Path Planning for Tottori City Bus Network using Two-stage Shortest Path Algorithm

Soonchun PARK*1, Ryosuke KAWASIMA*2, Takao KAWAMURA*2
and Kazunori SUGAHARA*2

*1Graduate School of Engineering Research Department,Tottori University,
Graduate School of Marine Traffic information Engineering Department,
Korea Maritime University
Korea Maritime University, Busan, 606-791, Republic of Korea
E-mail: pscjerry@hanmail.net

*2Department of Information and Knowledge Engineering, Faculty of Engineering
Tottori University, Tottori, 680-8552 Japan
E-mail: {s022020, kawamura, sugahara}@ike.tottori-u.ac.jp

Abstract: We have developed a path planning system for the bus network in Tottori as Web service. This system finds appropriate paths using the location information of the starting point and the destination obtained through GPS or landmark databases and the current time. Our path planning method can find paths including not only bus transfer but walking transfers. But it is not perfect to search the path by only Dijkstra's Shortest Path Algorithm. So, Two-stage search method is introduced to search for all the bus routes. The system searches the path by using the shortest path estimation gotten from Dijkstra's Algorithm in the 1st stage and selects the most suitable path by using several condition of lopping off branch in the 2nd stage. Experiments are examined to validate the system.

Key words: Path Planning, Dijkstra's Shortest Path Algorithm, Bus Network, Two-stage search system

1 Introduction

The number of people who use buses is decreasing year by year in every province of Japan. Especially, almost 70% of all bus routes operated in Tottori Prefecture was red bottom line in 1999 and, therefore, it became a big burden to the local government to maintain the bus routes. To increase the number of people who use buses, it is necessary to enhance the convenience in several points. The first thing of inconvenience is that it is difficult to get the information of bus route. In case of railroad station, passengers can easily get the route information from route map, time table and/or tariff, or by making inquiries at the station staffs. However, in case of bus stop, there is not enough space to notify detailed route information or no staff to ask route generally. Therefore, route information which can be gotten from the bus stop is limited considerably. In general, railroad stations are indicated very well on the signpost; however, bus stops are not clearly indicated on the guidepost and, therefore, it is very difficult for tourists or strangers to find the location of bus stops. Furthermore, there are route searching systems at railroad stations in general which can search detailed route information by inputting departure and arrival stations; on the other hand, there is not any system which can search route information for bus stops.

We developed a bus route searching system which can output the shortest path in time between 2 points collaborating with bus companies and the Tottori Chamber of Commerce and Industry[1]. Although the bus companies in Tottori City have recognized the urgent need of bus route searching systems in order to provide convenience to passengers, they could not develop such kind of systems due to the lack of experts

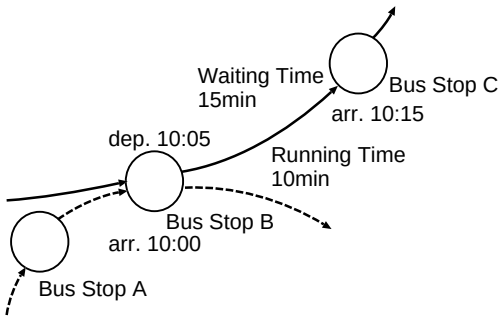

Figure 1: The Weight of an arc including waiting time.

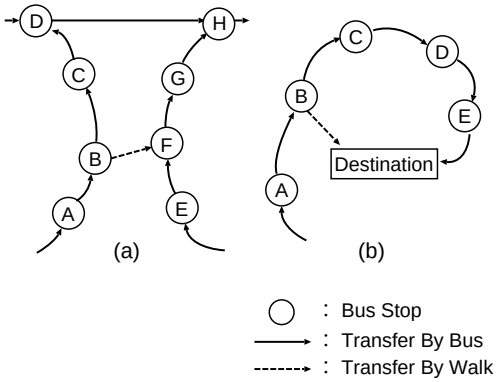

Figure 2: Bus routes including walking transfer.

and the huge amount of route data. Finally it can be realized by the cooperation with researchers in Tottori University who can deal the problem theoretically and logically and can develop large size of database. The Tottori Chamber of Commerce and Industry participated later, which resulted in industrial-educational cooperation, and made very important role in development of the system smoothly.

