A new method of proactive recovery mechanism for large-scale network failures

Takuro Horie*, Go Hasegawa†, Satoshi Kamei‡, Masayuki Murata*
*Graduate School of Information Science and Technology, Osaka University
Yamadaoka 1-5, Suita, 565-0871 Japan
E-mail: {t-horie,murata}@ist.osaka-u.ac.jp
†Cybermedia Center, Osaka University
Machikaneyama 1-32, Toyonaka, Osaka, 560-0043 Japan
E-mail: hasegawa@cmc.osaka-u.ac.jp
‡NTT Service Integration Laboratories, NTT Corporation
Midori-cho 3-9-11, Musashino, Tokyo, 560-0043 Japan
E-mail: kamei.satoshi@lab.ntt.co.jp

***Abstract*—This paper proposes a novel recovery mechanism from large-scale network failures caused by earthquakes, terrorist attacks, large-scale power outages and software bugs. Our method, which takes advantage of overlay networking technologies, pre-calculates multiple routing configurations to prevent possible simultaneous network failures and selects one configuration immediately after detecting the failures. Through numerical calculation results using actual AS-level topology, we show that our proactive method improves network reachability from 89% to 99%, while keeping the path length sufficiently short, when up to 8% of the nodes in a network are down simultaneously.**

***Index Terms*—Overlay network, routing, large-scale network failures, proactive failure recovery**

I. Introduction

Computer networks have already been regarded as an essential infrastructure, like water and gas utilities. Therefore, recovering from network failures and ensuring network connectivity are becoming an important challenge.

Generally, highly reliable networks can be realized by adding redundancy to network equipment. In this case, when active equipment goes down due to some failure, the network will recover from the failures by replacing them with the alternate equipment. Therefore, existing research on network recovery focuses generally on the trade-off between cost and performance, and concludes that, to increase the efficiency of the recovery mechanism within limited resources, we should add higher-level redundancy to the network equipment with a larger probability of failing. However, this traditional approach cannot be applied to the recovery mechanisms for large-scale network failures caused by earthquakes, terrorist attacks, large-scale power outages and software bugs, because the probability of such failures occurring is quite low and the implementation cost for preparing against such failures is very expensive. Consequently, most existing protection and recovery mechanisms assume a single-failure model, that is, only one failure occurs at one time, and there have been very few studies on protecting mechanisms against large-scale network failures in which many network elements go down simultaneously.

Furthermore, there have been few studies regarding large-scale failures in IP networks such as the Internet [1]. The main reason for this may be that IP itself has some mechanisms to quickly recover from small-scale network failures. However, recent investigations have revealed that the Border Gateway Protocol (BGP) [2], which operates inter-Autonomous-System (inter-AS) routing in the current Internet, requires considerable time (from a few minutes to several days) to converge routing tables, especially when large-scale failures occur or for certain kinds of network topologies [3, 4]. There is essentially no theoretical upper bound for the routing convergence time in BGP, and there are many situations in which the routing convergence time increases significantly, as in the count-to-infinity problem [5].

Therefore, various methods to improve routing convergence time in BGP have been proposed [6-8]. However, most of them require modifications to BGP and TCP/IP, meaning that they require standardization processes. Consequently, such modifications cannot be deployed to the current Internet in the near future.

Another problem of current BGP routing is the policy-based routing operated by Internet Service Providers (ISPs). ISPs generally have many links interconnecting with other ISPs, which have various monetary cost structures, such as peering and transit relationships [9, 10]. BGP routing configurations are very much affected by the ISPs' policies, which are driven by the cost structure of these links. This means that current BGP routing is not configured to maximize user-perceived performance, such as end-to-end delay and throughput [11, 12], as well as the network connectivity itself. We believe that this affects network performance, especially the network connectivity, under large-scale failures.

In this paper, we propose a novel recovery mechanism from large-scale network failures. By taking advantage of proactive network recovery mechanisms, our mechanism can work quickly, even when BGP requires a long time to recover the network reachability, or cannot completely recover from the failure. The main reason we utilize the overlay network approach is that we can deploy the proposed method easily and quickly since it does not require the standardization process. In addition, the application-level traffic routing which is operated by overlay routing can overcome the shortcomings in policy-based BGP routing.

Our method is based on a proactive recovery scheme which pre-calculates multiple routing configurations against possible network failures and shares the configurations throughout the network. When failures are detected, our scheme immediately selects one of the configurations according to the detected failures. In this paper, we propose various algorithms to calculate multiple routing configurations to accommodate large-scale failures in a network.

The effectiveness of our method is demonstrated by numerical evaluation results using the actual AS-level network topology of the current Internet. We show that our method improves the network reachability significantly in cases of single node (AS) failure and simultaneous multiple node failures. Furthermore, our method can keep the average path length after the recovery almost equal to the ideal value.

