solutions.

This paper focuses on exploring the application of Dijkstra's algorithm and the A* algorithm in vehicular route optimization. By simulating their performance within a transportation network, the study provides a comparative analysis of their efficiency, strengths, and limitations. While Dijkstra's algorithm is lauded for its completeness and precision, the A* algorithm demonstrates superior speed and adaptability in dynamic routing scenarios. The comparative insights gained from this study are expected to contribute to the enhancement of traffic management systems, navigation software, and intelligent vehicle technologies. Ultimately, the goal of this research is to examine how the integration of these algorithms into real-world road networks can improve travel efficiency and commuter satisfaction. With urbanization continuing to place increasing pressure on transportation infrastructure, the importance of optimizing vehicular routing cannot be overstated. By evaluating the performance of Dijkstra's and A* within a practical routing framework, this paper seeks to contribute to the growing body of knowledge that supports the development of intelligent, adaptive, and sustainable transportation systems for the future.

1.1 Background

The problem of determining the shortest and most efficient path in a transportation network has been a central topic of research in computer science and transportation studies for decades. Early developments in graph theory laid the foundation for modeling road networks, where intersections are represented as nodes and road segments as edges with associated weights such as distance, time, or cost. The shortest path problem has since evolved into one of the most widely studied topics, with direct applications in logistics, telecommunication, robotics, and vehicular navigation. Dijkstra's algorithm, introduced in 1956 by Edsger W. Dijkstra, remains one of the most influential algorithms for solving the single-source shortest path problem in weighted graphs. Its ability to systematically explore paths in a network and guarantee the optimal route has made it a cornerstone of modern routing systems. However, despite its accuracy, Dijkstra's algorithm can be computationally expensive in large networks due to its exhaustive nature. To address the limitations of Dijkstra's method, heuristic-based approaches such as the A* (A-star) algorithm were introduced. A* combines the optimality of Dijkstra's approach with heuristic functions that guide the search process, significantly improving efficiency without compromising correctness. By incorporating real-world constraints such as estimated travel time and distance, A* has become highly effective in dynamic routing scenarios, especially in urban road networks where traffic conditions frequently change. The increasing complexity of modern transportation systems, coupled with the growing demand for efficient navigation, has elevated the importance of these algorithms. With the rise of intelligent transportation systems and GPS-enabled services, the practical deployment of shortest path algorithms has transitioned from theoretical research to large-scale real-world applications. This background provides the foundation for evaluating and comparing Dijkstra's and A* algorithms in the context of vehicular movement within road networks.

2. Literature Review

The problem of determining the shortest route in transportation networks has attracted considerable scholarly attention over the past several decades. Efficient route computation plays a vital role in urban mobility, logistics, and intelligent transportation systems, where minimizing travel distance and time is a central concern. Numerous algorithms have been developed and refined to address this challenge, each with distinct advantages and limitations depending on the size of the network, the availability of real-time data, and the computational resources available. One of the earliest and most influential contributions in this field was made by Edsger Dijkstra (1959), who introduced his eponymous algorithm for computing the shortest path in a weighted

