OPTIMAL ROUTING IN COMMUNICATION NETWORKS WITH DELAY VARIATIONS

Masaki Aida, Ichizo Nakamura and Teruyuki Kubo

NTT Telecommunication Networks Laboratories
1-2356 Take Yokosuka-shi Kanagawa, Japan

ABSTRACT This paper addresses the problem of optimal routing in packet switched networks. Optimality is discussed in terms of end-to-end delay. We take into account the variance of delay as well as its mean value. Achieving optimal routing is necessary for multimedia networks to fully support real-time services. We model a network as a weighted graph with its link weights representing link delays. We assume that the delay statistics conform to a normal distribution. In the course of analysis, we show that this type of routing optimization problem can be formulated as a process of searching for a specific point in a coordinate system defined by the mean and variance of the end-to-end delay. This paper presents an efficient algorithm for finding the optimal point in this coordinate system.

I. INTRODUCTION

With the evolution of multimedia services, development of integrated transport mechanisms to support them is strongly desired. Among several candidate mechanisms, transport integration with a packet switching technique is the most promising. Transporting real-time traffic over a packet switched network, however, introduces new technical challenges; the most critical problem is to minimize the end-to-end delay of packets across a network.

In the past, various ways have been offered for routing in packet switched networks. Their main goal is to minimize the mean value of end-to-end delay across the network[1]. This class of problem is rather tractable and can be efficiently solved by applying the well-known Dijkstra's algorithm[2]. Network delay as perceived by end users, however, is determined not only by the mean value but also by its variance.

Associated with each telecommunication service, there exists a maximum permissible delay that should be guaranteed to obtain satisfactory quality. Let us take interactive voice communication as an example. Voice packets received by the listener side terminal are temporarily stored in a buffer before they are played back to smooth out the packet delay jitter. Play back is performed according to a continuous clock source. The required amount of storage must be long enough to make the chance of buffer overflow/underflow reasonably small; a situation that may occur with excessive packet delays. Furthermore, packets that experience delays longer than a certain value will be too late for play-back and must be discarded at the receiving buffer.

Thus, to obtain satisfactory service quality, the routing should be designed to take the fluctuation of delay into account. Studies of routing that consider delay fluctuations, effectively support various types of traffic on the same network[3]. As opposed to the optimal routing in terms of mean delay minimization, this class of problem requires a different formulation. As will be described later, Dijkstra's algorithm cannot directly be applied.

Based on the above analysis, we will first formulate the routing problem as the minimization of the probability that packets exceed the maximum permissible delay (*sec.2*). We model a network as a graph with its link weights representing delays and assume that the delay statistics obey a normal distribution. Our problem is regarded as a stochastic version of the shortest path problem on a weighted graph. Ichimori *et.al.*[4] treat a related problem which deals with a minimum spanning tree. We will show that the treatment proposed in their paper can be efficiently used as one tool to solve our problem (*sec.3*). Their treatment, however, is not sufficient to solve the stochastic shortest path problem by itself. A shortest path problem is a more difficult class of problem than the minimum spanning tree problem and requires some extensions. Next, we show that the optimization of routing in our problem is equivalent to searching for a specific point in a coordinate system determined by the mean and variance(*sec.4*) of end-to-end delay. Using this formulation, we construct a simple and efficient algorithm for the problem (*sec.5*) and evaluate performance in terms of the relationship between its time complexity and estimation error(*sec.6*).

II. OPTIMAL ROUTING PROBLEM

We represent a network as a graph G = G(V,E) where V is the set of nodes and E is the set of links connecting the nodes. Each link denoted by e(i) or simply **i** has a weight c(i) that represents delay experienced by packets on the link. In general, the value of c(i) fluctuates dynamically. We assume that c(i) is an independent normal random variable characterized by its mean m(i) and variance v(i). We further assume that the mean m(i) and variance v(i) are stable over time. We refer to c(i) as a stochastic weight. We treat a shortest path problem between given two nodes in the following part of this paper.

Let π denote a path between given two nodes in G and $\Pi=\{\pi\}$ be the set of all paths between the given two nodes.

