Network recovery method with pre-calculation of routing configurations for large-scale failures

Go Hasegawa
Cybermedia Center
Osaka University
1-32, Machikaneyama-cho, Toyonaka, Osaka 560-0043, JAPAN
Email: hasegawa@cmc.osaka-u.ac.jp
Takuro Horie and Masayuki Murata
Graduate School of Infomation Science and Technology
Osaka University
1-3, Yamadaoka, Suita, Osaka 560-0871, JAPAN
Email: {t-horie, murata}@ist.osaka-u.ac.jp

Abstract

This paper proposes a novel recovery method 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 for multiple network failures occurring simultaneously and selects the appropriate 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 87% to 98%, while keeping the path length sufficiently short when as many as 8% of the nodes in a network are down simultaneously. We also show that our method does not require frequent routing configuration calculations when new nodes and links join the network.

Index Terms

Overlay network, Routing, Large-scale network failures, Proactive failure recovery

I. Introduction

Computer networks are now regarded as essential infrastructure, much like water and gas utilities. Therefore, recovering from network failures and ensuring network connectivity are 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 a failure, the network recovers from the failure by replacing the faulty equipment with functioning equipment. Therefore, existing research on network recovery focuses primarily on the trade-off relationships between cost and performance, and concludes that the best solution is adding higher-level redundancy to network equipment that has a higher probability of failing.

However, this traditional approach cannot be applied to the recovery methods 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 preparations against such failures is very expensive. Consequently, most existing protection and recovery methods assume a single-failure model, that is, only one failure occurs at one time, and there have been very few studies on methods that would protect against large-scale network failures in which many network elements go down simultaneously.

Recent investigations on IP networks have revealed that the Border Gateway Protocol (BGP) (1), 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 for large-scale failures or certain types of network topologies (2; 3). Essentially, the routing convergence time in BGP has no theoretical upper bound, and there are many situations in which the routing convergence time increases significantly, as in the count-to-infinity problem (4). Various methods to improve the routing convergence time in BGP have been proposed (5; 6; 7). However, most of them require modifications to BGP and/or IP protocols, which means 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 of varying monetary cost structures, such as peering and transit relationships (8; 9). BGP routing configurations are very much affected by each ISP's policies, which are driven by the cost structure of the links. This means that current BGP routing is not configured to maximize user-perceived performance, such as end-to-end delay, throughput, and network connectivity (10; 11). We believe that the BGP configuration affects network performance, especially network connectivity, under large-scale failures.

In this paper, we propose a novel recovery method for large-scale network failures. Our method is based on a proactive recovery scheme that pre-calculates multiple routing configurations to protect against possible network failures and shares the configurations throughout the network. When failures are detected, our scheme immediately selects one of the configurations, depending on the detected failures. By taking advantage of proactive network recovery methods, our method can work quickly, even when BGP requires a long time to recover the network reachability or cannot completely recover from the failure. Moreover, we propose various algorithms to calculate multiple routing configurations to accommodate large-scale failures in a network. In addition, our method adapts overlay network techniques. The primary reasons we utilize the overlay network approach are as follows: we can deploy the proposed method easily and quickly because it does not require a standardization process, and the application-level traffic routing implemented by overlay routing can overcome the shortcomings in policy-based BGP routing.

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 network reachability significantly in cases of single node (AS) failure and simultaneous multiple node failures. Furthermore, we show that our method can sustain an average path length after recovery that is almost equal to the ideal value, and it does not require frequent recalculation of routing configurations when new nodes and links join the network.

The remainder of this paper is organized as follows. In Section II, we introduce the research background on overlay networks, overlay routing, and network recovery methods. In Section III, we give a brief explanation of the recovery method that is the basis of our method. In Section IV, we present design issues and detailed algorithms. 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

In this paper, overlay networks are defined as upper-layer networks built on the lower-layer IP network. These overlay networks provide special-purpose application services such as file sharing, grids, 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* (12; 13), which we exploit for the proposed method in this paper.

B. Recovery from large-scale network failures

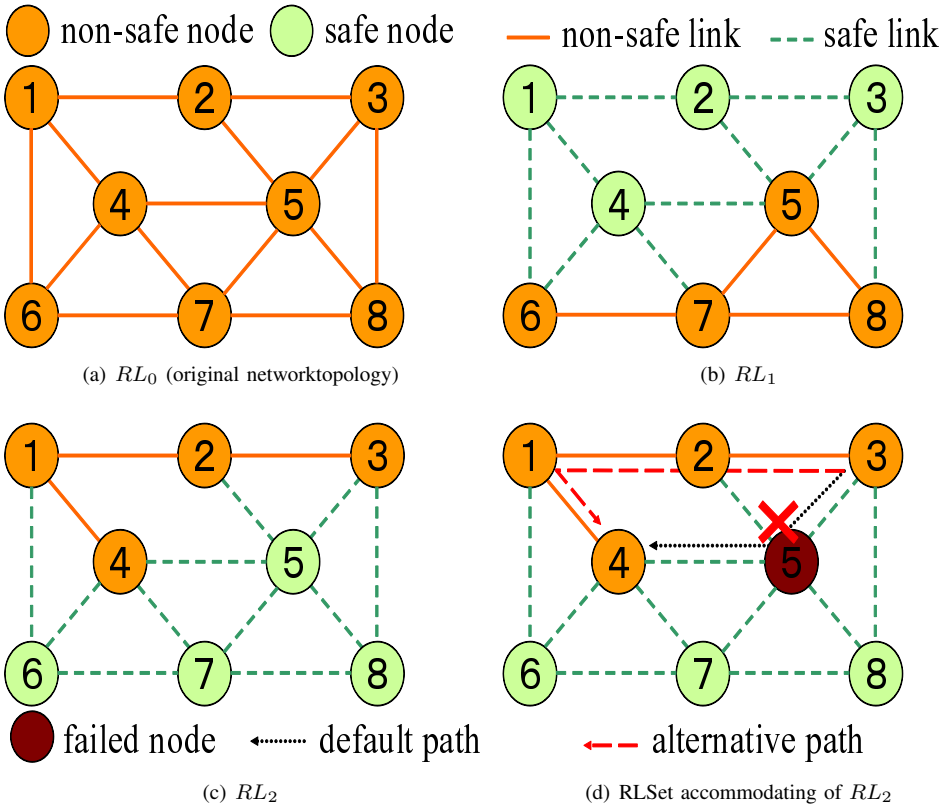
As in water and gas utilities, information networks are vulnerable to large-scale failures when disasters, such as earthquakes, terrorist attacks, and large-area power outages, occur. In addition, software bugs in major router operating systems may result in the simultaneous breakdown of many network 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 including, for example, 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 instead of multiple failures occurring simultaneously at any particular time. In general, the methods 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 methods for large-scale failures is the 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 methods. Thus, an effective, low-cost solution is necessary to deal with large-scale network failure in IP networks.