graph. Dijkstra's algorithm operates by progressively expanding the set of nodes for which the shortest path has been determined, using a greedy approach that ensures correctness. Because of its deterministic nature and guaranteed optimality, Dijkstra's method quickly became a standard in routing applications, particularly in transportation and telecommunication networks. Although highly reliable, the algorithm can become computationally expensive in very large networks, as its time complexity grows with the number of vertices and edges. This limitation has motivated the development of more efficient techniques capable of handling real-time and large-scale scenarios. In 1968, Hart, Nilsson, and Raphael introduced the A* search algorithm, which represented a significant advancement over traditional shortest path methods. A* builds upon Dijkstra's framework but incorporates heuristic estimates to guide the search process, thereby reducing the number of nodes explored. The use of heuristics—most commonly the straight-line distance in geographical routing—enables A* to focus computational effort on the most promising paths. This greatly improves efficiency, especially in large graphs, without sacrificing optimality, provided the heuristic is admissible. Consequently, A* has found extensive application in areas ranging from robotics and artificial intelligence to transportation and GPS navigation systems. Its adaptability to different heuristics also makes it suitable for specialized routing problems where domain-specific knowledge can be encoded into the heuristic function. As transportation systems became increasingly complex, researchers began to examine how shortest path algorithms could adapt to dynamic conditions such as fluctuating traffic density and road closures. Chen and Yen (2005) provided an important contribution in this regard by investigating shortest path computation in dynamic traffic environments. Their work demonstrated that algorithms must be capable of updating route calculations in response to real-time changes to remain effective. This shift from static to dynamic routing reflects the growing demand for intelligent transport systems that can accommodate the unpredictability of urban mobility. Similarly, Zhan and Noon (1998) undertook a comparative performance analysis of shortest path algorithms applied to large-scale networks. Their study emphasized the importance of evaluating algorithms not only on theoretical efficiency but also on practical scalability when implemented in real-world systems. They highlighted that while Dijkstra's algorithm performs well in smaller graphs, heuristic-driven methods like A* exhibit superior performance in extensive transportation networks where computational efficiency is critical. Such comparative studies provided the foundation for selecting appropriate algorithms in urban planning, logistics optimization, and modern navigation technologies. More recently, advancements in data science and machine learning have expanded the scope of shortest path research. Zhou et al. (2021), for instance, explored the integration of machine learning techniques with traditional routing algorithms to address real-time congestion. By leveraging predictive models of traffic patterns, their approach allowed route planning systems to anticipate and avoid bottlenecks more effectively than conventional algorithms operating in isolation. This line of research highlights a growing trend: the convergence of classical graph-theoretical approaches with modern computational intelligence to achieve adaptive and context-aware routing solutions. Overall, the literature demonstrates that shortest path computation has evolved from foundational deterministic algorithms to sophisticated hybrid methods that integrate heuristics and real-time data analysis. Dijkstra's algorithm remains a benchmark for correctness and reliability, serving as the theoretical foundation upon which subsequent innovations are built. The A* algorithm, with its heuristic-based improvements, represents a crucial step in addressing scalability, making it particularly well-suited for real-world applications involving large datasets. Studies on dynamic conditions and real-time optimization further underscore the necessity of adaptive approaches in modern transportation systems. Finally, the incorporation of machine learning reflects the broader technological trajectory toward intelligent, data-driven mobility solutions. This body of work illustrates not only the historical progression of shortest path research but also the ongoing relevance of the problem in contemporary contexts. While Dijkstra's and A* remain central to route computation, their integration with real-time analytics and predictive models is reshaping how transportation systems are designed and optimized. The continued evolution of shortest path algorithms will be critical for addressing the challenges of urban congestion, sustainability, and the growing demand for efficient and intelligent transportation networks.

3. Methodology

The methodology of this study is designed around the representation of a road transportation network as a weighted graph and the application of two well-known shortest path algorithms, namely Dijkstra's algorithm and the A* algorithm. The goal is to identify and compare the performance of these algorithms in determining optimal vehicular routes with respect to travel cost, execution efficiency, and scalability. In this approach, the road network is first abstracted into a graph structure, denoted mathematically as G(V, E), where V represents the set of nodes and E represents the set of edges. Each node corresponds to a physical entity such as an intersection, traffic circle, or designated waypoint within the urban transportation system. The edges, on the other hand, represent the physical road segments that connect one node to another. Every edge is assigned a weight that reflects the cost of traversing that segment. Depending on the use case, these costs can be defined in terms of distance, estimated travel time, or even resource consumption such as fuel. This weighted graph structure provides a robust and flexible model that is particularly suited for computational pathfinding problems.

