International Journal of Computational and Experimental Science and ENgineering
(IJCESEN)
Vol. 11-No.3 (2025) pp. 5713-5720
http://www.ijcesen.com
Research Article
ISSN: 2149-9144

Improved Yen's Algorithm for 2nd-Shortest Path Problem

Junlin Tian*

Lanzhou university-China
* **Corresponding Author Email:** tianjl2023@lzu.edu.cn - **ORCID:** 0000-0002-1409-9342

Article Info:

DOI: 10.22399/ijcesen.3524
Received : 22 May 2025
Accepted : 23 July 2025

Keywords

Path Optimization,
Graph theory,
k-shortest path problem,
Yen's algorithm,
A star algorithm,
multi-variable Optimization

Abstract:

This paper introduces an algorithm designed to find the shortest and second shortest paths between two nodes in a network. Path optimization plays a crucial role in various fields, such as urban road management, network communication, and traffic planning. Numerous algorithms and insights inspired by machine learning have been developed, leading to valuable outcomes in these areas. However, many existing algorithms are mainly focused on minimizing path duration, and they often neglect other important factors such as cost, motor heating, and battery level, which can substantially affect the process. This study presents an enhancement to Yen's algorithm with the A star algorithm, aiming not only to achieve a shorter time but also to reduce the associated costs. In many practical scenarios, algorithms that target the absolute shortest travel time often incur higher costs. Consequently, the second shortest path identified by Yen's algorithm is valuable because it not only meets the requirement for a shorter travel time but also tends to incur lower costs compared to the shortest path. Then this work try to proof this model can be a reduction of Knapsack problem.This study uses the main roads in Lanzhou City as an example, using network topology to establish a coordinate system. By considering the time(t) and cost(c) associated with different transportation options, this study aims to develop a solution that more accurately reflects practical conditions and expected outcomes. The results indicate that the proposed algorithm is more effective at meeting the time and cost requirements compared to the original algorithm.

1. Introduction

Path optimization is a aimed at determining the of steps from a starting point to a destination. This process involves considering various constraints and associated costs while its objective is to minimize various factors, such as distance, time, cost, energy usage, network density, and speed, or a combination of these factors[1] to satisfying specific constraints. This problem has been observed in several real- world applications where path optimization can applied, such as robotics and transportation (e.g., logistics, telecommunica- tions, gaming, and autonomous driving)[2, 3].

Path planning and optimization are crucial in robotics be- cause they enable autonomous robots to effectively navigate complex and dynamic environments. When determining the most efficient path for task completion, robots must consider various factors, including obstacles, terrain, and interactions with humans[4]. For instance, this is particularly signifi- cant in warehouse automation, where robots must navigate

Junlin Tian is with the School of Mathematics and Statistics, Mas- ter Candidate, Lanzhou University, Lanzhou 730000, China (e-mail: tianjl2023@lzu.edu.cn).

aisles, avoid collisions, and optimize picking routes. Robotic workspaces often contain various potential dangers that robots must navigate. However accurately determining the robot's exact positions is often impractical and not cost-effective. To address this issue, a multi-objective optimization (MOO) algorithm [5] based on particle swarm optimization (PSO) was developed for robot navigation. Furthermore, an intelligent optimization algorithm was designed[6] to facilitate multi- objective path optimization in industrial welding robots. With the help of artificial intelligence and machine learning, existing optimization algorithms have been able to solve most of the problems. However, when handling large-scale dynamic networks, multi-objective optimization, and real-time path recalculations, the performance of exciting algorithms will be substantially degraded. When solving optimization problems related to large-scale datasets, the methods used have to be very simplified due to the arithmetic limitations of the computing devices. Bai[7] propose a bilevel variable grouping (BLVG)-based framework to scaled back the data into two variable cells which can effectively reduce the computing time complexity. However, this subcomponents algorithm is only applicable when the point set size is large. It is still not very practical for large-scale data processing and complex interrelationships. Graph databases are a type of NoSQL database that stores data in nodes, edges, and properties to represent and structure information. Its advantages can easily fix the problems with complex queries and rich data models. Unfortunately, the graph query is hard due to the NP-complete nature of subgraph isomorphism. Zhao and his fellew[8] present a high performance graph indexing mechanism, SPath, to address the graph query problem on large networks in 2010. It performance very well but many large networks change rapidly over time, such that incremental update of graph indexing structures becomes important.

