Shortest Path Discovery in Consideration of Obstacle in Mobile Social Network Environments

Dawei Sun[1(✉)], Wentian Qu[1], Shang Gao[2], and Li Liu[3]

[1] School of Information Engineering, China University of Geosciences, Beijing 100083, People's Republic of China
sundaweicn@cugb.edu.cn, qunty496047796@163.com

[2] School of Information Technology, Deakin University, Geelong, VIC 3216, Australia
shang.gao@deakin.edu.au

[3] School of Automation and Engineering, University of Science and Technology Beijing, Beijing 100083, People's Republic of China
liuli@ustb.edu.cn

Abstract. The issue of shortest path discovery in consideration of obstacle is one of the problems for location-based services in mobile social network environments. Currently, most research focuses on quickly discovering the shortest path in obstacle free area with reasonable latency, while the obstacle issue, especially the obstacles that enter temporarily is not fully considered. This creates the need for investigation on shortest path discovery at the same time avoiding detected obstacles. In this paper, a shortest path discovery approach is proposed. The following contributions are made: (1) Modeling the shortest path discovery problem in consideration of obstacle. (2) Discovering the shortest path using an improved A-star algorithm with reasonable latency. (3) Evaluating the accuracy rate of shortest path discovery with acceptable latency for a location-based service in a mobile social network. Experimental results conclusively demonstrate the efficiency and effectiveness of the proposed approach.

Keywords: Mobile social network · Location-based service
Shortest path discovery · A-star algorithm

1 Introduction

1.1 Background and Motivation

In the era of big data, mobile social network, as one of the major application scenes, is playing a more and more important role in our daily lives. It is advised that nearly all services in a mobile social network are location-based. If a mobile social network scene can be abstracted as a large-scale graph, composed of vertex set and edge set, location-based services can be achieved by manipulating the large-scale graph. For example, the shortest path discovery is alway one of the most important search queries for large-scale graph, and has received intense attention owing to its broad applications, such as

I. Romdhani et al. (Eds.): CollaborateCom 2017, LNICST 252, pp. 637–646, 2018.
https://doi.org/10.1007/978-3-030-00916-8_58

path planning in mobile social network, vehicle routing in traffic network, and package routing in computing network [1, 2].

It is common that obstacles exist between locations in location-based in application scenarios. For example, part of an optimal path is under construction or there are road work on the path, so this "shortest path" is no longer an ideal shortest path. The shortest path only considering physical distance is not alway work well, and needs to take into account the actual obstacles, especially these temporary ones. The issue of shortest path discovery in consideration of obstacle is realistic and one to-be-resolved problem for location-based service. Currently, most research only considers quickly discovering the shortest path with reasonable latency, while the obstacle is not fully considered [3, 4]. It is necessary to consider the avoidance of existing and temporary obstacles while discovering the shortest path.

1.2 Contributions

Motivated by the above discussion, and our previous work [5], we propose the shortest path discovery approach in consideration of obstacle (ASPO) for location-based services in a mobile social network environment. In this paper, we move one step further to a real world application scenario in a large-scale graph based experimental environment. Three aspects of ASPO are covered and summarized as follows: (1) Modeling the shortest path discovery problem in consideration of obstacle. (2) Discovering the shortest path using an improved A-star algorithm with a reasonable latency. (3) Evaluating accuracy rate of the shortest path discovery approach with acceptable latency. Experimental results conclusively demonstrate the efficiency and effectiveness of the proposed approach.

1.3 Paper Organization

The remainder of this paper is organized as follows. In Sect. 2, the modeling of the shortest path discovery problem in consideration of obstacle is presented. Section 3 focuses on the detailed discussion of the ASPO. Section 4 addresses the experimental environment, parameter setup and performance evaluation of the ASPO. Conclusions are given in Sect. 5.

2 Problem Statement

This section focuses on modeling the shortest path discovery problem in consideration of obstacle in a mobile social network environment [6, 7].

The mobile social network environment can be described as a grid, and the location-based service can be represented as a connected graph G, as shown in Fig. 1. The graph G is composed of a vertex set and a directed edge set, denoted as $G = (V, E)$, where $V = \{v_1, v_2, \cdots, v_n\}$ is a finite set that contains n vertices. Each vertex represents a location of service or user. $E = \{e_{1,2}, e_{1,3}, \cdots, e_{n-i,n}\} \subset V \times V$ is a