C. Reactive and proactive recovery methods

In general, network recovery methods are categorized into two types: reactive and proactive. In reactive recovery methods, when network nodes detect network failures, they recalculate their 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 methods is the considerable time required for routing convergence after the failures, since new routing information is generally propagated in a hop-by-hop manner.

In contrast, proactive recovery methods pre-calculate recovery settings (e.g., routing tables) by assuming possible failures and distribute the settings throughout the network. Then, when a network failure is detected, the recovery method immediately selects one of the pre-calculated settings according to the detected failure. If the failure is covered by the pre-calculated settings,



(a) RL_0 (original networktopology) (b) RL_1

(c) RL_2 (d) RLSet accommodating of RL_2

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

proactive recovery does not require routing convergence time after the failure. However, if the failure has not been considered in the pre-calculation, the recovery method cannot completely recover. So, in the proactive method, 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 method. Especially, we focus on Resilient Routing Layers (RRL), proposed in (14), because RRL is simple and has 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 adaptability in accommodating large-scale failures.

III. Resilient Routing Layers (RRL) (14)

A. Overview

RRL pre-calculates multiple network topologies and routing tables, which are called Routing Layers (RLs), from the original network topology. In each RL, RRL assumes that a failure of the network node(s) will 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 do occur. RRL utilizes the original network topology as long as no failure occurs.

We refer to the node that is assumed to be down in each RL as a *safe node* and the calculated RLs as a *Routing Layer Set* (RLSet). With the exception of the original network topology, each RL 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 use as 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 an RL in which the failed node is safe. Once the adjacent node selects an appropriate RL from the RLSet, 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. In RL_2 in Figure 1(c), nodes 5, 6, 7, and 8 are safe. That is, every node in the network is safe in at least one RL in RLSet. Note that the weight of the dashed links in Figure 1 is set to the maximum value, since they are safe links that connect to safe nodes.

Using this figure, let us 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).

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 (14), the authors show the following evaluation results: as few as 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 number of safe links also increases, which increases the path length (hop count) between nodes in the RL.

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 can accommodate a larger number of failure patterns. However, increasing the number of RLs in RLSet increases memory usage and processing overhead. Therefore, for RRL to realize high recovery performance against large-scale network failures, we must carefully configure the number of RLs in RLSet, the number of safe nodes in each RL, as well as the selection of nodes as safe in each RL. However, to the best of our knowledge, no other research results have been reported on RRL-based proactive recovery methods for large-scale network failures.

C. Implementation issues

In (14; 15), the authors note that RRL can be implemented at various layers. In (15), they show an example of RRL running in a Multiprotocol Label Switching (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 implementation at the MPLS or IP layer is the required standardization process. The other problem is that RRL must be implemented for all network 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 think the advantages of overlay routing, which are summarized in Subsection II-A, outweigh the disadvantages.

IV. Proposed Method

A. RLSet construction

As described in Section III, no efficient method has been proposed to recover from simultaneous multiple failures by an RRL-based recovery mechanism despite its potential. To accommodate 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 to be safe in each RL. In the following, we present various construction algorithms of RLSet. In each construction algorithm, we assume that patterns of simultaneous multiple node failures occur, and the proposed algorithm enables recovery from all failure patterns.

In all the construction algorithms presented, we select node i to be safe from all nodes in the original network topology. This safe node satisfies the following three conditions:

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

Note that in all the 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 pseudo code of HUB is shown in Algorithm 1.

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 multiple 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 using this overlapping feature, we can expect improvement of the recovery performance when the number of RLs in RLSet increases. Algorithm 2 describes the pseudo code of HUB_o.

Algorithm 1 Hub-based construction

for $i = 1$ to $L_{hub} - L_{hub}^{rnd}$ **do**
  Make node n_{hub} which has the i-th largest degree be safe in RL_i
  for all nodes n connecting to n_{hub} **do**
    if n does not have the largest degree in adjacent nodes to n_{hub} **then**
      Make node n be safe in RL_i
    end if
  end for
end for
$i \leftarrow L_{hub} - L_{hub}^{rnd} + 1$
repeat
  for all nodes n does not become safe in any RL_k $s.t.$ $1 \leq k < i$ **do**
    Make node n be safe in RL_i
  end for
  $i \leftarrow i + 1$
until $i \leq L_{hub}$, or there exists a node which is not safe in any RL_k $s.t.$ $1 \leq k < i$

Algorithm 2 Overlapped hub-based construction

for all nodes n_{hub} that its degree $> Deg_{min}$ **do**
  for $j = 1$ to L_{hub_o} **do**
    Make n_{hub} which has the i-th largest degree be safe in RL_{ij}
    for all nodes n that is adjacent node to n_{hub} and does not become safe in any RL_{ik} $s.t.$ $1 \leq k < j$ **do**
      Make node n be safe in RL_{ij}
    end for
    for all nodes n that is adjacent node to n_{hub} and becomes safe in some of RL_{ik} $s.t.$ $1 \leq k < j$ **do**
      Make node n be safe in RL_{ij}
    end for
  end for
end for
$i \leftarrow 1$
repeat
  for all nodes n which is selected at random and does not become safe in any RL **do**
    Make node n be safe in RL_{random_i}
  end for
  $i \leftarrow i + 1$
until $i \leq L_{hub}^{rnd}$, or there exists a node which is not safe in any RL

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, or topological information. We then assume that 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 the same RL. As in HUB_o, ATR_o constructs the RLSet with the overlapping feature. The pseudo codes of ATR and ATR_o can be found in Algorithm 3.

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. This algorithm is simpler than the above two algorithms, so we consider DEC and INC for comparative purposes.

4) Random-based algorithm: The random-based construction algorithm (RND) randomly selects the node to be safe. Therefore, it may be effective against random network failures, such as age-related degradations of network equipment. One of the advantages of the algorithm is its simplicity. RND sets the number of RLs in RLSet as L_{rnd} and the number of safe nodes in each RL as $safe_{max}$. We select the safe nodes in each RL to have the overlapping feature. RND is suitable for networks in which many nodes tend to break down simultaneously.

B. RL selection

When packets are routed according to the proposed algorithms, there are two ways to select an RL from RLSet: static RL selection and dynamic RL selection. We summarize the details of each type of selection, since they significantly affect the

Algorithm 3 Attribute-based construction

```
for all a ∈ A do
  for i = 1 to L_atr do
    for all nodes n that its attribute is a and it does not become safe in any RL_ai do
      Make node n be safe in RL_ai
    end for
    if ATR_o is constructed then
      for all nodes n that its attribute is a and it become safe in some of RL_ai do
        Make node n be safe in RL_ai
      end for
    end if
  end for
end for
```

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. In detail, the source node selects an RL in which all failed nodes are safe. When more than one RL are found, the source node selects the one that 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 the RL that sets the largest number of failed nodes as safe. In this case, the proposed method cannot completely guarantee network reachability.

Obviously, static RL selection is simpler than dynamic RL selection, described below, since there is no need for the intermediate nodes to select an RL packet-by-packet.