Given any path π, based on the independence assumption,

CH3133-6/92/0000-0153 $3.00 © 1992 IEEE

the path weight $C(\pi)$, its mean $M(\pi)$, variance $V(\pi)$, and standard deviation $S(\pi)$ can be expressed as:

$$C(\pi) = \sum_{i\in\pi} c(i),\ M(\pi) = \sum_{i\in\pi} m(i),$$
$$V(\pi) = \sum_{i\in\pi} v(i) \text{ and } S(\pi) = \sqrt{V(\pi)}\ . \qquad (1)$$

The conventional routing problems have been formulated as finding a particular path π between two nodes from the set of paths Π that minimizes the mean end-to-end delay. In this formulation, the problem is expressed in the form:

$$\min_{\pi\in\Pi} M(\pi). \qquad (2)$$

This problem depends only upon the mean value of delay because eq.(2) does not take the variance of delay into consideration. Previous studies show that this type of shortest path finding problem can be solved efficiently by applying dynamic programming techniques; *e.g.* Dijkstra's algorithm.

In contrast, our aim is to take the variance of delay into consideration, so we formulate our optimal routing problem as follows:

$$\min_{\pi\in\Pi} Pr(C(\pi) > C_0), \qquad (3)$$

where C_0 is a given positive real constant representing the maximum permissible delay. The purpose of our optimization is to minimize the probability that the end-to-end delay exceeds a predefined value. Solving our problem expressed in this form is mathematically intractable because of the presence of the random variable $C(\pi)$ in eq.(3). After a little manipulation we can obtain an equivalent formulation [4]:

$$\max_{\pi\in\Pi} \frac{C_0 - M(\pi)}{S(\pi)}\ . \qquad (4)$$

This formulation consists only of deterministic variables $M(\pi)$ and $V(\pi)$ and thus is more tractable. Hereafter, we name the function to be maximized an evaluation function of the path π:

$$E(\pi, C_0) := \frac{C_0 - M(\pi)}{S(\pi)}\ . \qquad (5)$$

III. APPLICATION OF DYNAMIC PROGRAMING TECHNIQUES

Unfortunately, the evaluation function $E(\pi, C_0)$ still has the property that is hard to treat. This section explains what this difficulty is and how it can be overcome.

Imagine the situation that there are three nodes (A, B and C) and they are connected by three links as shown in fig.1. Each link and path have weights that are evaluated by $E(\pi, C_0)$. We further assume that link "b" is the optimal path between nodes B and C. It can easily be shown that, if the evaluation function $E(\pi, C_0)$ is used to evaluate the path/link weights, the optimal path between node A and C does not always include path "b" which is the optimal path between node B and C. It means that the principle of optimality[5], which is the basis for applying dynamic programming techniques[6], is not effective when this evaluation function is used.

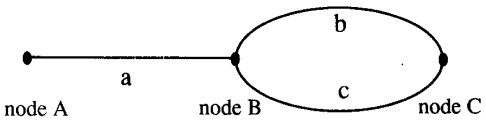

Fig.1 Example of Network I

Hence, the well-known, Dijkstra's algorithm cannot directly be applied to our problem because the algorithm employs the optimality principle.

To avoid the above difficulty, we introduce a new measure for the link and path weights. The following approach is after Ichimori *et.al.* [4] whose work deals with the stochastic minimum spanning tree problem. This technique is also effective in solving our problem. We introduce a positive parameter **x** (has a dimension of 1/(time)) and define, the weight (which we call x-weight) of a path π as:

$$W_x(\pi) := M(\pi) + x \cdot V(\pi)\ . \qquad (6)$$

Following this definition, the x-weight of the individual link e(i) can be expressed as:

$$w_x(i) = m(i) + x \cdot v(i)\ . \qquad (7)$$

With this formulation of path and link weights, an end-to-end path weight can be obtained by simply summing up the weights of all links that comprise the path, i.e.,

$$W_x(\pi) = \sum_{i\in\pi} m(i) + x \cdot \sum_{i\in\pi} v(i)$$
$$= \sum_{i\in\pi} (m(i) + x \cdot v(i)) = \sum_{i\in\pi} w_x(i). \qquad (8)$$

Hereafter, we call $w_x(i)$ the x-weight of link e(i).

Next, we define the notion of x-optimal path between a given pair of nodes. A path $\pi_{x\text{-opt}}(x)$ is x-optimal if its x-weight satisfies the following minimization for a given value of $x \geq 0$.

$$\min_{\pi\in\Pi} W_x(\pi)\ . \qquad (9)$$

In general, the configuration of the path $\pi_{x\text{-opt}}$ varies with the value of x. Let $\Pi_{x\text{-opt}}$ denote a set of such optimal paths and let π_{opt} denote the real optimal path that satisfies the original problem (eq.(4)).