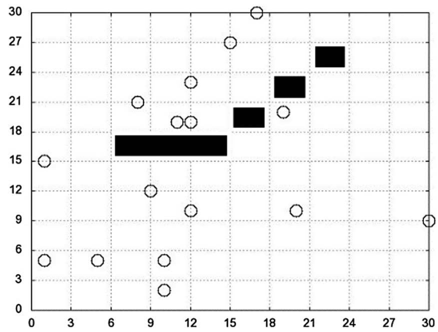

Fig. 1. Location-based services and obstacles in a grid (points of hollow circle are the location of service/user, and points of black squares are the location of existing or temporary obstacles)

finite set of directed edges, which represents reachable paths between two related vertices. If $\exists e_{i,j} \in E$, then $v_i, v_j \in V$, $v_i \neq v_j$, the weight $w_{i,j}$ associated with an edge $e_{i,j}$ represents the distance from v_i to v_j. If an obstacle is on $e_{i,j}$, then $e_{i,j}$ is a reachable path from v_i and v_j, and $e_{i,j}$ should be discovery immediately, and removed or avoided from the directed edge set E.

Let v_s and v_d be a source and destination vertex, respectively, and $P_{s,d} = \left\{p^1_{s,d}, p^2_{s,d}, \cdots, p^m_{s,d}\right\}$ be a set of available path set between v_s and v_d in the grid. Length $L^i_{s,d}$ of the ith path $p^i_{s,d}$ can be calculated by (1).

$$L^i_{s,d} = \sum_{e_k \in E^i(v_s, v_d)} l_{e_k}, \tag{1}$$

where $E^i(v_s, v_d)$ is the edge set of the ith path $p^i_{s,d}$, and l_{e_k} is the length of edge e_k.

The shortest path length between v_i and v_j in consideration of obstacle is defined as the average number of steps along the shortest paths for all available path set P between v_i and v_j. Updating the obstacle information in grid, if an obstacle is on a path between v_i and v_j, the path is removed from available path set.

The shortest path discovery problem [8, 9] is to discover the shortest path length $SL_{s,d}$ connecting two specified vertices v_s and v_d with reasonable latency, can be described as (2).

$$SL_{s,d} = \min\left(P_{s,d}\right), s.t. L^i_{s,d} < rs - lat. \tag{2}$$

where *rs-lat* is a pre-set acceptable latency threshold for one specific location-based service.

If the grid has millions of vertices or edges, millions of available paths can be selected, resulting in unbearable latency.

3 ASPO Overview

To provide a bird-eye view of the ASPO, this section focuses on the detailed discussion of ASPO, including the improved A-Star model, and A-Star algorithm [10, 11].

3.1 Improved A-Star Model

A-Star is considered as one of the widely used graphic algorithms for its simplicity and performance, as it combines the advantages of both breadth-first graphic search and depth-first graphic search mechanism [12, 13]. In this paper, we modify the algorithm and make it work better for the shortest path discovery while avoiding obstacle.

Evaluation function $f(n)$, described as (3), is the sum of function $g(n)$ and $h(n)$, used to guide the visiting order of vertices in the search space.

$$f(n) = g(n) + h(n), \tag{3}$$

where $g(n)$ can be calculated by Dijkstra's algorithm [14], described as (4), is the distance of the optimal path from the source vertex v_s to the current vertex v_n, and $h(n)$, calculated by (5), is the heuristic estimated distance of the optimal path from the current vertex v_n to the destination vertex v_d.

$$g(n) = g(n') + 1, \tag{4}$$

where $v_{n'}$ is direct predecessor vertex on the path from the source vertex v_s to the current vertex v_n.

$$h(n) = d(v_c, v_d, \alpha, \beta) + d'(V_c, v_d, \alpha, \beta), \tag{5}$$

where $d(v_c, v_d, \alpha, \beta)$ is a Manhattan distance [15] with adjustment parameter α and β. It can be calculated by (6). V_c is the set vertices of the vertices on the path from v_s to v_c, and $d'(V_c, v_d, \alpha, \beta)$ can be calculated by (7)

$$d(v_s, v_c, \alpha, \beta) = \alpha \cdot |x_{v_s} - x_{v_c}| + \beta \cdot |y_{v_s} - y_{v_c}|, \tag{6}$$

where $\alpha \in [0, 1]$, $\beta \in [0, 1]$, and $\alpha + \beta = 1$.

$$d'(V_c, v_d, \alpha, \beta) = \frac{1}{m} \sum_{i=1}^{m} d\left(v_{c(i)}, v_d, \alpha, \beta\right) \tag{7}$$

where $v_{c(i)} \in V_c$, and m is the number of vertices on the path from v_s to v_c.

