IEIE Transactions on Smart Processing and Computing, vol. 5, no. 6, December 2016
https://doi.org/10.5573/IEIESPC.2016.5.6.445

IEIE Transactions on Smart Processing and Computing

Candidate Path Selection Method for TCP Performance Improvement in Fixed Robust Routing

Yukinobu Fukushima, Takashi Matsumura, Kazutaka Urushibara, and Tokumi Yokohira

The Graduate School of Natural Science and Technology, Okayama University / Okayama 700-8530, Japan
{fukusima, yokohira}@okayama-u.ac.jp

* Corresponding Author: Yukinobu Fukushima

Received October 5, 2016; Revised November 8, 2016; Accepted November 28, 2016; Published December 30, 2016

* *Regular Paper*

* *Extended from a Conference: Preliminary results of this paper were presented at IEEE CQR 2011. This paper has been accepted by the editorial board through the regular review process that confirms the original contribution.*

***Abstract*:** Fixed robust routing is attracting attention as routing that achieves high robustness against changes in traffic patterns without conducting traffic measurement and performing dynamic route changes. Fixed robust routing minimizes the worst-case maximum link load by distributing traffic of every source-destination (*s*-*d*) router pair onto multiple candidate paths (multipath routing). Multipath routing, however, can result in performance degradation of Transmission Control Protocol (TCP) because of frequent out-of-order packet arrivals. In this paper, we first investigate the influence of multipath routing on TCP performance under fixed robust routing with a simulation using ns-2. The simulation results clarify that TCP throughput greatly degrades with multipath routing. We next propose a candidate path selection method to improve TCP throughput while suppressing the worst-case maximum link load to less than the allowed level under fixed robust routing. The method selects a single candidate path for each of a predetermined ratio of *s*-*d* router pairs in order to avoid TCP performance degradation, and it selects multiple candidate paths for each of the other router pairs in order to suppress the worst-case maximum link load. Numerical examples show that, provided the worst-case maximum link load is less than 1.0, our proposed method achieves about six times the TCP throughput as the original fixed robust routing.

***Keywords*:** Fixed robust routing, Multipath routing, TCP throughput, Worst-case maximum link load, Candidate path selection method

1. Introduction

With the spread of bandwidth-consuming applications (e.g., video streaming), traffic patterns in Internet service provider (ISP) networks have become more volatile. One way to accommodate volatile traffic patterns is to use dynamic routing based on traffic measurement. However, this increases operational complexity [1-4] and can lead to route instability [5].

An alternative way is to use fixed robust routing [1-4], which statically assigns multiple candidate paths to every source-destination (*s*-*d*) router pair and determines the fraction of the traffic routed onto each of the candidate paths; in other words, it uses multipath routing. Fixed robust routing determines the fraction based not on a specific traffic pattern but a set of traffic patterns that can actually occur as a result of traffic changes. The feasible region of the traffic patterns is derived by traffic variation models (e.g., the hose traffic model [6] and the pipe traffic model [3]). For such traffic patterns, fixed robust routing tries to minimize the worst-case performance, such as the worst-case maximum link load (i.e., the maximum link load for all feasible traffic patterns).

Although multipath routing, which is adopted by fixed robust routing, is effective at decreasing the worst-case maximum link load, it also degrades throughput of Transmission Control Protocol (TCP). Multipath routing frequently causes out-of-order packet arrivals, which cause duplicated acknowledgments (ACKs) at the source of a TCP connection. If the number of duplicated ACKs reaches three, the source recognizes that the network is congested, and decreases its packet transmission rate (first retransmission and first recovery). Because TCP is used by most Internet services, it is necessary for fixed robust routing to avoid TCP performance degradation.

In this paper, we first investigate the influence of multipath routing in fixed robust routing on TCP performance with a simulation. The simulation results clarify that TCP throughput greatly degrades under multipath routing. Based on the simulation results, we next propose a candidate path selection method to improve TCP throughput while suppressing the worst-case maximum link load to less than the allowed level under fixed robust routing. One way to improve TCP throughput is to adopt single-path routing, which can drastically decrease the occurrence of out-of-order packet arrivals. However, because single-path routing concentrates all traffic of each *s-d* router pair onto a single path, its worst-case maximum link load generally becomes higher than that of multipath routing. In order to optimize the trade-off between TCP throughput and worst-case maximum link load, our proposed method selects a single candidate path for each of a predetermined ratio of *s-d* router pairs to improve TCP throughput, while it sets multiple candidate paths for each of the other *s-d* router pairs to suppress the worst-case maximum link load. Then, we determine the fraction of traffic routed onto each of the candidate paths by solving the fixed robust routing problem.

In Section 2, we introduce related work. In Section 3, we explain traffic variation models, the fixed robust routing problem, and performance degradation of TCP over multiple paths. In Section 4, we investigate the influence of multipath routing on TCP performance under fixed robust routing with the simulation. In Section 5, we propose a candidate path selection method for fixed robust routing. In Section 6, we evaluate our method with the simulation. Section 7 concludes this paper.

2. Related Work

TCP throughput is theoretically analyzed using the fluid model [7]. In the paper, it is assumed that characteristics of the network (e.g., round trip time between source and destination, and packet-loss probability) are known beforehand and are constant. However, in such an analysis, we cannot consider degradation of TCP throughput due to out-of-order packet arrivals, which dynamically occur on a per-packet basis. On the other hand, in this paper, we evaluate TCP throughput with packet-level simulation, which can simulate out-of-order packet arrivals and the behavior of TCP to cope with it.

TCP modifications have been proposed to improve TCP throughput over multipath routing [8, 9]. Under one proposal [8], the authors adjusted the fast retransmission threshold (*dupthresh*), which means the number of duplicated ACKs received by the sender of TCP in order to detect packet loss (set to three, in general). They set the threshold in proportion to the number of paths used in multipath routing by the sender of TCP. They also delayed ACK for an out-of-order packet arrival at the receiver. The other proposal [9] dynamically adjusts *dupthresh* with consideration of both 1) positive impact of increasing *dupthresh:* throughput improvement by avoiding false fast retransmission caused by out-of-order packet arrivals, and 2) negative impact of increasing dupthresh: throughput reduction by increasing the frequency of timeouts caused by packet losses. Although these approaches are effective for improving TCP throughput under a multipath environment, these approaches require all users to apply the modifications to the TCP layer in their terminals. In this paper, we aim at improving TCP throughput under multipath environments by modifying the network layer from the viewpoint of the network operator, who adopts fixed robust routing as the routing scheme in the network.

