Regular Paper

Reactive Load Balancing During Failure State in IP Fast Reroute Schemes

Kazuki Imura[1] Takuya Yoshihiro[2,a)]

Received: December 5, 2013, Accepted: March 7, 2014

Abstract: To improve reliability of IP networks against link/node failure, several IP fast reroute schemes have been proposed so far. They proactively compute backup paths and activate them when failure occurs to prevent packets from losing at the failure link/node. However, it is known that network performance considerably degrades in the failure state of IP fast reroute schemes, because congestion hot spots often appear near the failure link/node. In this paper, we propose a reactive load balancing method that can be applied to the major IP fast reroute schemes that covers single failure. Our scheme works when an IP fast reroute scheme is activating its backup paths, and reduces the degradation of network performance due to failure. In our load balancing scheme, with the overhead of a few bit field on packet header, we can largely reduce the performance degradation in the failure state and mostly keep the throughput as it was in the normal state where no failure exists.

Keywords: load balancing, traffic control/analysis, routing protocols, IP Fast Reroute

1. Introduction

The Internet has grown as an essential social infrastructure in the world, for which high-level reliability is required. Even a short-time disruption of the network may reflect on significant cost because various indispensable communications depend on this high-speed network. However, when a link or a node failure occurs in an IP network, it is difficult to avoid service disruption for a certain time as long as we deploy a traditional routing protocol such as OSPF [1] and IS-IS [2]. Unfortunately, the frequency of failure is not low enough to be negligible, as Ref. [3] reported.

To augment reliability of IP networks, several IP Fast Reroute (IPFRR) techniques have been proposed [4], [5], [6]. They proactively compute backup paths and activate them when failure occurs to prevent packet loss at the failed links or nodes. Typically, they cover every single link or node failure, so that path disruptions in an IP network can be eliminated in every single link or node failure scenario. These IPFRR schemes complement the network performance during the failure state, i.e., during the time period until the failure is recovered.

However, with IPFRR schemes, it is known that using backup paths in face of failure brings congestion on links around failure that degrades the performance of the network [7]. Even if the network resources may be insufficient in failure state, it is strongly desired to reduce the degradation level of network performance.

In this paper, we propose a reactive load balancing method that works in the failure state of IPFRR schemes. Namely, the proposed method tries to reduce the degradation of network performance in the time period in which IPFRR is activating its backup paths. When a congestion occurs around the failure, our method utilizes unused backup paths of the employed IPFRR scheme to make a part of traffic escape from the congestion. With this two-level rerouting, we make use of unused resources of the network around failure to reduce degradation of network performance. Through traffic simulation, we show that the network throughput in case of failure is significantly improved to be a comparable level to the normal state where failure is not present. Note that this proposal is the first load-balancing method that works over full-coverage IPFRR schemes, which covers every single link or node failure, such as NotVia and FIFR.

This paper is organized as follows. In Section 2, we describe several major IPFRR schemes that cover single failure, and present several existing approaches for load balancing over IP networks as well as over IPFRR schemes. In Section 3, we describe the proposed method in detail, and give the results of our traffic simulation in Section 4. Finally, we conclude the work in Section 5.

2. Load Balancing in IP Fast Reroute Schemes

2.1 IP Fast Rerouting and Their Modeling

To augment reliability of IP networks against failure, many IP fast reroute (IPFRR) schemes have been proposed. We first describe the literature of IPFRR schemes as an underlying technology of the proposed method.

IPFRR is a scheme that proactively computes backup paths to prevent packet loss due to failure. In IPFRR, if a router detects the failure of the next-hop link (or node), the backup paths are immediately activated to forwards packets, instead of forwarding them to the failed components. In the early days of IPFRR studies, simple methods such as LFA (Loop-Free Alternate) were

1 Graduate School of Systems Engineering, Wakayama University, Wakayama 640–8510, Japan
2 Faculty of Systems Engineering, Wakayama University, Wakayama 640–8510, Japan
a) tac@sys.wakayama-u.ac.jp

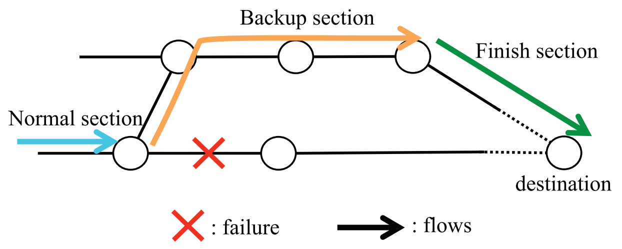

Fig. 1 Model of IP Fast Reroute schemes.

proposed [19]. They proactively prepare alternative next-hops to forward packets when the primary next-hop is unavailable due to failure. However, because every backup path is 1-hop long, they cannot cover every single link or node failure.

Currently, one of the well-agreed goals of IPFRR is to cover a single link/node failure with low overhead. Several schemes that achieved single-failure coverage are proposed so far. One of the major approaches for such IPFRR schemes is the *tunneling approach* such as NotVia [5], [8], [9]. NotVia computes a tunnel in advance for every single-failure scenario i.e., for protection against single-node failure, so that every node has its tunnel that reaches the next-next-hop node on the shortest path without visiting the next-hop failure node. When a packet meets failure, the packet is encapsulated and is forwarded into the tunnel to bypass the failed component.

Another major approach is the *two-table approach*, in which the secondary routing table is proactively computed to be activated in case of failure. FIFR (Failure Inferencing based Fast Rerouting) [4], [10] would be the representative proposal in this approach, which covers every single link/node failure scenario. In FIFR, routers infer failure according to the in-coming network interfaces of packets, i.e., if a packet for a destination arrives from an unusual interface, the packet is forwarded using the secondary routing table to avoid the failure node. To improve the computational cost of FIFR, Xi et al., proposed a time-efficient algorithm to compute the secondary routing table [11].

As an extension of the two-table approach, SBR (Single Backup-table Rerouting) was proposed [6], [12], [13]. SBR, which covers single link/node failure scenarios, achieved to prevent packet loops in case of multiple-failure to improve reliability of networks by utilizing a 2-bit field on packet header.