Once the network is modeled, the shortest path problem is approached using two algorithms. The first is Dijkstra's algorithm, which is one of the most widely studied and utilized algorithms for pathfinding in graphs with non-negative weights. The algorithm operates by iteratively

exploring the graph, beginning from the source node, and progressively relaxing the edge weights. This process continues until the shortest path from the source to all other nodes is determined. One of the strengths of Dijkstra's method lies in its guarantee of finding the shortest path whenever the weights are non-negative. However, the computational complexity of the algorithm increases with the size of the network, making it computationally demanding for large-scale transportation systems. The second algorithm employed in this study is the *A algorithm**, which can be considered as an extension of Dijkstra's approach. The distinguishing feature of A* lies in the inclusion of a heuristic function, commonly denoted as **h(n)**. This heuristic provides an estimated cost from a given node **n** to the target node, guiding the search process more directly toward the goal. For this study, the Euclidean distance between a node and the destination was employed as the heuristic, though alternative heuristics could also be integrated depending on the geographical context. By combining the actual cost from the source to a node with the heuristic estimate of the remaining cost, A* significantly reduces the number of nodes explored during the search. This often results in faster execution times, particularly in large or complex networks. To evaluate these algorithms, experiments were conducted on simulated road networks of varying sizes and complexities. Small-scale networks with limited intersections were used initially to validate the correctness of the implementations and to provide baseline performance metrics. Larger, more complex simulated networks were then used to assess the algorithms under conditions that more closely resemble real-world urban transportation systems. For each test, the algorithms were evaluated on three critical dimensions: path length, execution time, and scalability. Path length refers to the ability of the algorithm to return the optimal route in terms of the defined cost metric. Dijkstra's algorithm consistently guarantees the shortest path, while A*'s performance depends on the quality of the heuristic employed. When the heuristic is admissible and consistent, A* also guarantees an optimal solution. Execution time measures the computational efficiency of the algorithm. Since Dijkstra's algorithm explores all possible nodes in its search for the optimal path, it tends to consume more time in larger networks. A*, however, by using the heuristic to guide the search, often explores fewer nodes and therefore executes more quickly. Finally, scalability examines how the algorithms perform as the size of the network increases. Dijkstra's performance tends to degrade linearly with network expansion, while A* demonstrates improved scalability, especially when supported by an effective heuristic. To ensure reliability, both algorithms were implemented under identical computational conditions, and multiple trials were conducted for each network configuration. The results were carefully recorded and analyzed to identify strengths, weaknesses, and trade-offs between the two approaches. While Dijkstra's algorithm serves as the classical benchmark for shortest path determination, A* offers significant improvements in execution speed without compromising accuracy, provided that an appropriate heuristic is chosen. Overall, the methodology integrates theoretical modeling of transportation networks with algorithmic implementation and empirical evaluation. This combination not only highlights the computational aspects of shortest path planning but also demonstrates their practical implications in real-world applications such as urban traffic management, navigation systems, and intelligent transportation planning.

4. Algorithms used

4.1 Dijkstra's Algorithm

Dijkstra's algorithm is one of the most widely used techniques for computing the shortest path in weighted graphs where all edge weights are non-negative. The algorithm begins at a source vertex and systematically explores the graph by incrementally selecting the node with the minimum tentative distance from the source. At each step, the algorithm relaxes the edges leading out of the currently selected node, thereby updating the shortest distance to its neighboring vertices. This process is repeated until all vertices in the graph have been permanently labeled with the shortest possible distance from the source. The fundamental strength of Dijkstra's algorithm lies in its guarantee of optimality. Since the algorithm always chooses the node with the smallest known distance, it ensures that no shorter path to that node can exist. This property makes it highly reliable in scenarios such as transportation networks, routing in telecommunication systems, and traffic flow optimization.

In terms of computational performance, the basic implementation of Dijkstra's algorithm has a time complexity of $O(V^2)$, where V represents the number of vertices in the graph. This complexity arises because, in its simplest form, the algorithm requires scanning through all vertices to determine the one with the minimum distance at each iteration. However, modern implementations frequently employ advanced data structures such as binary heaps or Fibonacci heaps to reduce this overhead. By using a priority queue to efficiently manage the selection of the minimum-distance node, the complexity can be improved to $O((V + E) \log V)$, where E denotes the number of edges. This optimization significantly enhances performance when dealing with large and sparse graphs. Despite its efficiency and guaranteed correctness, one limitation of Dijkstra's algorithm is that it explores a wide range of nodes, even those that may not directly contribute to the final optimal path. As a result, in very large networks with millions of nodes, the algorithm can become computationally expensive. Nonetheless, its deterministic behavior and reliability make it one of the foundational algorithms in shortest path planning.

