

PAPER • OPEN ACCESS

Application of Dijkstra algorithm in finding the shortest path

To cite this article: Baoyi He 2022 *J. Phys.: Conf. Ser.* **2181** 012005

View the [article online](#) for updates and enhancements.

You may also like

- [Comparative Analysis between Dijkstra and Bellman-Ford Algorithms in Shortest Path Optimization](#)

Samah W.G. AbuSalim, Rosziati Ibrahim, Mohd Zainuri Saringat et al.

- [Optimization of ISP Service Maintenance Router Using Dijkstra and Flyod-Warshall Algorithm](#)

Wiwien Hadikurniawati, Edy Winarno, Aldy Hernawan et al.


- [Dijkstra and Bidirectional Dijkstra on Determining Evacuation Routes](#)

I G S Rahayuda and N P L Santiari

A promotional banner for 'Free the Science Week 2023' with a dark blue background and a futuristic circular interface. A hand is shown interacting with the interface, which features a central padlock icon. The text 'Free the Science Week 2023' is in large, light blue font, followed by 'April 2-9' in a smaller white font. Below this, the text 'Accelerating discovery through open access!' is displayed, with 'open access!' in a bold, light blue font. At the bottom left is the ECS logo and the website 'www.ecsdl.org'. At the bottom right is a blue button with the text 'Discover more!' in white.

Free the Science Week 2023 April 2-9

Accelerating discovery through
open access!

 www.ecsdl.org [Discover more!](#)

Application of Dijkstra algorithm in finding the shortest path

Baoyi He¹

ZJUI, ZheJiang University, Haining, ZheJiang, 314400, China

Email: baoyi.19@intl.zju.edu.cn

Abstract: As the development of Automotive, people are doing research on self-driving and innovation of public transportation. The coming of 5G will made these come true soon in the future. And this paper is about an algorithm useful on transportation. Using Dijkstra algorithm to match two people who have the similar source and destination, then find the shortest path for them. For each person, the source and destination are two regions but not two points. This paper includes some methods to implement Dijkstra algorithm in detail and two examples to display the results. The algorithm is written in C language.

1. Introduction

Dijkstra^[1] is an algorithm through which in a graph we can get all shortest paths from one point to all other points. This algorithm was pointed out at the year 1959, and it is useful to solve problems today. For example, the function of finding the shortest path in self-driving cars and finding a partner from a region around a person and which has the similar destination with the person. Both are shown in the following example, which is the core of this paper. Dijkstra algorithm is faster than other algorithms for it can calculate the shortest length to every point, and the example below can display the practicability of Dijkstra algorithm.

2. Find partner and the shortest path

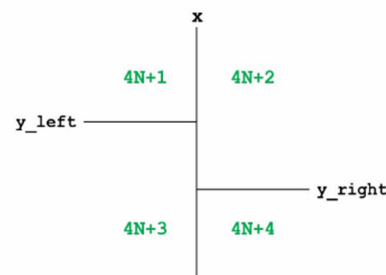
2.1 introduction to this example

This example can match two people who have the similar source and destination, then find the shortest path for them. For each person, the source and destination are two regions but not two points.^[2] The important structures included are pyramid tree and heap.

2.2 Find nodes

Firstly, in find nodes function, find all vertex nodes that in range and write them in a vertex set. For pyramid tree node, if it is a leaf node, check if it in range; if it is not a leaf node, recurse. According to professor Steven S. Lumetta, I use a structure called pyramid tree. For all nodes in a graph, we use a pyramid tree to mark. All real nodes in the graph are leaf nodes of pyramid tree, and internal nodes in the pyramid tree divide space into (up to) four quadrants (one node may have fewer children). For example, if one examines an internal node at array index N , array index $4N+1$ is a subtree in which all nodes have x values no greater than the x value of the internal node at array index N and y values no greater than the y value of the internal node at array index N .^[3] The structure of the pyramid tree is shown as follow. The figure right is from professor Lumetta.