2) Dynamic RL selection: 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 because it is using an inappropriate RL, the node changes the RL to be used so that the packet can be forwarded to the next-hop node. In general, such 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 that has a larger number of safe nodes than the current RL. Thus, the proposed method can forward the packet to the destination node unless RLSet has no other suitable RLs.

Compared with static selection, dynamic selection can increase the network reachability after recovery, even when there is no RL in RLSet that makes all the failed nodes safe.

C. Adding new network nodes

When a new network node joins the network, the proposed method adds it to each RL. The new node becomes safe in an RL when the three conditions described in Subsection IV-A are satisfied. After some nodes are added to the network, however, the proposed method may not recover the failures even when there is an RL in which all the failed nodes are safe. For example, when using an RL in which all nodes connecting to the newly added nodes are safe, the selected RL cannot recover the reachability from/to the added nodes. Therefore, the recalculation of RLSet at some intervals is required to maintain the performance of the proposed method when the network grows. We evaluate the performance of the proposed method with the above situation in Subsection V-E.

V. Evaluation Results

A. Evaluation method

To evaluate our proposed method, we utilize the AS-level network topology obtained from CAIDA (16). The topology data includes information about the relationships between ASes (transit or peering). 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 their adjacent node, as shown in Figure 2, because these nodes do not have any alternate path when the link is disconnected due to failures; this issue is not within the scope of this work. As a result, the CAIDA network topology consists of 259 nodes and 1162 links (84 peering links and 1078 transit links), and the average degree of network nodes is 4.4. To consider the ISPs' routing policies, we limit the usage of peering links for 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 this method is applied by overlay routing.

Table I summarizes the parameters of all the RLSet construction algorithms described in Subsection IV-A, and Table II lists the number of RLs in each RLSet. For the evaluation, we consider three cases of parameter values. For CASE 1, we assume that the network nodes have sufficient memory for storing routing configurations, and the number of RLs in each RLSet is set to be quite large. For CASE 2, we assume that the network nodes do not have enough memory and the number of RLs in

TABLE I
EVALUATION PARAMETERS OF RLSET CONSTRUCTIONS

Parameter	Explanation	Value		
		CASE1	CASE2	CASE3
L_{hub}	Number of RLs (HUB)	259	10	10
L_{hub}^{rnd}	Number of random RLs (HUB, HUB_o)	3	3	3
Deg_{min}	Minimum degree of hub node (HUB_o)	20	40	40
L_{hub_o}	Number of RLs (HUB_o)	38	2	2
A	Number of attributes (ATR, ATR_o)	4	2	2
L_{atr}	Number of RLs constructed from each attribute (ATR_o)	60	2	2
L_{rnd}	Upper bound of the number of RLs (RND)	2000	10	20
$safe_{max}$	Upper bound of the number of safe nodes in each RL (RND)	259	259	20

TABLE II
THE NUMBER OF RLS

RLSet	Number of RLs		
	CASE 1	CASE 2	CASE 3
HUB	260	11	17
HUB_o	269	11	16
ATR	254	13	21
ATR_o	254	13	21
RND	2000	10	20
DEC	12	12	22
INC	11	12	22

RLSet is small (approximately ten). We evaluate our method with static and dynamic RL selections for CASE 1, and dynamic RL selection for CASE 2. CASE 3 is utilized to evaluate the performance of the proposed method when new nodes and links join the network.

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, that is, the number of links across the subnetworks becomes the minimum.

F_LNK selects some nodes and we assume that failed links interconnect the selected nodes.

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. We further investigate the traffic ratio, which is the ratio of the traffic amount on a link in the network after failure recovery to that before failure recovery. Note that the amount of traffic of a link is defined as the number of node pairs whose route passes through the link.

In the evaluation results given in the following subsections, we plot the results of two extreme cases for comparison: ORG, which represents the results in the original topology without applying any failure recovery mechanisms, and IDEAL, which represents the results of the ideal case where we recalculate the routing tables after failure detection. ORG and IDEAL provide the lower limit and upper limit of the network reachability, respectively. Figure 3 illustrates the labels of each RLSet construction algorithm used in the following graphs.

B. Network reachability

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 much improves the network reachability against all failure patterns. For example, the network reachability after recovering from the failures increases from 98% to 99.99% against F_RND (Figure 4(a)) when two nodes fail. This is because RND has the largest number of RLs and safe nodes in each RL among all RLSets, as shown in Table II. Against F_ATR (Figure 4(c)), ATR_o largely improves the network reachability, and the network reachability after

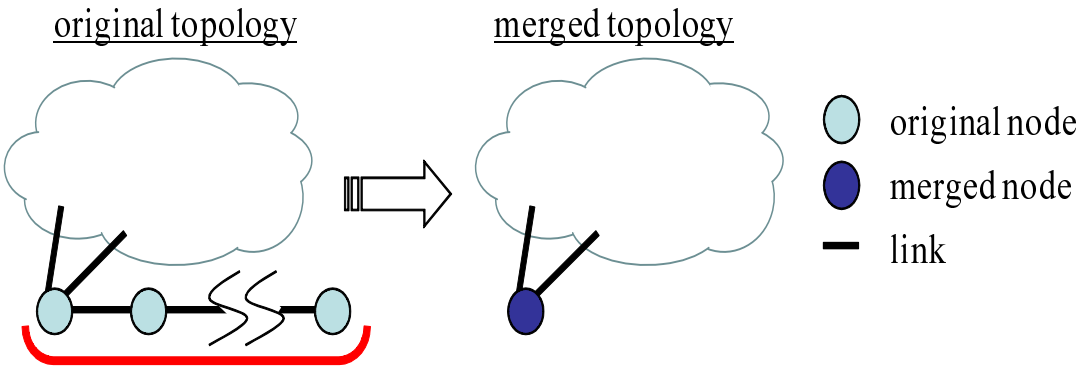


Fig. 2. Node merge on the AS-level network topology (obtained from CAIDA)

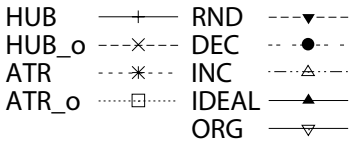


Fig. 3. Types of data lines for the RLSet construction algorithm used in the following graphs

recovering the failures increases from 98% to 99.9% when two node failures occur. These results mean that the attribute-based RLSet construction algorithm works well when network failures according to an attribute occur. The improvement of the network reachability against the failure pattern of F_LNK is larger when we utilize the degree-based algorithms (DEC and INC). This is because F_LNK causes failures of high-degree nodes and the degree-based algorithms are likely to make high-degree nodes safe in the same RL, which is effective against F_LNK. Furthermore, for all algorithms, when the number of simultaneously failed nodes increases, network reachability degrades significantly. This represents the performance limitation of static RL selection: we cannot find an appropriate RL in which all failed nodes are safe.

