Design of a Fast Algorithm for Bus Route Inquiry

Quanzhou Huang

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

qzhuang@xsyu.edu.cn

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

Abstract. 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. 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.

Introduction

The bus management information system is very important and useful, it's an important objective is to provide accurate information services so as to attract more travelers to use public transportation [1]. In cities, there are thousands of bus stations and hundreds of bus routes. Before traveling, when you are unsure of travel routes, if you visit bus route query system, you can quickly get information about the bus routes.

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.

Systems Design

Passengers and managers are users of the system. Station query, route query and transfer query is the main function of passengers. Station information management, route information management and system management is the main function of managers.

Use case diagrams can be used to describe the requirements of users. The use case diagrams of users and managers are the models of system requirement analysis, and it is also the foundation of system design. Transfer algorithm is the focus of the system design.

We need to create several databases to store user information, station information, route information, and relationship between the routes and stations.

The system is developed in B/S mode. According to the functions, the system is divided into two modules: front desk query module and background management module [4]. Query module at the front desk is open to all users, without logging in and permission setting. Management module determines whether the user has authority to enter the background management page by checking the user name and password. After a successful login managers will enter the background module. In this module managers can add, delete, and modify related information. The software architecture of bus management system diagram is shown in Fig. 1.

Design of transfer algorithm

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… 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…B_n}, and each B_i is a route through 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.

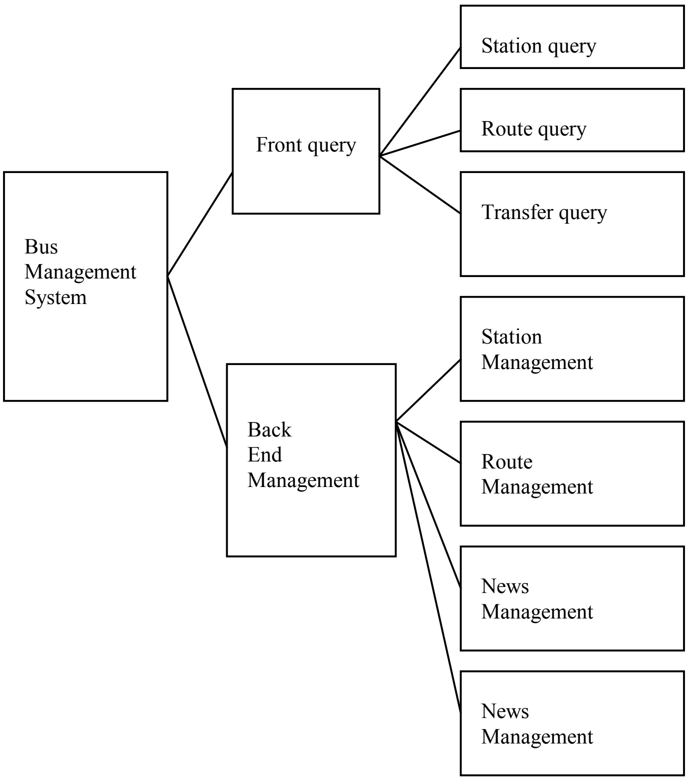

Fig.1 The software structure diagram

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, and calculates intersection routes between them one by one. Intersection stations are transfer stations (there may be multiple or 0). For example, to judge whether A_i and B_j have intersection stations, If so, the stations are transfer stations, If not, there is no way to reach its destination through A_i and $B_{j\,.}$ If the routes have multiple cross stations, then choose only the first cross site as transfer station.

The Design Flow Chart of Algorithm. System input data is the start station and termination station. After searching and comparing route set, the bus routes can be queried. If there is a direct line, it gives information of the routes. If there is no direct route, it then gives information of the transfer routes, as shown in Fig. 2.

Secondary Transfer Algorithm. Suppose a collection of routes through the start station is A, A= {A_1, A_2 ...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…B_n}, and each B_i is a route through the termination station.

Set a collection of other routes is C, C:{C_1, C_2 ...C_r}, and each C_i is a route not through the start station and termination station.

First, secondary transfer algorithm in C looks for intersection routes set with A, the set as D, D:{D_1, D_2 ... D_s}, each D_i is a route that has intersection stations with A, which are the transfer stations.

Then, Secondary transfer algorithm in B looks for intersection routes set with B, the set as E, E= {E_1, E_2 ... E_t}, each E_i is a route that has intersection stations with D and the intersection stations are the transfer stations.

Between two routes there may be more than one intersection point. In order to reduce the complexity of computation, the algorithm uses only the first intersection station as transfer station. If secondary transfer cannot be reached, according to the above algorithm, keep looking for the 3rd transfer stations. If so, complexity of computation and algorithms will greatly enhance. In general traffic or rail system, only a maximum of two transfers is taken into account.