In the field of transportation, path optimization problems based on urban transportation networks are widely considered as MOO tasks. Real-time traffic fluctuations, road closures, or multi-modal transport options impact optimization tasks more difficult. In real-world scenarios, factors such as travel time and other considerations, like remaining battery level, are crucial when selecting between different transportation op- tions. Given that congestion in urban transportation networks increases emissions, Nagurney[9] proposed a more efficient approach. E. Mandl[10]introduced a heuristic algorithm that effectively calculates distances and routes in large networks. For individuals with disabilities, Ferrari et al.[11] proposed an approach that leverages network science and spatio-temporal analysis to identify accessible routes. To mitigate the impact of crisis events, Chen et al. proposed a dynamic road network model based on Dijkstra algorithm [12] to facilitate vehicle evacuation.

When navigating an urban transportation network, the pri- mary considerations are typically the time and cost of reaching the destination. To determine the shortest path in a network flow, the Ford-Fulkerson[13] algorithm, which is based on the principle of maximum flow minimum cut, is a widely used approach. In recent years, more advanced algorithms such as Dijkstra and A*[14], and more sophisticated methods such as genetic algorithms and PSO, have been applied to address these problems. Wu[15] employed an enhanced Dijkstra al- gorithm to select vehicle routes in urban traffic. Considering potential disruptive events, the concept of transportation net- work resilience has emerged as a critical factor that influences path optimization outcomes[16]. Nevertheless, each existing path optimization algorithm has limitations. Lu[17] analyzed and summarized the strengths and weaknesses of current mainstream approaches, and also outlined future trends for static and dynamic path-planning algorithms. In some path optimization scenarios, traveling along the shortest path yields the optimal solution. However, real-world conditions can complicate this viewpoint because the shortest path often requires more resources. In situations where energy is limited, minimizing energy consumption to achieve the tar-get becomes a significant challenge. To address this problem,this study proposes a multivariate optimization problem. This study presents a combination of Yen's algorithm and the A star algorithm, aimed at minimizing energy expenditure or cost while ensuring shorter travel times. A proof that the problem can be seen as the reduction of the knapsack problem.

The primary contributions of this study are as follows:

- An improved Yen's algorithm combined with the A* algorithm is proposed to address path optimization in existing methods, making it more suitable for the 2nd- Shortest Path Problem (one component of k-shortest path problem(KSP)) in scenarios with a high number of paths and nodes.
- To address the challenge that the shortest distance often results in the highest cost in real-world situations, we propose a model that mitigates the limitations associated with traditional methods. Then proof this model can reduction to the Knapsack problem.
- The proposed algorithm was validated through experi- ments conducted on the urban road management of major roads in Lanzhou City, and the results demonstrated that the framework significantly enhances path optimization outcomes.

2. Related algorithm

This essay use 2nd-Shortest Path Problem to demonstrate the balance between time consumption and money cost in ur- ban traffic problem. The whole strategy of this demonstration is shown by figure 1.

Rapid advancements in computer technology and deep learning have made applying such techniques to traffic man- agement problems a prominent research area. The Dijkstra algorithm and its derivative, the A star algorithm, are widely recognized as a solution for the acyclic shortest path problem with non-negative weight. Originally introduced in 1968 by Peter Hart et al. at the Stanford Research Institute, the A star algorithm [18] is considered an extension of the Dijkstra algorithm. In addition, incorporating heuristic functions into the A star algorithm has been demonstrated to significantly improve its overall performance compared to the Dijkstra algorithm, particularly in networks with many edges.

First, it is essential to understand the function used to compute the priority of each node in the A star algorithm:

$$f(n) = g(n) + h(n) , \qquad (1)$$

where f(n) represents the general priority of node n. When selecting the next node to traverse, the node with the highest priority (i.e., the one with the smallest value of f(n)) is chosen. Here, g(n) denotes the cost of node n from the starting point and h(n) represents the expected cost of node n from the endpoint, which is the heuristic function of the A star algorithm.