In our previous paper, the bus route searching problem was regarded as a shortest path in time from a single staring point basically, and the Dijkstra's Shortest Path Algorithm[2], which is used generally in route searching, was applied to solve the problem. However, it is necessary to improve the algorithm or the data structure to construct more practical system, because it may be possible for passengers to move between bus stops by walk, which is the unique and peculiar characteristics of bus route searching. In this paper we will point out the peculiarity of bus route searching algorithm, then explain the development of the system, and finally confirm the validity and efficiency of the developed system by experiment.

2 The bus route search problem

The bus route search problem can be regarded basically as the shortest path from a single starting point of a network.

The network is constructed with nodes which denote bus stops, arcs which denote bus routes and weights of the arcs which denote the needed time between bus stops. The numbers of bus stops may be reduced to the direct line even in the same bus route due to the difference of starting time or the difference of the day of a week. In such a case, the direct line will be treated as a new route which differs from the original bus route. For example, if a bus in a certain bus route stops at the bus stops A and C generally except the starting time zone around 8:00 in which it stops at the bus stops A, B, and C, the network must be constructed as if two different bus routes -1 and -2 exist.

The shortest path in time from a certain node to another node can be obtained by applying Dijkstra's Algorithm to this network. However, in case of a bus-to-bus transfer, the waiting time for the next available bus at a bus stop should be considered. The example is displayed in the Fig.1. Let's suppose that a passenger leaves from a bus stop A through transfer bus stop B for the destination bus stop C. If he arrives the transfer bus stop at 10:00 and the next available bus for C departs at 10:05, then the weight of the arc between the nodes B and C will be 15 which is resulted from the addition of the original weight (10 minutes) and the waiting time (5 minutes). It is impossible to determine the weight of an arc as a fixed one because the waiting time can be changed according to the time of arrival.

Therefore, an algorithm which can calculate a weight dynamically to the next node in accordance with the arrival time of a certain node will be needed. In this study, the bus route search problem will be solved by applying the improved Dijkstra's algorithm.

In some cases, however, the route which is obtained with the improved algorithm may not coincide with the real shortest path in time, because the bus routes are not designed to get the shortest path between two bus stops but to pass the places where as many passengers as possible can use. As a result, the routes are very much complicated and the distances between bus stops are rather short in general and, consequently, a passenger can get off a bus at a bus stop and walk to another bus stop of different route to take a bus.

Figure 2(a) shows an example. If a passenger moves from bus stop A to H only by bus, the only available path is $A \rightarrow B \rightarrow C \rightarrow D \rightarrow H$. However, if he can move by walk between bus stops, the path

$A \rightarrow B \rightarrow F \rightarrow G \rightarrow H$ may be available. Distances between rail road stations are too far for passenger to move by walk generally and, therefore, this kind of problem can be regarded as a unique problem in searching bus routes.

To cope with the problem of transfer at bus stops other than the designated transfer bus stops, arcs by walk from each node to all other nodes of the network will be added. Besides, in some cases, it can be considered that the fastest way to a destination is not necessarily the nearest bus stop to the destination. In other words, it may be faster to get off at a bus stop, which is not the nearest one, and move by walk to the destination. As illustrated in Fig.2 (b), the nearest bus stop to the destination is E. If we do not consider the movement by walk, the only available route will be $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow$ Destination. However, if the movement by walk is considered, the route $A \rightarrow B \rightarrow$ Destination can be included in the target routes for search.

In the same manner, it must be considered that a passenger moves by walk to a bus stop other than the nearest one and takes a bus at the starting point. On the contrary to the railroad route search problem, in which only the paths between the stations are considered, routes between the real starting point and the real destination must be considered in the bus route search problem. To cope with this problem, the starting points and the destinations are inserted in the network as nodes and connected with all other nodes with arcs by walk.

After making the network including walk movement and bus transfer as described, the shortest path can be found by the Dijkstra's Algorithm which is improved for the dynamic calculation. But in many case, the Dijkstra's Algorithm may give complicated result as a root candidate, such as a root include many transfers from bus to bus. To solve the problem, we proposed Two-stage search system in this paper. In other words, this search system finds the estimated shortest path or time in the 1st stage and, in the 2nd stage, searches again the whole network with the estimated value from 1st stage and several search conditions. Then the system can output the most suitable path at a real time.