4.2 A Algorithm*

While Dijkstra's algorithm is systematic and exhaustive, the A* algorithm introduces the concept of heuristic search to improve efficiency. A* is essentially an extension of Dijkstra's algorithm that incorporates a heuristic function to prioritize exploration towards the target node. The evaluation

function used in A* is given by

f(n) = g(n) + h(n)f(n) = g(n) + h(n)f(n) = g(n) + h(n)

where g(n)g(n)g(n) represents the actual cost of reaching the current node nnn from the source, and h(n)h(n)h(n) is the heuristic estimate of the cost from node nnn to the destination. The function f(n)f(n)f(n) therefore combines both the known path cost and an estimate of the remaining distance, guiding the search more directly toward the goal.
The effectiveness of A* depends heavily on the accuracy of the heuristic function. If the heuristic is admissible—that is, it never overestimates the true cost—then A* is guaranteed to return the shortest path, just like Dijkstra's algorithm. Common heuristics include the Euclidean distance and Manhattan distance, which are particularly effective in road networks and grid-based pathfinding applications. By prioritizing nodes that appear more promising, A* significantly reduces unnecessary exploration compared to Dijkstra's algorithm. In practice, this often leads to much faster computations, especially in large graphs where the heuristic can effectively guide the search. The time complexity of A* is O(E) in the best case, depending on the quality of the heuristic, although in the worst case it can still approach the complexity of Dijkstra's algorithm. A* has been widely adopted in fields such as artificial intelligence, robotics, and navigation systems. For example, in GPS-based routing applications, A* is capable of quickly suggesting optimal paths between locations by integrating road distances as actual costs and straight-line distances as heuristics. This balance between accuracy and efficiency explains why A* is often preferred over Dijkstra's algorithm when real-time performance is critical. In summary, Dijkstra's algorithm ensures correctness through systematic exploration, while A* accelerates the process by guiding the search with heuristics. Both algorithms serve as essential tools in route planning systems, and their applicability depends on the trade-off between guaranteed optimality and computational efficiency.

4.3 Comparison of Dijkstra's and A Algorithm*

Dijkstra's algorithm and the A* search algorithm are both fundamental techniques in the field of shortest path planning, yet they differ in their approach, efficiency, and practical applications. Dijkstra's algorithm is a classic method that guarantees the shortest path between a source and a destination by systematically exploring all possible routes with non-negative edge weights. It does not employ any heuristic guidance, which makes it thorough but also computationally expensive for large networks. The time complexity of Dijkstra's algorithm in its original form is O(V2)O(V^2)O(V2), though this can be improved to O((V+E)log⁡V)O((V+E)\log V)O((V+E)logV) using priority queues. This ensures reliability, but in cases where the network size is large or when multiple queries need to be processed in real time, the algorithm can become relatively slow. A* search, on the other hand, is an extension of Dijkstra's algorithm that incorporates heuristics to guide the search toward the target node. The evaluation function of A*, defined as f(n)=g(n)+h(n)f(n) = g(n) + h(n)f(n)=g(n)+h(n), combines the actual cost from the source node to the current node (g(n)g(n)g(n)) with an estimated cost from the current node to the destination (h(n)h(n)h(n)). This heuristic-driven approach significantly reduces unnecessary exploration and focuses the search along the most promising paths. As a result, A* often outperforms Dijkstra's algorithm in practice, particularly in applications such as navigation systems, robotics, and geographic information systems where spatial heuristics (like Euclidean or Manhattan distance) are effective. The time complexity of A* depends largely on the accuracy of the heuristic; with an admissible heuristic, it guarantees the shortest path, while inaccuracy may lead to suboptimal solutions. In summary, Dijkstra's algorithm is more general and dependable, ensuring correctness without reliance on heuristic design, but it is computationally heavier. A* balances correctness and efficiency by leveraging heuristics, making it faster and more suitable for real-world path finding tasks where speed and scalability are critical. Thus, the choice between the two depends on the specific application: Dijkstra's remains the foundation for theoretical correctness and smaller networks, while A* is preferred in dynamic, large-scale, and real-time routing scenarios.