3. Fixed Robust Routing Problem for Variable Traffic

3.1 Traffic Variation Model

Fixed robust routing tries to accommodate a set of traffic patterns that can happen as a result of traffic changes. Under fixed robust routing, *traffic variation models* specify the set of traffic patterns. Conventional schemes adopt the hose model [6] and the pipe model [3] as traffic variation models, and we also adopt them in this paper.

The hose model imposes an upper bound on incoming (outgoing) traffic volume from (to) the outside network at every router. When we denote a set of routers as N, the upper bound on incoming traffic volume at router i as R_i, the upper bound on outgoing traffic volume at router j as C_j, and traffic volume from router i to router j as t_{ij}, the hose model assumes that any traffic pattern that satisfies the following inequalities can happen as a result of traffic changes:

$$\sum_{j \in N} t_{ij} \le R_i \quad \forall i \in N \tag{1}$$

$$\sum_{i \in N} t_{ij} \le C_j \quad \forall j \in N \tag{2}$$

The authors used the bandwidth of router i's input link from the outside network as R_i, and that of router j's output link to the outside network as C_j [6].

The pipe model imposes an upper bound on traffic volume between every *s-d* router pair. When we denote the upper bound on traffic volume from router i to router j as ω_{ij}, the pipe model assumes that any traffic pattern that satisfies the following inequality can happen as a result of traffic changes:

$$t_{ij} \le \omega_{ij} \quad \forall i, j \in N \tag{3}$$

The authors state that ω_{ij} can be determined based on traffic history or service level agreement (SLA) between the ISP and users [3].

3.2 Fixed Robust Routing Problem

Fixed robust routing statically determines paths and the fraction of the traffic routed onto each candidate path for each *s-d* router pair so that the best worst-case performance is achieved for those traffic patterns that follow the assumed traffic variation models. It adopts *multipath routing* where the traffic of every *s-d* router pair is routed onto multiple paths.

As a special case of the general fixed robust routing problem, the problems where cost function minimizes the worst-case maximum link load were formulated [1, 3]. In the first study [1], the authors showed the link-flow-based formulation where fractions of traffic are determined on a per-link basis, while in the latter study [3], the authors showed the path-flow-based formulation where the fractions are determined on a per-path basis. Because the path-flow-based formulation allows us to select the candidate paths based on quality of service (e.g., delay and bandwidth), we adopt the path-flow-based formulation in this paper.

The fixed robust routing problem that is formulated on a per-path basis is a *linear semi-infinite programming (LSIP) problem* [3], which has an infinite number of constraints. In the problem, given the set (D) of all traffic matrices that follow the hose and pipe traffic models, the set (P) of candidate paths for all *s-d* router pairs, the set (P_{ij}) of candidate paths for *s-d* router pair (i, j), the set (Π_l) of candidate paths traversing link l, the set (L) of all links in the network, and the capacity ($c(l)$) of link l, we derive the fraction (x_p) of traffic of the corresponding *s-d* router pair (i, j) routed onto path p so that the worst-case maximum link load (t) is minimized. The formulation of the LSIP problem is as follows:

$$\min t$$
subject to
$$\sum_{p \in P_{ij}} x_p = 1 \quad \forall i\ j \in N \tag{4}$$
$$\sum_{i,j \in N} \sum_{p \in (P_{ij} \cap \Pi_l)} \frac{x_p t_{ij}}{c(l)} \le t \quad \forall l \in L, \forall T \in D \tag{5}$$

Constraint (4) confirms that valid routing is performed for every *s-d* router pair. The left-hand side of constraint (5) is the load of link l for traffic matrix T where the load of a link is defined as the sum of traffic volumes that pass along the link, divided by the capacity of the link. Constraint (5) guarantees that every link load is less than or equal to t for every traffic matrix. Because an infinite number of traffic matrices (T) follow the hose and pipe models, the problem is an LSIP problem.

As one way to solve the above-mentioned LSIP problem, the exchange method [11] is used [3]. The method solves the LSIP problem using the following steps.

• Step 1 We simplify the original LSIP problem to a linear programing (LP) problem by replacing D with D', which includes only a limited number of traffic matrices in D.

• Step 2 We solve the simplified LP problem and obtain worst-case maximum link load t and routing x_p for the current D'.

• Step 3 In order to check whether routing x_p obtained in Step 2 satisfies constraint (5) for any feasible traffic matrix, we solve the following subproblem regarding x_p obtained in Step 2 as a constant and t_{ij} as a variable, and obtain the worst-case maximum link load and the worst case traffic matrix T_l' for every link l.

$$\max \sum_{i,j \in N} \sum_{p \in (P_{ij} \cap \Pi_l)} \frac{x_p t_{ij}}{c(l)}$$
subject to
$$\sum_{j \in N} t_{ij} \le R_i \quad \forall i \in N \tag{6}$$
$$\sum_{i \in N} t_{ij} \le C_j \quad \forall j \in N \tag{7}$$
$$t_{ij} \le \omega_{ij} \quad \forall i, j \in N \tag{8}$$

If the worst-case maximum link load obtained with the subproblem is less than or equal to t obtained in Step 2 for every link, we finish the exchange method. Otherwise, we add traffic matrix T_l' to D' and go back to Step 2.