3 The bus routes search system

The bus routes search system is developed as a CGI program which can be accessed through WWW. C language is used for developing the system.

Data used in the system are based on the 98 bus routes and 1839 bus stops, which is really operated by the Hinomaru Bus Limited and Nihon Kotsu Co.LTD. Four databases are developed with the data as follows and managed by the relational data base server MySQL;

(1) Bus stop table,

(2) Separated route table,

(3) Connection table,

(4) Starting table.

Each record in item (1) corresponds to a bus stop and consists of 4 parts; that is, bus stop ID, the name of the bus stop, the north latitude and the east longitude. Each record in item (2) corresponds to a separated bus route and consists of 3 parts; that is, separated route ID, starting bus stop ID and bit sequence. A bit sequence shows when the data is affective in a weekday, the Saturday in an odd week, the Saturday in an even week, Sunday and school term. The school term is a period of an elementary school year. Each record in item (3) corresponds to a bus stop in the separated bus route and consists of 4 parts; that is, separated route ID, bus stop ID, next bus stop ID and bus running time(minutes) to the next bus stop. Each record in item (4) corresponds to a departure time of separated route and consists of 2 parts; that is, separated route ID and departure time.

These 4 kinds of data are built as a static part of network. So bus route searching network is completed by adding the starting point and the destination in every time of searching and bus route search is conducted by the improved Dijkstra Algorithm calculating dynamically weights of the arcs.

$T_{a:b}$, Required time(minutes) from bus stop A to bus stop B by walk is as follow.

$$T_{a:b} = \lceil N \times d(A, B) \rceil, \tag{1}$$

$d(A, B)$ is the crow flies(m) from bus stop A to bus stop B, N is a positive number.

Fundamentally, it is necessary to connect all bus stops with arcs by walk but it is not realistic to add an arc that has longer time than a certain value. And, therefore, an appropriate threshold value T1 should be set and only the bus stops between which move time by walk is shorter than T1 should be connected.

It is also necessary to connect the starting point to all bus stops with arcs by walk and, in the same manner, to connect all bus stops to the destination with the arcs. However, due to the similar reason, an appropriate threshold value T2 should be set and only the bus stops which have shorter move time by walk from the starting point or the destination should be connected.

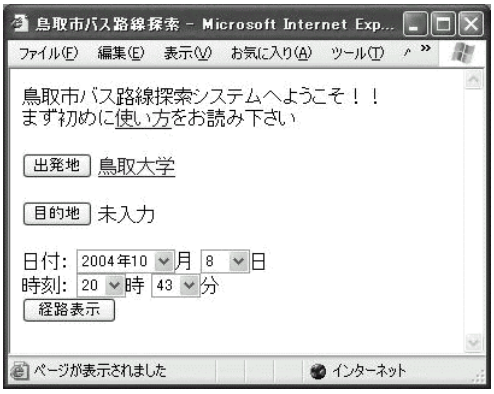

Figure 3: Opening Page of the Proposed System.

For the network constructed like the upper, Two-stage search system is developed to search the most suitable path as quickly as possible. In the 1st stage of the system, it finds the estimated shortest path by the improved Dijkstra's Algorithm which is improved to calculate dynamic accounts. And, in the 2nd stage, this system seeks for the optimum bus route using the estimated shortest path obtained by the 1st stage.

3.1 The estimated shortest path from the Dijkstra's Algorithm (the 1st stage)

The calculation time in Dijkstra Algorithm as the 1st stage depends on the calculation time for scanning the node of the shortest distance among the ones which are connected to a certain node (Extract-Min operation, hereinafter). Currently, the Priority Queue utilizing Fibonacci heaps is the fastest one among the data structures that have been utilized in this operation. In the priority queue, the calculation amount for one Extract-Min operation is O(log n) when there are n nodes. On the other hand, in this system, we do not utilize the Fibonacci Heaps but the data structure called bucket method[3]. In this method, queues are prepared for every value which is expected to be the shortest path, and the calculation amount is O(1) for one Extract-Min operation. However, the memory consumption by this method is increased with the number of expected shortest paths; it is useful only for cases where the expected shortest paths are limited to a certain number. The bus route search problem is an adequate problem for the bucket method because the expected shortest path is given in the unit of minutes and, further more, the maximum time in minutes is limited within the longest move time by bus.