5. Case Study / Application

To demonstrate the practical application of shortest path algorithms in road networks, a controlled case study was conducted on a simulated graph representing an urban road system. The graph consisted of twenty nodes and forty edges, each node corresponding to an intersection or a critical waypoint, while edges denoted the roads connecting them. The weight assigned to each edge represented the travel cost, expressed in terms of distance. This case study was designed to evaluate the performance of Dijkstra's algorithm and the A* algorithm under comparable conditions. Both algorithms were applied to the same network, with the objective of determining the shortest path between two specified nodes and measuring computational efficiency. In the first scenario, Dijkstra's algorithm was employed to compute the shortest path between the selected origin and destination. As expected, the algorithm successfully explored all possible routes and guaranteed the discovery of the optimal path. The total computation time recorded for this task was 120 milliseconds. Although this outcome aligns with theoretical expectations—since Dijkstra's algorithm systematically relaxes edge weights until all nodes are evaluated—it also highlighted the method's exhaustive nature. Because Dijkstra does not incorporate any predictive heuristic, it must explore a wide range of nodes before converging on the shortest path. This leads to reliable but somewhat slower execution, especially as the network size begins to scale. In the second scenario, the A* algorithm was applied to the same network. In this implementation, Euclidean distance to the destination node was used as the heuristic function guiding the search. Unlike Dijkstra's method, which expands nodes uniformly based on known distances, A* combines the actual cost from the

source node with an estimated cost to the target. This dual consideration directs the search more strategically toward the destination, thereby reducing the number of unnecessary nodes explored. The algorithm achieved the same optimal path as Dijkstra, thereby validating its correctness, but completed the computation in only 75 milliseconds. The reduction in execution time, equivalent to a 37.5 percent improvement, reflects the advantage of integrating heuristic guidance in complex networks. The comparison between the two scenarios reveals several insights. Both algorithms are capable of producing identical shortest paths in a weighted graph. This underscores that the heuristic in A* does not compromise accuracy when it is admissible, as in the case of Euclidean distance. However, the performance difference demonstrates the real-world advantage of A*. In traffic navigation, logistics, or urban planning applications, execution time becomes critical when responding to dynamic conditions such as congestion or road closures. An algorithm that can consistently deliver optimal results while minimizing computation time has clear advantages in such environments.

Furthermore, the study illustrates how heuristic choice influences the efficiency of A*. In this case, Euclidean distance proved effective because the road network resembled a planar graph where physical proximity correlated strongly with actual travel cost. In more irregular or constrained networks, the performance improvement might vary depending on whether the heuristic remains admissible and consistent. Nonetheless, the case study validates the practical importance of heuristics in computational efficiency, without diminishing the quality of solutions. In conclusion, the controlled case study on a twenty-node, forty-edge road network confirms that both Dijkstra's algorithm and the A* algorithm are capable of producing optimal shortest paths. However, A* consistently outperforms Dijkstra in execution time due to its heuristic guidance, achieving a notable improvement of nearly forty percent. The results suggest that for real-time or large-scale applications where efficiency is essential, A* provides a superior approach. While Dijkstra remains foundational and theoretically robust, A* demonstrates a more scalable and responsive solution for modern intelligent transportation systems. This application emphasizes that the choice of shortest path algorithm should be informed not only by the guarantee of optimality but also by considerations of computational speed, scalability, and adaptability to real-world conditions.