In this paper, we present an integrated model that accounts for most of those IPFRR schemes that covers single link/node failure, in order to propose a general method that can be applied to many IPFRR schemes. In this model, (i) we first assume that every node uses a shortest-path based algorithm to compute the primary routing table, and (ii) every node has a backup path to forward packets without visiting the next-hop link or node. Also, (iii) the traveling paths of packets in these schemes can be splitted into three sections as shown in **Fig. 1**. Namely, the first section is the path along the shortest path that is used by the packets that have not met failure, the second section is the path along the backup routing configuration that is used after packets meet failure, and the third section is the path along the shortest path that the packets escaped from the second section use to reach their destination. We call those three sections as *normal, backup*, and *finish* sections, respectively.

We hereafter assume this integrated model rather than treating each individual IPFRR scheme in order to propose a general framework of load balancing for IPFRR schemes.

Note that, MRC (Multiple Routing Configuration) [30] is also known as an IPFRR scheme that covers any single link or node failure. However, the framework of MRC is far capable than our IPFRR model so that it is not involved in our study. MRC computes multiple trees over the network, and consequently each node has multiple next-hops for a single destination. MRC potentially is more capable than NotVia, FIFR, and SBR that are included in our model, in exchange for its heavy overhead. Due to its capability, it is reported that MRC has good load-balancing performance in presence of failure [31]. However, as is mentioned here, MRC requires too heavy overhead for practical use.

2.2 Related Work on Load Balancing in IP Networks

Several load balancing methods have been proposed for shortest-path based IP networks, which is independent of IP fast reroute schemes. We first introduce these load balancing methods because they are deeply related to the load balancing mechanisms over IP fast reroute schemes.

For instance, Antić et al., proposed TPR (Two Phase Routing) [14], which distributes traffic into several paths using intermediate nodes, i.e., TPR once forwards packets to some intermediate nodes using IP tunnels and then forwards them to their destinations using normal shortest paths. Mishra et al., proposed S-OSPF (Smart-OSPF) [15], in which the source node of a flow distributes traffic to its neighbor nodes to balance traffic load among those neighbors, i.e., among the shortest paths that start from the neighbors to the destination. They work effectively with relatively low overhead. However, in case of failure under IPFRR schemes, they do not work well immediately because they have to decide the ratio of traffic to distribute among several possible paths, based on the measured traffic load given as the demand matrix of the network. Namely, when a failure occurs, they first have to measure the traffic load over the network, and then compute the optimal distribution ratio. This process naturally brings considerable delay before these load balancing schemes work, even though IPFRR schemes achieve recovery from failure as quick as 50 msec.

There is a few load balancing method that works in the failure state of IPFRR schemes. They introduce mechanisms similar to TPR [14] and S-OSPF [15] into the failure state of IPFRR schemes. Ho et al., proposed a new IPFRR scheme that incorporates load balancing function, in which packets are distributed to several intermediate nodes using IP tunnels when a flow meets failure [18]. Wang et al., presented a load-balanced IPFRR scheme based on LFA, which distributes traffic over multiple alternative next-hops in face of failure [17]. Because they distribute traffic among multiple backup paths, they require optimization processes to control traffic distribution over multiple backup paths, resulting in the heavy computational load and the delay that is required to obtain a traffic demand matrix. Another drawback of these methods is that they cannot cover even a single link or node failure, because they are based on relatively simple IPFRR strategies such as naive tunneling or LFA.

With full-coverage IPFRR schemes that cover any single link or node failure such as NotVia, FIFR, and SBR, few load balancing mechanisms have been proposed. Arai et al., proposed a load balancing method to utilize backup paths of IPFRR not only in the failure state, but also in the normal state to provide a load balancing function [29]. They distribute traffic into both the primary and the backup next-hops at each node, in which an optimized distribution ratio of traffic is applied at each node so as to minimize the maximum congestion ratio over the network.

As another approach, Hara et al., also proposed a method for load balancing in the normal state of IPFRR schemes [16]. Different from [29], they basically use the shortest-path-based routing unless traffic load exceeds the capacity in some links because network operators usually manage their networks using shortest-path based routing. Not to change the usual operation as less frequent as possible, they use backup paths only in face of congestions.

In their scheme, every node monitors congestions through observation of the output queue lengths, and when a congestion is detected in the primary link, packets are rerouted into the corresponding backup path. Note that the rerouted packets may cause other congestion. If a congestion caused by rerouting packets invokes another rerouting of packets, this endless chain of rerouting leads harmful confusion over the network. To prevent this, the rerouted packets are given less priority than the original (unrerouted) packets, and they are dropped by priority in face of congestion. This method enables us to utilize backup paths effectively in the normal state. However, their load balancing mechanism does not work in the failure state because backup paths are occupied to protect routes against failure and are not available for load balancing.

In the current state of the art, no load balancing method that works in the failure state in full-coverage IPFRR schemes has been proposed. In the full-coverage schemes, only one backup path for a destination is generally available against any single link or node failure. Thus, instead of the optimization process that controls traffic distribution among several backup paths, we require a new mechanism to distribute traffic within the limited number of backup paths.

In this paper, we propose a load balancing method for full-coverage IPFRR schemes. Our method is the first proposal to provide a load-balancing function during the failure state in full-coverage IPFRR schemes. In our method, in face of failure, we not only repair the failed paths using the corresponding backup paths, but also we reduce congestions caused by the rerouted traffic, by using the backup paths whose start nodes are on the backup paths that the rerouted packets travel. Our method provides a load balancing function over the limited number of backup paths without any optimization processes.

2.3 Base Load Balancing Scheme over IPFRR Schemes

In this section, we give an in-depth description of the load balancing scheme of Hara et al. [16] since the method that we propose in this paper is an extension of their method.

As mentioned in Section 2.1, packets in IPFRR schemes travel through the three sections of the forwarding paths. In Ref. [16], they utilize a 2-bit field in the packet header to mark packets so as to distinguish the sections in which they are belonging to. We call this mark the *state* of packets, and the states corresponding to the three sections of backup paths are called as *normal*, *backup*, and *finish* states.