Input data into the system is composed with following 3 data ; i) departure time, ii) starting point, iii) destination as in the Fig. 3. The starting point and destination can be appointed by using land marks. Important 2,085 facilities in Tottori prefecture are managed as land marks by MySQL. And the staring point can be inputted by GPS when using the cell phone with GPS.

3.2 Search for the suitable path at real time according to the search conditions (the 2nd stage)

The suitable path can be obtained by searching again all the paths of bus routes network using the estimated shortest path from 1st stage (by Dijkstra's Algorithm), but the search time may be too long due to huge amount of data. Accordingly, now, we introduce the idea of the separated bus stop to search the whole network and find the suitable path and, at the same time, the search conditions (lopping off branch) are suggested to reduce the searching time. And useful conditions to do that are presented. This is the 2nd stage.

The search method is considering the walking transfer between bus stops and the number of switching to another bus.

The search system that considers the shortest path in time only can produce the routes of the shortest time because it is based on the Dijkstras Algorithm; however, if there are plural number of routes which have the same time length, the output route among them wholly depends on the order of search sequence. It may be unpractical in some cases. In other words, the system may produce the route which has too long move time by walk or too many switching numbers to another bus.

We developed a search method which can find the optimum route by searching the whole paths of bus route network again after limiting the search range with the route obtained from the preliminary search considering only the shortest path in time. The system stops when it finds the limiting conditions during the search operation. The limiting conditions in this system are as follows;

(1) When the arrival time at a certain node exceeds the time limit which results from the addition of the shortest time between starting point and destination to the departure time at the starting point.

(2) When the number of switching to another bus at a certain node exceeds the number obtained by the search method considering only the shortest path in time.

To restrict the search range more, the upper condition 1 is changed into more strict one such that arrival time at a certain node exceeds the time of adding the departure time to d of that node. Here, d is the shortest time from a starting point to a destination by the search method considering only the shortest path in time. In the network stated in the chapter 2, it is impossible to make the conditions more strict. It is explained at Fig. 4. supposing that there are the route that doesn't go any more and the route that goes to the bus stop B at bus stop A. Arrival time at the bus stop A on the route is 8:10 and arrival time at bus stop A on the route is 8:15. In the case of arrival on the route , people must wait 5 minutes at bus stop A and then transfer on the route to go to the bus stop B. But, in the case of arrival on the route , it is possible to go to the bus stop B by the same bus without transfer. Both of them arrive on the bus stop B at 8:25. d of bus stop A is 8:10 because it is the fastest arrival time of that node. So, if applying the more strict condition, the path on the route does not become a target for the search but can become a more practical one. To correspond to this matter, the definition of d is changed as "the fastest one among departure times after arriving from the starting point to the node". At the Fig. 4, the newly defined d becomes 8:15 and the path on the route becomes a target for the search.

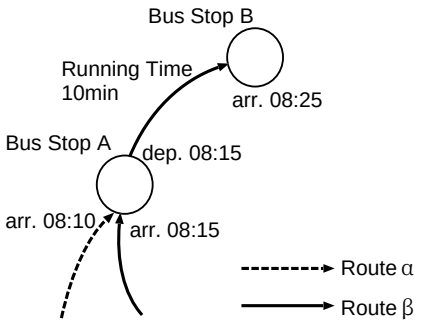

Figure 4: Search range can not be restricted by d.

If plural arcs from a certain node come out, the departure time of that node differs according to which arc to be faced. For that reason, the number of arc coming out from a node must be 1 to apply the condition mentioned before. For that, the new network which has the separated node is constructed. The network before division shows at Fig. 5(a) and the new network after division shows at Fig. 5(b). X_i(i=1, 2, 3) indicates weights of arcs and W_i(i=2, 3) indicates waiting time at nodes. As shown at the Fig. 5, the waiting time at nodes is added as weights of arcs of that node in the old network. But waiting time is

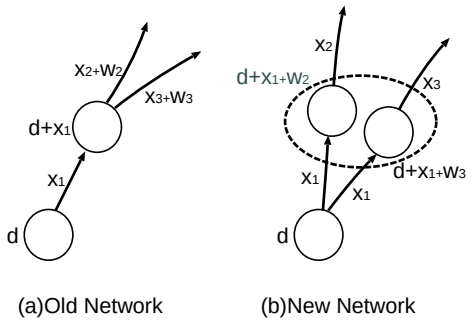

Figure 5: Old network and new network including the separated nodes.

added at the d of nodes in the new network.