In order to solve the LSIP problem in one step instead of iterative steps in the exchange method, the authors showed the polynomial size single LP problem [3]. In that problem, the infinite number of constraints in (5) is replaced with duals of the subproblems in Step 3 of the exchange method. The formulation of the single LP problem is as follows.

$$\min t$$
subject to
$$\sum_{p \in P_{ij}} x_p = 1 \quad \forall i\ j \in N \tag{9}$$
$$\sum_{i \in N} R_i r_i(l) + \sum_{j \in N} C_j y_j(l) + \sum_{i,j \in N} \omega_{ij} q_{ij}(l) \le t \quad \forall l \in L \tag{10}$$
$$\sum_{p \in (P_{ij} \cap \Pi_l)} x_p - c(l)\left(r_i(l) + y_i(l) + q_{ij}(l)\right) \le 0 \ \forall l \in L\ \forall i\ j \in N \tag{11}$$
$$r_i(l) \ge 0 \quad \forall i \in N\ \forall l \in L \tag{12}$$
$$y_j(l) \ge 0 \quad \forall j \in N\ \forall l \in L \tag{13}$$
$$q_{ij}(l) \ge 0 \quad \forall i\ j \in N\ \forall l \in L \tag{14}$$

where $r_i(l)$, $y_j(l)$, and $q_{ij}(l)$ are dual variables that correspond to constraints (6), (7), and (8), respectively. The left-hand side of (10) is an objective function of duals of the subproblem in Step 3 of the exchange method. Due to the strong duality of LP, the optimal value of the subproblem is equal to that of the dual of the subproblem. Thus, satisfying constraint (10) means that routing x_p obtained with the above single LP minimizes the worst-case maximum link load.

Given the parameter values (R_i, C_j, and ω_{ij}) in the hose and pipe traffic models, and a set (P) of candidate paths for all source-destination router pairs, we can derive the minimum of worst-case maximum link load t and optimal routing x_p by solving the single LP in one step.

3.3 Performance Degradation of TCP over Multiple Paths

Although fixed robust routing can minimize the worst-case maximum link load for traffic patterns that follow the assumed traffic variation models [1, 3], they may result in performance degradation of TCP because of frequent out-of-order packet arrivals caused by multipath routing.

TCP achieves reliable communications between source-destination hosts with acknowledgment. When a destination host of TCP receives a packet, it sends an ACK to the source. As a result, all in-order packet arrivals are acknowledged. The source detects packet loss by either timer expiration or duplicated ACKs. When it receives three duplicated ACKs, it retransmits the lost packet (fast retransmission) and reduces the congestion window (fast recovery), which leads to a decrease in transmission rate [8].

Figs. 1 and 2 show examples of out-of-order packet arrivals and reception of three duplicated ACKs over multiple paths, respectively. In these examples, packet 1 is routed onto a longer path, while packets 2, 3, and 4 are routed onto a shorter path. As a result, the latter packets arrive at the destination earlier than the first packet, and consequently the source receives three duplicated ACKs.

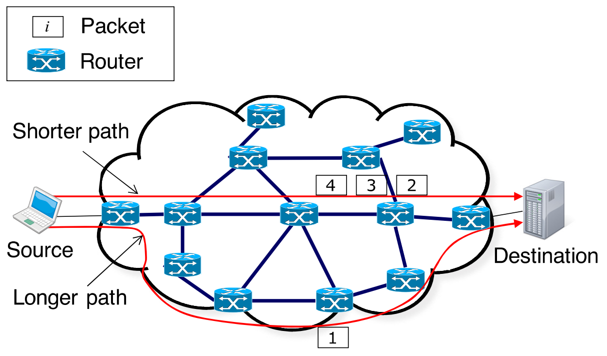


Fig. 1. Out-of-order packet arrivals over multiple paths.

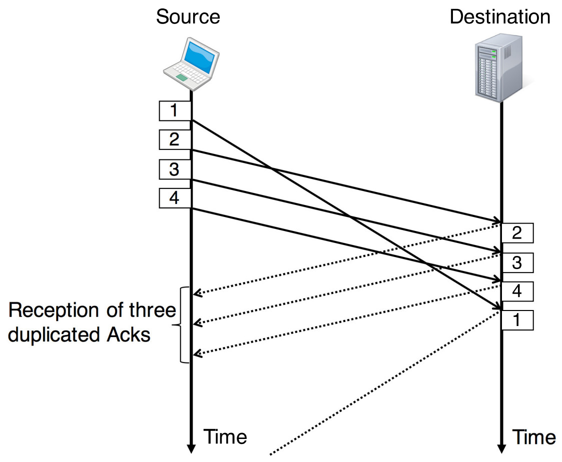


Fig. 2. Reception of three duplicated ACKs.

4. Evaluation of TCP Performance over Multiple Paths

The frequency of fast recovery in TCP over multiple paths will depend on delay differences among multiple paths and the ratio of shorter paths among the multiple paths, because they affect the frequency of out-of-order packet arrivals. Thus, we investigated by simulation the effect on TCP throughput of the delay differences and that ratio.

4.1 Simulation Model

We use the ns-2 network simulator [11] for our evaluation. Fig. 3 shows the network topology in our simulation. Access links between end hosts and the router have a bandwidth of 50 Mbps, while paths between routers have a bandwidth of 100 Mbps. The propagation delay of access links is set to 0.2 ms, while those of the paths range from 2.0 ms to 2.0 + d ms, where parameter d determines the delay difference among multiple paths. The delay on a path only consists of a propagation delay under the assumption that a path corresponds to a lightpath of wavelength-routed networks. A lightpath does not have any queueing delay at intermediate nodes due to circuit switching.

We use four or ten paths between router R_1 and router R_2. At R_1, packets belonging to a single TCP connection are randomly and uniformly distributed onto one of the multiple paths; that is, the fraction of the traffic routed onto a path is 1/4 for four paths and 1/10 for 10 paths. In order to investigate the effect of delay differences among multiple paths on TCP throughput, we adopted three kinds of path: a long path (L), a middle path (M), and a short path (S). Delays for L, M, and S are set to 2.0 + d ms, 2.0 + d/2 ms and 2.0 ms, respectively.

Each end-host pair (S_i, D_i) ($1 \leq i \leq 5$) sets up one TCP (NewReno) connection. Each source's data transmission rate follows a Poisson distribution with a rate 20 Mbps. The segment size is set to 1 KB. Please note that the average intersegment time is 0.4 ms (= 1 s / 2,500 segments) because a source transmits 2,500 (= 20 Mbps / 1 KB) segments per second. Routers adopt a drop-tail queueing policy, and their queue size is set to 1000 segments.