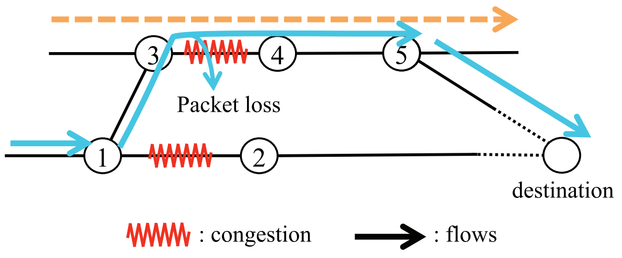

Fig. 2 Load balancing using IP Fast Reroute [16].

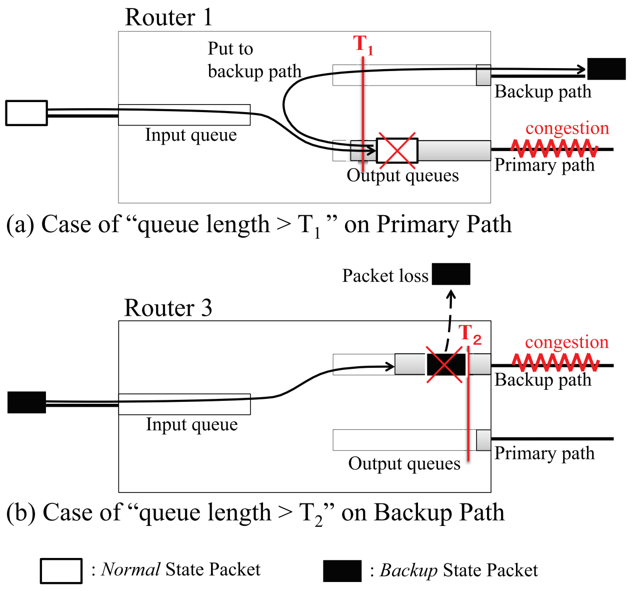

Fig. 3 Load balancing behavior based on queue length.

For the mechanism of Ref. [16], see **Fig. 2**. When a packet of *normal* state meets congestion at router 1, it is forwarded into the backup section with its state changed to *backup* state. To detect congestion at router 1, the router watches the length of the output queues: if the queue length for the primary next-hop interface is longer than threshold T_1, the router detects the congestion, and enqueue the packet into the queue of the backup interface (See **Fig. 3** (a) for the behavior of router 1.). When a packet of *backup* state reaches the end of the backup section (at router 5 in Fig. 2), the state is changed to *finish* state and the packet is forwarded along the shortest path to reach its destination.

As we mentioned in Section 2.2, the packets forwarded into the backup section may cause other congestions. To prevent the harmful influence of this rerouting chain, their scheme gives less priority to the *backup*-state packets to make them dropped as soon as they meet a congestion again. For the specific mechanism, see Fig. 3 (b). When the *backup*-state packets reach Router 3, if the queue length for the backup-path interface, which connects to Router 4, is longer than threshold $T_2(< T_1)$, the *backup*-state packets are silently dropped because of their less priority. It is shown in Ref. [16] that, due to this priority control, their scheme scarcely reduces the performance of the original flows (i.e., flows that are not rerouted) in both delay and throughput even in a high-load network.

In multipath load balancing schemes, we have to be careful to prevent loops of packets. In this scheme, if we allow reroutes for *finish*-state packets (i.e., if we allow packets to be rerouted more than once), it may cause routing loops in case of multiple congestions or failures. To prevent these harmful loops, we should prohibit reroutes of *finish*-state packets, or, as is proposed in Ref. [16], we may limit the number of reroutes that a packet can experience.

3. Load Balancing Method for Failure State

3.1 Basic Idea

We propose a reactive load balancing method that works in the failure state of IPFRR. As base IPFRR schemes, we assume the integrated model of IPFRRs described in Section 2.1 and Fig. 1. Therefore, the proposed method works over most of the full-coverage IPFRR schemes including NotVia, FIFR, and SBR. In this model, we assume that every node has a single backup path for each destination to forward packets without using the next-hop node (or link) along the shortest path. Note that, in our model, no further backup path is available for the packets in a backup section, because they are travelling with the backup (i.e., secondary) routing configurations and the third routing configuration does not exist in our model.

Our method is designed as an extension of the mechanism of Ref. [16]. Our basic idea is to allow secondary rerouting of packets to reduce congestion caused by the rerouting packets coming from failure. Similar to Ref. [16], our concept is to provide a load-balancing function in the failure state of IPFRR while preserving usual network behavior of the shortest-path-based IP routing unless congestion does not present. In other words, our method adds a load-balancing function to existing IPFRR schemes without changing the forwarding paths of the original routing paths as long as possible, so that network operators have less cost to employ the method. In the failure state, the IPFRR scheme forwards packets into the backup paths at the failure point to avoid the failed node or link. Note that these packets that are forwarded into backup paths may invoke another congestion. Only if such congestion occurs in the backup paths, our mechanism works; it uses backup paths again at the congested point to reduce packets going into the congestion. Here, our idea to reduce the congestion is to change the forwarding paths of the packets that do not use the failed component but travel around the failure.

The idea of this two-level rerouting is illustrated in **Fig. 4**. Assume that the failure of link (1, 2) occurs and accordingly that the packets in the flows that primary next-hop is node 2 (such as flow 1 in Fig. 4) are forwarded into the backup section at node 1 using the mechanism of IPFRR. Also assume that the rerouted packets arrive at node 3 and link (3, 4) is congested as a result. (Note that there may be more than one congested links such as (3, 4) because more than one flows may reroute packets due to the failure.) With the conventional method [16], the rerouted packets are likely to be dropped with high probability because the rerouted packets are given less priority. This leads the result that only a limited part of rerouted packets can be saved by IPFRR, which degrades the network performance in the failure state.

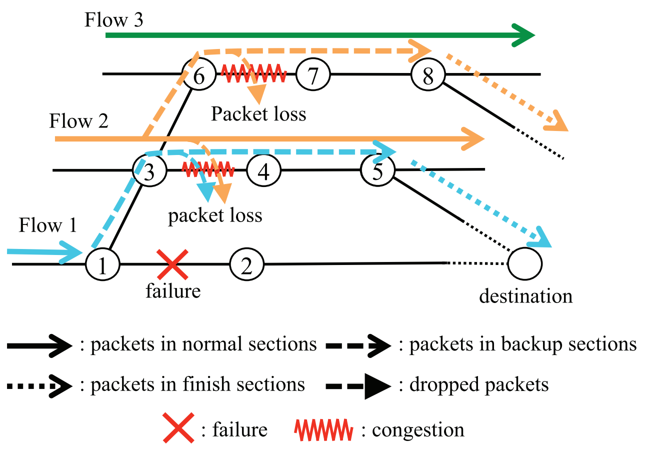

Fig. 4 Two-level rerouting in the proposed method.

