Selective Protection: A Cost-Efficient Backup Scheme for Link State Routing

Meijia Hou
Tsinghua Univ

Dan Wang
Hong Kong Polytechnic Univ

Mingwei Xu
Tsinghua Univ

Jiahai Yang
Tsinghua Univ

Abstract

In recent years, there are substantial demands to reduce packet loss in the Internet. Among the schemes proposed, finding backup paths in advance is considered to be an effective method to reduce the reaction time. Very commonly, a backup path is chosen to be a most disjoint path from the primary path, or in the network level, backup paths are computed for all links (e.g., IPRFF). The validity of this straightforward choice is based on 1) all the links may fail with equal probability; and 2) facing the high protection requirement today, having links not protected or sharing links between the primary and backup paths just simply look weird. Nevertheless, indications from many research studies have confirmed that the vulnerability of the links in the Internet is far from equality. In addition, we have seen that full protection schemes may introduce high costs .

In this paper, we argue that such approaches may not be cost effective. We first analyze the failure characteristics based on real world traces from CERNET2, the China Education and Research NETwork 2. We observe that the failure probabilities of the links is heavy-tail, i.e., a small set of links caused most of the failures. We thus propose a selective protection scheme. We carefully analyze the implementation details and the overhead for general backup path schemes of the Internet today. We formulate an optimization problem where the routing performance (in terms of network level availability) should be guaranteed and the backup cost should be minimized. This cost is special as it involves computation overhead. Consequently, we propose a novel Critical-Protection Algorithm which is fast itself. We evaluate our scheme systematically, using real world topologies and randomly generated topologies . We show significant gain even when the network availability requirement is 99.99% as compared to that of the full protection scheme.

1. Introduction

The proliferation of the real time and loss sensitive applications such as virtual lease line services, video services, and stock exchange data services today are far less tolerant to packet loss [5]. Internet failures, however, are routine events rather than exceptions [16]. To bridge this gap, many schemes are under active investigation in the network layer to improve the overall Internet performance. One direction focuses on reactive schemes which reduce the network convergence time after failures occur [19]. Another direction is to pre-establish backup paths [11]. There are also new protocols proposed, which fundamentally switched away from the existing routing paradigms [15]. While we believe that providing a satisfiable failure recovery in the Internet is a joint force of different schemes, in this paper, we focus on pre-establishing backup paths; as it provides first reaction upon failures [14].

One very natural backup path design is to find a most disjoint path from the primary path [18]; or in the network level, to find backup paths for all links [3]. The reason may be just as simple as not making this look weird. Nevertheless, the validity needs an assumption that, all links fail with equal probability, or at least, due to the high protection requirement of the Internet today, not protecting[1] a link makes the entire backup scheme futile.

It can be costly to build an overall protection for the network layer [14]. The computation cost is high for each router, and all the rerouting needs to be stored. Thus, in this paper, we question whether it is possible (i.e., cost efficient) to have the links selectively protected. We conduct a trace analysis on the link failures in CERNET2 (the China Education and Research NETwork 2 [4]). We have observed that majority of the link failures in CERNET2 are caused by a small set of unstable links. Such effect has also been observed previously from the ASes of Sprint [16]. Consequently, we propose a *selective protection* scheme as a cost effective solution.

In this paper, we confine our study to intra-domain link state routing systems such as OSPF; and we focus on link failures. We pre-establish backup paths to improve the network availability, i.e., for a packet, the network has a route from the source to the destination, either on the primary or the backup path, to deliver it. We do not consider bandwidth reservation as our study is strictly on the network layer.

An Example. Before exposition, we illustrate our selective protection using an example in Figure 1. The topology in the figure is a sub-set of the CERNET2 topology. The link costs and all the other parameters are from CERNET2 or its history data. The number of shortest paths which traverse any link e_{ij} $(1 \leq i, j \leq 6)$, s_{ij}, and the normalized

E-mail addresses: houmeijia@csnet1.cs.tsinghua.edu.cn (M. Hou), csdwang@comp.polyu.edu.hk (D. Wang), xmw@csnet1.cs.tsinghua.edu.cn (M. Xu), yang@cernet.edu.cn (J. Yang).

[1] In this paper, we use protection and backup interchangeably.