Typically, two methods are employed when calculating distances. The first is the Manhattan distance, which is used when only four directions are allowed in the graph: up, down, left, and right. The second is the Euclidean distance, which is applicable when any direction is allowed in the graph. There are two notable special cases: when g(n)= 0, which indicates that the algorithm is incorrect and degenerates into a greedy algorithm that may not yield the shortest; and when h(n)=0, which indicates that the algorithm has no heuristic function and reverts to the Dijkstra algorithm. As the value of h(n) decreases, the algorithm traverses a larger number of nodes, which results in a slower algorithmic process.

The KSP concept was first proposed by Hoffman and Pavley in 1959[19]. KSP is generally categorized into two types: restricted KSP and unrestricted KSP. The restricted KSP requires that the set of shortest paths must not contain any loops, whereas the unrestricted KSP does not impose such a limitation on the identified shortest paths. In 1971, Yen [20] reviewed the computational requirements and memory addresses related to algorithms for determining the k-shortest loopless paths. Based on this analysis, he introduced a new algorithm known as Yen’s algorithm.

Yen’s algorithm comprises three distinct phases. In the first phase, the shortest path, denoted as P(1), is calculated. The remaining k-1 shortest paths are then calculated sequentially, with each subsequent path depending on the previous one. In the second step, the (i+1)th shortest path, P(i+1), is determined by considering all nodes on P(i), except the terminal node, as deviation nodes. The algorithm then calculates the shortest path from each deviation node to the terminal node. The shortest deviation path is determined by integrating the previous paths from the initial node to the deviation nodes on P(i) to form a potential path. Let N denote the number of nodes and M represent the number of edges. Consequently, the time complexity isO(NM*log(N+M)+M^2[20] , which can be effectively managed based on M. Nevertheless, the number of edges on the shortest path is significantly smaller than the number of edges in the graph, which results in more accurate outcomes than expected. To address this limitation, Aljazzar [21] introduced a heuristic approach with an on-the-fly search known as the “A star algorithm.” The time complexity of this algorithm is O(M+NlogN+k), where k represents the order of one path in the sequence of the shortest paths.

3. An improved yen’s Algorithm

A. Overall Structure

The proposed algorithm for optimizing urban traffic paths is described below. Algorithm1 and Algorithm2 are the original algorithms to find the shortest path that presented in article[14]. The algorithm3 is the main idea of Yen’s algorithm[20] to remove the specified edge from the optimal path and find the optimal path. Last two algorithms are conventional Yen’s algorithm and the improvement of it proposed by this work.

Algorithm 1 Heuristic Estimate For A* Algorithm.[14]

INPUT: START POINT AND DESTINATION POINT
Compute the Euclidean distance between the start point and destination point
OUTPUT: DISTANCE

Algorithm 2 A star algorithm.[14]

INPUT: START POINT AND DESTINATION POINT
Initialize start as node with start_id
Initialize end as node with end_id

Initialize open_set as an empty priority queue, visited as an empty set
Push (0,start.id,empty_path, 0, 0) into open_set While open_set is not empty, do
(f_n, current_id, path, total_time, total cost) = open_set
If current_id is in visited, do
Continue to the next iteration.
End if
Add (current_id, None) to path
Add current_id to visited
If current_id is equal to end.id, do
Obtain the shortest path, total time and total_cost
"Terminal!"
End if
If current_id is in self.edges ,do
for each edge in self.edges [current id], do
Update the last element of path to (current_id, edge)
Calculate heuristic as the estimated cost
from current node to end using algorithm 1 Calculate new_time = total_time+edge.time Calculate new cost = total_cost + edge.cost
Calculate f_n = new_time + heuristic
Update (f_n, edge.end, updated_path, new_time,new cost) into open_set
End for
End if
End while
"the shortest path is not exists!"
OUTPUT: THE SHORTEST PATH

Algorithm 3 Remove the specified edge from the optimal path and find the optimal path for the current graph.

INPUT: GRAPH , start_id, end_id, edge_to remove
Remove edge_to_remove from graph
Calculate the shortest path, total time, and total cost for the current graph using algorithm 2.
OUTPUT: SHORTEST PATH , TOTAL TIME , TOTAL COST

Algorithm 4 Conventional Yen's algorithm for finding the top 2 shortest paths.