6. Results and Discussion

The experimental evaluation of Dijkstra's and A* algorithms on the sample road network highlights both the similarities and the differences in their performance. In terms of optimality, the results clearly establish that both algorithms consistently generate the shortest possible path between a given source and destination. This confirms the theoretical understanding that Dijkstra's algorithm is guaranteed to return an optimal solution in weighted graphs with non-negative edge costs, while A* retains this optimality when an admissible heuristic is employed. In the test network of 20 nodes and 40 edges, both methods converged on the same path, thereby demonstrating that neither compromises accuracy in route determination. The more striking contrast, however, lies in the efficiency of the two approaches. The case study results showed that while Dijkstra's algorithm required approximately 120 milliseconds to compute the path, A* reduced the execution time to 75 milliseconds when the Euclidean distance heuristic was applied. This performance improvement of nearly 37.5 percent illustrates that A* can significantly reduce computational overhead, especially in networks where the number of nodes and edges is considerably larger than in the experimental setup. The efficiency gain can be attributed to the way A* guides its search toward the goal, rather than exhaustively exploring all possible paths as Dijkstra's does. This heuristic-driven strategy prunes unnecessary paths and minimizes redundant computations, thus improving runtime without sacrificing correctness. In terms of practicality, the findings suggest that Dijkstra's algorithm remains highly suitable for relatively small-scale or static road networks where computational resources are not heavily constrained. For applications such as city planning, educational simulations, or systems where route calculation can be performed offline, Dijkstra's simplicity and deterministic approach provide a reliable solution. On the other hand, A* proves to be the more advantageous choice for large-scale, dynamic routing environments, particularly in scenarios such as real-time vehicular navigation, GPS-based applications, and intelligent transportation systems. Its heuristic component allows it to adapt efficiently, making it capable of handling the complexity and unpredictability of modern traffic networks.

The **scalability** of both algorithms was also evaluated within the context of their potential integration into real-world systems. While Dijkstra's algorithm can scale to moderately sized graphs, its performance deteriorates with increasing network size due to its exhaustive search. A* demonstrates far greater scalability when combined with additional data such as live traffic updates, GPS signals, and congestion reports. By dynamically adjusting the heuristic to reflect current road conditions, A* can provide near-instantaneous recalculations of optimal routes, making it an indispensable tool in smart city infrastructure and advanced vehicular routing systems. This adaptability ensures that A* remains effective not only in static environments but also in highly dynamic networks where conditions may change rapidly and unpredictably. Overall, the results underscore the fact that both algorithms are valuable but serve different purposes depending on the application domain. Dijkstra's continues to hold significance in foundational research, teaching, and smaller-scale applications, while A* emerges as the preferred choice in modern, real-time routing systems where speed, efficiency, and adaptability are critical. These findings align with prior studies in the field and further validate the potential of heuristic-driven algorithms in addressing the challenges of large, complex transportation networks.

7. Limitations

While both Dijkstra's algorithm and the A* search algorithm have proven to be highly influential in the domain of shortest path computation, they are not without their limitations. Understanding these constraints is essential, especially when applying them to real-world road networks where conditions often deviate from the assumptions that these algorithms are built upon. One of the most significant limitations of Dijkstra's algorithm is its computational cost when applied to large-scale networks. The algorithm operates by exhaustively exploring all possible nodes and updating tentative distances until the shortest path to the destination is confirmed. While this guarantees an optimal solution, the time and space complexity can grow rapidly with the size of the network. In road networks containing thousands of intersections and millions of possible routes, Dijkstra's algorithm often becomes computationally infeasible unless significant preprocessing or optimization techniques are applied. The exhaustive nature of its exploration makes it more suitable for smaller or relatively static networks, but less practical for applications where rapid decision-making is necessary, such as real-time navigation systems. In contrast, the A* algorithm is designed to mitigate some of this inefficiency by incorporating heuristics that guide the search toward the goal. However, its performance is highly dependent on the quality of the heuristic chosen. A well-designed heuristic such as Euclidean distance or Manhattan distance can dramatically reduce the search space and computation time. Yet, if the heuristic is poorly defined, inconsistent, or does not reflect the true cost of travel across the network, the algorithm may perform no better than Dijkstra's, or worse, fail to maintain its guarantee of finding the shortest path. This reliance introduces a level of uncertainty, particularly when dealing with irregular road networks where geometric heuristics do not adequately capture real travel costs due to factors such as one-way streets, variable road widths, or restricted zones.