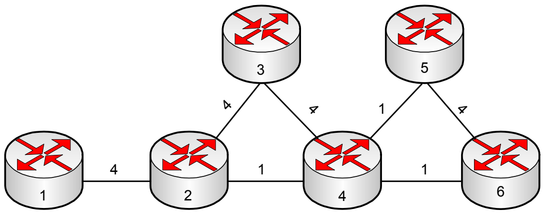


Figure 1. A sub-set of the CERNET2 topology.

	s_{ij}	f_{ij}	$s_{ij} \times f_{ij}$	$rank_{ij}$
e_{12}	5	13.344	66.72	2
e_{23}	2	1.000	2.00	6
e_{24}	6	2.001	12.01	5
e_{34}	3	30.025	90.08	1
e_{45}	5	4.580	22.90	4
e_{46}	5	6.764	33.82	3
e_{56}	0	11.113	0	7
sum			227.53	

Table 1. Parameters for Figure 1.

protected links	protection cost	protected times	relative performance
all$-\{e_{12}\}$	6Λ	160.81	100%
e_{34}	1Λ	90.08	56.02%
e_{34}, e_{46}, e_{45}	3Λ	146.80	91.29%

Table 2. Comparison for a fully protection scheme and a selective protection scheme.

number of failures on e_{ij}, f_{ij}, are shown in Table 1. These failures are taken from the history record of CERNET2 in July 2008. Due to commercial reasons, we normalized the number of failures on each link to the number of failures on link e_{23}, which has the smallest number of failures in July 2008. The fourth column of Table 1, $s_{ij} \times f_{ij}$, is the total times (normalized) of additional rerouting if the link e_{ij} goes down. Intuitively, this shows the impact on the network availability if having link e_{ij} protected ; the formal definition of the network availability is in Section 3.

We compare full protection scheme[2] with a selective protection scheme. From the last row of Table 1, we see that, during the period, there are 227.53 times of end-to-end rerouting in total due to link failures. For simplicity, we temporarily assume the cost for protecting each link to be Λ. Thus, for full protection, where all links except e_{12} are protected, the protection cost is 6Λ and a total of 160.81 end-to-end reroutings are avoided.[3] For a selective protection, if e_{34} is protected, a total of 90.08 reroutings are avoided. This is 56.02% of the performance of the full protection scheme ($90.08/160.81 = 56.02\%$), with cost of only 1Λ. If we have a budget of 3Λ, a careful selection on the protected links (e.g., e_{34},e_{46} and e_{45}) can achieve a relative performance of 91.29%; see Table 2 for a summary.

The above example is not special; from analysis and experiments, we find that different links have different impact on the network availability. This suggests that a brute force protection of all links may not be the best choice, especially when the resource is limited. Nevertheless, to fully explore this opportunity, many issues need to be carefully addressed. Practically, the network availability should be formally defined, the cost should be carefully justified, and the practical issues in the implementation should be considered. Algorithmically, the selection of the links to be protected should optimize system performance.

In this paper, we for the first time provide a systematic study for selective protection for the link state routing. We first analyze the failure traces of CERNET2. We then study the implementation details and overhead for pre-establishing backup paths. We formulate the selective protection problem. A special challenge that we face is that one major cost for pre-establish backup paths is the computational overhead. Thus, the selection algorithm should itself be of low complexity so that it will not dominate over computing backup paths. We propose a novel Critical-Protection Algorithm that successfully achieves this goal. Our evaluation on different topologies constructed by BRITE topology generator [2] and real world topologies has shown the cost-efficiency of our scheme: in most conditions, with an availability requirement of 90%, the cost of our scheme is less than 5% to that of the full protection scheme; and even with an availability requirement of 99.99%, the cost of our scheme is around 60% to that of the full protection scheme.

The remaining part of the paper proceeds as follows. In Section 2, we present the background and related work. Trace studies, implementation details and the problem formulation are specified in Section 3. Section 4 is devoted to the algorithm designs. We evaluate our scheme in Section 5. Section 6 concludes the paper and discusses future work.