In contrast, our method adds the mechanism of two-level rerouting that achieves load-balancing function in the scheme. Our method allows node 3 in Fig. 4 to reroute a part of packets in other flows (such as flow 2) that has been using the congested link (3, 4) as their primary paths. This secondary rerouting makes room on link (3, 4) for the rerouted packets of flow 1 that come from router 1, which shrinks the congestion on link (3, 4). Note that the secondary rerouted flows like flow 2 may have different destinations as flow 1. So, if another link (1, 3) or (4, 5) is congested, we would find the flows (like flow 2) that can shrink the congestion by rerouting a part of their packets at node 1 or 4, respectively. That is to say, we utilize the resources of the secondary backup paths to reduce the degradation of network throughput that is caused by the congestion coming from the rerouted flows in the failure state.

Note that we do not allow the third reroutes, in order to prevent the harmful chain of congestion. This is because reroutes of packets include both benefits and risks over the performance in communications; if we allow higher-level reroutes (i.e., more than secondary reroutes), then the throughput may be improved by utilizing the backup paths of larger number of nodes, while the redundant rerouted traffic that may degrade communication quality increases in the wider area of the network. Because we regard it important to clarify the benefit of the proposed method with the most elementary condition, we in this paper examine the two-level case first. Consequently, we give less priority to the secondary rerouted traffic (e.g., packets in flow 2 rerouted at node 3 in Fig. 4) in the same way as Ref. [16] to have them immediately dropped on the congested link (6, 7), to prevent effecting on flow 3, which uses link (6, 7) in its primary path.

3.2 State Transitions

In the conventional scheme [16], they use three states (i.e., *normal, backup*, and *finish*) of packets to perform priority processing of packets. In our scheme, because we have to distinguish the primary rerouted packets from the secondary ones, we introduce another state: when a packet is rerouted due to failure (i.e., the first-level rerouting is invoked at router 1 in Fig. 4), the packet state changes to *backup* state. In addition, when a packet of *normal* state is rerouted due to the followed congestion, (i.e., the second-level rerouting invoked at router 3 in Fig. 4), the packet

Table 1 Forwarding rule of proposed scheme.

Conditions			Operations	
packet state	failure	queue length	forward to	state change
normal	No	$l(p) \geq T_1$	backup Path	To *secondary*
		$T_1 > l(p)$	primary path	-
	Yes	-	backup path	To *backup*
backup	No	$l(b) \geq T_1$	drop	-
		$T_1 > l(b)$	backup path	-
	Yes	-	drop	-
secondary	No	$l(b) \geq T_2$	drop	-
		$T_2 > l(b)$	backup path	-
	Yes	-	drop	-

* $l(p)$: queue length for primary path, $l(b)$: queue length for backup path
* "Failure" means the failure existence on primary nexthop for *normal*-state packets, on backup nexthop for *backup*- or *secondary* -state packets.

state changes to *secondary* state.

Note that we in this paper do not distinguish the backup section of the rerouting path from the finish section; both sections in the first-level reroute correspond to the *backup* state, and those in the second-level correspond to the *secondary* state. This is because, as described in Section 2.3, the role of the *finish* section in Ref. [16] is to limit the number of reroutes that a packet experiences. This function plays an important role when we deploy a load balancing function in the practical scenes, however, to simplify the description of our scheme in this paper, we do not refer to this function hereafter. Namely, we regard that the *backup* section and the *finish* section correspond to the same state.

3.3 Packet Priority Control

In our scheme we give each packet a proper priority to prevent the harmful congestion chains as well as to take the fairness among flows into account.

We first point out that, at the congestion link that caused by failure (e.g., link (3, 4) in Fig. 4), the original flows and the rerouted flows should be treated equally. It is because failure should be concealed from users in IPFRR schemes, and consequently those flows should be fairly processed. Thus, we apply the rule of T_1 threshold instead of T_2 (as described in Section 2.3 and Fig. 3 (b)) to both the *normal*- and *backup*- state packets. It means that the drop probability of *normal* and *backup* packets (i.e., the normal flow-2 packets and the backup flow-1 packets at node 3 in Fig. 4) is the same in the congested link.

In contrast, we have to prevent the rerouting chain that causes harmful congestion so that we limit reroutes within two levels. Specifically, at the congested link that is caused by the first-level rerouting (e.g., link (6, 7) in Fig. 4), we apply the rule of T_2 threshold into *secondary*-state packets to drop them immediately when they meet another congestion.

The formal packet forwarding rule is shown in **Table 1**. This table shows the behavior of routers when a packet comes in, under every possible conditions.

3.4 Enabling Secondary Rerouting

In our scheme, we enable the secondary rerouting only when failure occurs. In the practical scenes of network management, multi-path load balancing mechanisms include several inconveniences such as large jitter or packet reordering. Thus, many network operators would not wish to use load-balancing functions as usual. To reduce such degradation of networks in normal state, we introduce a mechanism to enable the secondary rerouting mechanism only when the first-level rerouting is occurring.

To control the load-balancing function, we introduce a 1-bit flag for each interface on every router that indicates the load-balancing is enabled or not. Namely, only if the flag of the primary link is *true* (i.e., the flag of the corresponding interface is *true*), a *normal*-state packet that meets congestion on the primary link is forwarded into the backup section with its state changed to *secondary* state.

The 1-bit flag is initially *false*, and is changed to *true* when a *backup*-state packet that uses the corresponding interface arrives at the router. The flag becomes *false* when no such *backup*-state packet arrives at the router in a certain period of time.