The remainder of this paper is organized as follows. In Section II, we introduce the research background on overlay routing mechanisms and network recovery mechanisms. In Section III, we give brief explanation of the recovery mechanism which is the basis of our method. In Section IV, we present the design issues and detailed algorithms of our method. We confirm the effectiveness of our method using extensive numerical examples in Section V. Finally, Section VI summarizes the conclusions of the present study and discusses areas of future consideration.

II. Related Work

A. Overlay networks and overlay routing

Overlay networks are defined as upper-layer networks built on the lower-layer IP network, and they provide special-purpose application services such as P2P networks, grid networks, IP-VPN services and Content Delivery/Distribution Networks (CDNs). In overlay networks, the endhosts and servers that run application programs become overlay nodes that form the upper-layer logical network with logical links between the overlay nodes, and the overlay nodes control the application traffic to satisfy their requirements and policies.

Some overlay networks do not assume specific upper-layer applications and concentrate only on the routing of overlay network traffic. We refer to such application-level traffic routing as *overlay routing* [13, 14], which we exploit for the proposed method in this paper. The primary reason for utilizing overlay routing is that it does not require a standardization process since it runs at the application-layer. In addition, the application-level traffic routing which is operated by overlay routing can overcome the shortcomings in policy-based BGP routing.

B. Recovery from large-scale network failures

As in water and gas utilities, information networks are vulnerable to large-scale failures when disasters, such as earthquakes, terrorist attacks and large-area power outages, occur. In addition, software bugs in major router operating systems may result in the simultaneous breakdown of many network nodes (e.g., routers and switches) in a network. In such emergency situations, it is vital to quickly restore network connectivity and to prioritize emergency communications such as 911 calls. Although many studies have considered the restoration of network connectivity, which is also the focus of this paper, most assume a single-failure model, not multiple failures occurring at any particular time. In general, the mechanisms for single failures are not effective for coping with large-scale network failures during which many network elements simultaneously break down.

A further problem associated with recovery mechanisms for large-scale failures is cost/performance trade-off. Since the probability of large-scale network failures occurring is quite low and the implementation cost for preparing against such failures is very high, it is difficult to introduce appropriate recovery mechanisms. Thus, an effective low-cost solution is necessary to deal with large-scale network failure in IP networks.

C. Reactive and proactive recovery mechanisms

In general, network recovery mechanisms are categorized into two types: reactive and proactive. In reactive recovery mechanisms, when network nodes detect network failures, they re-calculate the routing configurations and propagate them throughout the network to converge the routing. The nodes can accommodate various kinds of network failures flexibly without failure prediction by utilizing dynamic mechanisms in calculating and propagating alternate paths after detecting the failures. One of the main shortcomings of reactive recovery mechanisms is that they require considerable time for routing convergence after the failures since new routing information is generally propagated in a hop-by-hop manner.

In contrast, proactive recovery mechanisms pre-calculate recovery settings (i.e., routing tables) by assuming possible failures and then distribute the settings throughout the network. When a network failure is detected, the recovery mechanism immediately selects one of the pre-calculated settings according to the detected failure. So, when the failure is covered by the pre-calculated settings, proactive recovery does not require routing convergence time after the failure. However,

when the failure has not been considered in the pre-calculation, the recovery mechanism cannot completely recover from the failure. So, in the proactive mechanism, we must carefully select the network failures assumed to occur in pre-calculating the recovery settings.

Since our goal is to recover from large-scale network failures in a short time, we employ the proactive network recovery mechanism. Especially, we focus on the Resilient Routing Layers (RRL) proposed in [15], because RRL are simple and have high-flexibility and applicability. We extend RRL to accommodate large-scale network failures. In Section III, we briefly explain the mechanism of RRL and its difficulty in accommodating large-scale failures.