Pyramid tree node structure for node N. Leaf nodes (with no children: $4N+1 \geq \#$ of nodes) represent graph vertices as (x, y_{left}) . Nodes with children subdivide space into up to four parts. Note that children in any quadrant can be located on the lines (equality is allowed in both directions).

Figure1. pyramid tree generated by Lumetta

2.3 Dijkstra algorithm and Heap

In Dijkstra algorithm, from the initial point, mark all next points we can get directly from initial point, and put all these points in a heap with their distances. Find the point with smallest distance, delete it from the heap, do what do on the initial point before, then update the shortest distance of some vertexes^[4]. Repeat this many times until the heap is empty. I use a heap to pop the node with the smallest distance to the root of the top(H(0)), and delete it. Then after once recurrence, pop again. The new nodes added in the heap are as the leaf nodes. What's more, because the source in this example is a region including many vertex but not a vertex, for all vertexes in source, set their distance zero, so we can see them equally.

2.4 show it in graph

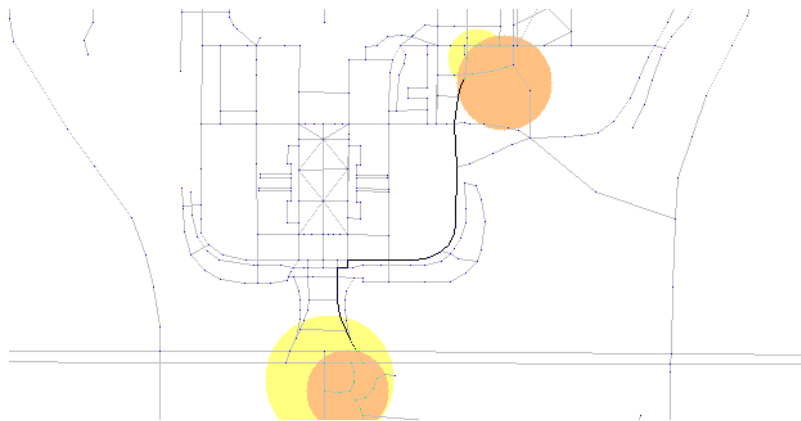


Figure2 example using ZJUI map data

The two orange ranges are the source and destination of the person 1. The two yellow ranges are the source and destination of the person 2. They are matched because there are superpositions of both sources and destination. And the shortest path is shown in the graph. To finish this function, we set a array to store people. When a new person want to match someone, check if he or she can match someone already in the array, if not, put the person into the array to wait. Two matched people can go their shortest path together as shown above. The input data are taken from OpenStreetMap data for the ZJUI campus area.

2.5 Improve performance

The algorithm above still need to improve to cope with larger figure. There are four points I think can

be researched in the future. Firstly, make use of dynamic allocation in several ways^[5], the structures in this examples including arrays and heaps can be dynamic allocate, so their sizes are related to the data in them, which can save some spaces. At the same time, dynamic allocation is also time consuming, so a practicable method should be used.

Secondly, the Dijkstra algorithm can be improved by using one-dimensional array.^[6] And Ping Li, the author, compared the improved Dijkstra algorithm with Prim algorithm^[7] and Huffman coding algorithm, and stated that it does save storage space and improves the running rate.

Thirdly, at the find partner part, it can be improved using a special number. This number can be 32 bits, 64 bits, 128 bits or so on, which means the size of this number is related to the number of vertexes in the graph. This number divides the graph in several regions as the number of bits. If this region have vertex in the source, it is represent as 1, otherwise 0. So the situation of one person's source can be represented as a number, AND two people's source numbers, if it is 0, means they are not match. The same as the destination region. This filter partner candidates quickly, which improve the algorithm largely.

Finally, there are many forms of heaps, so the heap used can also be optimized.^[8] However, this need more research in the future, I still can't implement is in code.