4. Performance Evaluation

4.1 Simulation Scenario

We evaluate the proposed method using the ns-2 simulator [20]. As a base IPFRR scheme, we chose a link protection scheme SBR-LP (Single Backup-table Rerouting - Link Protection) [13] because SBR-LP is convenient to cooperate with the proposed load balancing method in that it utilizes a 2-bit field on the packet header. Note that SBR-LP provides backup paths against every single link failure, where the algorithm to compute backup paths is similar to Ref. [11]. We implemented SBR-LP and the proposed method over SBR-LP on ns-2.

In our evaluation, we designed two scenarios in which two different types of topologies are used. In scenario 1, we use randomly generated topologies that model the Internet. In scenario 2, we use topologies of real networks retrieved from a public database.

We first describe scenario 1. The topologies used are random networks of Waxman [21] model topology generated by topology generator BRITE [22], where the number of nodes is 30 and that of links is 60. The bandwidth of every link is 1.0 Mbps and its transmission delay is 1 msec. Note that, although the link speed is extremely low compared to the practical networks, we can estimate the throughput of high-speed networks because the performance is in proportion of link speed. We generate CBR (Constant Bit Rate) flows randomly, i.e., we select a source and a destination node randomly from the network. The packet size is 1 Kbytes and the transmission rate of every flow is 200 Kbps. The output queue size is as large as 50 packets. Threshold T_1 and T_2 is set at 10% and 100% of the queue capacity, respectively. We vary the number of flows generated between 20 and 150 with interval of 5. For each case of flow numbers, we performed 30 trials of the simulation; we use 5 different random topologies with 6 different random seeds and compute the average of them. Note that the number of generated flows is relatively small compared to the number of node pairs in the network. As is often seen in the practical scenes, our scenario simulates the situation where traffic amount varies among node pairs, by generating only dominant flows randomly in the networks. The overview of simulation settings in scenario 1 are shown in **Table 2**.

On the other hand, in scenario 2, evaluation is done through simulations using real network topologies. We used five topologies, Telstra, EBONE, Tiscali, Exodus, and Abovenet, obtained from topology database site Rocketfuel [23]. We set the bandwidth of every link to be in inverse proportion to its weight. Specifically, the link bandwidth is set to 5/weight Mbps. Flows are generated in the configuration similar to scenario 1. The transmission rate of every flow is 200 kbps. The packet size is 1 Kbytes and the output queue size is as large as 50 packets. Threshold T_1 and T_2 are set at 10% and 100% of the queue capacity, respectively. We vary the number of flows generated between 50 and 150 with interval of 10. For each case of flow numbers, we performed 12 trials of the simulation with different random seeds. The overview of the simulation settings in scenario 2 is shown in **Table 3**.

In both scenarios 1 and 2, we measure the performance in case of single link failure. For this purpose, we generated flows at 1 second from start, and made a single link failure randomly 5 seconds later. We measured the throughput of two 4-second periods in both before and after the failure occurs; the measurement is done in the time period between 2–6 seconds from start as the performance of the normal state, and in the time period between 7–11 seconds from start as that of the failure state. Note that, we did not measure the throughput of all flows: we measured the throughput of the flows that are affected by the failure. Specifically, we define the *affected flows* as the flows whose packets are rerouted by the proposed scheme in the failure state, and measured the throughput of them to compare the performance of two schemes.

Table 2 Simulation settings in random networks.

Item	Value
Topology	Waxman Model
#Nodes	30
#Links	60
Link Bandwidth	1.0 Mbps
Link Delay	1 msec
Flow Type	CBR
Rate of a Flow	200 kbps
Packet size	1 kbytes
Output Queue size	50 packets
Threshold T_1	10% of queue size
Threshold T_2	100% of queue size
#flows	20–150

Table 3 Simulation settings in rocketfuel networks.

Item	Value				
ISP Name	Telstra	EBORN	Tiscali	Exodus	Abovenet
#Nodes	104	87	161	79	138
#Links	151	161	328	147	372
LinkBandwidth	5 / link weight Mbps				
Link Delay	Obtained from Rocketfuel				
Flow Type	CBR				
Rate of a Flow	200 kbps				
Packet size	1 kbytes				
Output Queue size	50 packets				
Threshold T_1	10% of queue size				
Threshold T_2	100% of queue size				
#flows	50–150				

4.2 Results

4.2.1 Random Topology Scenarios

We describe the result of scenario 1. **Figure 5** shows the average throughput of the *affected flows* for each case of flow numbers in the failure state. The throughput is shown as the ratio compared to the throughput in the normal state, i.e., 100% means the same throughput as the normal state. Although the throughput of the conventional method (i.e., SBR-LP) decreases as the flow number increases, the throughput of the proposed method keeps the same level. Note that in several cases the throughput of the proposed method in the failure state is larger than the normal state, i.e., it exceeds 100% in Fig. 5. This is because the proposed method can reduce the level of the congestion that had been occurring in the normal state, because the secondary reroutes work only when failure occurs due to the mechanism described in Section 3.4.

Figure 6 indicates the average link occupancy of each case of flow numbers in the normal state, and **Fig. 7** shows the ratio of rerouted flows, i.e., the ratio of the flows that did not experience reroutes, that experienced reroutes caused by failure, and that ex-

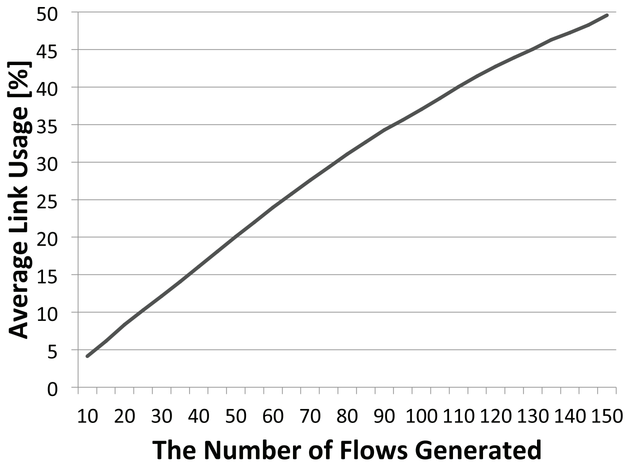

Fig. 6 Average link occupancy in normal state.