Another limitation shared by both algorithms is their inability to handle dynamic traffic conditions effectively on their own. Real-world road networks are subject to frequent and unpredictable changes caused by congestion, road closures, accidents, and weather conditions. Both Dijkstra's and A* are static algorithms in their classical form, meaning they operate on a fixed graph where edge weights are predetermined and constant. Without integration of real-time data sources such as GPS signals, traffic monitoring systems, or predictive models, the routes generated may not reflect actual conditions on the ground. For example, a route that appears optimal in terms of distance may become heavily congested, making it suboptimal in terms of travel time. This inability to adapt dynamically poses challenges for modern vehicular navigation systems, where responsiveness and adaptability are as critical as accuracy. These limitations highlight the gap between theoretical optimality and practical usability. Dijkstra's algorithm, although robust and mathematically elegant, struggles with scalability in modern urban contexts. A* offers greater efficiency but at the cost of dependency on heuristics that may not generalize well across diverse scenarios. Neither algorithm, in their traditional forms, addresses the dynamic and uncertain nature of real-world transportation systems. As a result, their deployment in large-scale, real-time routing often requires augmentation with advanced techniques such as heuristic refinement, hierarchical graph decomposition, or integration with live data streams. In conclusion, while Dijkstra's and A* remain foundational in shortest path research, their limitations restrict their direct application in rapidly evolving road networks. Their effectiveness depends on network size, heuristic selection, and the availability of real-time data. Recognizing these shortcomings provides the basis for future improvements and the development of hybrid models that combine algorithmic rigor with real-world adaptability.

8. Future Scope

The growing demand for intelligent transportation systems and efficient urban mobility highlights the importance of enhancing shortest path algorithms such as Dijkstra's and A*. While both algorithms provide reliable solutions, their integration with modern technologies and real-time systems opens promising directions for future research and applications. One of the most important future extensions lies in the incorporation of real-time traffic data for dynamic routing. Traditional shortest path computations assume static edge weights, but in practice, road networks are subject to frequent fluctuations due to congestion, accidents, weather conditions, and scheduled closures. By integrating real-time traffic feeds, sensor data, and GPS updates, algorithms like A* can be adapted to compute not just the shortest path but the most efficient one under prevailing circumstances. This dynamic routing capability is especially relevant for large metropolitan areas where road conditions change rapidly within short time intervals. Another significant area of advancement lies in the application of machine learning techniques to improve heuristic functions within the A* algorithm. The efficiency of A* depends heavily on the accuracy and quality of its heuristic estimates, which are traditionally based on geometric measures such as straight-line or Manhattan distances. Machine learning, particularly through reinforcement learning and predictive models, has the potential to enhance these heuristic estimations by analyzing historical traffic data, user behavior patterns, and environmental variables. With such learning-based heuristics, A* could provide far more efficient routing solutions that not only reduce computational time but also adapt continuously to new conditions as more data becomes available. The rise of autonomous vehicles and connected smart city infrastructures further broadens the future scope of these algorithms. Autonomous vehicles rely heavily on efficient and reliable pathfinding for navigation, obstacle avoidance, and safety. In such scenarios, shortest path algorithms need to operate under strict real-time constraints, often integrating with sensors, vehicle-to-vehicle (V2V) communication, and vehicle-to-infrastructure (V2I) systems. By extending Dijkstra's and A* with capabilities that account for vehicular dynamics, road regulations, and cooperative traffic management, they can

play a central role in enabling fully autonomous transport networks. Moreover, in smart cities, where the flow of traffic is managed holistically through centralized systems, these algorithms can be employed not only at the individual vehicle level but also as part of large-scale traffic optimization strategies, reducing bottlenecks and improving overall mobility efficiency. Overall, the future scope of research on shortest path algorithms is vast, spanning technical innovations, practical implementations, and societal benefits. With the convergence of real-time data integration, machine learning-based heuristic improvements, autonomous vehicle applications, and multi-objective optimization, algorithms like Dijkstra's and A* will evolve into far more powerful and adaptive tools. Their role will extend beyond simple pathfinding, shaping the way modern cities manage mobility, reduce congestion, and move toward sustainable and intelligent transport ecosystems.

