

Design and Optimization of Bus Route Query Algorithm

Quanzhou Huang

School of Computer Science, Xi'an Shiyou University, Xi'an, Shaanxi, 710065, China

qzhuang@xsyu.edu.cn

Keywords: Query algorithm, Bus Route, Transfer Algorithm, Hub stations

Abstract. The bus route search algorithm is the key technical problem of bus route query system. This paper analyses the basic needs of passengers and design requirements of the transfer algorithm, and analyses the realization conditions and influence factors for effective running. It introduces the shortest path problem and Dijkstra algorithm, and analyzes the drawbacks of its application in bus route query. The general transfer algorithm is based route set theory, but it can also be improved and optimized. Then the optimal travel route algorithm of based on hub stations is proposed, and the implementation of the transfer algorithm is given. This algorithm is simple and especially useful, use it you can quickly get bus routes with less amount of calculation and comparison.

Introduction

In cities, there are a large number of bus routes and stations. Bus information management system is a public system that people often use, and the route query is one of the key technologies[1].

People have done a lot of research on bus route search algorithms, including a matrix algorithm for the shortest distance, the shortest path algorithm, minimum time, minimum transfer times, costs and other algorithms [2, 3]. These algorithms have their own advantages, but they are too complex to achieve. In this regard, our method is to find a simple, practical and fast algorithm of transfer by building station and route sets.

This paper mainly discusses a transfer algorithm based on set theory, and the design process of the algorithm is given. Then, improved algorithms are presented. The transfer algorithm based on hub stations greatly reduce the amount of calculation, and may be the most practical algorithm. These algorithms are simple and effective, and may improve the speed of the system.

The Basic Needs of Passengers and Design Requirements of Transfer Algorithm

From the user's point of view, transfer algorithm must achieve the following requirements:

- Transfer times is lesser, not more than 2 times.

- Passengers like familiar transfer stations.

- System choose bigger stations to transfer as much as possible.

- Transfer price is relatively cheap.

- Don't be too much trouble.

In view of the user requirements, we proposed the following design principles of transfer algorithms:

- The query algorithm must be fast and reliable.

- Only consider transfer to one or two.

- Give priority to transfer from hub stations.

- No big hub site, it only considers general search strategy.

- A route have more than one stations of transfer, it gives priority to the hub stations.

- It must achieve the shortest distance and less cost.

The Shortcoming of Dijkstra Algorithm in the Bus Route Query

Dijkstra algorithm is the theory basis to solve the shortest path problem. The algorithm is based on network graph model in graph theory. With label method it can calculate shortest path from v_1 to v_n ,

and get the the shortest paths from v_1 to other various nodes. The algorithm searches all the network nodes and do lots of computation, its time cost is $O(n^2)$. When node number N is greater, its computation time is very greater. In the bus route query we need to quickly get the route between two nodes, but Dijkstra algorithm is very slow. For passengers to the destination in the shortest time, the priority is the number of transfer, the second is the length of the route. The shortest path in the general case is not necessarily suitable for bus routes.

The general transfer algorithm based route set theory is very useful, and many developers use the algorithm to design route query program[4], but it can also be improved and optimized.

Also, a matrix algorithm for the shortest distance, the shortest path algorithm, minimum time, minimum transfer times, and other algorithms are traditional query algorithms, and very complex[5,6]. Our goal is to find a simple, practical and fast algorithm of transfer, and meet the needs of passengers.

Transfer Algorithm Based on Hub Stations

Hub Stations. It can be seen in the above discussion, that every time a lot of comparison is involved, which is compute-intensive and inefficient. In the cities there are places of cultural, tourist and commercial hubs, which are the dense district and central areas of bus stations and routes. In the city of Xian, for example, such places as Bell Tower, Big Wild Goose Pagoda are transport hubs. The hub sites can be sorted out, as the main transfer collection stations. Each query is to look for a station from the collection as a transfer station and each calculation will be greatly reduced, thereby enhancing the efficiency of the query algorithm.

Hub Stations Network. In a city, there are many of hub stations in all directions. Hub stations contact each other, and form a transportation network. Each bus route actually passes through some hub stations. Bus hub stations network diagram is shown in Fig. 1.