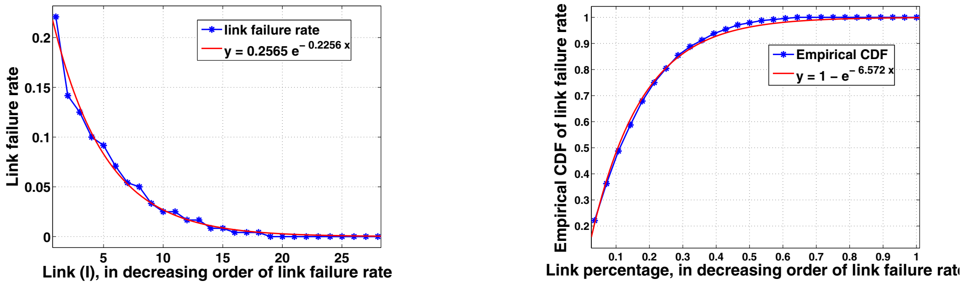
2. Background and Related Work

Failures are very common events in Internet today [16]. In a typical link state network (e.g., OSPF), when a router detects a failure, it will propagate this information to the network and each router will recalculate its routing table. During this interim period, routers will have inconsistent views of the network. Loops or black holes may occur and cause packet loss. The packet loss can be huge; for example, three million packets (average packet size 1KB) would be lost if an OC-48 link goes down for 10 seconds [18]; such disruption is unacceptable for current real time applications.

While there are studies that try to shorten the convergence period, in link state routing, pre-establishing backup paths

[2] In this paper, a full protection scheme is a scheme, where 1) the backup path is a most disjoint path from the primary path; and/or 2) a backup path is computed for each link.

[3] We note that not all the links can be protected, e.g., e_{12} (this reflects a true situation in CERNET2). This indicates that even for a full protection scheme, we can only achieve a protection ratio of 70.68% ($160.81/227.53 = 70.68\%$) . As we have argued, in many situations, a joint force of multiple schemes is needed to handle Internet failures; for example, robust physical protection is required for failures on e_{12}. We restrict the scope of this paper, however, on backup path.



(a) Per-link failure rate, in descending order.

(b) The CDF of the link failure rate.

Figure 2. Failure rate of links in CERNET2.

is more effective to reduce the service disruption time to as short as the failure detection time. Finding disjoint paths has been long a research topic in theoretical computer science with various objectives [21] and many such problems are NP-hard. In practice, backup path can be pre-established by MPLS, where both the primary path and the backup path can be reserved [11]. This approach introduces large overhead for the virtual paths establishment and requests the infrastructure to make MPLS capable.

Another approach that is currently in heavy investigation is IP Fast Reroute (IPFRR). The IPFRR framework [20] is proposed where alternative paths are identified and entries are added in the FIB of each router for rerouting. Several simple IPFRR techniques such as equal cost multiple paths (ECMP), loop-free alternates (LFA), etc request only modification on the forwarding tables of the routers that are adjacent to the failures. The ECMP or LFA paths, however, may not be found even though an alternative path that can avoid the failure exists. Other IPFRR schemes build tunnels, or specially, use not-via addresses, where each router computes backup paths for each link. When a failure occurs, the packets are encapsulated in a not-via address. All the routers will use this address to forward the packets, until the failure is bypassed. An evaluation of several IPFRR techniques is in [8]; and a number of associated techniques for IPFRR, e.g., avoiding mini-loops, can be found in [1][3][18].

Pre-establishing backup paths is also used as a building block for many schemes, e.g., R-BGP [13]; where the most disjoint path of the primary one is used as the protection.

In all these previous protection schemes, an intrinsic assumption is that all links are equally treated. Based on our observation, we find that some links undertake more traffic and/or are more vulnerable than others. We argue that facing resource limitation, the primary and the corresponding backup paths should be disjoint at such links; in other word, it is more cost efficient for these links to be protected.

3. Selective Protection: The Motivation and Problem Formulation

3.1. Data Trace Study

We conducted trace study on the backbone AS of CERNET2, the China Education and Research NETwork 2, which consists of 25 nodes, 28 links and spans across most major cities in China. We collected the network failures from Oct. 10, 2008 to Nov. 2, 2008. During this period, we have observed 240 failures. The failure rate of each link is shown in Figure 2 (a). We can clearly see that the failure shows a heavy-tail behavior and matches an exponential function. In Figure 2 (b), we can see that almost 60% of failures are caused by a small set of links, e.g., only 14% (4 out of 28 links). This behavior is also observed from the ASes of Sprint [16]. Thus a natural question is that as the majority failures are caused by a small set of links, whether a full protection scheme is cost effective.