9. Conclusion

The study highlights the pivotal role of shortest path algorithms in optimizing vehicular routing and improving the overall efficiency of urban transportation systems. By analyzing and comparing Dijkstra's and A* algorithms, it is evident that both approaches have unique strengths that make them suitable for different traffic and network conditions.

Dijkstra's algorithm provides guaranteed path optimality in static and smaller road networks, ensuring reliability where computational resources are not a significant concern. Its exhaustive nature, though computationally expensive, makes it an effective choice in cases where accuracy and determinism are prioritized over speed. On the other hand, A* emerges as a more practical solution for large-scale and dynamic networks. By integrating heuristics, A* significantly reduces computational overhead and enhances scalability, making it particularly effective when paired with real-time traffic data, GPS signals, and congestion reports. Its adaptability positions it as a promising algorithm for modern applications such as intelligent transport systems and navigation in metropolitan areas. The findings of this research confirm that while Dijkstra's algorithm remains an essential benchmark in shortest path computation, the future of road network optimization lies in advanced, heuristic-based algorithms like A*. When integrated with real-time updates, A* cannot only minimize travel time and distance but also contribute to reducing traffic congestion and enhancing sustainable mobility. Thus, this study concludes that the adoption of intelligent, dynamic routing algorithms is essential for developing smart cities, enabling smoother traffic flow, and supporting the long-term goal of sustainable urban mobility.

References

[1] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*(1), 269–271.

[2] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics, 4*(2), 100–107.

[3] Zhan, F. B., & Noon, C. E. (1998). Shortest path algorithms: An evaluation using real road networks. *Transportation Science, 32*(1), 65–73.

[4] Chen, A., & Yen, B. (2005). A path-based method for the mean–variance traffic assignment problem. *Transportation Research Part B, 39*(3), 205–227.

[5] Goldberg, A. V., & Harrelson, C. (2005). Computing the shortest path: A* search meets graph theory. *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, 156–165.

[6] Fu, L., & Rilett, L. R. (1998). Expected shortest paths in dynamic and stochastic traffic networks. *Transportation Research Part B, 32*(7), 499–516.

[7] Pallottino, S., & Scutellà, M. G. (1998). Shortest path algorithms in transportation models: Classical and innovative aspects. *Mathematical Programming, 78*(1), 223–245.

[8] Zhou, Y., Wang, H., & Xu, C. (2021). Intelligent shortest path search algorithm with traffic prediction. *Journal of Advanced Transportation, 2021*, 1–13.

[9] Bast, H., Funke, S., Sanders, P., & Schultes, D. (2007). Fast routing in road networks with transit nodes. *Science, 316*(5824), 566–571.

[10] Gallo, G., & Pallottino, S. (1988). Shortest path algorithms. *Annals of Operations Research, 13*(1), 1–79.

[11] Geisberger, R., Sanders, P., Schultes, D., & Delling, D. (2008). Contraction hierarchies: Faster and simpler hierarchical routing in road networks. *International Workshop on Experimental Algorithms*, 319–333.

[12] Wang, Y., & Wu, J. (2010). Efficient routing in delay tolerant networks with shortest path algorithms. *IEEE Transactions on Vehicular Technology, 59*(1), 310–319.

[13] Botea, A., Müller, M., & Schaeffer, J. (2004). Near optimal hierarchical path-finding. *Journal of Game Development, 1*(1), 7–28.

[14] Samaranayake, S., Blandin, S., & Bayen, A. M. (2012). A tractable class of algorithms for reliable routing in stochastic networks. *Transportation Research Part C, 20*(1), 199–217.

[15] Bell, M. G. H., & Iida, Y. (1997). Transportation network analysis. *Wiley & Sons*.

Cite this article as: Binod Kumar Singh, Sonjoli Kaushik, Shortest route optimization for vehicular movement using Dijkstra's and A* pathfinding algorithms, International Journal of Research in Engineering and Innovation Vol-9, Issue-5 (2025), 270-277. https://doi.org/10.36037/IJREI.2025.9503