The x-weight thus defined has a useful characteristic which allows our problem to be solved. We show this in the following theorem.

Theorem 1 : The real optimal path π_{opt} is included in the set $\Pi_{x\text{-opt}}$ of x-optimal paths.

$$\pi_{opt} \in \Pi_{x\text{-opt}}\ . \qquad (10)$$

Proof : We prove this by contradiction. Suppose the real optimal path π_{opt} is not included in the set $\Pi_{x\text{-opt}}$. For any path $\pi_{x\text{-opt}}$, including those in $\Pi_{x\text{-opt}}$, we can show the following inequality.

$$M(\pi_{x\text{-}opt}) - M(\pi_{opt})$$

$$> E(\pi_{opt}, C_0)\cdot(S(\pi_{opt}) - S(\pi_{x\text{-}opt})) . \quad (11)$$

We can arbitrarily select the value of x as $x' = E(\pi_{opt}, C_0)/(2S(\pi_{opt}))$, and there should exist a corresponding x-optimal path $\pi_{x\text{-}opt}(X')$ in $\prod_{x\text{-}opt}$ and it should satisfy

$$W_x(\pi_{x\text{-}opt}(x')) < W_x(\pi_{opt}) . \quad (12)$$

However, it follows that

$$W_x(\pi_{x\text{-}opt}(x')) - W_x(\pi_{opt})$$

$$= M(\pi_{x\text{-}opt}(x')) + x\cdot V(\pi_{x\text{-}opt}(x')) - (M(\pi_{opt}) + x\cdot V(\pi_{opt}))$$

$$> \left\{ \frac{E(\pi_{opt}, C_0)}{2S(\pi_{opt})} \right\} (S(\pi_{opt}) - S(\pi_{x\text{-}opt}(x')))^2 > 0 . \quad (13)$$

Thus the contradiction. Q.E.D.

This theorem dictates that we can take the following steps to solve our problem: first, we solve for the set of x-optimal paths for the x-weight problem, to which the optimality principle holds and can be solved efficiently with the Dijkstra's algorithm; then we search for the real optimal path from among the x-optimal paths.

IV. REPRESENTATION OF PATHS ON (M,V)-PLANE

The main difficulty with the above derived approach is how to find out all the x-optimal paths for a given graph. To overcome this difficulty, we introduce, in this section, a new coordinate system to analyze our problem. It turns out to offer a useful means to systematically search for the x-optimal paths and the real optimal path.

Specifically, our problem is formulated as searching for a specific point in the plane.

In the following, we show how each x-optimal paths π_x's are represented as points in the plane. We then examine the geometrical properties of the points in the plane that can be used to shrink the domain where the target point should be searched for.

The x-weight defined by eq.(7) is a linear function of **x**. In general, any linear function is completely determined by specifying two parameters, *i.e.*, its slope and intersection at the vertical axis. Here, we introduce a plane that is defined by M-axis and V-axis, representing the intersection and slope of the x-weight function respectively, which in turn represent the mean delay of the corresponding path and its variance respectively. On this plane, referred to as (M,V)-plane, each path π is represented by a single point $P=(M(\pi),V(\pi))$ as shown in fig.2.

Next let us examine the contours of the original evaluation function, eq.(5), on the (M,V)-plane. For a given value E of the evaluation function, the corresponding contour is expressed as

$$V = \frac{1}{E^2}(C_0 - M)^2. \quad (14)$$

The resulting shape of the contour is parabolic. Several sample contours are also shown in fig.2.

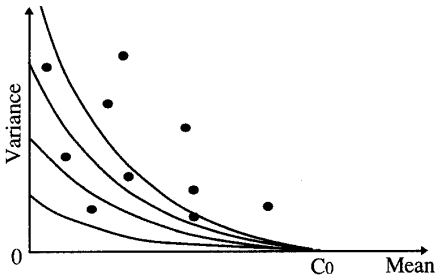

Fig.2 Distribution of Paths and Contours for Evaluation Function

Following the above development, our purpose can be restated as finding for a point with the largest evaluation. The point is expected to be found towards the lower left corner on the (M,V)-plane.

To systematize this searching process, we first examine basic relationships between two points, P_1 and P_2, on the (M,V)-plane with the following two lemmas. In the following discussion, a path π_i is represented by the point P_i whose Cartesian coordinate is given by (M_i, V_i), where $M_i = M(\pi_i)$ and $V_i = V(\pi_i)$.