Set second_shortest_path = Null
Set second_total_time = infinity
Set second_total_cost = infinity
Calculate the shortest path for the original graph using conventional Yen's algorithm.
OUTPUT: SHORTEST PATH , SHORTEST TIME , SHORTEST COST, SECOND SHORTEST PATH , SECOND TOTAL TIME , SECOND TOTAL COST.

Algorithm 5 Improved Yen's algorithm for finding the top 2 shortest paths.

Calculate the shortest path for the original graph
using algorithm 2.
Set second_shortest_path = Null
Set second_total_time = infinity
Set second_total_cost = infinity
for each edge in shortest path, do if edge[1] is not Null, do
Generate graph g by deeply copying the original
graph, and removing the current edge.
Calculate the shortest path, total time and total cost of the current graph g using algorithm 2.
if total_time < second_total time or (total_time == second_total_time and total_cost < second_total_cost), do
Update second_shortest_path = path Update second_total time = total_time
Updatesecond_total_cost = total_cost
end if
end if
end for
OUTPUT: SHORTEST PATH , SHORTEST TIME , SHORTEST COST, SECOND SHORTEST PATH , SECOND TOTAL TIME , SECOND TOTAL COST.

B. Enhanced Algorithm

The main idea of algorithm 5 is to combines Yen's algorithm with the A star algorithm. Yen's algorithm determines each step of the shortest path by the Dijkstra algorithm. Conversely, in the context of urban traffic road path optimization, the number of roads and nodes substantially increases. The traditional Dijkstra algorithm cannot obtain an optimal solution in a timely manner. Therefore, the A star algorithm is considered a suitable alternative for such operations. In addition, the flexibility of the A star algorithm facilitates a trade-off between the time required and the cost of the results. Calculations revealed the time complexity of the new algorithm is determined to be $O(N^2M * logN)$ (N points by M edges and the number of iteration is ($NlogN$), indicating that it effectively eliminates the influence of the number of edges in the original algorithm. Furthermore, a comparison of the computational times revealed the significant impact of the improved algorithm on overall time complexity.

In addition, the disparity between the theoretical and prac- tical applications of the algorithms in real-world scenarios requires further exploration. The primary challenge lies in the balance between time and resource expenditure, as the shortest time to a destination often requires the highest resource utiliza- tion. Consequently, a viable strategy involves minimizing costs while reducing time or distance. This study proposes selecting the path with the lowest cost among the shortest paths. When the shortest paths have the same cost, the second shortest path with the lowest cost is selected.Based on the previous elaboration, it can be concluded that the time complexity of the improved algorithm is related to the scale of the input (number of vertices). This suggests that it falls into the category of NP-complete problems. It follows that the problem posed in this paper is potentially reducible to a known problem.

4. Reduction to the knapsack problem

The Knapack Problem is a well-known optimization prob- lem that models a scenario where you have a set of items, each with its own weight and value, and a knapsack with a maximum weight capacity. The objective is to select items to include in the knapsack such that the total value of the chosen items is maximized, but the total weight does not exceed the knapsack's capacity. It finds applications in resource alloca- tion, portfolio optimization, and scheduling, among others.

For the optimal problem raise by this article, let the the battery or the fuel volume a constant number G. Faster and shorter transportation will be seen by the items with higher weight. Given two values T_i , C_i and D_i to transportation i to denote the travel time T, cost C and the distance D_i. The exact ruduction function can be set as equation 2.

$$g_i = \frac{T_i}{D_i} \quad (2)$$

where gi denoted by the weight of transportation i. Then the problem is reducted into a knapsack problem with the maximum weight capacity G and the value of each item is Ci. By rnduction, it can be obtained that the problem presented in this paper is NP-complete, which is consistent with the previous analysis on the time complexity of the improved algorithm.