It becomes possible to apply condition 1 at the new network. Accordingly, the search range is limited largely.

And next, the proper and reasonable search conditions are proposed as follows to shorten the search time of the whole network to get the suitable bus route. The search time can be reduced greatly by lopping off unnecessary branches from the search tree using these conditions.

1) Successive walking transfer should be cut. Move by walk must form a branch only.

2) It should be cut if the elapsed time from the departure to the current point exceeds the estimated shortest time to destination obtained by Dijkstra's algorithm.

3) The case that the number of bus transfer exceeds a setting number should be excluded. A setting number starts from 2 and, if there is no solution with the setting number, will be repeatedly set in increasing order up to the maximum number of transfer time obtained by Dijkstra's Algorithm. There may be many shortest routes with the same arrival time in the network; however, the number of bus transfer in the effective bus route should not exceed the number obtained by Dijkstra's Algorithm.

4) It should be excluded if the elapsed time from the departure to the current point exceeds the estimated shortest time to virtual bus stop obtained by Dijkstra's algorithm. In case (2), the elapsed time is compared with the estimated shortest time to destination. On the other hand, in this case, the elapsed time is compared with the estimated shortest time to a virtual bus stops which locate on the way to destination. The searching time can be reduced greatly by cutting off the branches in advance which are judged unnecessary by the comparison to the destination as well as to the virtual bus stops.

5) It should be excluded when the arrival time to a

virtual bus stop obtained in current search exceeds the arrival time to the same bus stop obtained by the past search on condition that the number of bus transfer must be the same. For example, if the number of bus transfer to a virtual bus stop is the same number 3, and the arrival time in current search is 1035 while the time in the past search is 1030, then the arrival time should be excluded.

4 New concepts for this system

The new concepts, the Zone and the Serializable Network are used in developing this search system. A concept of Zone is introduced to solve the matter of walk movement and a Serializable Network including static part of network is made to shorten the search time when people do a search for bus routes by using this system.

(1) Zone

(2) Serializable Network

A zone is made to find an virtual bus stop which a transfer by walk is possible from a certain point. It is constructed with the horizon which denotes an latitude and the vertical which denotes longitude, and Tottori City is divided by it as a space of 5 minutes walk. When constructing the network, the positions of a certain virtual bus stop and an accessible virtual bus stop by walk in 5 minutes in its neighboring 8 zones are connected by zone. In the case of searching the network, the search expands as 8 near zones (level 1), 16 near zones (level 2), 32 near zones (level3) until when an virtual bus stop is found from the coordinates of a staring point and a destination. If an virtual bus stop is found, the system searches for one more level and memorizes all the bus stops in the zone. If move time by walk to the nearest bus stop puts as 'n', only the accessible bus stops within 5*(n/5 +1) are connected with a starting point or a destination. Therefore, it becomes easy to find the reachable bus stop by walk by making "the zone".

When people actually use the system, the network and the search parts are sectioned and the network part is prepared in advance to shorten a search time. Static part of network is drawn up and saved in files previously. In searching time, a system reads the network and adds a starting point and a destination. Pointers is excluded by using cursors to preserve the network in files.

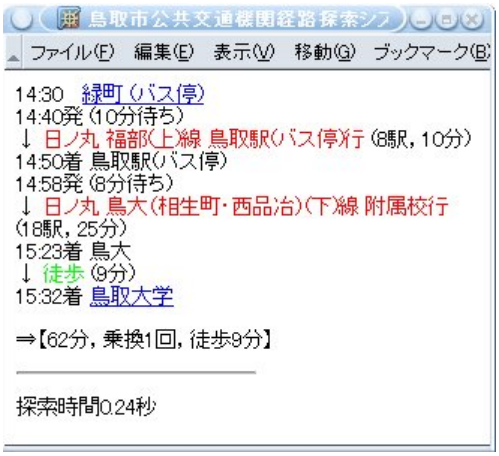

Figure 6: Path planning result not including walking transfer between bus stops.