As a performance metric, we use total throughput (that is, the sum of all TCP connections' throughputs). TCP throughput means an average of the received TCP segments per second, except for duplicated ones at a destination.

4.2 Results

Fig. 4 depicts total throughput as a function of the delay difference (d) when four paths are used between routers. We use four path combinations: SLLL, SSLL, SSSL, and SSSM.

All the combinations achieve total throughput of almost 100 Mbps when d is less than or equal to 1.2 ms for

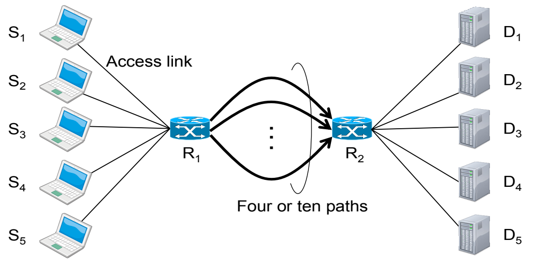


Fig. 3. Network model.

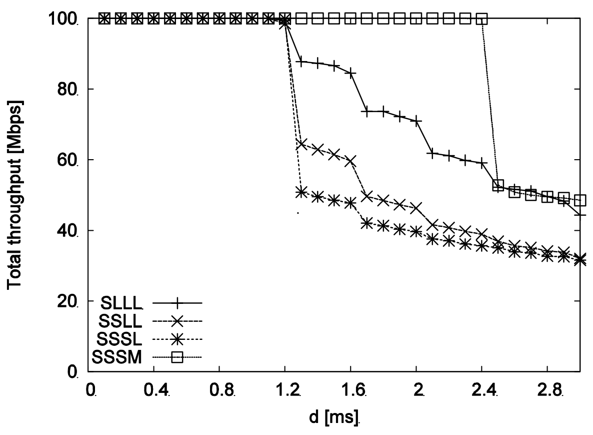


Fig. 4. Total throughput (four paths).

SLLL, SSLL, and SSSL, and 2.4 ms for SSSM. This is because delay differences between a long path (L) and a short path (S) or between a middle path (M) and a short path (S) (i.e., d = 1.2 ms for SLLL, SSLL, and SSSL, $d/2$ = 1.2 ms for SSSM) are less than three times the average intersegment time (0.4 ms) at the source, and consequently, the number of packets that may overtake the preceding packet is less than three, which does not cause fast recovery at the source, as shown in Fig. 5. The number of candidates for overtaking packets is equal to the number of packets where arrival time at the destination can be less than the arrival time of the focused preceding packet. Out-of-order packet arrival can happen when the preceding packet is routed onto a longer path, while the succeeding packet is routed onto a shorter path. In such a situation, the arrival times of the preceding packet and the nth succeeding packet are calculated as $t_0 + d_L$ and $t_0 + n \times i + d_S$, respectively (t_0: transmission time of the preceding packet at the source, d_L: propagation delay of a longer path [L or M], d_S: propagation delay of a shorter path [S], and i: an average of intersegment time). If $t_0 + d_L > t_0 + n \times i + d_S$ is satisfied, the nth succeeding packet overtakes the preceding packet. Thus, the number of candidates for overtaking packets is calculated as $(d_L - d_S)/i$.

When d exceeds three times the average intersegment time (1.2 ms for SLLL, SSLL, and SSSL, and 2.4 ms for SSSM), the total throughput is degraded because the number of the candidates for overtaking packets reaches three, and fast recovery at the source can happen when the first packet selects a longer path and the three succeeding packets select a shorter path (Fig. 6).

As d further increases by 0.4 ms, total throughput is

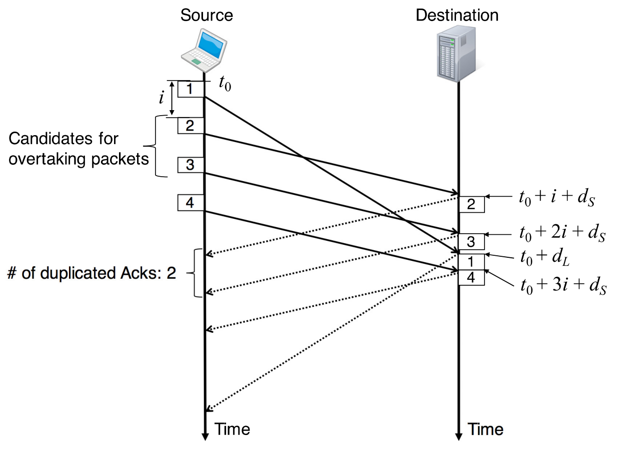


Fig. 5. Out-of-order packet arrivals when delay difference is less than 1.2 ms.

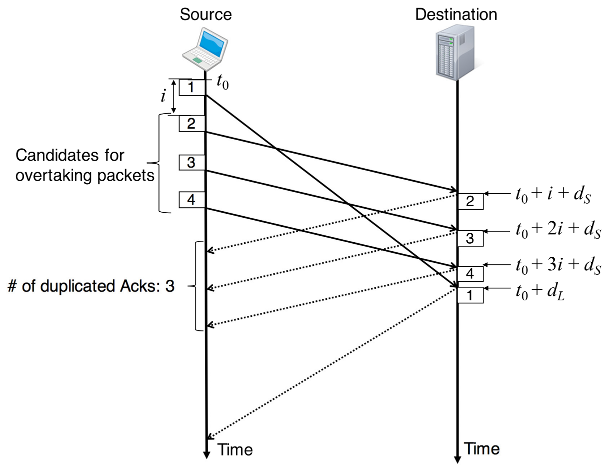


Fig. 6. Out-of-order packet arrivals when delay difference is between 1.2 ms and 1.6 ms.

largely degraded in each combination. This is because the number of candidates for overtaking packets increases by one, and the probability of fast recovery increases. Even if the increase is less than 0.4 ms, total throughput is slightly degraded. This is due to transmission of segments where intersegment time is less than the average.