Lemma 1 : Let π_1 and π_2 be different paths belonging to $\prod_{x\text{-}opt}$. Let points P_1 and P_2 on the (M,V)-plane represent the paths π_1 and π_2 respectively. If there is some $x = x_0 > 0$ that satisfies

$$W_x(\pi_1) = W_x(\pi_2) \text{ for } x = x_0, \quad (15)$$

then the slope of the line segment P_1P_2 is negative and its value is $-(1/x_0)$.

Proof : By applying eq.(6) into eq.(15), x_0 is calculated as

$$x_0 = -\frac{M_1 - M_2}{V_1 - V_2} . \quad (16)$$

Therefore the slope of the line segment P_1P_2 is:

$$\text{slope} = \frac{V_1 - V_2}{M_1 - M_2} = -\frac{1}{x_0} < 0. \quad \text{Q.E.D.} \quad (17)$$

Lemma 2 : If

$$W_x(\pi_1) = W_x(\pi_2) \text{ for } x = x_0, \text{ and}$$

$$W_x(\pi_1) > W_x(\pi_2) \text{ for } x > x_0, \quad (18)$$

then

$$M_1 < M_2 \text{ and } V_1 > V_2 . \quad (19)$$

Proof : By applying eq.(15) to eq.(18), we have

$$(M_1 - M_2) + x\cdot(V_1 - V_2) > 0 \text{ for } x > x_0 , \quad (20)$$

Using the result of lemma 1

$$V_1 - V_2 = -\frac{M_1 - M_2}{x_0} . \quad (21)$$

Substituting the above relation into eq.(20),

$$(M_1 - M_2)(1 - \frac{x}{x_0}) > 0 \text{ for } x > x_0 > 0 . \quad (22)$$

Thus

$$M_1 - M_2 < 0 \text{ and } V_1 - V_2 > 0 . \quad \text{Q.E.D.} \quad (23)$$

We next examine the geometrical distribution of points P_i's, representing the collection of x-optimal paths, over the (M,V)-plane. This distribution has an outstanding property described by the following theorem. Before entering into details, we should mention some properties that can be intuitively understood:

1) There exists only a countable number of x-optimal points.
2) Each point presents x-optimality for a certain range of x, *i.e.*, for $x_i \le x < x_{i+1}$. So, supposing that the complete set of x-optimal paths has been obtained, we can number the points in the manner that P_i is optimal for $x_i \le x < x_{i+1}$, where i = 1,2,3,...

Theorem 2 : Suppose that the complete set of the x-optimal points have been obtained and they are numbered in the manner that a point presenting the optimality with larger value of x is given a greater number. Connect these P_i's on the (M,V) plane by straight lines in ascending order and the resulting shape is convex and monotonically decreasing [see fig.3].

Proof : Any pair of points from $\prod_{x\text{-opt}}$ satisfy eq.(15) for some x > 0. Thus, following lemma 1, the curve is monotonically decreasing.

Let P_i, P_j and P_k be different points on the (M,V)-plane, where they are determined to be x-optimal for the value x of x_i, x_j and x_k respectively, and $0 < x_i < x_j < x_k$.

Let slope(P_i, P_j) and slope(P_j, P_k) denote the slopes of line segments P_iP_j and P_jP_k, respectively. Because these points correspond to paths in $\prod_{x\text{-opt}}$, we can always find some real values **a**, **b** for x such that

$$W_x(\pi_i) = W_x(\pi_j) \text{ for } x = a, \qquad W_x(\pi_j) = W_x(\pi_k) \text{ for } x = b. \quad (24)$$

For these to be satisfied, **a** and **b** must satisfy the relationship:

$$0 < x_i < a < x_j < b < x_k . \quad (25)$$

From lemma 1, we have

$$\text{slope}(P_i,P_j) = -\frac{1}{a}, \quad \text{slope}(P_j,P_k) = -\frac{1}{b}, \quad (26)$$

hence

$$\text{slope}(P_i,P_j) < \text{slope}(P_j,P_k) < 0 . \quad (27)$$

That is, the slope is less steep on the right side of line connected lines.

This property holds true for any combination of x-optimal points and hence the shape is convex. Q.E.D.

Using the above results, we now show an algorithm that locates the optimal point in the (M,V)-plane. Our algorithm starts with a known set of x-optimal points on the plane. The main steps carried out in the algorithm consist of:

1) determining regions in which the candidate points are to be searched for,
2) determining the region in which the search process is first applied, and
3) shrinking the search regions based on the newly obtained x-optimal points.