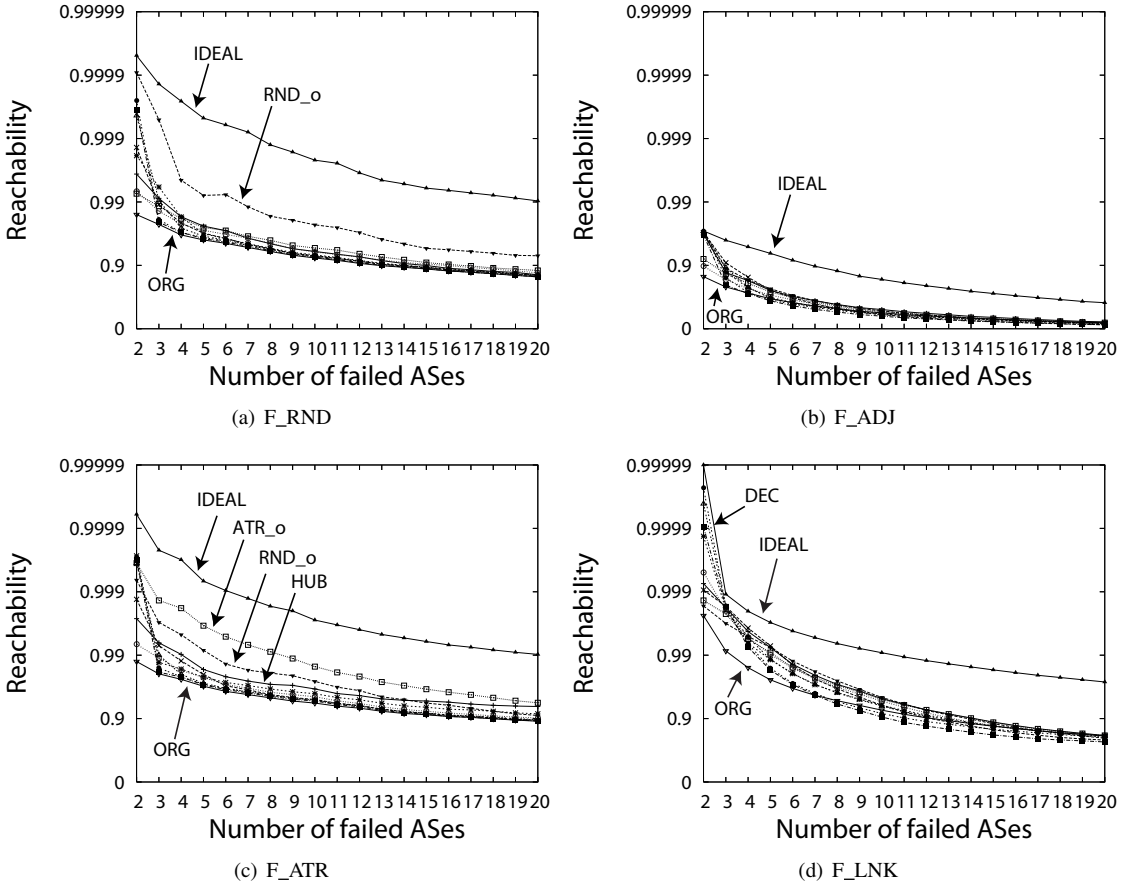
Figure 5 represents the changes in network reachability as a function of the number of failed nodes with dynamic RL selection for CASE 1. We can observe from this figure that the reachability of all RLSet construction algorithms is greatly improved compared with that with static RL selection, and it 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 from 89% to 99%, even when 20 nodes go down simultaneously. However, against F_ADJ (Figure 5(b)), the degree of reachability improvement is not as 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 the same RL. We also note that by employing dynamic RL selection, all RLSet construction algorithms show a similar performance. This represents the strong effect of dynamic RL selection. These results indicate that dynamic RL selection can result in high recovery performance, even with a simple RLSet construction algorithm such as RND.

Figure 6 shows the corresponding results for CASE 2 with dynamic RL selection. 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 a dynamic RL selection, we can expect good performance with a small number of RLs in RLSet.

C. Average path length

Table III summarizes the average path length with static RL selection for CASE 1 under two node failures. These results show that the average path length of RND is the longest for all failure patterns. This is because the number of available links in each RL of RND is quite small since the number of safe nodes in the RLs is large. Combining this with the results of 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 that can generally be found in proactive failure recovery methods. However, the other RLSets can reduce the average path length in comparison with the original topology, especially because 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 the ASes (nodes) interconnected by the link. These results clearly show the effectiveness of overlay routing for proactive failure recovery.

Table IV summarizes the corresponding results to Table III with dynamic RL selection. These results show that our method keeps the average path length sufficiently small, as in the case with static RL selection.



(a) F_RND (b) F_ADJ

(c) F_ATR (d) F_LNK

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

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

RLSet	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	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