3.2 Improved A-Star Algorithm

The improved A-Star model can be further described by an improve A-Star algorithm. The improved A-Star algorithm maintains two tables [16], an OPEN table and a CLOSE table. The OPEN table is used to keep the vertices to be searched with least evaluation function $f(n)$ to select next optimal vertex based on priority. The CLOSE table is used to keep the vertices that have already been searched and examined [17, 18].

The improved A-Star algorithm is described in Algorithm 1.

Algorithm 1: improved A-Star algorithm.

Input: location-based service in a grid environment, the source vertex v_s, and the destination vertex v_d of the service.

Output: the shortest path in consideration of obstacle from source vertex v_s to destination vertex v_d in the grid.

Initialize OPEN table, CLOSE table.
Add source vertex v_s to OPEN table. Set CLOSE table is NULL.
while OPEN table is not NULL **then**
  Delete the highest priority vertex v_h from OPEN table
  Add the vertex v_h to CLOSE table.
  if vertex v_h is the destination vertex v_d **then**
    break.
  end if
  if vertex v_h can be spreading **then**
    for each spreader vertex v_i of vertex v_h **do**
      Check whether vertex v_i is in OPEN table or CLOSE table.
      if vertex v_i is not in the OPEN table and CLOSE table and not a obstacle **then**
        Add the vertex v_i to the OPEN table.
        Calculate priority by evaluation function $f(n)$.
      else
        if vertex v_i is the on the path from source vertex v_s to vertex v_h **then**
          Delete vertex v_i.
        else
          Spread vertex v_i and calculate priority by evaluation function $f(n)$.
        end if
      end if
    end for
    Sort all vertices in OPEN table by priority in ascending order.
  end if
end while
return the shortest path in consideration of obstacle from source vertex v_s to destination vertex v_d in the grid.

The inputs of the improved A-Star algorithm are location-based service in a grid environment, the source vertex v_s, and the destination vertex v_d of the service. The output is the shortest path in consideration of obstacle from source vertex v_s to destination vertex v_d in the grid. Step 3 to step 25 are to discover the shortest path from source vertex v_s to destination vertex v_d.

4 Experiments and Performance Evaluation

Experiments are conducted to evaluate the performance of the proposed ASPO. Experimental environment and parameter settings are firstly discussed in this section, followed by performance evaluation results.

4.1 Environment and Parameter Setup

In the experimental environment, one machine is employed. The machine runs Windows 10 with dual 4-core, Intel Core i5-4200, 2.5 GHz, 8 GB Memory, and a 100 Mbps network interface card. The mobile social network environment is simulated as a 1000 × 1000 grid environment.

4.2 Performance Evaluation

The experiment evaluates the impact of search time and search depth.

(1) Search time

The search time is the time to discover the shortest path in consideration of obstacle in a grid environment.

Relationship of search depth and search time with different distances between source vertex v_s and obstacle is shown in Fig. 2. The distance between destination vertex v_d and obstacle is 300 units, and the length of obstacle is 1600 units. When varying the distance between source vertex v_s and obstacle from 50 to 300 units, the corresponding average search time is 1.1 s, 1.2 s, and 1.3 s, respectively. The shortest path is discovered within 5 depths.

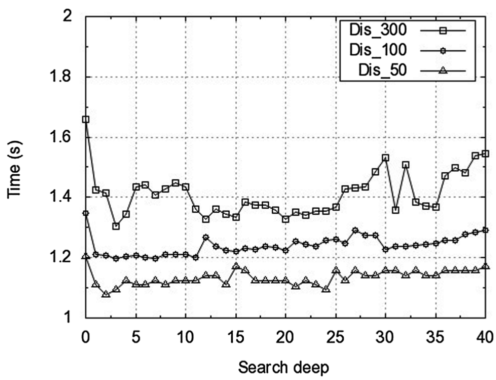

Fig. 2. Relationship of search depth and search time with different distances between source vertex v_s and obstacle

Relationship of search depth and search time with different distances between destination vertex v_d and obstacle is shown in Fig. 3. The distance between source vertex v_s and obstacle is 300 units, and the length of obstacle is 1600 units. When varying the distance between destination vertex v_d and obstacle from 50 to 300 units, the corresponding average search time is 2.3 s, 2.2 s, and 1.4 s, respectively. The shortest path is discovered within 5 depths.