We next investigate the impact on total throughput of the ratio of shorter paths among multiple paths. As the ratio of S in the combination becomes lower, total throughput becomes higher (SLLL: 87 Mbps, SSLL: 64 Mbps, and SSSL: 50 Mbps with d = 1.3 ms). A lower ratio of S leads to a lower probability of incurring three out-of-order packet arrivals. The probabilities are about 0.012 = $0.75\times(1-0.75)^3$ for SLLL, 0.063 = $0.5\times(1-0.5)^3$ for SSLL and 0.11 = $0.25\times(1-0.25)^3$ for SSSL. As the probability of three out-of-order packet arrivals becomes lower, the frequency of fast recovery at the source decreases. In our simulation, the frequencies of fast recovery at the source are 403, 710, and 1042 under SLLL, SSLL, and SSSL, respectively.

Fig. 7 depicts the total throughput when 10 paths are used between routers. We use three path combinations: SLLLLLLLLL, SSSSSLLLLL, and SSSSSSSSSL. Similar

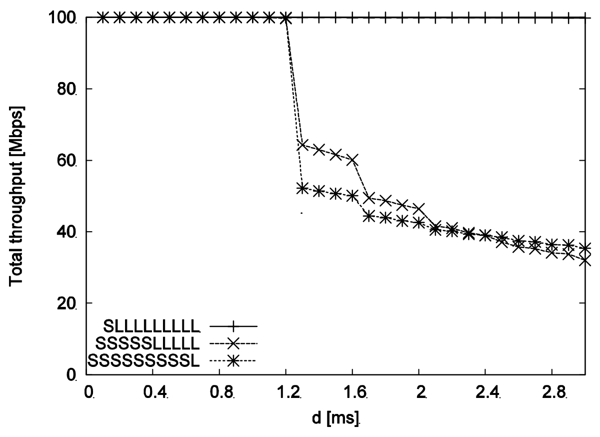


Fig. 7. Total throughput (10 paths).

to Fig. 4, the combination with a lower ratio for S achieves higher total throughput.

5. Candidate Path Selection Method for Improving TCP Throughput

As shown in the previous section, TCP throughput greatly degrades under multipath routing in fixed robust routing. One way to improve TCP throughput is to adopt single-path routing, which can drastically decrease the occurrence of out-of-order packet arrivals. However, because single-path routing concentrates all traffic for each *s-d* router pair onto a single path, its worst-case maximum link load generally becomes higher than with multipath routing.

In order to improve TCP throughput with fixed robust routing while suppressing the worst-case maximum link load to less than the allowed level, we propose a candidate path selection method for fixed robust routing. Our proposed method performs single-path routing for each of a predetermined ratio of *s-d* router pairs in order to improve TCP throughput, while it provides multipath routing for each of the other pairs in order to suppress the worst-case maximum link load.

The procedure of our proposed candidate path selection method is as follows.

- Step 1 We select the K shortest paths as candidate paths for every *s-d* router pair.
- Step 2 For every *s-d* router pair, we select the (longest) path with the maximum propagation delay among the K paths, and we determine the top X percent of *s-d* router pairs that have longer longest paths; then, we consider the single shortest path of each of the top X percent of the *s-d* pairs as the candidate path.

In Step 1, we select K shortest paths as candidate paths for every *s-d* router pair. As to the criteria for selecting the K shortest paths, we use two: hop count on the path, and propagation delay on the path (i.e., the sum of propagation delays for the links along the path). Hereafter, we call the proposed method using the former criterion *HYBRID-HOP*, and we call the proposed method using the latter criterion *HYBRID-DELAY*.

In Step 2, as the target of single-path routing, we give high priority to selecting *s-d* router pairs where maximum propagation delay among its K candidate paths is long. This is because a longer maximum propagation delay results in a need for more time to recover the congestion window size of the TCP connection after fast recovery is caused by out-of-order packet arrivals, which results in more degradation of TCP throughput.

After obtaining the candidate paths for every *s-d* router pair with our proposed method, we solve the fixed robust routing problem in Section 3, determine the fraction (x_p) of traffic routed onto every candidate path, and obtain the worst-case maximum link load.

We show an example of candidate path selection with HYBRID-HOP, one of our proposed methods, on the 14-node NSFNET (Fig. 8) in Figs. 9 and 10. We assume that five *s-d* router pairs—pair 1 (R_1, R_2), pair 2 (R_1, R_3), pair 3 (R_1, R_4), pair 4 (R_1, R_5), and pair 5 (R_1, R_6)—try to transmit their traffic, and K and X are set to 2 and 60, respectively.

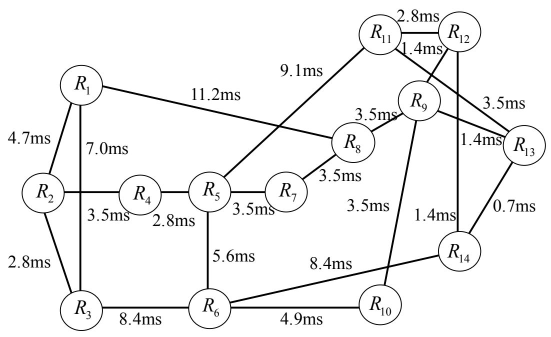


Fig. 8. 14-node NSFNET.