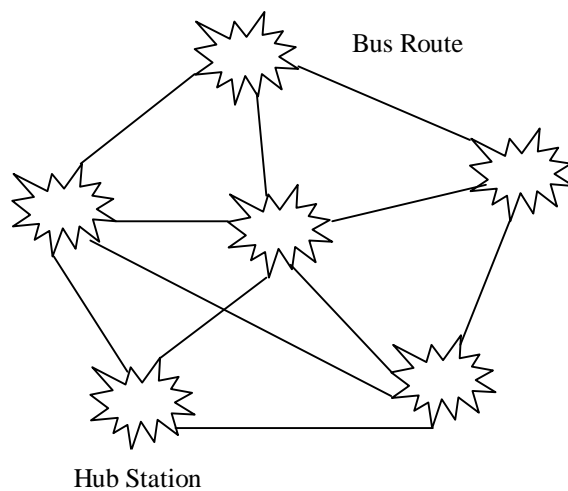


Fig.1 The hub stations network diagram

The lines represent the bus routes in the Figure and pass through many stations. The star mark represent the bus hub stations, each hub station links many bus routes, between Hub stations there are many bus routes. Hub stations are bigger and famous, and passengers are familiar with the hub stations, so use them as a transfer point very well.

All routes are analyzed by calculating the number of bus routes through the station. It selects most frequently several sites as transfer stations, or you can manually specify hub sites as transfer stations. Suppose hub station collection is H set, $H: \{H_1, H_2 \dots H_k\}$, where each H_i is a hub station, and you can also specify hubs in all directions of the city as the transfer station collection.

Design of the Transfer Algorithm Based on Hub Stations

Once Transfer Algorithm. Here only direct and once transfer algorithm is taken into consideration, as described below:

Suppose S_1 is the start station, S_2 is the termination station.

Suppose a collection of routes through the start station is A , $A: \{A_1, A_2 \dots A_m\}$, and each A_i is a route through the start station.

Suppose a collection of routes through the termination station is B , $B: \{B_1, B_2 \dots B_n\}$, and each B_i is a route through the termination station.

Suppose a collection of hub stations is C , $C: \{C_1, C_2 \dots C_p\}$, and each B_i is a hub station from the start station to the termination station.

First it searches for direct routes. In this step, it is only necessary to search all the routes that include the start station and termination station, these routes are direct routes.

If there is no direct route, it searches for a transfer routes. According to two station names, it searches the collection of routes A through start station and collection of routes B through termination station, using each B_i in the collection of hub stations C as a transfer stations, and calculates intersection hub stations between them one by one. Intersection hub stations are transfer stations (there may be multiple or 0). It looks for all routes from start station to termination station with each hub station as a transfer stations. The route query algorithm diagram is shown in Fig.2.