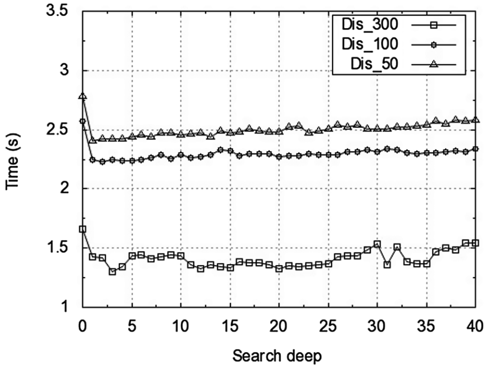

Fig. 3. Relationship of search depth and search time with different distances between destination vertex v_d and obstacle

Relationship of search depth and search time with different length of the obstacle is shown in Fig. 4. The distance between source vertex v_s and obstacle is 300 units, and the distance between the destination vertex v_d and obstacle is also 300 units. When varying the length of the obstacle from 1500 to 1800 units, the corresponding average search time is 1.8 s, 1.4 s, 3.4 s, and 4.3 s, respectively. The shortest path is discovered within 5 depths.

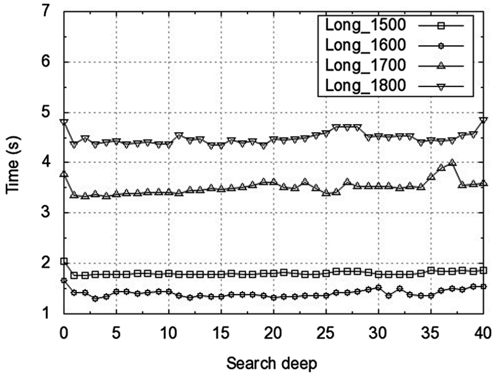

Fig. 4. Relationship of search depth and search time with different long of the obstacle

(2) Search depth

The search depth is the level of search depth to discover the shortest path in consideration of obstacle in a grid environment.

Relationship of search depth and search time with different distances between source vertex v_s and obstacle is shown in Fig. 5. The distance between destination vertex v_d and obstacle is 300 units, and the length of obstacle is 1600 units. When varying the distance between source vertex v_s and obstacle from 50 to 300 units, the corresponding average search steps are 55000 steps, 57000 steps, and 59000 steps, respectively. The shortest path is discovered within 3 depths.

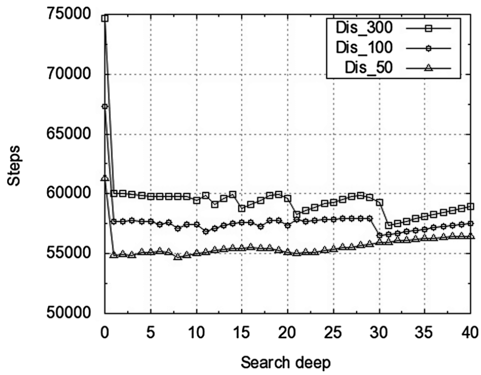

Fig. 5. Relationship of search depth and search steps with different distances between source vertex v_s and obstacle

Relationship of search depth and search time with different distances between destination vertex v_d and obstacle is shown in Fig. 6. The distance between source vertex v_s and obstacle is 300 units, and the length of obstacle is 1600 units. When varying the distance between destination vertex v_d and obstacle from 50 to 300 units,

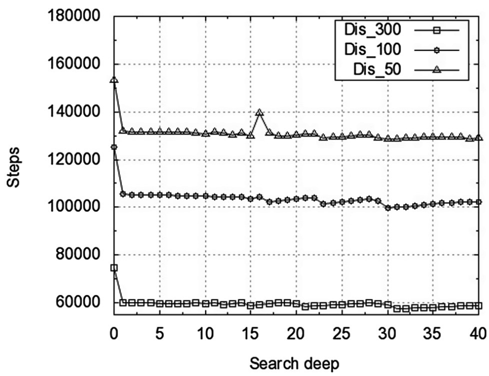

Fig. 6. Relationship of search depth and search steps with different distances between destination vertex v_d and obstacle

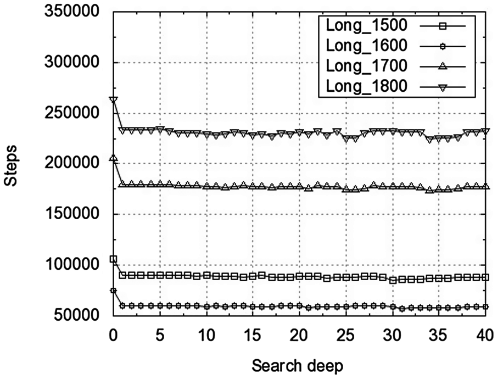

Fig. 7. Relationship of search depth and search steps with different length of the obstacle

the corresponding average search steps are 130000 steps, 110000 steps, and 59000 steps, respectively. The shortest path is discovered within 2 depths.