5. Experiment

A. Dataset

As a typical river-valley city,the unique geographical en- vironment and complex traffic of Gansu Lanzhou layout bring great challenges to traffic management. Given that the enhanced algorithm exhibits reduced sensitivity to the number of edges M, and considering that urban paths can typically be represented with a limited number of nodes (denoted as N) and numerous traffic paths (denoted as M), this study presents an urban traffic path optimization problem as an experimental case. The data and graphs used in this experiment were derived from a two-dimensional coordinate system of major roads in Lanzhou City. All the data are from the publicly available datasets [22] and China Digital Elevation Map[23] . The origin of the coordinates can be any node in the graph. Seven well- known locations were selected as nodes in the graph, with node A denoted as the initial point and node G as the final point. The types of transportation used between various points, travel time, and associated costs were provided. As illustrated in Figure 2, a binary array is used to represent the duration of time spent and the costs incurred between two locations, with three colors indicating different modes of transportation.

The primary objective of this model is to identify the short- est and the second shortest time-consuming paths from node A to node G using two distinct algorithms: the enhanced Yen's algorithm and the standard version. A comparison of the time durations used by the two algorithms is essential. Moreover, the costs associated with these paths will be evaluated to determine if they align with real-world scenarios, specifically assessing whether the second shortest time-consuming path requires less cost than the shortest path.

B. Experimental Details

To facilitate manipulation and align more closely with practical usage scenarios, the A star algorithm employs the 2-dimensional Euclidean distance as its metric distance. The formula for this distance is given by:

$$\rho = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}, |X| = \sqrt{x_1^2 + y_1^2}. \quad (3)$$

Where ρ represents the Euclidean distance between point (x_1,y_1) and point (x_2,y_2), and |X| denotes the Euclidean distance between point (x_1,y_1) and the origin of the coordi- nates. The edge and node data were input into the proposed algorithm, yielding the results presented below.

C. Test Results

Figures 3 and 4 show the detailed image and data results of the time-consuming shortest and second time-consuming shortest paths using the improved algorithm. To clearly illus- trate the differences between these two paths, the shortest path is marked in red and the second shortest path is marked in yellow in Figure 4. The experimental findings indicate that the proposed algorithm provides valuable insights for addressing urban transportation path optimization problems.

As shown in Figure 3, the shortest time-consuming path starts from node A, passes through nodes B and D, and then ends at node G, with a total travel time of 10.5 units and a total cost of 12.5 units. The second shortest time-consuming path also starts from node A, traverses nodes B and D, and then ends at node G, with a total time of 11.5 units and a total cost of 11.5 units. The primary difference between these two paths is the mode of transportation between nodes A and B.

In the second shortest time-consuming path, a bus is chosen as the mode of conveyance. This slight change significantly reduced commuting expenses, with only a minimal increase in travel time. In addition, this result supports the conjecture and conclusions of this study.

In addition, Figure 3 illustrates the trajectories of the short- est path and the second shortest path. The total computational time of the enhanced algorithm was approximately 0.0147 second, whereas that of the unmodified algorithm was ap- proximately 0.0534 second. The experimental results indicate that the enhanced algorithm exhibits superior time complex- ity, particularly when solving problems involving numerous nodes and edges, such as urban traffic path optimization. The proposed algorithm demonstrates high generalizability. Transportation path optimization problems involving numer- ous nodes and edges simply require inputting the connected relationship of each node and the binary array of vehicles represented by each edge into the enhanced algorithm.