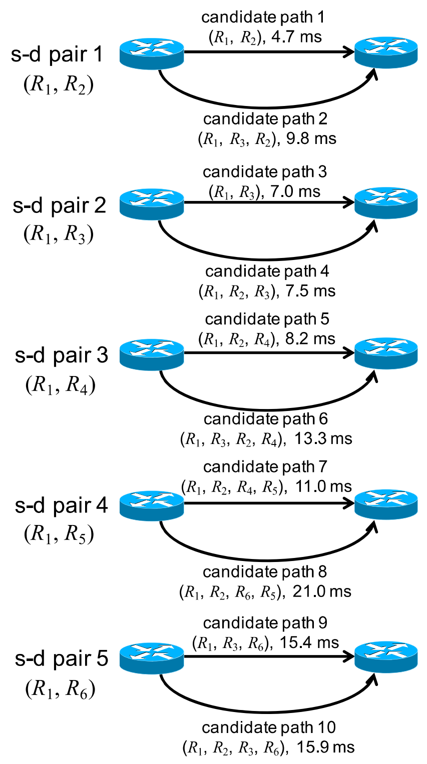
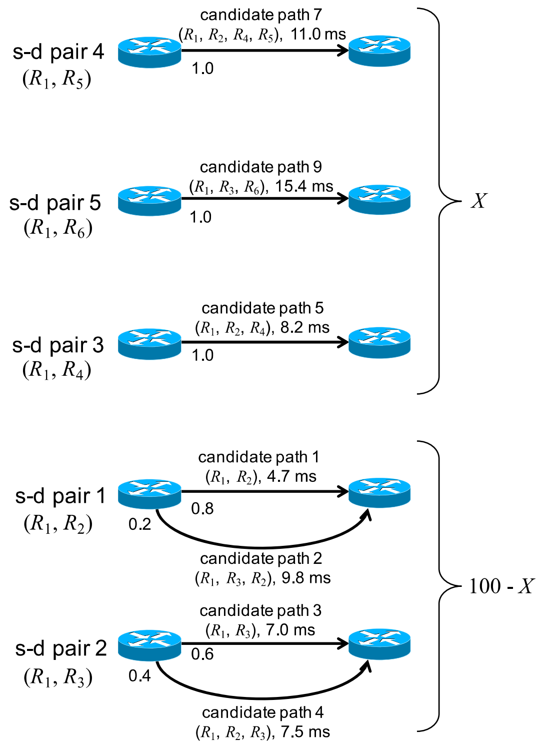


Fig. 9. Example of Step 1 in the proposed method.



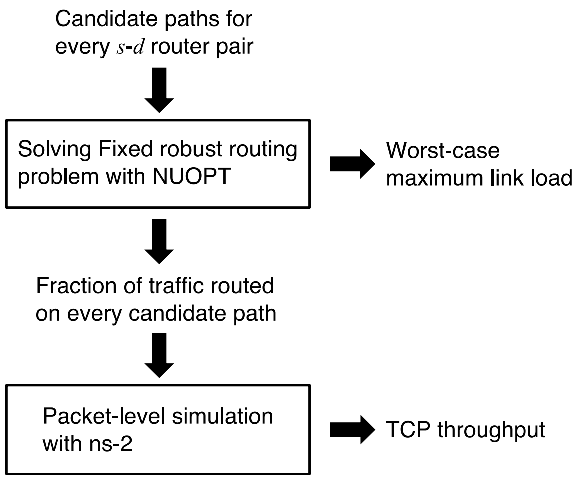
s-d pair 4 (R_1, R_5)
candidate path 7 (R_1, R_2, R_4, R_5), 11.0 ms
1.0
s-d pair 5 (R_1, R_6)
candidate path 9 (R_1, R_3, R_6), 15.4 ms
1.0
s-d pair 3 (R_1, R_4)
candidate path 5 (R_1, R_2, R_4), 8.2 ms
1.0
X
s-d pair 1 (R_1, R_2)
candidate path 1 (R_1, R_2), 4.7 ms
0.8
0.2
candidate path 2 (R_1, R_3, R_2), 9.8 ms
s-d pair 2 (R_1, R_3)
candidate path 3 (R_1, R_3), 7.0 ms
0.6
0.4
candidate path 4 (R_1, R_2, R_3), 7.5 ms
100 - X

Fig. 10. Example of Step 2 in the proposed method.

In Step 1, we select the shortest and the second shortest hop-count paths as candidate paths for every *s-d* router pair, as shown in Fig. 9. In Step 2, we first calculate the propagation delay of the longest candidate path for every *s-d* router pair. In Fig. 9, the delays for pairs 1, 2, 3, 4, and 5 are 9.8 ms, 7.5 ms, 13.3 ms, 21.0 ms, and 15.9 ms, respectively. We then sort the pairs in descending order of delay, select the top 60 percent of the pairs as the target of single-path routing, and reduce their candidate paths to the single shortest hop-count paths. In Fig. 10, the candidate paths of pairs 3, 4, and 5 are reduced to their shortest hop-count paths. After determining the candidate paths of each router pair, we solve the fixed robust routing problem in Section 3, and obtain the fraction of traffic routed onto every candidate path. In this case, we assume that the fractions are determined as shown in Fig. 10. For example, in *s-d* pair 1, the fractions of traffic routed onto candidate paths 1 and 2 are set to 0.8 and 0.2, respectively.

6. Evaluation

We evaluate the fixed robust routing that adopts our proposed candidate path selection methods (HYBRID-HOP and HYBRID-DELAY) in terms of worst-case maximum link load and TCP throughput. As a comparison, we use a fixed robust routing scheme that selects all the paths as the candidate paths for every *s-d* router pair (ALL). In HYBRID-HOP and HYBRID-DELAY, we set the number of candidate paths (K) for every *s-d* router pair to 2, and the percent of the number of *s-d* router pairs (X) that perform single-path routing to either 0, 20, 40, 60, 80, or 100.



Candidate paths for every s-d router pair
Solving Fixed robust routing problem with NUOPT
Worst-case maximum link load
Fraction of traffic routed on every candidate path
Packet-level simulation with ns-2
TCP throughput

Fig. 11. Flow of evaluation.

Table 1. Parameter settings.

Parameter	Value
Network model	14 node NSFNET
Link bandwidth	500 Mbps
Upper bound (R_i) on incoming traffic volume at every router	130 Mbps
Upper bound (C_j) on outgoing traffic volume at every router	130 Mbps
Upper bound (ω_{ij}) on traffic volume between an *s-d* router pair	50 Mbps

Fig. 11 depicts the flow of evaluation. We first derive the fraction of traffic routed onto every candidate path for HYBRID-HOP, HYBRID-DELAY, and ALL, and obtain the worst-case maximum link load among them by solving the fixed robust routing problem where the parameters are set as shown in Table 1. As a network model, we use the 14-node NSFNET in Fig. 8. We use NUOPT solver [12] to solve the problem. Then, given the fraction of traffic routed onto every candidate path, we evaluate the TCP performance of HYBRID-HOP, HYBRID-DELAY, and ALL with the simulation using ns-2. In the simulation, we establish one TCP connection for every *s-d* router pair. We assume that an application on each source sends packets of 1 KB, and the packet inter-arrival time follows an exponential distribution with a mean of 0.8 ms, which results in every TCP connection's data transmission rate at the maximum of 10 Mbps.