III. Resilient Routing Layers (RRL) [15]

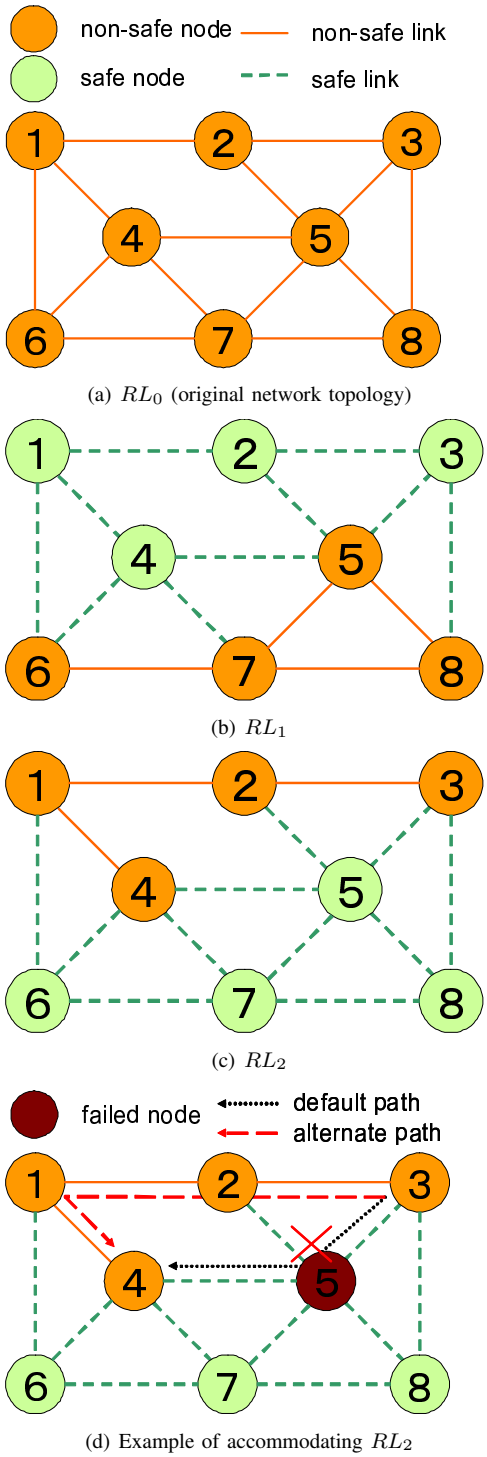
A. Overview

RRL pre-calculates multiple network topologies and routing tables, which are called Routing Layers (RLs), from the original network topology to which RRL is applied. In each RL, RRL assumes a failure of the network node(s) to occur, and configures the network topology to recover the failure without degrading the reachability of other parts of the network. All nodes in the network share the calculated RLs, and select the same one RL when network failures occur. When no failure occurs, RRL utilizes the original network topology.

We refer to the node which is assumed to be down in each RL as a *safe node*, and calculated RLs as a *Routing Layer Set* (RLSet). Each RL, except the original network topology, has at least one safe node. The weight of the link connected to the safe node is set to the maximum value so that the safe node is prevented from being used in a route between other nodes. That is, the links connecting to the safe node are used only when the safe node is either the source node or destination node. We refer to such links as *safe links*. When a node failure is detected by its adjacent node, the adjacent node selects one RL from the RLSet, in which the failed node is safe. Once the adjacent node selects an appropriate RL, all transmitted packets can avoid the failure.

Figure 1 illustrates an example of the application of RRL to the network topology. Figure 1(a) represents the original network topology RL_0. RL_0 is utilized while no failure is detected in the network. In RL_1 in Figure 1(b), nodes 1, 2, 3, and 4 are safe nodes, and nodes 5, 6, 7, and 8 are safe in RL_2 in Figure 1(c). That is, all nodes in the network are safe in at least one RL in RLSet. Note that the weight of the dashed links in Figure 1(c) is set to the maximum value, since they connect to the safe nodes.

Here, consider a data transmission from node 3 to node 4. When there is no failure in the network, RL_0 is utilized and the route becomes 3-5-4 since each RL utilizes the route by Dijkstra's shortest path algorithm. When node 5 is down, the route from node 3 to node 4 becomes unavailable since it includes the failed node. In this case, RL_2 is utilized since node 5 is safe in RL_2. Then the route from node 3 to node 4 becomes 3-2-1-4, as shown in Figure 1(d).



(a) RL_0 (original network topology)

(b) RL_1

(c) RL_2

(d) Example of accommodating RL_2

Fig. 1. Example of RLSet. Shown are RL_0, RL_1 and RL_2

B. Accommodating large-scale network failures

As described above, RRL can recover from a single-node failure completely, meaning that it can keep the reachability of all nodes except the failed node. This is because each node in the network is safe in at least one RL in the RLSet. In [15], the authors show the following evaluation results: up to tens of RLs are needed to keep all nodes in the network safe in at least one RL, even when the network has thousands of nodes. In addition, when multiple nodes which are safe in the same RL fail simultaneously, the failures can be recovered by utilizing the RL. Therefore, as the number of safe nodes in each RL increases, the probability that multiple node failures can be recovered increases. However, as the number of safe nodes in each RL increases, the number of available links in the network decreases, since the links connected to the safe nodes become unavailable due to the nodes' maximum weight. The path length (hop count) also increases between nodes in the RL.

Furthermore, the number of RLs in the RLSet also affects the recovery performance of RRL. When we utilize many RLs and each node in the network becomes safe in multiple RLs, the RLSet will accommodate a larger number of failure patterns. However, increasing the number of RLs in RLSet will increase the memory usage and processing overhead.

Therefore, for RRL to realize high recovery performance after large-scale network failures, we must carefully configure the number of RLs in RLSet, the number of safe nodes in each RL, and the selection of nodes as safe in each RL. However, to the best of our knowledge, no research results have been reported on RRL-based proactive recovery mechanisms for large-scale network failures.