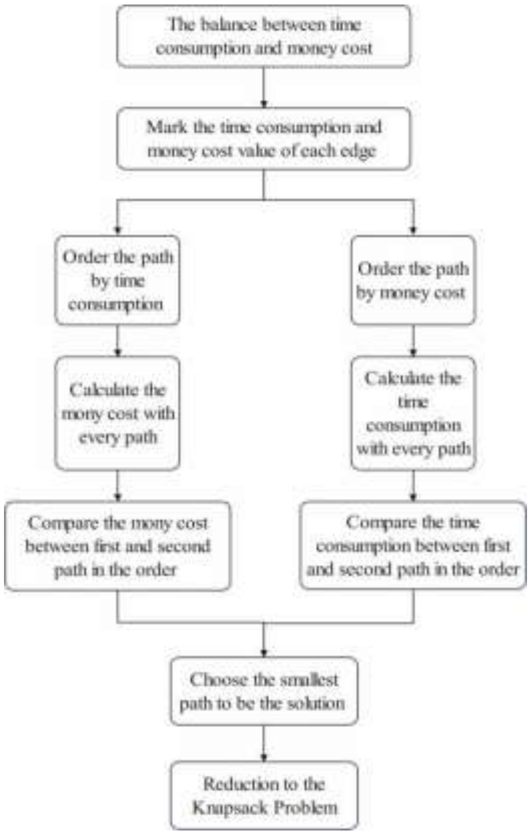

Figure 1. *The whole strategy of the demonstration.*

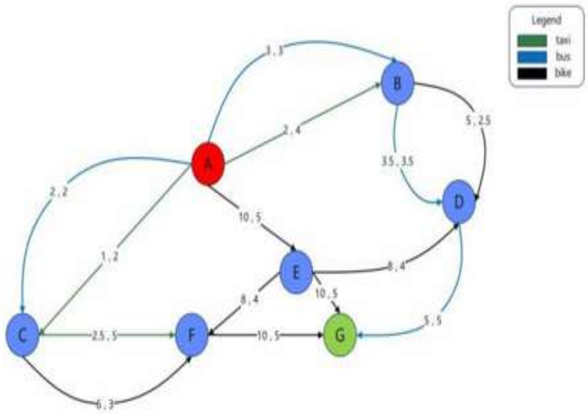

Figure 2. *Graph of the main road in Lanzhou.*

```
['A->taxi->B', 'B->bus->D', 'D->bus->G']
time-consuming:  10.5
cost:  12.5
the second shortest path:
['A->bus->B', 'B->bus->D', 'D->bus->G']
time-consuming:  11.5
cost:  11.5
running time of improved algorithm: 0.0147 sec
running time of Yen's algorithm: 0.0534 sec
```

Figure 3*. Comparison of two paths.*

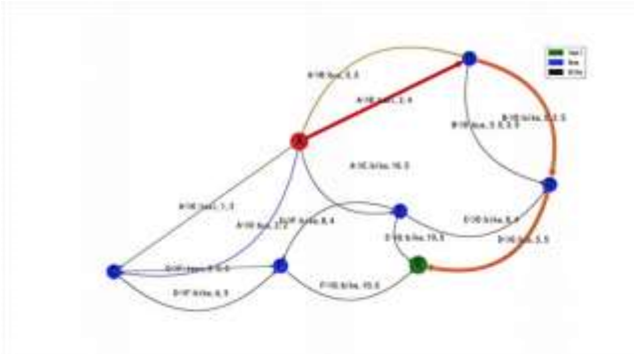

Figure 4.The result graph.

4. Conclusions

This study introduces an optimized version of Yen's algo- rithm for solving KSP using the A star algorithm. By providing pseudo-code and using it as illustrative examples, the im- proved algorithm exhibits significant computational efficiency, particularly for urban traffic path optimization involving nu- merous edges. After applying the improved Yen's algorithm to solve the KSP, the hypotheses regarding the shortest and second shortest time-consuming paths were confirmed to be accurate. Notably, the second shortest time-consuming path offers cost savings while ensuring a reasonable travel time. The inherent flexibility of the algorithm also allows its application to various path optimization problems, extending beyond its
initial application to urban road paths. However, the ruduction tells the problem proposed is NP-complete. To address this limitation, future work will explore a new computational problem of urban networks path optimization and try to fix it with acceptable time complexity.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1] Y. Ji and N. Geroliminis, "On the spatial partitioning of urban transportation networks," *Transportation Research Part B: Methodological*, vol. 46, no. 10, pp. 1639–1656, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261512001099 I

[2] A. Sadeghi-Niaraki, M. Varshosaz, K. Kim, and J. J.Jung, "Real world representation of a road network for route planning in gis," *Expert Systems with Applications*, vol. 38, no. 10, pp. 11 999–12 008, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417410014867 I

[3] H. Jin, X. Zhu, and C. lin Zhao, "Computation offloading optimization based on probabilistic sfc for mobile online gaming in heterogeneous network," *IEEE Access*, vol. 7, pp. 52 168–52 180, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:139298407 I