Figs. 12 and 13 depict the worst-case maximum link load for ALL, HYBRID-HOP, and HYBRID-DELAY. ALL shows the lowest load of 0.455. In Fig. 12, HYBRID-HOP achieves loads of 0.68, 0.82, 0.98, 1.08, 1.08, and 1.08 when X is set to 0, 20, 40, 60, 80, and 100, respectively. In Fig. 13, HYBRID-DELAY achieves loads of 0.88, 1.04, 1.24, 1.50, 1.56, and 1.56 when X is set to 0, 20, 40, 60, 80, and 100, respectively.

The loads for HYBRID-HOP and HYBRID-DELAY become higher as ratio X of *s-d* router pairs performing single-path routing increases because more *s-d* router pairs performing single-path routing results in concentrating

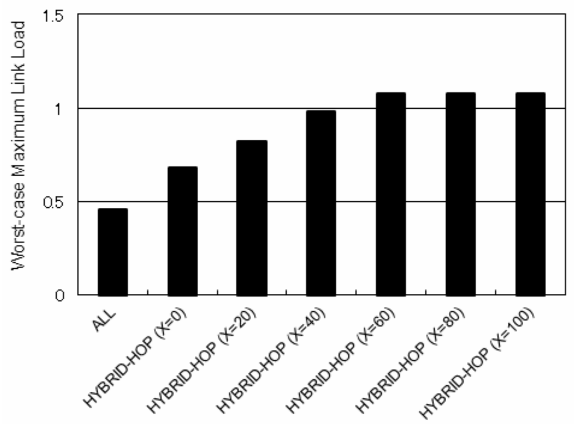


Fig. 12. Worst-case maximum link load (HYBRID-HOP and ALL).

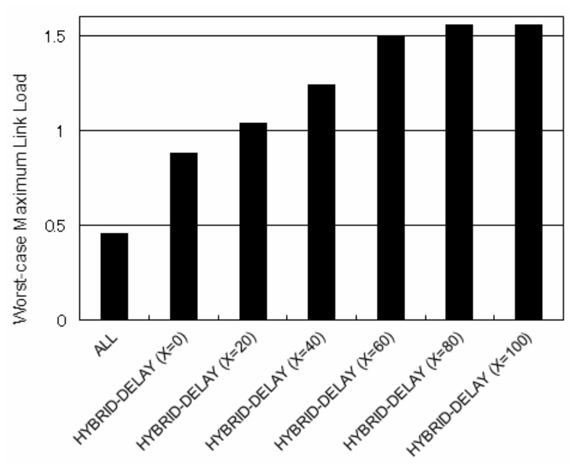


Fig. 13. Worst-case maximum link load (HYBRID-DELAY and ALL).

more traffic onto a single path.

HYBRID-DELAY shows higher worst-case maximum link load than HYBRID-HOP for each value of *X* because it adopts propagation delay as a criterion for *K* shortest paths. The shortest propagation delay path tends to have longer hop counts than the shortest hop count path, and consequently, the shortest propagation delay burdens more links along the path.

In HYBRID-HOP, the load is higher than 1.0 when *X* is higher than or equal to 60, and consequently, we suffer from bandwidth shortage at the bottleneck link when the worst-case traffic pattern happens. We can avoid the bandwidth shortage by setting *X* lower than or equal to 40. On the other hand, in HYBRID-DELAY, we can avoid the bandwidth shortage by setting *X* lower than or equal to 0.

Figs. 14 and 15 show the average TCP throughput per connection for ALL, HYBRID-HOP, and HYBRID-DELAY. ALL shows the lowest throughput of 1.44 Mbps. In Fig. 14, HYBRID-HOP shows a throughput of 4.65 Mbps, 6.89 Mbps, 8.51 Mbps, 9.28 Mbps, 10.0 Mbps, and 10.0 Mbps when *X* is set to 0, 20, 40, 60, 80, and 100, respectively. In Fig. 15, HYBRID-DELAY shows throughput of 8.28 Mbps, 8.32 Mbps, 8.58 Mbps, 9.03 Mbps, 9.84 Mbps, and 10.0 Mbps when *X* is set to 0, 20,

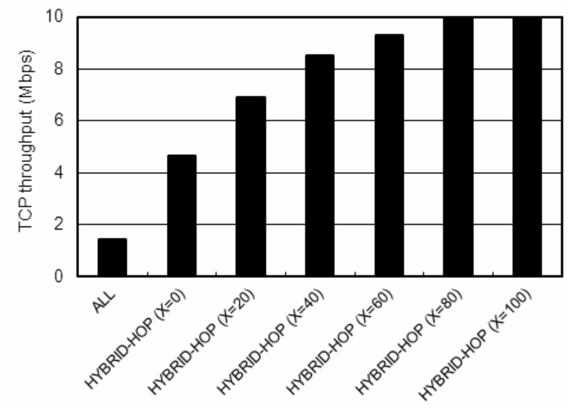


Fig. 14. Average TCP throughput per connection (HYBRID-HOP and ALL).

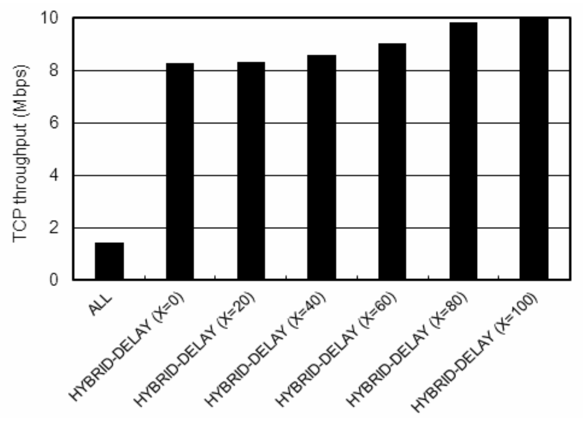


Fig. 15. Average TCP throughput per connection (HYBRID-DELAY and ALL).

40, 60, 80, and 100, respectively.