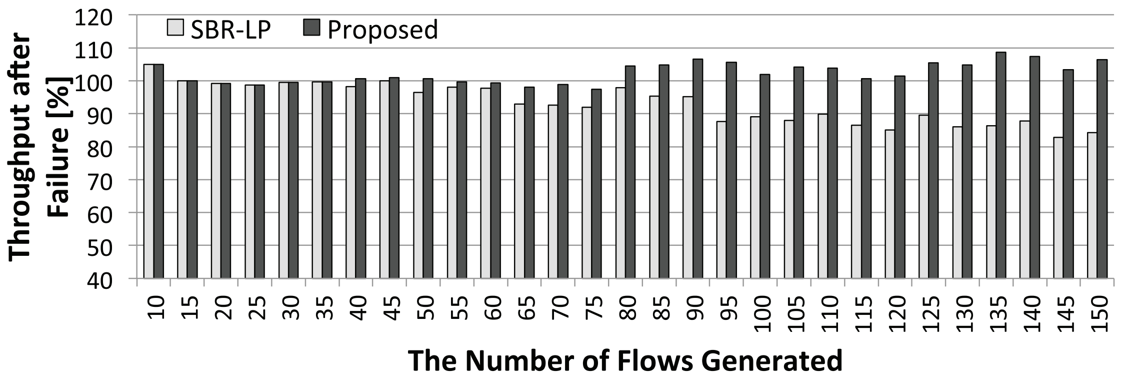

Fig. 5 Throughput with various random network load (average of 30 trials).

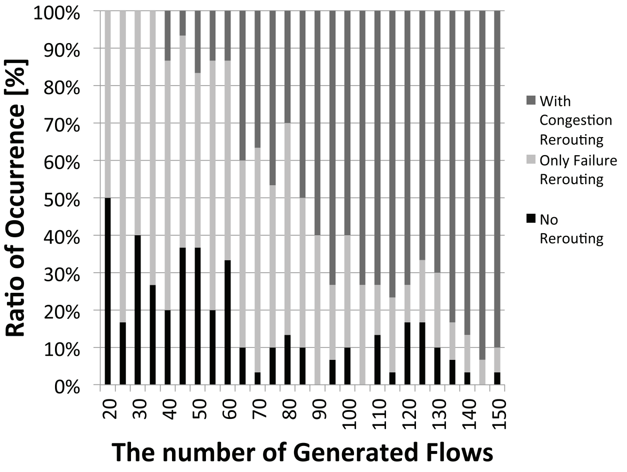

Fig. 7 Ratio of rerouted flows in proposed scheme.

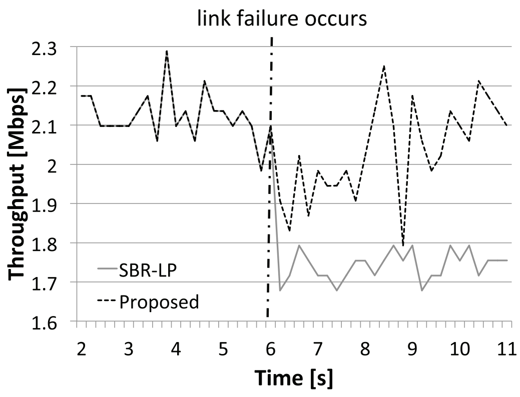

Fig. 8 Throughput in time series.

perienced reroutes caused by congestion, in the failure state of the proposed scheme. These figures show that the flows rerouted due to congestion increase as the link occupancy grows, and accordingly the difference of throughput between the conventional and the proposed schemes grows larger. Also, note that the average link usage is less than 50% throughout the evaluation. It indicates that, although networks are usually designed with adequate margin in link capacity, congestion may occur in the failure state and the proposed method works effectively in such situation.

Figure 8 shows the time series of throughput measured with 0.2 second interval, in a typical case of the 150-flow scenarios. As the figure shows, the throughput of SBR-LP degrades immediately when failure occurs, while the proposed method keeps the same level of throughput. It shows that the proposed scheme works immediately after failure without delay. Note that the throughput of the proposed method in a few second just after failure is a little lower than the following several seconds. This is caused by the queuing delay that rapidly increases when failure occurs due to the congestion on the backup paths. Although the effect of the queuing delay is kept in a few second in this scenario, the queuing delay in practice will shrink significantly as the link speed is far larger than that. Note that in this period of time the throughput also fluctuates a little larger than usual. This is because several backup paths work even under a single failure, and each backup paths have different queuing delays.

Figure 9 shows the ratio of drop packets of the *affected flows* in the same scenario. Although there are several exceptions, most

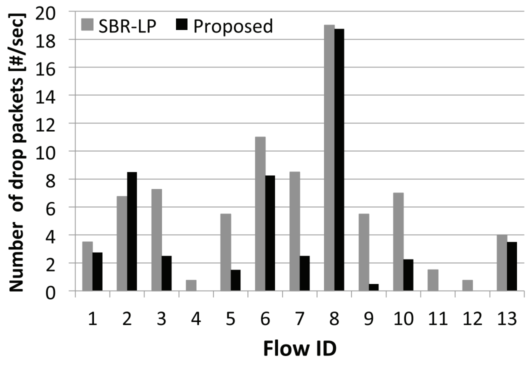

Fig. 9 Ratio of drop packets per flow.

flows reduce the drop packet ratio compared to SBR-LP. This is the gain brought from the secondary rerouting.

4.2.2 Real Topology Scenarios

We describe the result of scenario 2. **Figures 10–14** show the throughput of the *affected flows* in the failure state in each of five topologies. Same as Fig. 5, these figures show the ratio of the throughput in the failure state compared to the normal state.

Throughput of the proposed method is higher than SBR-LP in all topologies. These results show that the proposed method is also effective in the real topology. On the other hand, the performance of both SBR-LP and the proposed method is quite different in different networks. Although we could not find any relation between topology characteristics and the performance, the results show the range of performance under variations of network topologies.

In both SBR-LP and the proposed method, there are the cases where the throughput after failure exceeds the throughput before failure. In SBR-LP, such a case occurs when the rerouting packets reduce the traffic in the shortest paths, and consequently reduce congestion in these paths. In the proposed method, this tendency goes greater than SBR-LP. This is because the secondary rerouting in the proposed method often reduces the congestion that has been occurring before failure.