C. RRL implementation as an overlay network

In [15, 16], the authors note that RRL can be implemented at various layers. For example, in [16], RRL runs in an MPLS network. In an IP network, RRL can be implemented by utilizing unused bits of the IP packet header to designate the index of the currently used RL. One of the significant shortcomings in the implementation at the MPLS or IP layers is that the standardization process is required. The other problem is RRL must be implemented for all nodes (MPLS switches or IP routers) in the network.

In this paper, we assume that the proposed method is implemented at the application layer. That is, we exploit overlay networking technologies to implement the proposed method. In general, overlay routing requires additional overhead in terms of processing delay at overlay nodes and application-level encapsulation. However, we take the advantages of the overlay routing which are summarized in Subsection II-A.

IV. Proposed Method

A. RLSet construction

As described in Section III, no efficient method is proposed to recover from simultaneous multiple failures by RRL-based recovery mechanism despite its potential. For accommodating such failures by RRL-based recovery mechanisms, we must carefully choose the following: the number of RLs in RLSet, the number of safe nodes in each RL, and the selection of nodes as safe in each RL. In this subsection, we present various construction algorithms of RLSet. In each construction algorithm, we assume patterns of simultaneous multiple node failures occur, and the proposed algorithm aims at recovery from all the failure patterns.

In all construction algorithms, we select node n to be safe from the nodes in the original network topology, which satisfies the following three conditions:

- n connects to at least one non-safe node.
- All adjacent safe nodes to n connect to at least one non-safe node excluding n.
- The network topology after making n safe maintains the network connectivity. That is, all non-safe nodes in the network can reach the other non-safe nodes without passing through the safe links and safe nodes.

Note that in all construction algorithms, all network nodes are safe in at least one RL in RLSet. Furthermore, in some algorithms, we make some nodes safe in multiple RLs in RLSet. We call this feature an *overlapping feature*.

1) Hub-based algorithm: The hub-based construction algorithm (HUB) assumes failures that greatly affect the network reachability, that is, failures of high-degree nodes (hub nodes) and their adjacent nodes. HUB constructs RLs so that a hub node and as many of its adjacent nodes are as safe as possible. The rest of the nodes are safe in additional RLs, from which we select safe nodes randomly. Note that each node in the network is safe at only one RL in RLSet.

The overlapped hub-based construction algorithm (HUB_o) constructs multiple RLs for each hub node, whereas HUB constructs only one RL for each hub node. Specifically, HUB_o prepares RLs for a hub node so that all of its adjacent nodes become safe in those RLs. As a result, some nodes in the network are safe in multiple RLs; that is, there is overlapping. By this overlapping feature, we can expect improvement of the recovery performance when the number of RLs in RLSet increases.

2) Attribute-based algorithm: The attribute-based construction algorithm (ATR) and overlapped attribute-based construction algorithm (ATR_o) assume that each node in the network has an attribute such as location, vendor name, version of node OS, and topological information. We also assume that in large-scale failures, the nodes with the same attribute will break down simultaneously. ATR tries to construct RLs so that the nodes with the same attribute are safe in a single RL. ATR_o

constructs the RLSet in a way similar to HUB_o with the overlapping feature.

3) Degree-based algorithm: Degree-based construction algorithms select the nodes to be safe in order of their degree. We consider degree algorithms to be effective against network failures caused by intentional human attacks to the network. We consider two algorithms: DEC and INC, which select the safe nodes in decreasing and increasing order of the node degree. DEC and INC do not utilize the overlapping feature.

4) Random-based algorithm: Random-based construction algorithms randomly select the node to be safe. Therefore, they may be effective against random network failures, such as age-related degradations of network equipment. One of the advantages of these algorithms is their simplicity. Unlike HUB and ATR, they require neither the topology information nor the nodes' attributes; they only utilize the list of nodes in the network. We present the following three construction algorithms differentiating in terms of the policies of selecting safe nodes in each RL.

The filled random construction algorithm (RND_f) constructs an RLSet so that each RL makes as many nodes as possible safe. RND_f does not use the overlapping feature. Since RND_f can keep a small number of RLs in RLSet, it is suitable for networks in which memory usage is limited.

The uniform random construction algorithm (RND_u) constructs RLs so that the number of safe nodes in each RL is less than or equal to the threshold ($safe_{max}^{uni}$). This limitation controls the number of safe links in each RL, which affects the path length after recovering the failure. The overlapping feature is not used in RND_u. We consider that RND_u is suitable for networks in which the number of nodes that break down simultaneously is not large.

The overlapped random construction algorithm (RND_o) sets the number of RLs in RLSet to L_{OL_rnd} and the number of safe nodes in each RL to $safe_{max}^{OL\_rnd}$. We select the safe nodes in each RL to have the overlapping feature. RND_o is suitable for networks in which many nodes tend to break down simultaneously.