The throughputs of HYBRID-HOP and HYBRID-DELAY become higher as ratio *X* of *s*-*d* router pairs performing single-path routing increases because more *s*-*d* pairs performing single-path routing decreases the occurrence of out-of-order packet arrivals. Note that TCP throughput is limited by the data transmission rate (i.e., 10 Mbps) of the source host. Thus, HYBRID-HOP shows the same TCP throughput of 10 Mbps for *X*=80 and *X*=100.

HYBRID-DELAY shows higher TCP throughput than HYBRID-HOP for each value of *X* because it adopts propagation delay as a criterion for *K* shortest paths. The shorter the propagation delay of a path, the shorter the time a TCP connection needs in order to recover the congestion window size after first recovery is caused by out-of-order packet arrivals.

In HYBRID-HOP, we can realize higher throughput than ALL for any value of *X*, and the improvement compared to ALL is 322%, 478%, 591%, 644%, 695%, and 695% when *X* is set to 0, 20, 40, 60, 80, and 100, respectively. In HYBRID-DELAY, we can also realize higher throughput than ALL for any value of *X*, and the improvement compared to ALL is 575%, 578%, 596%, 627%, 683%, and 695% when *X* is set to 0, 20, 40, 60, 80, and 100, respectively.

In order to maximize TCP throughput under the condition that we suppress the worst-case maximum link load to less than 1.0, we should set X to as high a value as possible while satisfying the condition. In this evaluation, the throughputs of HYBRID-HOP and HYBRID-DELAY are maximized while keeping the worst-case maximum link load lower than 1.0 when X is set to 40 and 0, respectively. In this case, HYBRID-HOP and HYBRID-DELAY achieve throughputs of 8.51 Mbps and 8.28 Mbps, respectively. Therefore, in terms of TCP throughput, provided we suppress the worst-case maximum link load to less than 1.0, HYBRID-HOP shows better performance than HYBRID-DELAY in this evaluation.

7. Conclusions

In this paper, we first investigated TCP performance under fixed robust routing, and clarified that TCP throughput is greatly degraded due to out-of-order packet arrivals in the multipath routing environment of fixed robust routing. We then proposed a candidate path selection method to improve TCP performance while maintaining robustness against traffic changes. Evaluation results showed that fixed robust routing that adopts our proposed method can achieve about six times the TCP throughput as the original fixed robust routing provided that the worst-case maximum link load is less than 1.0.

References

[1] M. Kodialam, T. V. Lakshman, and S. Sengupta, "Maximum throughput routing of traffic in the hose model," in Proc. of INFOCOM, pp. 1-11, Apr. 2006. Article (CrossRef Link)

[2] M. Kodialam, T. V. Lakshman, J. B. Orlin, and S. Sengupta, "A versatile scheme for routing highly variable traffic in service overlays and IP backbones," in Proc. of INFOCOM, pp. 1-12, Apr. 2006. Article (CrossRef Link)

[3] V. Tabatabaee, A. Kashyap, B. Bhattacharjee, R. J. La, and M. A. Shayman, " Robust routing with unknown traffic matrices," in Proc. of INFOCOM, pp. 2436-2440, May 2007. Article (CrossRef Link)

[4] M. Kodialam, T. V. Lakshman, J. B. Orlin, and S. Sengupta, " Oblivious routing of highly variable traffic in service overlays and IP backbones," IEEE/ACM Transactions on Networking, vol. 17, iss. 2, Apr. 2009. Article (CrossRef Link)

[5] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the tightrope: Responsive yet stable traffic engineering," in Proc. of ACM SIGCOMM, pp. 253-264, Aug. 2005. Article (CrossRef Link)

[6] J. A. Fingerhut, S. Suri, and J. S. Turner, "Designing least-cost nonblocking broadband networks," Journal of Algorithms, vol. 24, no. 2, pp. 287-309, 1997. Article (CrossRef Link)

[7] M. Pagano and R. Secchi, "A survey on TCP performance evaluation and modeling," in Proc. of HET-NET, July 2004. Article (CrossRef Link)

[8] Y. Lee, I. Park, and Y. Choi, "Improving TCP performance in multipath packet forwarding networks," Journal of Communications and Networks, vol. 4. no. 2, pp. 148-157. 2002. Article (CrossRef Link)

[9] M. Zhang, B. Karp, S. Floyd, and L. Peterson, "RR-TCP: A reordering-robust TCP with DSACK,", in Proc. of ICNP, pp. 95-106, Nov. 2003. Article (CrossRef Link)

[10] R. Hettich and K. O. Kortanek, "Semi-infinite programming: Theory, methods, and applications," SIAM Review, vol. 35, pp. 380-429, Sept. 1993. Article (CrossRef Link)

[11] "Network simulator - ns (version 2)". Article (CrossRef Link)

[12] "NUOPT". Article (CrossRef Link)

Yukinobu Fukushima received his B.E., M.E. and Ph.D. degrees from Osaka University, Japan, in 2001, 2003 and 2006. He is currently an assistant professor of the Graduate School of Natural Science and Technology, Okayama University. His research interest includes optical networking, P2P live streaming and cloud computing. He is a member of IEEE, ACM, OSA, and IEICE.

Takashi Matsumura received his B.E. and M.E. degrees from Okayama University, Japan, in 2009 and 2011.

Kazutaka Urushibara received his B.E. and M.E. degrees from Okayama University, Japan, in 2010 and 2012.

Tokumi Yokohira received the B.E., M.E. and Ph.D. degrees in information and computer sciences from Osaka University, Osaka, Japan, in 1984, 1986 and 1989, respectively. From April 1989 to May 1990, he was an assistant professor in the Department of Information Technology, Faculty of Engineering, Okayama University, Okayama, Japan. From May 1990 to December 1994 and from December 1994 to March 2000, he was a lecturer and an associate professor, respectively, in the same department. From April 2000 to June 2003, he was an associate professor in the Department of Communication Network Engineering of the same faculty. Since July 2003, he has been a professor in the same department. His present research interests include performance evaluation and improvement of computer networks and communication protocols, design algorithm of optical networks and network securities. He is a member of the IEEE Communication Society, the Institute of Electronics, Information and Communication Engineers and Information Processing Society of Japan.