4.3 Discussion

We proposed a load balancing method for the failure state of IPFRR schemes. In the state where packets are rerouted by IPFRR against failure, the proposed method further reroutes packets at the congested points that are caused by the rerouted packets, using the detour paths of IPFRR schemes to shrink the congestion. Through the evaluation, we confirmed that the proposed method reduces the congestion that occurs in the backup paths, and prevents the degradation of throughput performance in case of a single failure.

Note that, although the proposed method shrinks the congestion that occurs on backup paths using secondary rerouting, the proposed method compels flows that are not directly affected by link failure to reroute. Thus, the proposed method may cause larger jitter and packet reordering, which may degrade communication quality.

Through the simulations, we have shown that the throughput in the failure state is as large as, or more than that of the normal state with the proposed method. Throughput is actually the

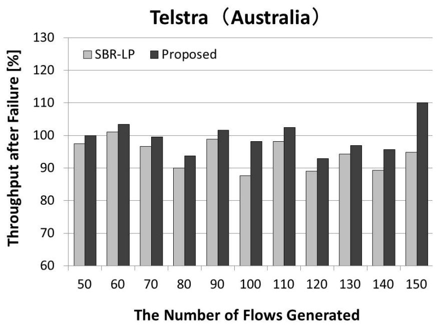

Fig. 10 Throughput in Telstra network.

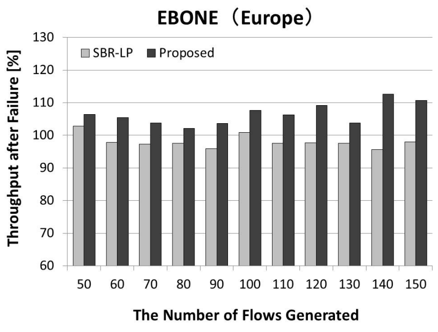

Fig. 11 Throughput in EBONE network.

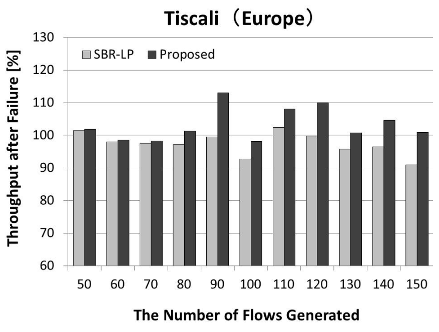

Fig. 12 Throughput in Tiscali network.

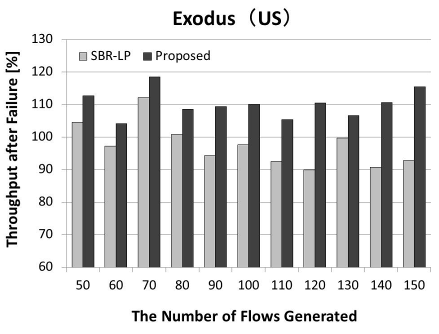

Fig. 13 Throughput in Exodus network.

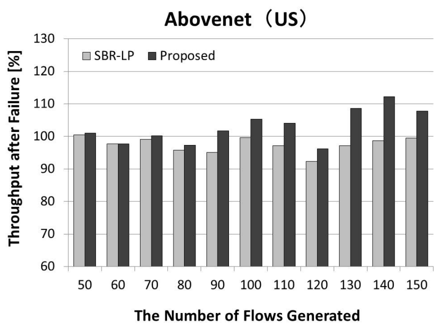

Fig. 14 Throughput in Abovenet network.

most important criterion to measure network performance, and so the proposed method is proved to be effective to improve the network performance. However, the degradation of communication quality from the viewpoint of jitter and packet reordering is not negligible.

For this kind of quality degradation, several methods that reduce the harmful influence of packet reordering have been proposed in the literature. In general, in this area of research, the granularity of load distribution has been discussed as one of the essential tradeoff issues; if we distribute traffic in the per-packet fashion, the effect of reordering goes significant, and if we do it with larger granularity, the load balancing performance degrades. One of the major methods to reduce the effects of reordering is to use the granularity of flows instead of that of packets, using a hash function [24], [25]. Several studies provide more sophisticated packet manipulations for traffic distribution that are robust against packet reordering [26], [27]. New TCP protocols to endure packet reordering are also presented [28]. One of the future tasks is to apply them to the proposed method, and evaluate the performance against packet reordering.

5. Concluding Remarks

In this paper, we proposed a reactive load balancing scheme to reduce degradation of network performance in the failure state of IPFRR schemes. Our scheme works over most of the major IPFRR schemes that cover single failure, and reuses their backup paths reactively to distribute traffic into wider area of networks around the failure component. Therefore, it requires small overhead of packet marking compared to the existing load balancing techniques that instead requires the overhead of estimating traffic matrix. The evaluation using a network simulator clarified that we can achieve at least the same level of the throughput as the normal state even in the failure state.

One of the problems in this scheme would be the problem of packet reordering. If a traditional TCP is used with the proposed method, the packet reordering degrades the throughput of flows, which reduce the effect of the proposed method. Accordingly, one of the future tasks is to apply existing techniques [26], [27], [28] that reduce the harmful influence of packet reordering, and to evaluate the performance of the proposed method against packet reordering. One of the essential tasks for the future is to grasp the level of harmful influence of packet reordering in comparison with the good aspect of the proposed method over throughput.

References

[1] Moy, J.: OSPF Version 2, IETF RFC2328 (Apr. 1998).

[2] ISO/IEC: Intermediate System to Intermediate System intra-domain routing information exchange protocol for use in conjunction with the connectionless-mode network service (ISO 8473), ISO/IEC, Technical Report 10589:2002(E) (Apr. 2002).

[3] Markopoulou, A., Iannaccone, G., Bhattacharyya, S., Chuar, C.N. and Diot, C.: Characterization of failures in an IP backbone network, *IEEE/ACM Trans. Netw.*, Vol.16, No.4, pp.749–762 (2008).

[4] Zhong, Z., Nelakuditi, S., Yu, Y., Lee, S., Wang, J. and Chuah, C.N.: Failure inferencing based fast rerouting for handling transient link and node failures, *Proc. IEEE Global Internet* (Mar. 2005).