Another interesting observation lies in simultaneous link failures. Simultaneous link failures are the events that not less than one link fails during the time period when the first link fails and the network has not yet to be re-stabilized. It is difficult to exactly estimate the time when the network is re-stabilized. Even though we capture the time when each router in the network receives the LSA and recomputes the forwarding table, the time clock at each router may still be different. In Figure 3, if we set the network re-stable time to 10 seconds, which is considered as a conservative number [6], the number of two simultaneous failures is almost zero. The probability for three or more simultaneous failures are negligible in the 10-second scale.

These observations have motivated us to explore a selective protection scheme, and emphasize on single link failure.

3.2. Implementation Details and Overhead for Pre-establishing Backup Paths

3.2.1. An Overview of the Implementation Framework.

The framework of our selective protection scheme operates as follows:

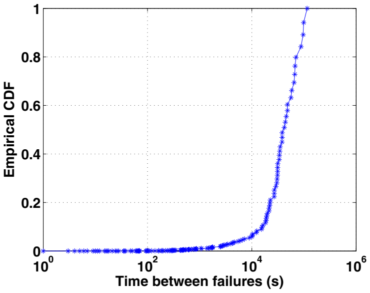


Figure 3. CDF of the time between two adjacent failures in CERNET2.

Step 1: Given a network topology, each node computes the set of links to be protected, $\mathcal{L}$; algorithm to be detailed in Section 4.

Step 2: For each link $e_{ij} \in \mathcal{L}$, each node will compute the backup path by first removing e_{ij} from the network and then computing the shortest path tree. Each node inserts the backup path information into its FIB.

Step 3: During the convergence stage of a link e_{ij} failure:

(a) A special header is used for all the packets that should be forwarded on this link.

(b) When a node receives a packet with the special header it will forward the packet using the backup path.

The not-via address [3] can be used for this step. We emphasize, however, that our scheme is not restricted to a specific technique.

3.2.2. The Overhead. Based on the current link state routing (e.g., OSPF), there are three types of overheads.

Computational Overhead: Each time the topology changes, each router computes backup path for each link to be protected (Step 2). Note that for a full protection scheme, the computational cost can be $O(|E| \times SPT)$ where $|E|$ is the number of links and SPT is the computational cost for the shortest path. This poses a high load for routers [14].

Memory Overhead: All the routers on the backup path of a selected link need to store additional entries in its forwarding table (Step 2). The more links protected, the more memory overhead is required [14].

Control Overhead: A router needs to configure itself to recognize specific headers of the potential re-directed packets on the backup paths (Step 3 (b)), and to add specific headers when it needs to re-directed packets for protected and failed links who adjacent to it (Step 3 (a)). This control process can be processed by hardware [17], however. Therefore, this cost is usually negligible.

In this paper, we will develop a selective protection scheme which can effectively reduce these costs as compared to a full protection scheme. We admit that in practice, a pure full protection scheme seldom exists alone. For example, in IPFRR, backup paths through not-via addresses are applied only on the links that cannot be protected by the less-costly ECMP and LFA schemes. Our work is also designed to be general enough so that it can be incorporated as a building block in various frameworks.

3.3. The Selective Protection Problem

We can model the communication network as an undirected and connected graph $G(V, E)$, where $V = \{v_1, v_2, \cdots, v_n\}$ is the set of nodes, and E is the set of links; $e_{ij} = (v_i, v_j) \in E$. We use P_{sd} to denote the primary path from s to d (in OSPF, this is the shortest path). We use P_{ij}^B to denote the backup path for link e_{ij}; $P_{ij}^B = \Phi$ if a backup path cannot be found for this link. Due to the observations of Section 3.1, we assume failures on e_{ij} and any link of P_{ij}^B will not occur simultaneously.

Our objective is to find a set of links to be protected to reduce the overhead and maintaining high network performance. Formally, we are looking for a link selection scheme

$$B = \{b_{ij} \,|\forall e_{ij} \in E, (1 \le i, j \le n), \exists b_{ij} \in B\} \tag{1}$$

where

$$b_{ij} = \begin{cases} 1 & e_{ij} \text{ has a backup path} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

We quantify our problem more specifically as follows.

3.3.1. Network Availability. The basic function of the network layer is packet delivery. We use network availability to quantify the network performance.