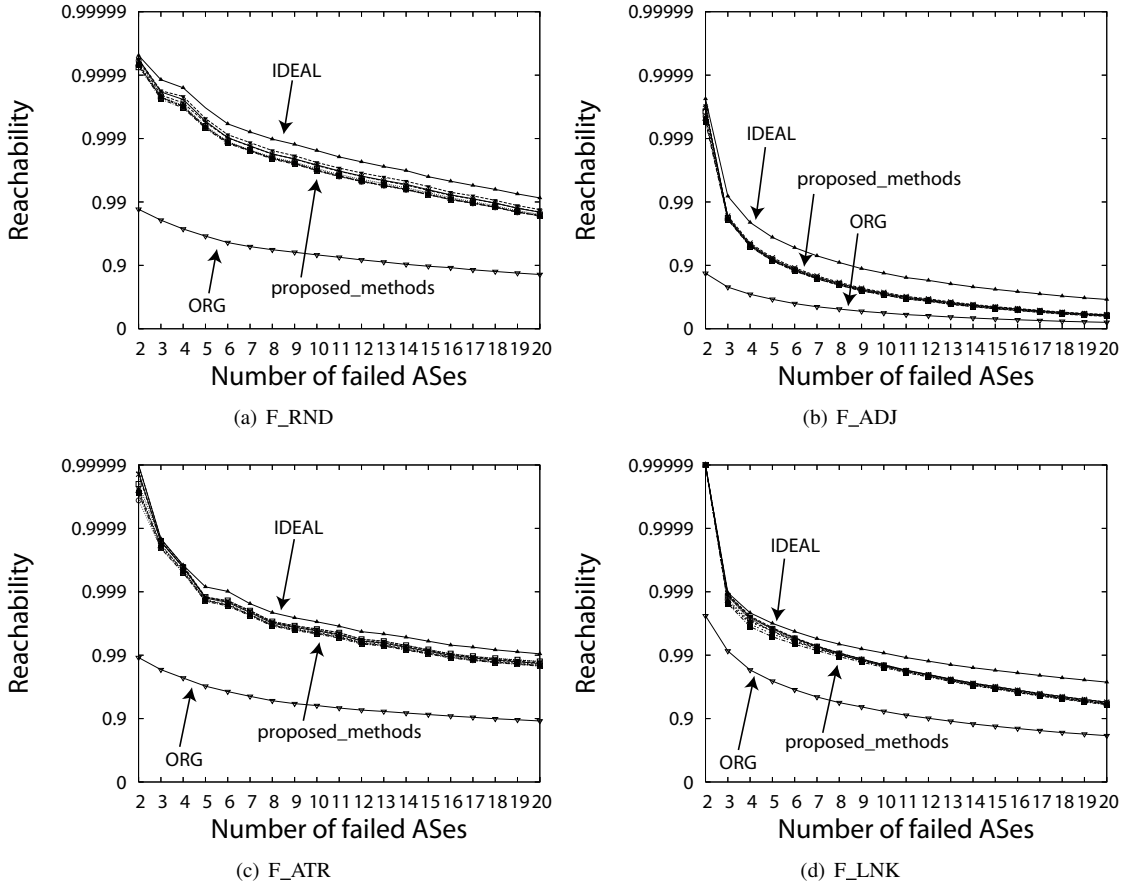
D. Traffic ratio

Figure 7 illustrates the distribution of the traffic ratio of each link in the network with static RL selection for CASE 1 against two node failures. We see that some of the links in the network suffer from the great increase of traffic after failure recovery. This may be because of the large network change by link cutting in a selected RL and the use of peering links. We also find that RND and HUB_o have a larger increase of the traffic ratio on specific links in comparison with other algorithms. This is because these algorithms isolate more nodes and their adjacent links in each RL and therefore decrease the number of available links in the selected RL's network topology.

Figure 8, which shows dynamic RL selection, illustrates the corresponding results to Figure 7. We can observe that dynamic RL selection gives a smaller increase of the traffic ratio compared to that with static RL selection because, by utilizing hop-by-hop RL selection mechanisms, the routes of node pairs are distributed throughout the network.

E. Performance with network growth

Finally, we investigate the performance of the proposed method when new nodes and links join the network. For the initial network topology, we use the network topology generated according to the BA model (17). We start with a network topology



(a) F_RND (b) F_ADJ

(c) F_ATR (d) F_LNK

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

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

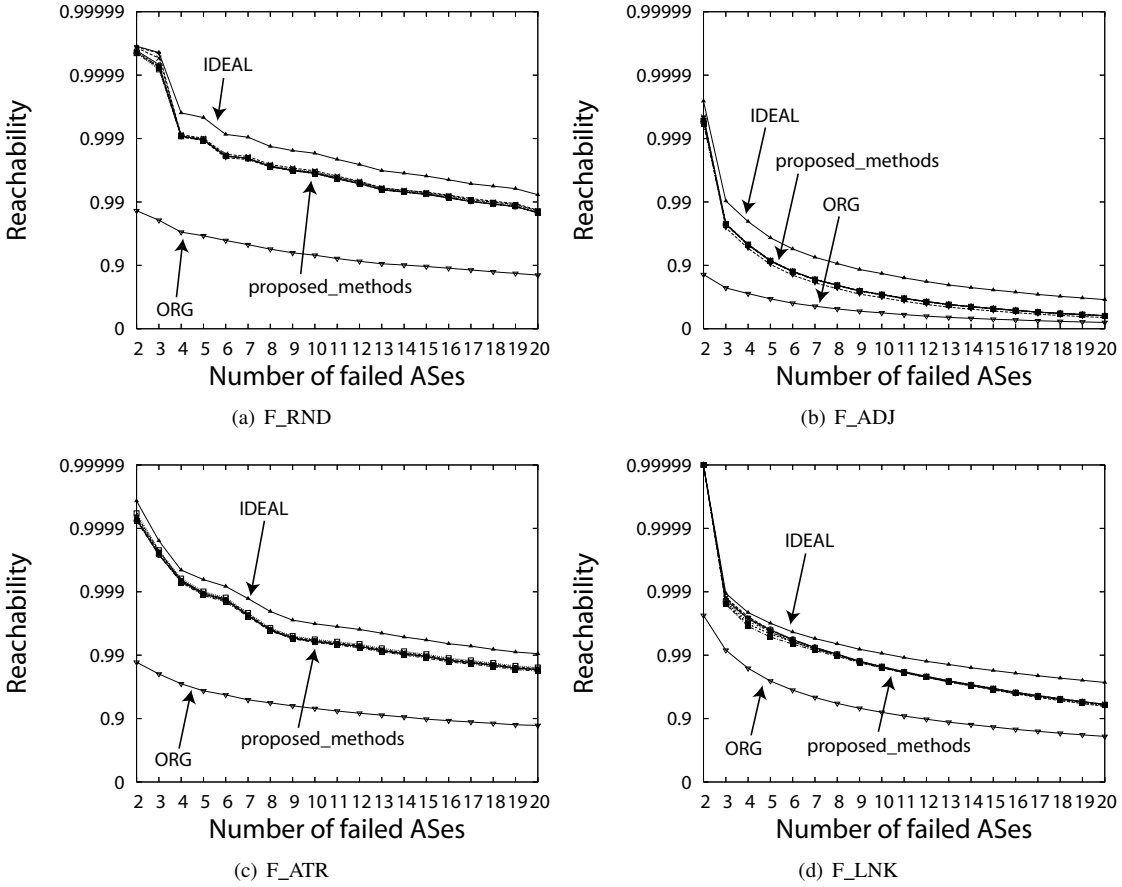
RLSet	F_RND	F_ADJ	F_ATR	F_LNK
HUB	2.88	2.90	2.87	2.82
HUB_o	2.85	2.89	2.85	2.89
ATR	2.84	2.91	2.76	2.83
ATR_o	2.89	2.92	2.88	2.98
RND	3.10	2.96	2.96	2.95
DEC	2.74	2.92	2.74	2.80
INC	2.79	2.93	2.78	2.80
ORG	2.84	2.83	2.84	2.85
IDEAL	2.70	2.79	2.70	2.70

of 259 nodes and 1030 links, and add a new node with four links to the network in a one-by-one manner until the network has 359 nodes and 1430 links. For constructing RLSet, we use the parameters for CASE 3 listed in Table I.

Figure 9 represents the network reachability as a function of the number of added nodes with static RL selection against two node failures when we use HUB and ATR. Note that the other construction algorithms have a similar performance. The label *plain* is the case when we do not recalculate the RLSet, and *recal* is the case when we recalculate the RLSet every time a new node joins the network. From the figure, the recalculation of RLSet does not affect network reachability. This is because static RL selection has lower reachability compared with that for dynamic RL selection, so the problem described in Subsection IV-C does not differentiate the performance of recalculation.

Figure 10 shows the results corresponding to Figure 9 when we use dynamic RL selection. From the figure, we find that the network reachability without recalculation decreases when more than ten nodes join the network. This means that with dynamic RL selection, we should recalculate the RLSet when ten nodes join the network in order to maintain the performance of the proposed method.

Figure 11 shows the average path length as a function of the number of added nodes with static RL selection against two node failures. The results show the average path length of the recalculated RLSets is smaller than that of the non-recalculated



(a) F_RND (b) F_ADJ

(c) F_ATR (d) F_LNK

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

RLSets, especially when the number of added nodes is larger than ten. This means that the recalculation of RLSets affects path lengths significantly. Moreover, we can observe the average path length increases as the number of added nodes increases in both cases. This is caused by enlargement of the topology diameter.