[5] Shand, M., Bryand, S. and Previdi, S.: IP Fast Reroute Using Not-via Addresses, draft-ietf-rtgwg-ipfrr-notvia-addresses-04.txt (2009).

[6] Yoshihiro, T. and Jibiki, M.: Single Node Protection without Bouncing in IP Networks, *IEEE 13th Conference on High Performance Switching and Routing* (*HPSR2012*), pp.88–95 (2012).

[7] Dasgupta, S., de Oliveira, J.C. and Vasseur, J.P.: A Performance Study of IP and MPLS Traffic Engineering Techniques under Traffic Variations, *IEEE Globecom2007*, pp.2757–2762 (2007).

[8] Li, A., Francois, P. and Yang, X.: On Improving the Efficiency and Manageability of NotVia, *Proc. ACM CoNext 2007* (2007).

[9] Enyedi, G., Szilágyi, P., Rétvári, G. and Császár, A.: IP Fast ReRoute: Lightweight Not-Via without Additional Addresses, *Proc. IEEE INFOCOM2009*, pp.2771–2775 (2009).

[10] Wang, J. and Nelakuditi, S.: IP Fast Reroute with Failure Inferencing, *Proc. SIGCOMM Workshop* (*INM 2007*), pp.268–273 (2007).

[11] Xi, K. and Chao, H.J.: IP Fast Rerouting for Single-Link/Node Failure Recovery, *Proc. IEEE BROADNETS2007*, pp.142–151 (2007).
[12] Ito, H., Iwama, K., Okabe, Y. and Yoshihiro, T.: Single backup table schemes for shortest-path routing, *Theoretical Computer Science*, Vol.333, No.3, pp.347–353 (2005).
[13] Yoshihiro, T.: A Single Backup-Table Rerouting Scheme for Fast Failure Protection in OSPF, *IEICE Trans. Commun*, Vol.E91-B, No.9, pp.2838–2847 (2008).
[14] Antić, M. and Smiljanić, A.: Oblivious Routing Scheme Using Load Balancing Over Shortest Paths, *Proc. IEEE ICC 2008*, pp.5783–5737 (2008).
[15] Mishra, A.K. and Sahoo, A.: S-OSPF: A Traffic Engineering Solution for OSPF based Best Effort Networks, *Proc. IEEE Globecom 2007*, pp.1845–1849 (2007).
[16] Hara, M. and Yoshihiro, T.: Adaptive Load Balancing based on IP Fast Reroute to Avoid Congestion Hot-spots, *IEEE International Conference on Communications* (*ICC 2011*), pp.1–5 (2011).
[17] Wang, N., Fagear, A. and Pavlou, G.: Adaptive post-failure load balancing in fast reroute enabled IP networks, *12th IFIP/IEEE International Symposium on Integrated Network Management* (*IM2011*), pp.470–477 (2011).
[18] Ho, K., Wang, N., Pavlou, G. and Botsiaris, C.: Optimizing post-failure network performance for IP Fast ReRoute using tunnels, *5th International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness* (*QShine08*) (2008).
[19] Atlas, A. and Zinin, A.: Basic Specification for IP Fast Reroute: Loop-free Alternate, IETF RFC 5286 (2008).
[20] The network Simulator NS-2, http://www.isi.edu/nsnam/ns/.
[21] Waxman, B.: Routing of Multipoint Connections, *IEEE J. Select. Areas Commun.* (Dec. 1988).
[22] Medina, A., Lakhina, A., Matta, I. and Byers, J.: BRITE: An approach to universal topology generation, *Proc. IEEE MASCOTS*, pp.346–353 (Aug. 2001).
[23] Spring, N., Mahajan, R., Wetherall, D. and Anderson, T.: Measuring ISP topologies with rocketfuel, *ACM SIGCOMM'02*, pp.133–145 (Aug. 2002).
[24] Thaler, D. and Hopps, C.: Multipath Issues in Unicast and Multicast Next-Hop Selection, IETF RFC2991 (2000).
[25] Hopps, C.: Analysis of an Equal-Cost Multi-Path Algorithm, IETF RFC2992 (2000).
[26] Kandula, S., Katabi, D., Sinha, S. and Berger, A.: Dynamic Load Balancing Without Packet Reordering, *ACM SIGCOMM Computer Communication Review*, Vol.37, No.2, pp.51–62 (2007).
[27] Martin, R., Menth, M. and Hemmkeppler, M.: Accuracy and Dynamics of Hash-Based Load Balancing Algorithms for Multipath Internet Routing, *Proc. IEEE International Conference on Broadband Communication, Networks, and Systems* (*BROADNETS*) (2006).
[28] Bohacek, S., Hespanha, J.P., Lee, J., Lim, C. and Obraczka, K.: A new TCP for persistent packet reordering, *IEEE/ACM Trans. Networking* (*TON*), Vol.14, No.2, pp.369–382 (2006).
[29] Arai, Y. and Oki, E.: A Routing Scheme Distributing Traffic using Primary and Backup Ports of IP Fast Reroute, *IEICE Communications Express*, Vol.2, No.3, pp.93–97 (2013).
[30] Kvalbein, A., Hansen, A.F., Cicic, T., Gjessing, S. and Lysne, O.: Multiple routing configurations for fast IP network recovery, *IEEE/ACM Trans. Networking*, Vol.17, No.2, pp.473–486 (2009).
[31] Kvalbein, A., Cicic, T. and Gjessing, S.: Post-Failure Routing Performance with Multiple Routing Configurations, *26th IEEE International Conference on Computer Communications* (*INFOCOM 2007*), pp.98–106 (2007).

Kazuki Imura received his B.E. and M.E. degrees from Wakayama University in 2012 and 2014, respectively. He is currently working with Benic Solution Corporation.

Takuya Yoshihiro received his B.E., M.I. and Ph.D. degrees from Kyoto University in 1998, 2000 and 2003, respectively. He was an assistant professor in Wakayama University from 2003 to 2009. He has been an associate professor in Wakayama University from 2009. He is currently interested in the graph theory, distributed algorithms, computer networks, wireless networks, medical applications, bioinformatics, etc. He is a member of IEEE, IEICE, and IPSJ.