We first specify the availability of a link. Let the failure rate Δ_{ij} of link e_{ij} represent the degree that link failure events on e_{ij} impact on the network connectivity during the concerned period. Formally, let f_{ij} be the total number of link failures observed on e_{ij}; let $\mathcal{F} = \sum\limits_{\forall e_{ij} \in E} f_{ij}$; then $\Delta_{ij} = \frac{f_{ij}}{\mathcal{F}}$. The *available rate* of a link is thus defined as $\delta_{ij} = 1 - \Delta_{ij}$.[4]

The end-to-end availability of a node pair (s, d), $A(s, d)$, is defined to reflect the probability that the packets from s to d can be successfully delivered along the primary path or additionally with the backup paths of the links on the primary path. As we assume the original link and any link on its backup path will not fail simultaneously, the probability that link e_{ij} on this path is available is considered to be 1 if this link has a backup path, or δ_{ij} otherwise. Thus, $A(s, d)$ can be defined to be the multiplication of the available rate of each link on this path; formally,

$$A(s, d) = \prod_{\forall e_{ij} \in P_{sd}} [1 - (1 - b_{ij}) \cdot \Delta_{ij}] \tag{3}$$

Finally, we define the network availability as

$$Availability(G) = \frac{\sum\limits_{\forall s, d \in V \wedge s \neq d} A(s, d)}{|V|(|V| - 1)} \tag{4}$$

[4] Note that we define the available rate according to the frequency not the length of the duration of link down as the link protection algorithm is invoked for every link failure event or link restore event.

3.3.2. Cost of the Network Protection. We quantify the cost for a protection scheme, which consists of the computational cost and memory cost (we neglect the control cost). The computational cost C_e^c for protecting link e on one node is to compute the SPT for network $G - e$ and is the same for all nodes and all links.

The memory cost is the total number of entries added in the nodes of G. The memory cost for the backup path of a single link e_{ij} is $c_{ij}^m = |P_{ij}^B|$, where $|P_{ij}^B|$ represents the numbers of nodes on P_{ij}^B.

3.3.3. The Problem. To protect e_{ij}, our backup cost c_{ij}^B is finally defined as:

$$c_{ij}^B = \lambda_1 n C_e^c + \lambda_2 c_{ij}^m \tag{5}$$

where λ_1 and λ_2 are the weights associated with the two types of costs.

Given the network G and a non-negative network availability requirement Ω, the problem is to search for a link protection scheme B which satisfies the network availability requirement and minimizes the total cost for protecting the network. Formally,

$$\text{Minimize} \qquad C = \sum_{\forall e_{ij} \in E} (c_{ij}^B \cdot b_{ij}) \tag{6}$$

$$\text{s.t.} \qquad Availability(G) \geq \Omega \tag{7}$$

4. The Critical-Protection Algorithm

Unlike the conventional optimization problem, a very unique challenge of our problem is that one major overhead of pre-establishing backup paths is the computational overhead. As such, intrinsically, this requests that the computation of the algorithm for link selection, i.e., the Step 1, must be itself very fast; and hopefully, negligible as compared to the computation of the backup paths, i.e., the Step 2. Thus, the selection algorithm to be developed must be in smaller order to $O(|E| \times SPT)$.

One observation is that the costs (both the computation and the memory) are highly correlated to the number of links that need to be protected; and generally, the less, the better. This leads to a development that focusing more on the availability constraint. We develop a novel Critical-Protection Algorithm. The key observation is that the impact of every link on the network availability is not the same.

We thus introduce *criticality* ρ_{ij} for e_{ij} which reflect the impact of this link on the network availability.

$$\rho_{ij} = s_{ij} \cdot \Delta_{ij} \tag{8}$$

where s_{ij} is the number of shortest paths in G that traverse through e_{ij}. Intuitively, a link is more critical if 1) the failure rate of this link is high and/or 2) the number of routes that this link undertakes is high. We would like to comment that one reason we finally choose this criticality for our algorithm development is also due to its simplicity.

We develop the Critical-Protection Algorithm (See Figure 4), which iteratively protects links that are more critical. In our algorithm, we combined Step 1 and Step 2; that is, the output of the CP algorithm (Critical-Protection Algorithm) contains not only the link selected, but also the backup paths computed for these links. The reason for combining these two steps is that each time a link e_{ij} is selected, we must verify whether a backup path for e_{ij} exists (See Figure 4 (a) line 5). Otherwise, such selection is not valid.