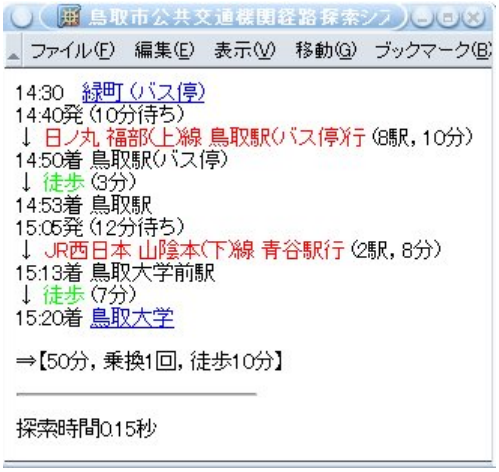

Figure 7: Path planning result including walking transfer between bus stops.

5 Experimentation

Experimentation has done to check that moving time from a start point to a destination can be reduced by considering walking movement. It is supposed that N in the previous chapter 3 is 1/50, T1 & T2 are 20. If the bus stop can not be found, T2 is increasing 5 each step until finding it.

In the case of starting point; "Midori Machi", destination; "Tottori University" and departure time; 14:30, the path considering walking movement between bus stops and the path not considering it are indicated in the Fig. 6 & 7. This example corresponds to Fig. 2(a). When a passenger transfer buses between "Tottori Station bus stop" and "Tottori JR Station"(in walking movement) is 12 minutes shorter than when

Figure 8: A Result of the 1st stage search.

passenger transfer at the only bus stops including plural routes.

The next experimentation has done to check that it is possible to get more practical bus route by 2nd stage of Two-stage search system. The case is with starting points; "Higasi Junior High School", destination; "Tottori Sand Dune" and departure time; 16:56, bus route in Fig. 8 is gotten from Dijkstra's Algorithm(1st stage search) and bus route in Fig. 9 is gotten from 2nd stage search. Travel time from a starting point to a destination is same as 65minutes at Fig. 8 & 9, but transfer time is 3 in Fig. 8 and 1 in Fig. 9. So, it is known that the case of Fig. 9 by 2nd stage search is more practical because it has a few transfer time. And all the route search time is Fig. 6, 7, 8 and 9 have not exceeded 1 second in calculation. They are very practical speed.

It is verified that there is a possibility to find the path which the needed time is shorter one by considering walking movement and to get the usable bus route actually through 2nd stage of Two-stage search system.

6 Conclusion

A bus route search of practical speed becomes possible after that we improve Dijkstra's Algorithm to

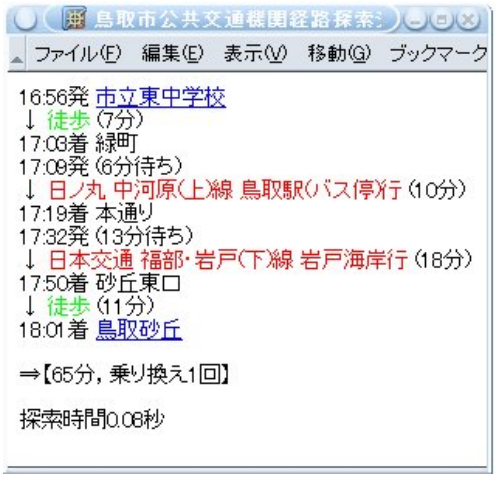

Figure 9: A Result of the 2nd stage search.

calculate a weight of the arcs corresponding with arrival time at the arcs and to select high-speedily the next arc which has the value estimated as the shortest path. Possibility to print the more practical path that required time is shorter by considering move by walk between bus stops in the search algorithm and that transfer time is much smaller by using Two-stage search system is known. Specially, Two-stage search system makes for people actually to refer to the suitable path served from our system very useful. Available bus route search system actually in Tottori City is constructed on the ground of the Algorithm developed cooperating with bus company in Tottori City.

References

[1] Takao Kawamura, et.al:Path Planning System for Bus Network Including Walking Transfer, IPSJ Journal, Vol.46, No.5,pp.1207-1210(2005).

[2] Dijkstra, E.W.: A Note on Two Problems in Connection with Graphs, Numerische Math.1, pp.269-271 (1959).

[3] Goldberg, A.V. and Silverstein, C.: Implementations of Dijkstra's Algorithm Based on Multi-Level Buckets, Technical Report 95-187, NEC Research Institute, Inc. (1995).

(Received September 30, 2005)