B. RL selection

When packets are routed according to the proposed methods, there are two ways to select an RL from RLSet. We summarize the details of each type of selection since they significantly affect the performance of our method.

1) Static RL selection: In static RL selection, when packets are generated at a source node, the source node selects an RL from RLSet according to the detected failed nodes and keeps using the RL until packets arrive at the destination node. The source node selects an RL in which all failed nodes are safe. When all of the failed nodes are safe in multiple RLs, the sender node selects one RL which has the smallest number of safe nodes. In this case, the proposed method can guarantee full network reachability. Conversely, when there is no RL in which all of the failed nodes are safe, the source node selects one RL which sets the largest number of failed nodes as safe. Note that, in this case, the proposed method cannot completely guarantee network reachability.

Static selection is suitable for low-latency networks since there is no need for the intermediate nodes to select an RL packet-by-packet.

2) Dynamic RL selection: The dynamic RL selection permits the intermediate nodes to change the RL to be used. In detail, when one of the intermediate nodes finds that it cannot forward a packet to the next-hop node due because it is using an inappropriate RL, the node will change the RL to be used so that the packet can be forwarded to the next-hop node. In general, this on-demand selection of an RL creates a routing loop by repeated changes of the RLs in some intermediate nodes. However, in the proposed method, we avoid the routing loop by forcing the node to use a new RL which has larger number of safe nodes than the current RL. The proposed method can forward the packet to the destination node unless RLSet are not exhausted.

This dynamic mechanism can increase the network reachability after recovery, even when there is no RL in the RLSet which makes all the failed nodes be safe.

TABLE I
EVALUATION PARAMETERS OF RLSET CONSTRUCTIONS

Parameter	Explanation	Value	
		CASE 1	CASE 2
L	Number of RLs (HUB)	259	10
L_{random}	Number of random RLs (HUB, HUB_o)	3	3
Deg_{min}	Minimum degree of hub node (HUB_o)	20	40
L_{hub}	Number of RLs (HUB_o)	38	2
A	Number of attributes (ATR, ATR_o)	4	2
L_{attr}	Number of RLs constructed from each attribute (ATR_o)	60	2
$safe_{max}^{uni}$	Upper bound of the number of safe nodes in each RL (RND_u)	26	26
L_{OL_rnd}	Number of RLs (RND_o)	2000	10
$safe_{max}^{OL\_rnd}$	Upper bound of the number of safe nodes in each RL (RND_o)	259	259

V. EVALUATION RESULTS

A. Evaluation method

To evaluate our proposed method, we utilize the AS-level network topology provided by CAIDA [17]. The topology data includes information about the relationships between ASes (transit or peering) in the current Internet. For simplicity, we extract the topology with ASes administrated by the Japan Network Information Center (JPNIC). Note that we merge the nodes which have only one link to the their adjacent node,

TABLE II
THE NUMBER OF RLS

RLSet Construction Algorithms	Number of RLs	
	CASE 1	CASE 2
Hub-based construction (HUB)	260	11
Overlapped HUB (HUB_o)	269	11
Attribute-based construction (ATR)	254	13
Overlapped ATR (ATR_o)	254	13
Filled random construction (RND_f)	7	7
Uniform random construction (RND_u)	12	12
Overlapped random construction (RND_o)	2000	10
Degree decreasing construction (DEC)	12	12
Degree increasing construction (INC)	11	12

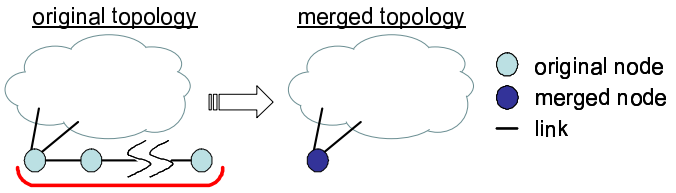


Fig. 2. Node merge on the AS-level network topology provided by CAIDA

as shown in Figure 2, because the nodes do not have any alternate path when the link is disconnected due to failures. As a result, the network topology consists of 259 nodes and 1162 links (84 peering links and 1078 transit links), and the average degree of the network nodes is 4.4. For taking ISPs' routing policies into account, we limit the usage of peering links in the IP routing as follows. Each peering link can be utilized only by two ASes which are interconnected by the peering link. In the proposed method, however, all ASes can utilize all peering links since it is operated by overlay routing.

We consider the following four types of network failures:

F_RND selects failed nodes randomly.

F_ADJ selects failed nodes so that the selected nodes are directly connected to each other.

F_ATR selects failed nodes with the same attributes. In this paper, we set the attribute of each network node as follows: we divide the network into two or four subnetworks with the minimum cut size, meaning that the number of links across the subnetworks becomes the minimum.

F_LNK selects some nodes and we assume that the links interconnecting the selected nodes become failures.