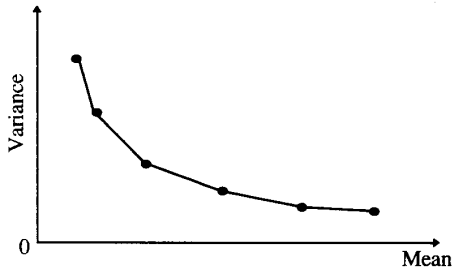

Fig.3 Distribution of x-optimal Paths

Assume that x-optimal points P_i, P_j, P_k and P_l are known on the (M,V)-plane, where $x_i < x_j < x_k < x_l$. We consider the problem of searching for a new point P_n for $x_j < x < x_k$. The domain where P_n should be searched for is a triangle [see fig.4] and its location is determined in the following manner. Two vertices of its triangular domain are trivially found to be P_j and P_k. We define the new vertex Q_n as the cross point of two extended lines, P_iP_j and P_kP_l . The reason for this is easily understood by considering the fact that if P_n is to exist for $x_j < x < x_k$, $P_jP_nP_k$ the connecting line must be convex as dictated by theorem 2.

Within this triangle, the candidate points that offer the highest value in terms of the evaluation function eq.(5) are the aforementioned three vertices. If the evaluation with the new point Q_n gives a larger value than with the other two points, the region is worth searching.

For a given set of known points, we can identify as many triangles as there are line segments. Each newly found vertex must have a corresponding evaluation. The triangle with the new vertex which is evaluated to have the largest value, is the most promising place to search for a new x-optimal point. This is because, if we can find a new x-optimal point in this region, the point is expected to yield the largest evaluation value.

Once we find a new x-optimal point, we can restart the algorithm with the increased number of known points. In this case, following the same reasoning as the above, we can shrink the search domain as shown in fig.5. Note that finding a single point results in the shrinkage of not only in the original triangle but also in the two adjacent triangles. As the cycles of the algorithm repeat, the search domain continues to shrink until the the point with the highest evaluation, or the real optimal point, is determined.

For a given triangle $\Delta(P_j,P_k,Q_n)$, we can find a new x-optimal point in the following manner. We use

$$x = -\frac{1}{\text{slope}(P_j,P_k)} \quad (28)$$

to solve for the x-optimal path by applying Dijkstra's algorithm. Its substantiation is explained in Appendix I. If we

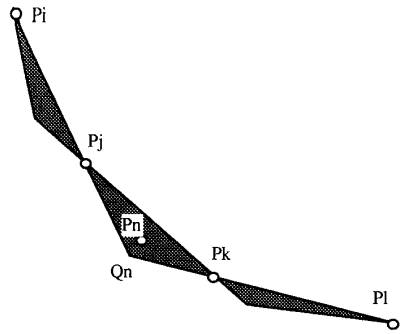

Fig.4 Search Domain I

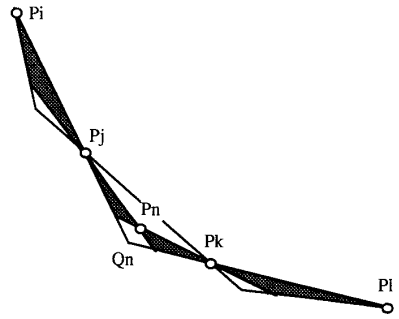

Fig.5 Search Domain II

can find a new path other than those corresponding to P_j or P_k, this would be the one with the highest evaluation value. Otherwise, there are no x-optimal points in this region, and this region should be excluded from further investigation.

V. FORMAL DESCRIPTION OF ALGORITHM

In this section, based upon the previous discussion, we introduce a formal description of our algorithm. The algorithm is as follows:

step 1 Given two nodes on G, find initial x-optimal paths between them with x set to 0 and infinity. The path finding processes can be done using Dijkstra's algorithm. Calculate the corresponding evaluations $E(\pi, C_0)$ by applying eq.(5) and set E_{opt} and E_{exp} to the largest value.

step 2 Determine search domains from a set of known x-optimal points, and evaluate $E(\pi, C_0)$ for each new vertex Q_n. For the domain $\Delta(P_j, P_k, Q_n)$ with the highest evaluation, E_{exp} update, and find a new x-optimal path applying Dijkstra's algorithm with x set to $-(1/\text{slope}(P_j, P_k))$. If there remains no candidate domain, go to step 4.

step 3 If a new path exists, we plot the corresponding point P_n on the (M,V)-plane. Calculate $E(\pi, C_0)$ for the new point P_n and if it is larger than E_{opt}, update E_{opt} to this value. Exclude from the candidate list those domains whose evaluation values are smaller than E_{opt}.