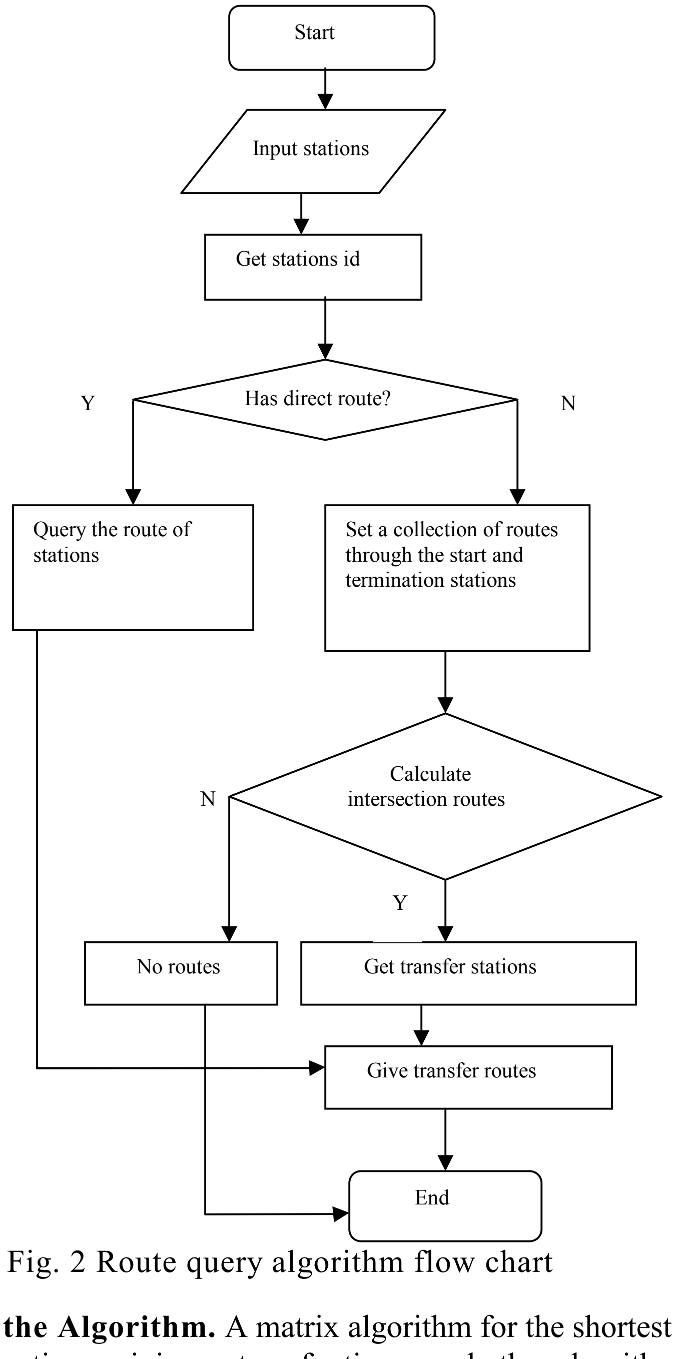
Start
Input stations
Get stations id
Has direct route?
Y
N
Query the route of stations
Set a collection of routes through the start and termination stations
Calculate intersection routes
N
Y
No routes
Get transfer stations
Give transfer routes
End

Fig. 2 Route query algorithm flow chart

Evaluation of the Algorithm. 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]. The proposed algorithm above is simpler and quicker than these.

Once transfer algorithm mainly searches through the start station routes set and termination stations set to determine whether it has intersection, if there is one, it is transfer station. Secondary transfer algorithm is the same as the one, but with much greater computation. This algorithm is simple and practical, you can quickly query bus routes, but with a high number of complicated calculation and comparison. Especially, if there a lot of routes and stations, the query is less efficient.

Further optimization of the transfer algorithm

Transfer Algorithm Based on 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.

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: $\{H_1, H_2 ... H_k\}$, 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.

As to once transfer algorithm, it looks for routes from start station to termination station with each hub as a transfer stations.

As to secondary transfer algorithm, such algorithm requires secondary applications. Principles with the above algorithm are basically the same, I won't go into details about it.

The Shortest Distances Transfer Algorithm. 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 [6,7]. However, for not familiar transfer stations, users often don't choose.

Conclusions

This paper briefly discusses software architecture of bus management system, and mainly proposes a transfer algorithm based on set theory, and its design process. 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. If distances between stations are taken into account, the algorithm can be further optimized. Compared with the traditional complex 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)

[7] Information on http://www.cnblogs.com/