Table I summarizes the parameters of all RLSets construction algorithms described in Subsection IV-A and Table II shows the number of RLs in each RLSet. For the evaluation, we consider two cases, CASE 1 and CASE 2, to set the parameter values. For CASE 1, we assume that our method is applied to large-memory networks, so that the number of RLs in each RLSet is unlimited. For CASE 2, we assume that our method is applied to small-memory networks; that is, we limit the number of RLs in RLSet to a small value (approximately ten). We evaluate our method with static and dynamic RL selections for CASE 1, and dynamic RL selection for CASE 2.

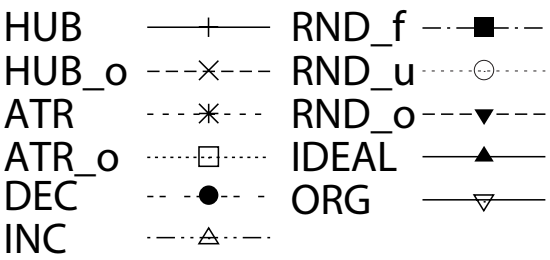


Fig. 3. Labels of each RLSet used in the following graphs

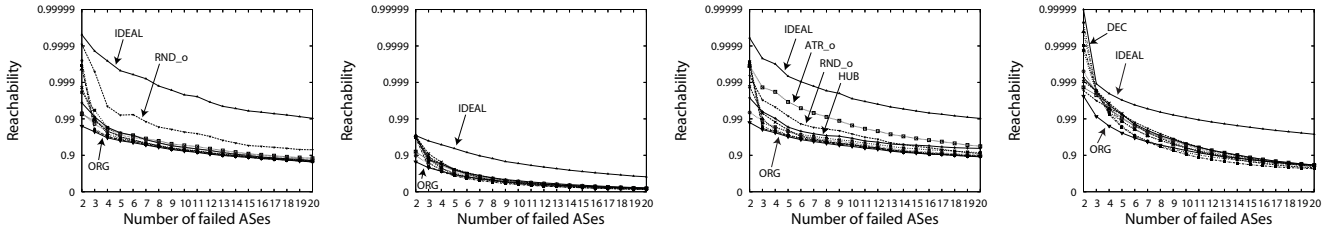
TABLE III
AVERAGE PATH LENGTH WITH STATIC RL SELECTION FOR CASE 1 AGAINST TWO-FAILURES

RLSet type	F_RND	F_ADJ	F_ATR	F_LNK
HUB	2.78	2.80	2.79	2.74
HUB_o	2.83	2.87	2.84	2.89
ATR	2.73	2.79	2.73	2.81
ATR_o	2.79	2.80	2.83	2.86
RND_f	2.78	2.88	2.78	2.90
RND_u	2.75	2.76	2.75	2.79
RND_o	2.99	2.99	2.99	2.98
DEC	2.72	2.79	2.71	2.79
INC	2.72	2.80	2.72	2.80
ORG	2.84	2.82	2.84	2.85
IDEAL	2.70	2.78	2.70	2.70

We evaluate the network reachability that represents the ratio of reachable node pairs after recovering from the failure, to all node pairs in the network except the failed nodes. We also evaluate the average path length between all reachable node pairs. In the evaluation results in the next subsection, we plot the results of two cases for comparison: ORG, which represents the results in the original topology without failure recovery, and IDEAL, which represents the results of the ideal case where we re-calculate the routing tables after failure detection. ORG and IDEAL provide the lower-limit and upper-limit of the network reachability. Figure 3 illustrates the labels of each RLSet used in the following graphs.

B. Results of static RL selection

Figure 4 shows the evaluation results of the network reachability as a function of the number of failed nodes with static RL selection for CASE 1. We observe that RND_o much improves the network reachability against all failure patterns, and it especially improves the network reachability after recovering the failures from 98% to 99.99% against F_RND (Figure 4(a)) when two nodes fails. This is because RND_o has the largest number of RLs and safe nodes in each RL among all RLSets. Against F_ATR (Figure 4(c)), ATR_o largely improves the network reachability, and the network reachability after recovering the failures is increased from 98%



(a) F_RND (b) F_ADJ (c) F_ATR (d) F_LNK

Fig. 4. Network reachability with static RL selection for CASE 1

to 99.9% when a two-node failure occurs. These results mean that the attribute-based RLSet construction algorithm works well when network failures according to the attribute occur. Against the failure pattern of F_LNK, the improvement of the network reachability is larger when we utilize degree-based methods (DEC and INC). This is because F_LNK causes failures of high-degree nodes and the degree-based methods are likely to make high-degree nodes safe in a single RL, which is effective against F_LNK.

Furthermore, for all algorithms, when the number of simultaneous failed nodes increases, network reachability degrades significantly. This represents the performance limitation of the static RL selection: we cannot find an appropriate RL in which all failed nodes are safe.