Relationship of search depth and search time with different lengths of the obstacle is shown in Fig. 7. The distance between source vertex v_s and obstacle is 300 units, the destination vertex v_d and obstacle is also 300 units. When varying the length of the obstacle from 1500 to 1800 units, the corresponding average search steps are 89000 steps, 59000 steps, 170000 steps, and 220000 steps, respectively. The shortest path is discovered within 2 deeps.

5 Conclusions

In this paper, the shortest path discovery approach ASPO in consideration of obstacle for location-based services in a mobile social network environment is proposed. Our work and contributions are summarized as follows: (1) Modeling the shortest path discovery problem in consideration of obstacle. (2) Discovering the shortest path in a location-based mobile social network via an improved A-star algorithm with reasonable latency. (3) Evaluating accuracy rate of shortest path discovery with acceptable latency for a location-based service.

Acknowledgment. This work is supported by the National Natural Science Foundation of China under Grant No. 61602428 and 61370132; the Fundamental Research Funds for the Central Universities under Grant No. 2652015338.

References

1. Hong, J., Park, K., Han, Y., Rasel, M.K., Vonvou, D., Lee, Y.K.: Disk-basedshortest path discovery using distance index over large dynamic graphs. Inf. Sci. **382–383**, 201–215 (2017)
2. He, K., Xu, Z.Z., Wang, P., Deng, L.B., Tu, L.: Congestion avoidance routing based on large-scale social signals. IEEE Trans. Intell. Transp. Syst. **17**(9), 2613–2626 (2016)
3. Lai, C.N.: Constructing all shortest node-disjoint paths in torus networks. J. Parallel Distrib. Comput. **75**, 123–132 (2015)
4. Wang, X., Li, J., Li, X., Wang, H.: Applying the locality principle to improve the shortest path algorithm. Clust. Comput. **20**(1), 301–309 (2017)
5. Qu, W.T., Sun, D.W.: A fast search strategy to optimize path finding in big data graph computing environments. Int. J. Wirel. Mobile Comput. **13**(2), 139–143 (2017)
6. Mao, G., Zhang, N.: Fast approximation of average shortest path length of directed BA networks. Phys. A Stat. Mech. Appl. **466**, 243–248 (2017)
7. Huang, W., Wang, J.: The shortest path problem on a time-dependent network with mixed uncertainty of randomness and fuzziness. IEEE Trans. Intell. Transp. Syst. **17**(11), 3194–3204 (2016)
8. Feng, G., Korkmaz, T.: Finding multi-constrained multiple shortest paths. IEEE Trans. Comput. **64**(9), 2559–2572 (2015)
9. Sang, Y., Lv, J., Qu, H., Yi, Z.: Shortest path computation using pulse-coupled neural networks with restricted autowave. Knowl. Based Syst. **114**, 1–11 (2016)
10. AlShawi, I.S., Yan, L., Luo, W., Pan, W., Luo, B.: Lifetime enhancement in wireless sensor networks using fuzzy approach and A-star algorithm. IEEE Sens. J. **12**(10), 3010–3018 (2012)
11. Wei, Q., Liang, X., Fang, J.: A new star identification algorithm based on improved hausdorff distance for star sensors. IEEE Trans. Aerosp. Electron. Syst. **49**(3), 2101–2109 (2013)
12. Li, B., Sun, Q., Zhang, T.: A star pattern recognition algorithm for the double-FOV star sensor. IEEE Aerosp. Electron. Syst. Mag. **30**(8), 24–31 (2015)
13. Huang, J., Sun, L., Du, F., Wan, H., Zhao, X.: Genetic adaptive A-star approach for ttrain trip profile optimization problems. In: Proceedings of the 2014 IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems, CIVTS 2014, pp. 1–6. IEEE Press (2014)
14. Zhang, X.G., Chan, F.T.S., Yang, H., Deng, Y.: An adaptive amoeba algorithm for shortest path tree computation in dynamic graphs. Inf. Sci. **405**, 123–140 (2017)
15. Chiu, W.Y., Yen, G.G., Juan, T.K.: Minimum manhattan distance approach to multiple criteria decision making in multiobjective optimization problems. IEEE Trans. Evolut. Comput. **20**(6), 972–985 (2016)
16. Ghaffari, A.: An energy efficient routing protocol for wireless sensor networks using A-star algorithm. J. Appl. Res. Technol. **12**(4), 815–822 (2014)
17. Cui, X., Shi, H.: A*-based pathfinding in modern computer games. Int. J. Comput. Sci. Netw. Secur. **11**(1), 125–130 (2011)
18. Ducho, F., et al.: Path planning with modified a Star algorithm for a mobile robot. Procedia Eng. **96**, 59–69 (2014)