Figure 12 shows the corresponding results to Figure 11 when we use dynamic RL selection. In contrast to static RL selection, the effect of recalculation is negligible because, by using dynamic RL selection, the routes for node pairs become quite different from the shortest path. Furthermore, we find that the path lengths with the dynamic RL selection are shorter than that with the static RL selection. This is because that when there is no RL in which all failed nodes are safe, the dynamic RL selection utilizes the RL that has many available links for packet forwarding in spite that the static RL selection utilizes the RL with few available links.

From these results, we conclude that we should recalculate the RLSet when approximately ten nodes join the network to maintain both high reachability and short path length.

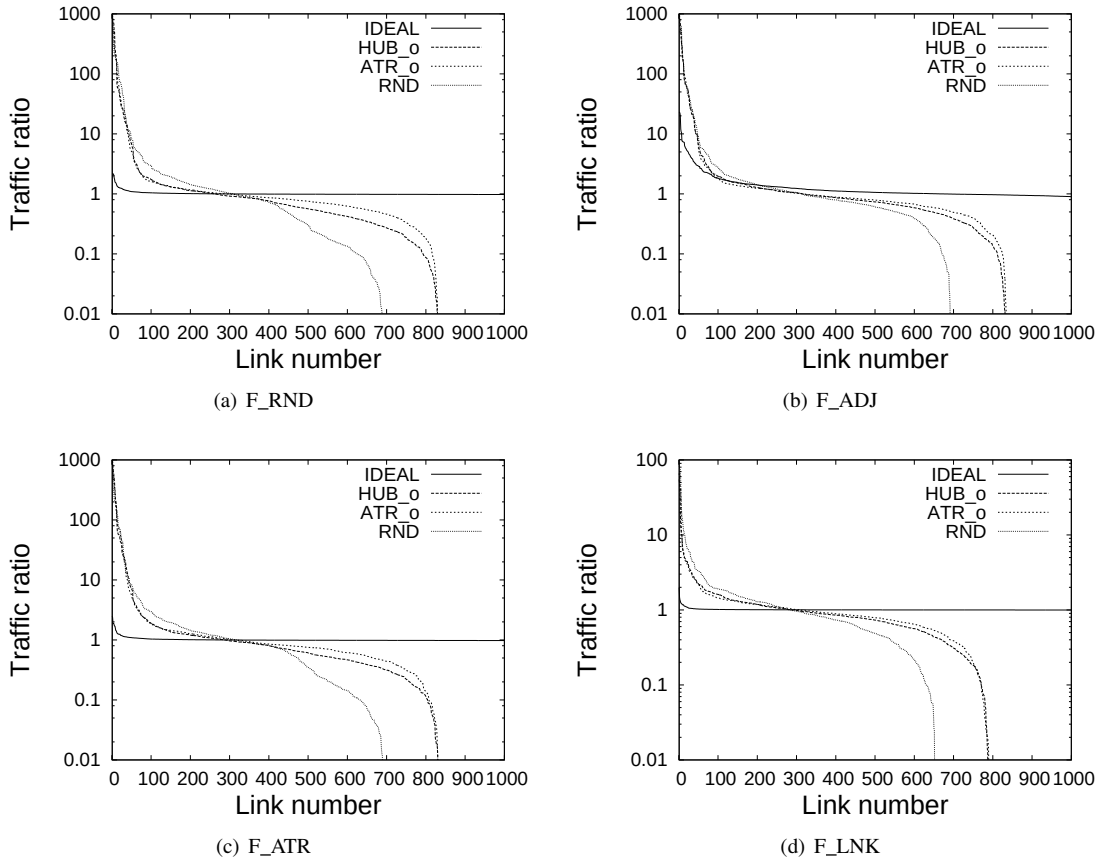
VI. Conclusion

This paper proposes a novel recovery method from large-scale network failures. By taking advantage of proactive network recovery methods and overlay networking technologies, our method constructs routing configurations to accommodate possible large-scale network failures. 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. We also plan to consider RLSet construction algorithms to decrease the traffic ratio increase mentioned in Subsection V-D while maintaining recovery performance.

For future work, we plan to investigate the effectiveness of the RLSet construction algorithms, especially when multiple hub nodes break down simultaneously.

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.



(a) F_RND (b) F_ADJ

(c) F_ATR (d) F_LNK

Fig. 7. Traffic ratio with static RL selection for CASE 1 against two node failures

REFERENCES

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

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

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

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

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

[6] 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.

[7] 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.

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

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

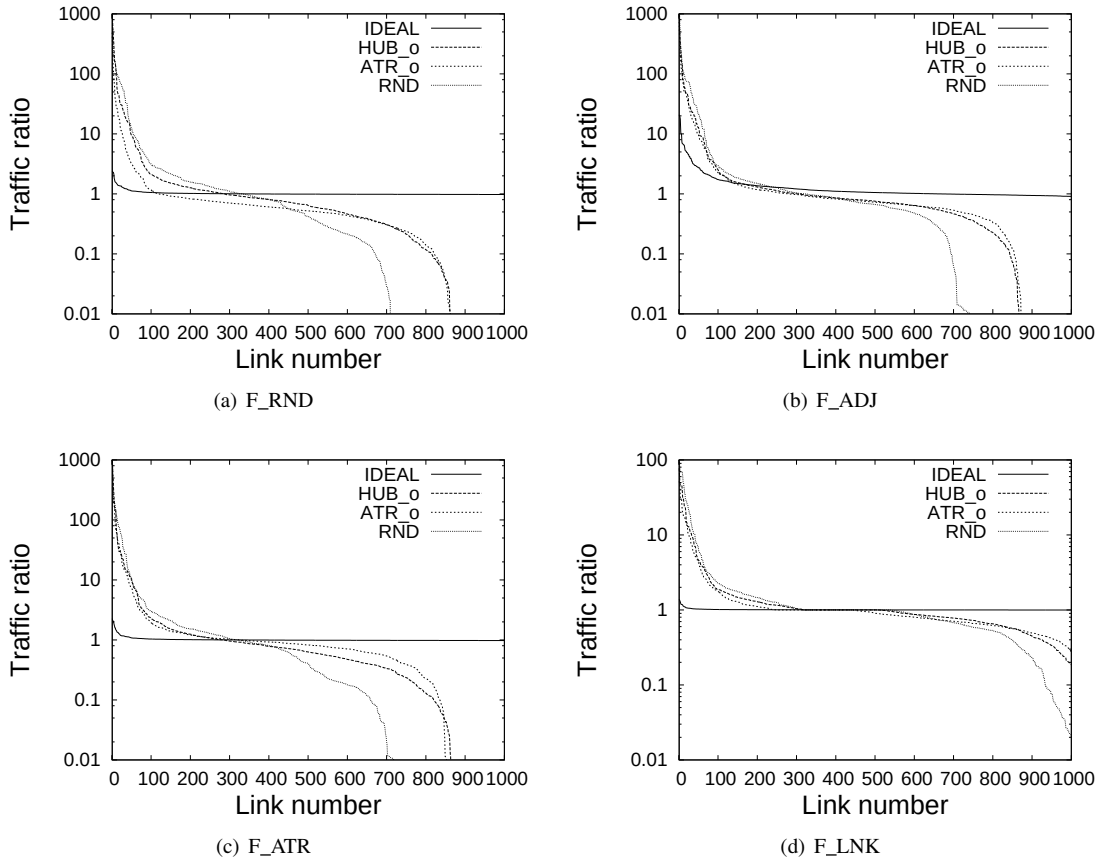
[10] Y. Zhu, C. Dovrolis, and M. Ammar, "Dynamic overlay routing based on available bandwidth estimation: A simulation study," *Computer Networks Journal*, vol. 50, pp. 739–876, Apr. 2006.