Table III summarizes the average path length with static RL selection for CASE 1 when the number of simultaneous failed nodes is 2, 10, and 20. These results show that the average path length of RND_o is the longest against all failure patterns. This is because the number of available links in each RL of RND_o is quite small since the number of safe nodes in the RLs is large. Comparing the reachability in Figure 4, we can conclude that when the number of safe nodes in each RL is large, the reachability improves as the average path length degrades. This is the trade-off relationship which can generally be found in proactive failure recovery mechanisms.

However, the other RLSets can reduce the average path length in comparison with the original topology. One reason for this is that the proposed method can fully utilize the peering links in the network, whereas the original IP routing can utilize peering links only when the source and destination of the traffic are ASes (nodes) which are interconnected by the link. These results clearly show the effectiveness of overlay routing for proactive failure recovery.

C. Results of dynamic RL selection

Figure 5 represents the changes in network reachability as a function of the number of failed nodes with dynamic RL selection for CASE 1. We can observe from this figure that the reachability of all RLSet construction algorithms is close to the ideal case (IDEAL) against F_RND (Figure 5(a)), F_ATR (Figure 5(c)), and F_LNK (Figure 5(d)). For example, against F_ATR (Figure 5(c)), each RLSet increases the network reachability after recovering the failures from 89% to 99%, even when 20 nodes go down simultaneously. However, against F_ADJ (Figure 5(b)), the degree of the reachability improvement is not so large, because F_ADJ tends to cause multiple simultaneous failures of hub nodes, and no RLSet construction algorithm makes two or more hub nodes safe in a single RL. We also note that by employing dynamic RL selection, all RLSet construction algorithms show similar performances. This represents the strong effect of dynamic RL selection, and so we presume that dynamic RL selection can result in high recovery performance even with a simple RLSet construction algorithm such as RND_o.

TABLE IV
AVERAGE PATH LENGTH WITH DYNAMIC RL SELECTION FOR CASE 2 AGAINST TWO-FAILURES

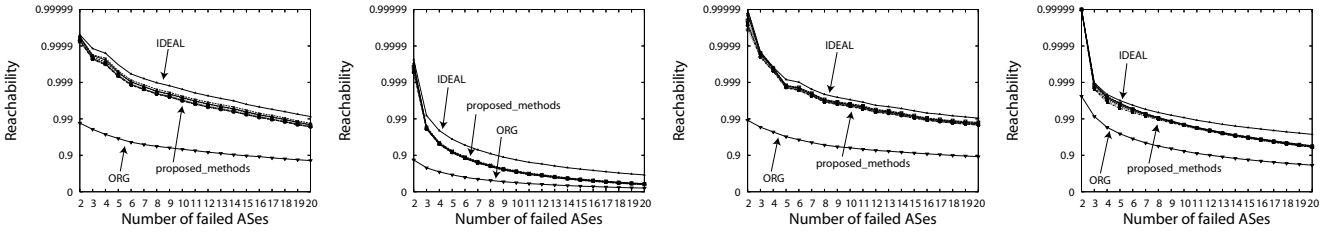
RLSet type	F_RND	F_ADJ	F_ATR	F_LNK
HUB	2.88	2.90	2.87	2.82
HUB_o	2.85	2.89	2.85	2.89
ATR	2.84	2.91	2.76	2.83
ATR_o	2.89	2.92	2.88	2.98
RND_f	2.77	2.92	2.86	2.86
RND_u	2.85	2.93	2.85	2.83
RND_o	3.10	2.96	2.96	2.95
DEC	2.74	2.92	2.74	2.80
INC	2.79	2.93	2.78	2.80
ORG	2.84	2.83	2.84	2.85
IDEAL	2.70	2.79	2.70	2.70

Figure 6 shows the corresponding results for CASE 2. This figure shows that the degree of reachability improvement is almost the same as that for CASE 1. This result means that when we employ dynamic RL selection, we can expect good performance with a small number of RLs in RLSet.

Table IV summarizes the average path length with dynamic RL selection for CASE 2 when the number of simultaneous multiple failures is 2, 10, and 20. These results shows that our method keeps the average path length sufficiently small, as in the case of static RL selection.

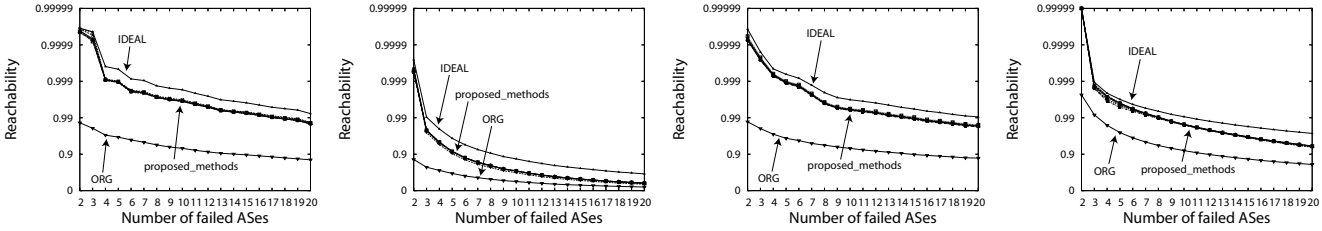
VI. CONCLUSION

This paper proposes a novel recovery mechanism from large-scale network failures. Our method, by utilizing proac-