Improve this example with the first, the second and the third improvement, I find that it can cope with more complex graph and faster. Following is also data taken from OpenStreetMap data for the ZJUI campus area but with more vertexes. This is generated using the first, the second and the third improvement.

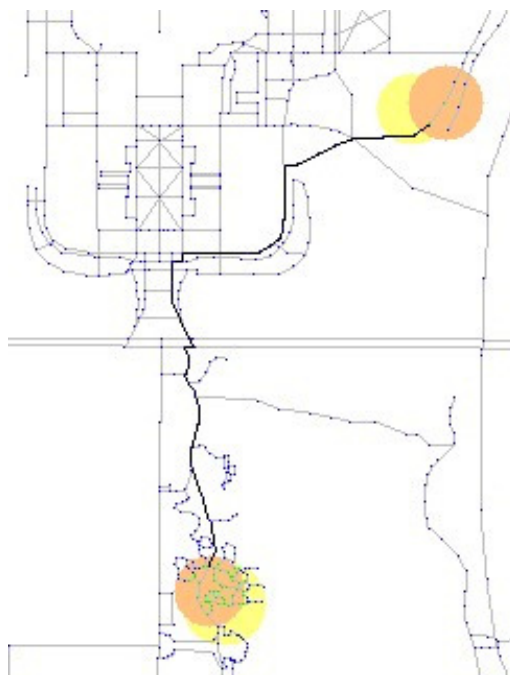


Figure3: More complex graph with improvment

3. Conclusion

This application of Dijkstra algorithm may be used to new form of transportation, such as the combination of self-driving and public transit. This paper illustrates the Dijkstra example about the structure pyramid tree and heap, and then show the results in graph. Then point out four points that have been improved or will be improved in the future. This algorithm is being improved and will be able to cope with more complex graph.^[9] In conclusion, this paper can be a base on transportation innovation.^[10]

Reference

- [1] Xiaoling Liu, Hui Li, Zhiguo Guo. Evaluation and implementation of dynamic production capacity in the workshop based on Dijkstra algorithm [J]. Microcomputer information, 2006(12):96-98 .
- [2] Xiaolong Yan, Dongcai Qu. A track optimization method based on genetic simulation annealing algorithm [J]. Sichuan Journal of Military Engineering, 2013, 34(12):66-70
- [3] Steven S. Lumetta, 2017. Pyramid Tree.
<https://baike.baidu.com/item/pyramid%20tree/23164617?fr=aladdin>.
- [4] HUANG Y. Research on the improvement of Dijkstra algorithm in the shortest path calculation[C]//Proceedings of the 2017 4th International Conference on Machinery, Materials and Computer, Xi'an, China, Atlantis Press, 2017.
- [5] TAMATJITA E N, MAHASTAMA A W. Shortest path with dynamic weight implementation using Dijkstra's algorithm[J]. ComTech Computer Mathematics and Engineering Applications, 2016, 7(3): 161-171.
- [6] Ping Li. Improvement and implementation of the Dijkstra algorithm [J]. Information construction, 2016(02):345.
- [7] Zhiming Yang. Prim algorithm and Dijkstra algorithm's comparison and analysis[J]. Baoshan Teacher's Journal, 2009, 28(05):73-75.
- [8] Zhilin Wang, Xinhui Qiao, Xu Ma, Yan Yan. A Dijkstra shortest path optimization method based on binary heaps [J]. Journal of Engineering Mathematics, 2021, 38(05):709-720.
- [9] Hua Wang. The improvement of The Dijkstra algorithm using combination technology is discussed[J]. Mapping science, 2014, 39(02):52-54. DOI:10.16251/j.cnki.1009-2307.2014.02.009.
- [10] Bo Zhang, Guozhong Cheng. The design and implementation of the DeJesustra algorithm in the traffic consulting platform [J]. Information and Computers (Theoretical Edition), 2016(10):76-77.