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

[12] 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.

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

[14] 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.



(a) F_RND

(b) F_ADJ

(c) F_ATR

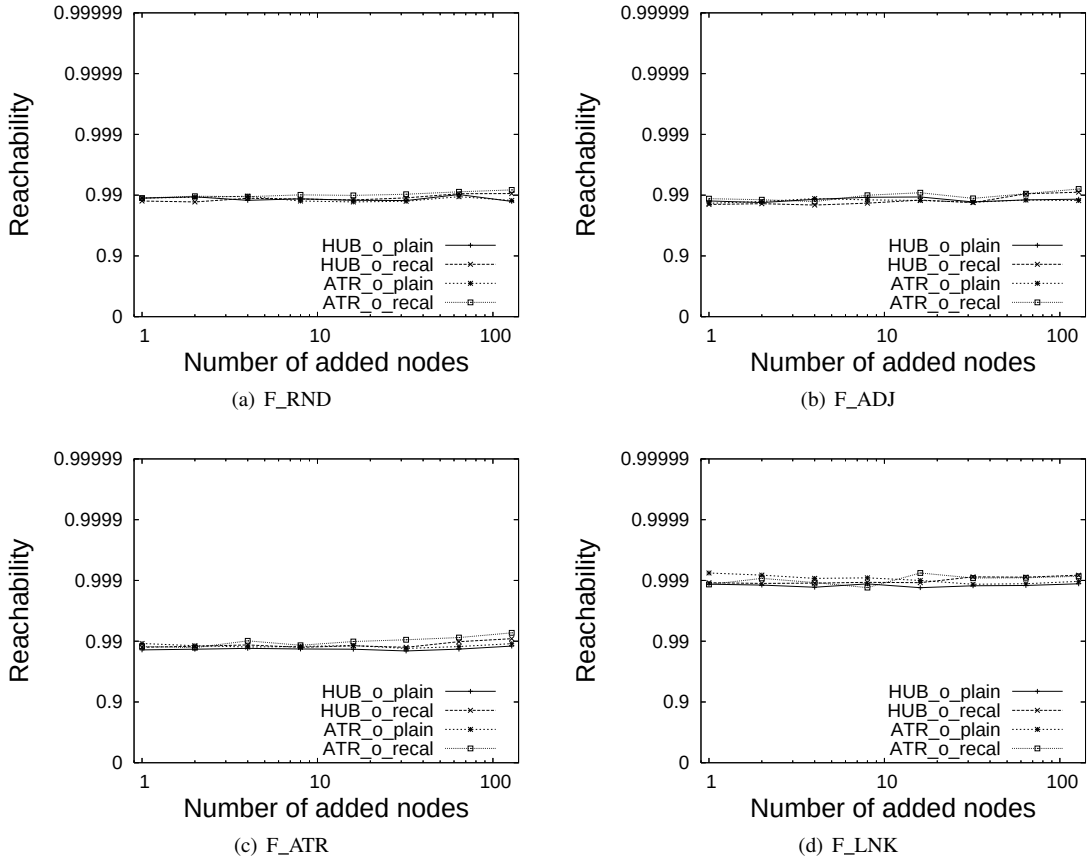
(d) F_LNK

Fig. 8. Traffic ratio with dynamic RL selection for CASE 1 against two node failures

[15] 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.

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

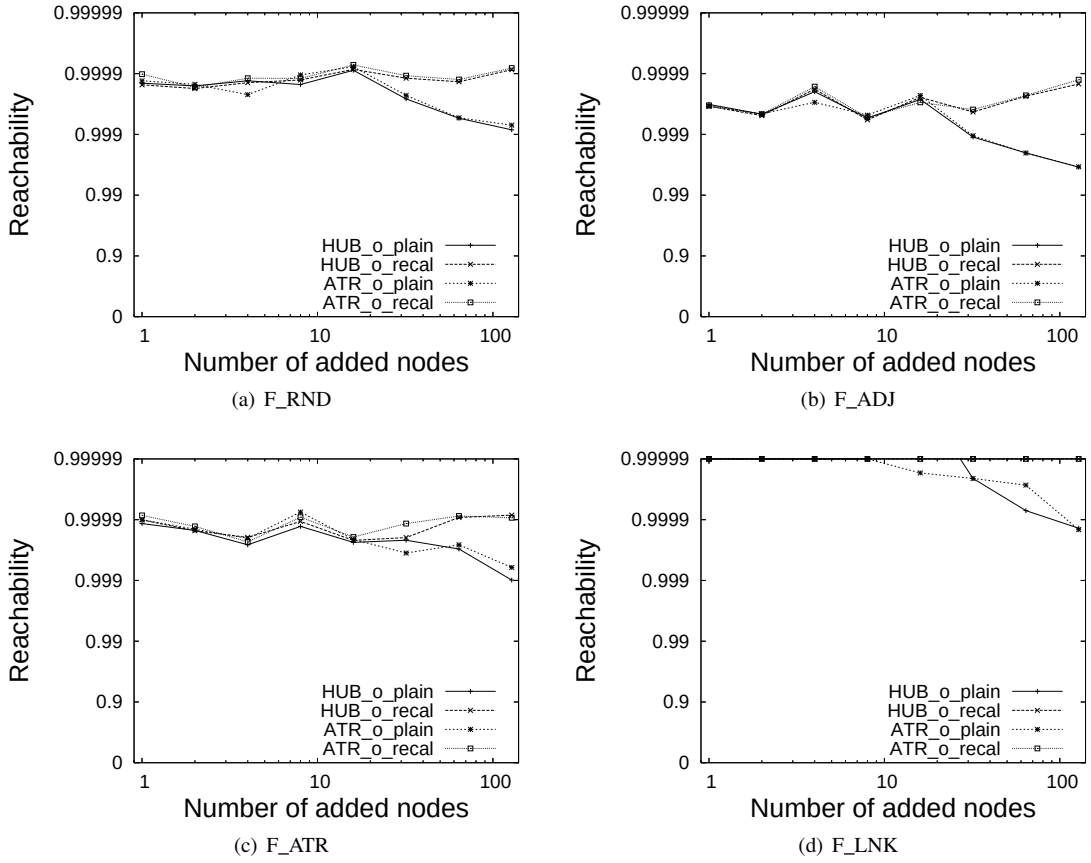
[17] A. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, Oct. 1999.