[4] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092188901300167X I

[5] Y. Zhang, D. wei Gong, and J. hua Zhang, "Robot path planning in uncertain environment using multi-objective particle swarm optimization," *Neurocomputing*, vol. 103, pp. 172–185, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231212007722 I

[6] X. Wang, X. Zhou, Z. Xia, and X. Gu, "A survey of welding robot intelligent path optimization," *Journal of Manufacturing Processes*, vol. 63, pp. 14–23, 2021, trends in Intelligentizing Robotic Welding Processes. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1526612520303078 I

[7] H. Bai, R. Cheng, D. Yazdani, K. C. Tan, and Y. Jin, "Evolutionary large-scale dynamic optimization using bilevel variable grouping," *IEEE Transactions on Cyber- netics*, vol. 53, no. 11, pp. 6937–6950, 2023. I

[8] P. Zhao and J. Han, "On graph query optimization in large networks," *Proc. VLDB Endow.*, vol. 3, no. 1–2, p. 340–351, Sep. 2010. [Online]. Available: https://doi.org/10.14778/1920841.1920887 I

[9] A. Nagurney, "Congested urban transportation networks and emission paradoxes," *Transportation Research Part D: Transport and Environment*, vol. 5, no. 2, pp. 145–151, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920999000310 I

[10] C. E. Mandl, "Evaluation and optimization of urban public transportation networks," *European Journal of Operational Research*, vol. 5, no. 6, pp. 396–404,1980. [Online]. Availablehttps://www.sciencedirect.com/science/article/pii/0377221780901265 I

[11] L. Ferrari, M. Berlingerio, F. Calabrese, and J. Reades, "Improving the accessibility of urban transportation networks for people with disabilities," *Transportation Research Part C-emerging Technologies*, vol. 45, pp. 27– 40, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:54938827 I

[12] Y. zhou Chen, S. fei Shen, T. Chen, and R. Yang, "Path optimization study for vehicles evacuation based on dijkstra algorithm," *Procedia Engineering*, vol. 71, pp. 159–165, 2014, 2013 International Conference on Performance-based Fireand Fire Protection Engineering, Wuhan (ICPFFPE 2013). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S187770581400441X I

[13] L. R. Ford and D. R. Fulkerson, "A simple algorithm for finding maximal network flows and an application to the hitchcock problem," *Canadian Journal of Mathematics*, vol. 9, p. 210–218, 1957. I

[14] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968. I, III-A, 1, 2

[15] W. Hongbo, "Analysis of urban traffic vehicle routing based on dijkstra algorithm optimization," *Journal of Beijing Jiaotong University*, vol. 43, no. 4, 2019. [Online]. Available: https://jdxb.bjtu.edu.cn/EN/10.11860/j.issn.1673-0291.20180109 I

[16] M. Z. Serdar, M. Koc¸, and S. G. Al-Ghamdi, "Urban transportation networks resilience: Indicators, disturbances, and assessment methods," *Sustainable Cities and Society*, vol. 76, p. 103452, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210670721007253 I

[17] L. Dongxiang, "A survey of intelligent transportation path planning algorithms," *Electronic Science and Tech- nology*, vol. 35, no. 07, pp. 22–26, 2022. I

[18] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968. II

[19] W. Hoffman and R. Pavley, "A method for the solution of the nth best path problem," *J. ACM*, vol. 6, no. 4, p. 506–514, Oct. 1959. [Online]. Available: https://doi.org/10.1145/320998.321004 II

[20] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, pp. 712–716, 1971. [Online]. Available: https://api.semanticscholar.org/CorpusID:121444315 II, III-A

[21] H. Aljazzar and S. Leue, "K*: A heuristic search algorithm for finding the k shortest paths," *Artificial Intelligence*, vol. 175, no. 18, pp. 2129–2154, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370211000865 II

[22] H. Jiuyuan and M. Yuyu. [On- line]. Available: http://www.ncdc.ac.cn/portal/metadata/f5cc0627-50d2-4232-bfe2-b87d55086e54 V-A

[23] T. Guoan, "Digital elevation model of china (1km)," 5 2019. [Online]. Available: https://dx.doi.org/ V-A