(a) F_RND (b) F_ADJ (c) F_ATR (d) F_LNK

Fig. 5. Network reachability with dynamic RL selection for CASE 1



(a) F_RND (b) F_ADJ (c) F_ATR (d) F_LNK

Fig. 6. Network reachability with dynamic RL selection for CASE 2

tive network recovery mechanisms, takes advantage of overlay networking technologies. Through numerical evaluation results, we confirmed that our method can improve network reachability while keeping the average path length sufficiently small. Especially, by employing dynamic RL selection, we can provide almost the same level of network reachability as in the ideal case, even when we utilize a simple RLSet construction algorithm.

For future work, we plan to investigate the effectiveness of the RLSet construction algorithms, especially when multiple hub nodes break down simultaneously. We will further evaluate the effect of recalculation of RLSet when nodes or links are added or removed. The effect of the ratio of overlay nodes in the network (assumed 100% in this paper) and the load-balancing problem after failure recovery are also the interesting issues to be pursued.

ACKNOWLEDGMENT

This work was partly supported by "Special Coordination Funds for Promoting Science and Technology: *Yuragi Project*," Grant-in-Aid for Scientific Research on Priority Areas 18049050 in Japan.

REFERENCES

[1] A. Sahoo, K. Kant, and P. Mohapatra, "Characterization of BGP recovery time under large-scale failures," in *Proceedings of ICC 2006*, June 2006.

[2] Y. Rekhter and T. Li, "A border gateway protocol 4 (BGP–4)," *RFC 1771*, Mar. 1995.

[3] C. Labovitz, A. Ahuja, A. Abose, and F. Jahanian, "Delayed Internet routing convergence," in *Proceedings of ACM SIGCOMM 2000*, vol. 9, pp. 293–306, Aug. 2000.

[4] B. Zhang, D. massey, and L. Zhang, "Destination reachability and BGP convergence time," in *GLOBECOM 2004*, vol. 3, pp. 1383–1389, Apr. 2004.

[5] A. S. Tanenbaum, *COMPUTER NETWORKS*. Upper Saddle River, New Jersey 07458: Prentice-Hall International, Inc., third ed., 1996.

[6] C. Labovits, A. Ahuja, R. Wattenhofer, and S. Venkatachry, "The impact of Internet policy and topology on delayed routing convergence," in *Proceedings of INFOCOM 2001*, pp. 537–546, Dec. 2001.

[7] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz, "Route flap damping exacerbates Internet routing convergence," *ACM SIGCOMM Computer Communication Review*, vol. 32, pp. 221–233, Oct. 2002.

[8] D. Pei, M. Azuma, D. Massey, and L. Zhang, "BGP-RCN: Improving BGP convergence through root cause notification," Tech. Rep. CO80523-1873, UCLA CSD, Dec. 2004.

[9] W. Norton, "Internet service providers and peering." available at http://www.equinix.com/pdf/whitepapers/PeeringWP.2.pdf.

[10] W. Norton, "A business case for peering." available at http://www.equinix.com/pdf/whitepapers/Business_case.pdf.

[11] Y. Zhu, C. Dovrolis, and M. Ammar, "Dynamic overlay routing based on available bandwidth estimation: A simulation study," *Computer Networks Journal*, vol. 50, pp. 739–876, Apr. 2006.

[12] D. G. Andersen, A. C. Snoeren, and H. Balakrishnan, "Best-path vs. multi-path overlay routing," in *Proceedings of ACM SIGCOMM conference on Internet Measurement*, pp. 91–100, Oct. 2001.

[13] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proceedings of the 18th ACM Symposium on Operating Systems Principles*, Oct. 2001.

[14] Z. Xu, M. Mahalingam, and M. Karlsson, "Turning heterogeneity into an advantage in overlay routing," in *Proceedings of INFOCOM 2003*, vol. 2, pp. 1499–1509, Apr. 2003.

[15] A. Hansen, A. Kvalbein, T. Čičić, and S. Gjessing, "Resilient routing layers for network disaster planning," *Lecture notes in computer science*, vol. 3421, pp. 1097–1105, Apr. 2005.

[16] A. Hansen, A. Kvalbein, T. Čičić, S. Gjessing, and O. Lysne, "Resilient routing layers for recovery in packet networks," in *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, pp. 238–247, July 2005.

[17] The CAIDA Web Site. available at http://www.caida.org/home/.