Otherwise no new path can be found, and we exclude the domain from the search domain list.

In both cases, go back to step 2.

step 4 Terminate the algorithm.

VI. WORST CASE ANALYSIS OF THE ALGORITHMS

To assess the computational complexity of our algorithm, we evaluate the number of required calculation steps, and compare it with the brute force search method. In the worst case, our algorithm must evaluate all x-optimal paths. Using a complete digraph for comparison, the number of x-optimal paths, I(n), is upper-bounded by

$$I(n) \le \frac{3^{n-2}+1}{2}. \tag{29}$$

Derivation of the above equation is given in Appendix II.

When a brute force search is applied, all paths between the given two nodes must be evaluated, where the number of the paths can be upper-bounded by

$$H(n) \le \sum_{k=0}^{n-2} \frac{(n-2)!}{k!}. \tag{30}$$

Since $I(n) < H(n)$, the algorithm can be run in less time than the brute force search. However, unfortunately, the time to compute the algorithm is proportional to the n-th power of 3 for the worst case and the computation does not terminate in polynomial time.

The main strength of our algorithm is that we can control the residual error caused by premature termination of the algorithm. Let us consider following two probabilities, that end-to-end delay exceeds the maximum permissible delay;

$$PR_{opt} = \frac{1}{\sqrt{2\pi}} \int_{E_{opt}}^{+\infty} \exp(-\frac{x^2}{2})\, dx, \tag{31}$$

$$PR_{exp} = \frac{1}{\sqrt{2\pi}} \int_{E_{exp}}^{+\infty} \exp(-\frac{x^2}{2})\, dx, \tag{32}$$

where E_{opt} is current optimal value and E_{exp} is expected optimum value obtained during the algorithm.

We define the error as the difference of above probabilities, $PR_{opt} - PR_{exp}$. The residual error can be calculated as:

$$\text{error} < PR_{opt} - PR_{exp}$$

$$= \frac{1}{\sqrt{2\pi}} \int_{E_{opt}}^{E_{exp}} \exp(-\frac{x^2}{2})\, dx. \tag{33}$$

As the algorithm iterates, E_{exp} is decreased and E_{opt} is increased (or remains unchanged). Therefore the error is decreased with each cycle in the algorithm and so we can stop the algorithm when error is sufficiently small.

VII. CONCLUSION AND DISCUSSION

We have presented in this paper an efficient algorithm for solving the stochastic shortest path problems whose optimization function is given by eq.(3). Let us briefly

summarize the primary consequences of this paper. First we introduced the notion of x-weight to make the application of dynamic programming techniques possible.

Next, we introduced the (M,V)-plane and re-state the problem in a graphical context, in which the problem is formulated as the process of searching for a specific point on the (M,V)-plane. Based on these developments, we proposed an efficient algorithm to search for the point on the (M,V)-plane. Finally we evaluated the time complexity of this algorithm. In this process, we clarified the relationship between repetition count of the algorithm and the remaining estimation error.

In general, a packet switched network is modeled as a directed graph. Since Dijkstra's algorithm is valid not only for undirected graphs but also for directed graphs, our algorithm is valid also for directed graphs.

In the worst case, there are a large number of paths, proportional to 3, in $\prod x$. However, this would be a rare case. Thus, generally, we can expect that the algorithm converges fairly rapidly in practical networks. $E_{exp} - E_{opt}$ is a monotonically decreasing function of repetition count of the algorithm and error eq.(33) is also a monotonically decreasing function of repetition count. Therefore, we can stop the algorithm taking into consideration various factors, such as the network size, available processing powers, and so on, while controlling the remaining estimation error caused by the immature termination of the algorithm.

ACKNOWLEDGEMENTS

We would like to thank H. Ito at NTT's Network Traffic Laboratory, Musashino for enlightening discussions and invaluable comments. We are thankful to J. Yamagata, T. Saito and I. Nishikado at our Laboratory, Yokosuka for exhaustive and critical reading of the manuscript. Encouragement of all the members of our Laboratories is appreciated.