(a) F_RND (b) F_ADJ

(c) F_ATR (d) F_LNK

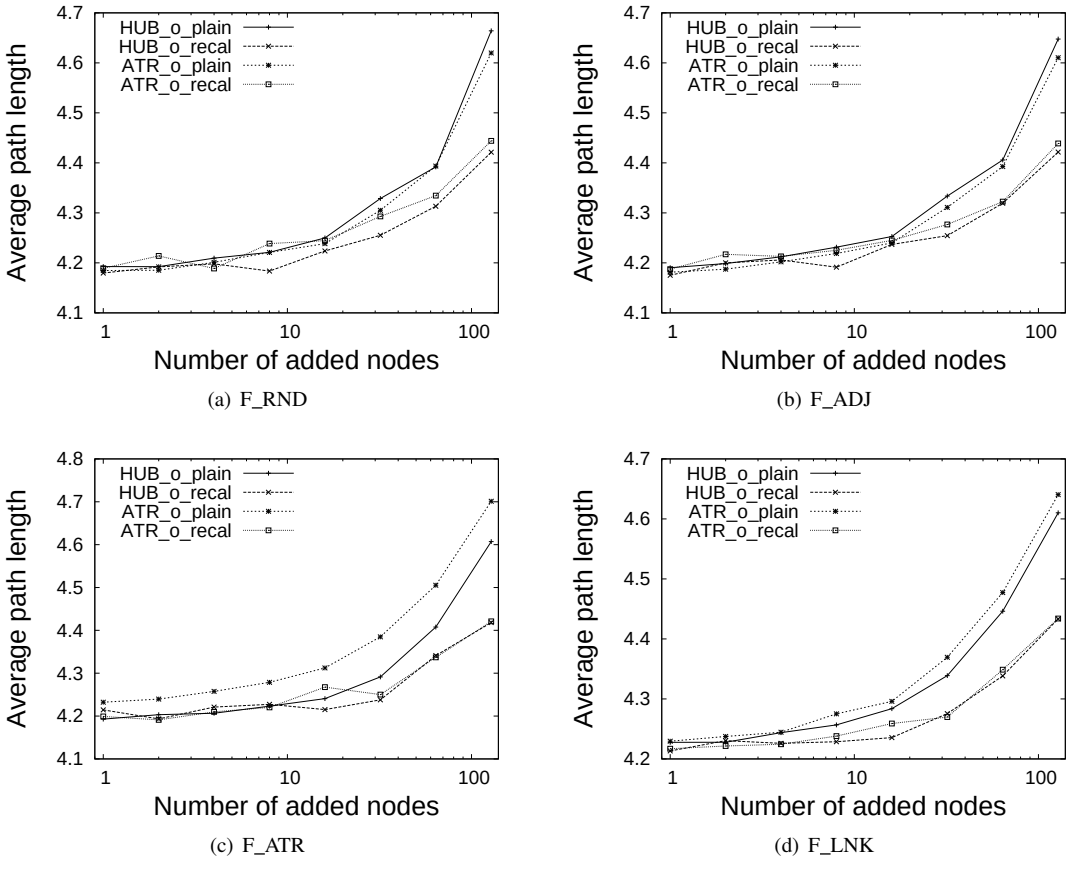
Fig. 9. Network reachability with static RL selection against two node failures



(a) F_RND (b) F_ADJ

(c) F_ATR (d) F_LNK

Fig. 10. Network reachability with dynamic RL selection against two node failures



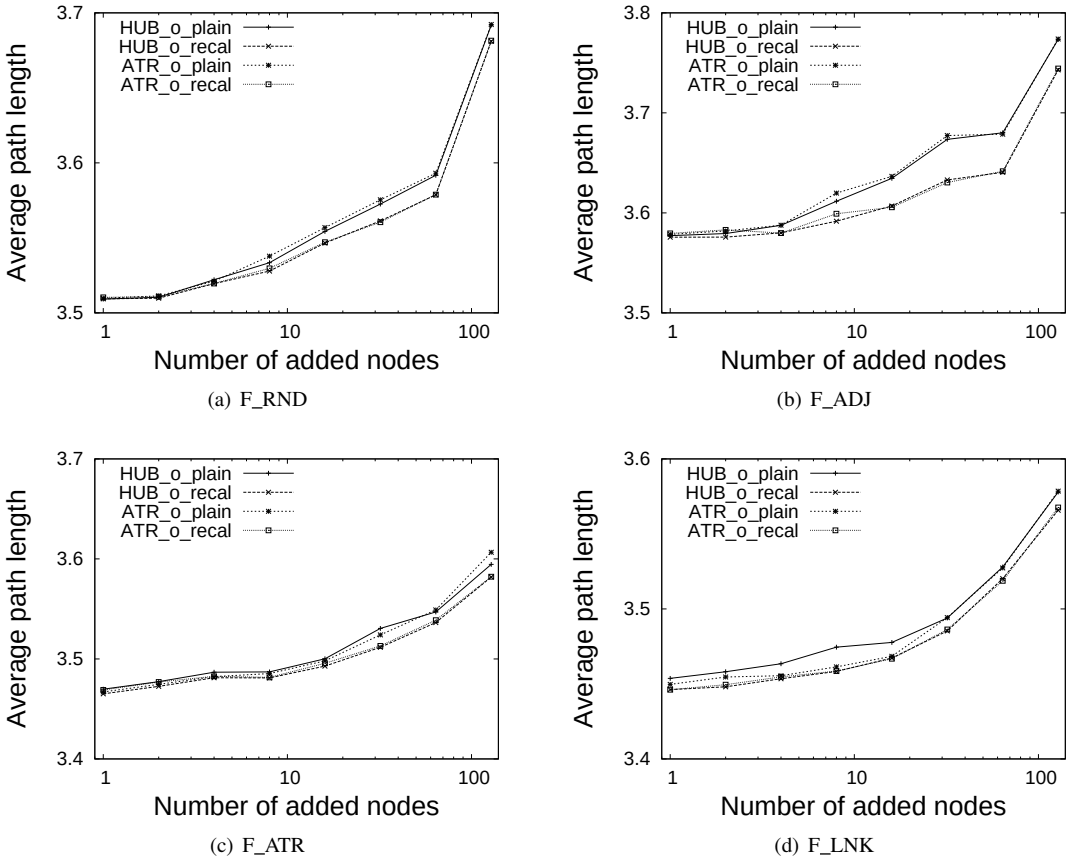
(a) F_RND

(b) F_ADJ

(c) F_ATR

(d) F_LNK

Fig. 11. Average path length with static RL selection against two node failures



(a) F_RND

(b) F_ADJ

(c) F_ATR

(d) F_LNK

Fig. 12. Average path length with dynamic RL selection against two node failures