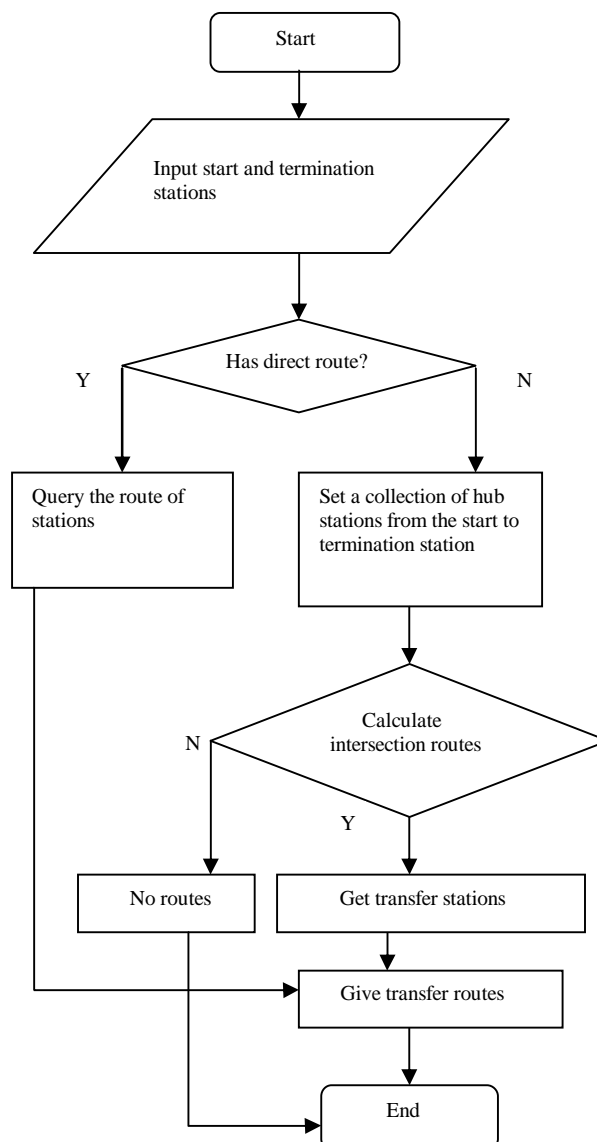


Fig. 2 Route query algorithm diagram

Secondary Transfer Algorithm. Similarly, secondary transfer algorithm may find out two transfer hub stations D_1, D_2 . First, From the start station to the first transfer station D_1 , then from D_1 to D_2 , finally from D_2 to the termination station.

Suppose a collection of hub stations is $C, C: \{C_1, C_2 \dots C_p\}$, and each C_i is a hub station from the start station to the termination station.

First, secondary transfer algorithm in C looks for intersection routes set with the start station, the set as $D, D: \{D_1, D_2 \dots D_s\}$, each D_i is a hub station that has intersection routes with the start station, which are the first transfer stations.

Then, secondary transfer algorithm in C looks for intersection routes set with the termination station, the set as $E, E= \{E_1, E_2 \dots E_t\}$, each E_i is a hub station that has intersection routes with the termination station, and which are the second transfer stations.

Finally, secondary transfer algorithm looks for the routes between D and E . If there is at least a route between station D_i in set D and station E_j in set E , then D_i as the first transfer station, E_j as the second station. The relationship of stations is shown in Fig.3.

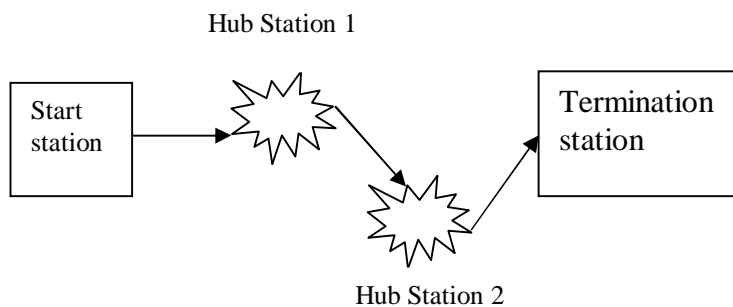


Fig. 3 The relationship of stations diagram

In fact, different routes have different distances. If the distances between the stations are calculated in above algorithms, you will get a shortest distance transfer route [6].

Conclusions

This paper briefly discusses the main problems involved in the bus route query system, and analyses the shortcomings of the commonly used route query algorithms. Then, improved algorithms are presented. The transfer algorithm based on hub stations greatly reduce the amount of calculation, and may be the most practical algorithm. The algorithms is simple and effective, and may improve the speed of the system. If distances between stations are taken into account, the algorithm can be further optimized. Compared with the traditional algorithms, this algorithm is very simple and practical.

References

- [1] L.M. Mu and G.F. Liu: Computer Applications and Software, Vol. 27(2010) No. 8, P.79. (In Chinese)
- [2] Q. Zhou and Y.G. Gou: Hainan University, Vol. 28(2010) No. 2, P.105. (In Chinese)
- [3] H.J. Zhou: Jilin University, Vol. 23(2009) No. 2, P.35. (In Chinese)
- [4] H. Chen and Y.T. Xi: Geomatics & Spatial Information Technology, Vol. 33(2010) No. 3, P48. (In Chinese)
- [5] J. Shen: Automation Instrumentation, Vol. 33(2012) No.1, P12. (In Chinese)
- [6] W.L. Su, S. Ye and Z.Y. Zhang: Computer Knowledge and Technology, Vol. 8(2012) No. 16, P3869. (In Chinese)