APPENDIX I : DETERMINATION OF THE VALUE OF x

Suppose the two x-optimal paths, P_1 and P_2 have been found for $x=x_1$ and x_2, respectively. The x-weight function of each path P can be represented on the (x,W_x)-plane by a straight line.

Path P is x-optimal for $x_i < x < x_j$ if and only if

$$W_x(P) < W_x(P') \text{ for } x_i < x < x_j , \qquad \text{(I-1)}$$

where P' represents any other path. Let us assume that the two lines intersect with each other at $x = x_3$.

If there exists another x-optimal path P_3 within $x_i < x < x_j$, the corresponding x-weight function should have the following property.

$$\begin{aligned} &W_x(P_1) < W_x(P_3) \text{ for } x = x_1 . \\ &W_x(P_2) < W_x(P_3) \text{ for } x = x_2 . \\ &W_x(P_3) < W_x(P_2) \text{ and } W_x(P_3) < W_x(P_1) \\ &\qquad \text{for } x_1 < x_i < x < x_j < x_2 . \end{aligned} \qquad \text{(I-2)}$$

That is, the corresponding line should reside within the shaded region in fig.I-1. It can easily be understood that we

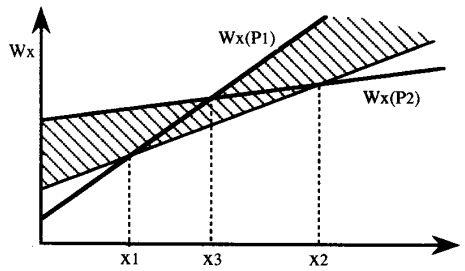

Fig.I-1 Path Representation on (x,Wx)-Plane

can determine the existence of such a path by calculating an x-optimal path with x set to x_3. The value of x_3 is calculated as

$$x_3 = \frac{1}{\text{slope}(P,P')} . \qquad \text{(I-3)}$$

APPENDIX II : DERIVATION OF I(n)

We assign two independent weights on each link in the graph and use a linear combination of weights on each link with same coefficient. Then we can take the path that minimizes the summation of the linear combination of weights with some coefficient through its path between two given nodes on the graph. Then we have the following problem.

Problem : If the coefficient is varied, we assume the number of different paths between two given nodes on the graph are at most I(n). Search for I(n).

First, we consider following a weighted graph [see fig.II-1]. We assume that, there are n independent paths from A to B and m, from B to C. From theorem 2 , there are n+m-1 difference paths from node A to C.

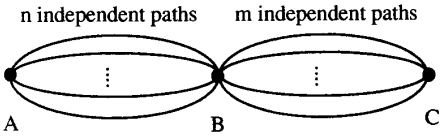

Fig.II-1 Example of Network II

Next we consider the nth order complete directed graph G(n) and name each node 1,2,...,n. We assume I(n) different paths connect two nodes from n to 1 in G(n). If we add a new node n+1 and 2n links and form the (n+1)th order complete directed graph G(n+1), then I(n+1) is represented as follows:

$$I(n+1) = 1 + (2I(n) - 1) + (2I(n-1) - 1) + \cdots + (2I(2) - 1) . \qquad \text{(II-1)}$$

The first term of eq.(II-1) corresponds the direct path from node n+1 to 1. The second term corresponds to the number of paths via node n using the above result. Similarly, the third term corresponds to the number of paths via node n-1 and without node n. Furthermore I(2)=1. Therefore, we can get eq.(29).

REFERENCES

[1] M. Schwartz and T. E. Stern, Routing Techniques Used in Computer Communication Networks., IEEE Trans. Commun. COM-28, '80, 539-552.

[2] See, *e.g.*, D. Bertsekas and R. Gallager, Data Networks., PRENTICE-HALL, INC., Englewood Cliffs, NJ, '87, 322-323.

[3] J. Burgin, Broadband ISDN Resource Management., 6th ITC Seminar, 12.2, '89.

[4] T. Ichimori *et al*, Minimal Spanning Tree with Normal Variates as Weight., Jour. Opns. Res. Vol. 24, No. 1, '81, 61-65.

[5] See, *e.g.*, R. Bellman, Dynamic Programming., Princeton Univ. Press, Princeton NJ, '57.

[6] S. Iwamoto, Recent Progress in Dynamic Programming., Proceedings of the Second RAMP Symposium, Kyoto, '90, 129-140., (in Japanese).