We next comment on the computation complexity for network availability, which is another major design issue. The complexity of computing $Availability(G)$ is $AvgL \times |V|^2$ where $AvgL$ is the average length of the network end-to-end paths in G. Note that if we select one link in each iteration, we have to evaluate $Availability(G)$ for each link. To balance such computational overhead, we select a subset of links in each iteration (See Figure 4 (a) line 4), so that the total computation for availability will be amortized. The number of links to be selected in each step is initially set to one and increases by $stepsize$ for each iteration. If $stepsize = 1$, only one link is selected in each iteration. If $stepsize$ is too large, we may over protect many links in the final iteration. As we need to compute the associated backup paths, over protection will suffer from computational overhead too. Obviously, increasing the number of links to be protected exponentially in each iteration is not a good choice. In our algorithm, we increase $stepsize$ linearly in each iteration. This results in the increase of the number of the links selected in each iteration (i.e., $stepsize$) to be $\sqrt{\cdot}$ to that of the total link selected (the total number of links selected after each iteration is 1, 3, 6, 10, 15, ...). This also results in an over protection of at most $\sqrt{\cdot}$.

Another additional optimization is that note that only a small set of $A(s,d)$ need to be recalculated each time. We trade-off space to time by storing the $A(s,d)$ table; and only modify the $A(s,d)$ that need to be changed.

5. Simulation

5.1. Simulation Setup

To evaluate our Critical-Protection Algorithm, we construct a variety of topologies generated by BRITE [2]. The numbers of nodes of these topologies range from 40 to 200. The average number of links per new node is set to be 2 or 3, and the mode is set to be Router only. The bandwidth on each link follows a heavy-tail distribution ranging from 100 to 1024 [8] and the link cost is an inverse function of the bandwidth. We adopt the same assumption with [8] that the topologies are static in terms of link costs and the link costs are symmetric.

We also follow the study of [9] to generate two topologies similar to the real DFN and AT&T networks. Besides, we evaluate our scheme on the real ABILENE topology [7]. The details of the parameters are listed in Table 3. The numbers

```
Function ComputeBP ($lower$, $upper$)
$t = lower$; $stepsize = 1$;
while ($Availibility(G) < \Omega$) and ($t \le upper$)
    $tbound = min\{upper, t + stepsize - 1\}$
    Compute BPs for $\{e^t, e^{t+1}, \ldots, e^{tbound}\}$;
    if BP for any link $e$ found, $b_e = 1$ else $b_e = 0$;
    $lower = t$;  $t = tbound$ + 1;  $stepsize$++;
$upper = t - 1$;
if $lower < upper$,
    $b_e = 0$ for any $e \in \{e^{lower}, e^{lower+1}, \cdots, e^{upper}\}$.
```

(a) Sub-function to select a set of links and compute the associated backup paths.

```
Algorithm Critical-Protection ($G$, $\Omega$)
Initialization: construct a link list in descending order
according to an adjusted criticality, $(e^1, e^2, \cdots, e^{|E|})$;
$lower = 1$; $upper = |E|$;
while ($lower < upper$)
    ComputeBP($lower$, $upper$);
Output: The set of selected links to be protected (i.e.,
$B$) AND the associated backup paths (i.e., $P_{ij}^B$).
```

(b) The main problem for Critical-Protection Algorithm.

Figure 4. Critical-Protection Algorithm.

of nodes and edges of the topologies we used for simulation are summarized in Table 4.

For each link e, the number of link failures on e, f_e, is generated with negative exponential distribution, whose probability density function is $y = e^{-x}$. Let $\mathcal{F} = \sum_{\forall e \in E} f_e$; the link e failure rate is $\Delta_e = \frac{f_e}{\mathcal{F}}$. As an illustration, the failure rate generated for the AT&T topology is shown in Figure 5.

To evaluate our scheme, we compare our results with a full protection scheme where backup path is computed for each link. To have a fair comparison, the costs of the CP algorithm is calculated as follows. For computational cost, we record all the backup paths we have computed. Note that the backup paths we computed may be more than the links selected; especially, as for verification, we may compute backup paths for some links that do not have backup paths. The memory cost is straightforward as the overhead will be proportional to the number of links we select to protect.

The major evaluation criterium we use is *backup-cost rate*, i.e., the ratio of the cost of CP algorithm to that of the full protection scheme. In our simulation, we use $C/MRate$ to denote different impact between the computational cost and the memory cost. That is $C/MRate = \lambda_1 C_e^c / \lambda_2$. We also use *availability bound* to denote the ratio of Ω to the $Availability(G)$ of the full protection scheme.

All the simulations are conducted on a PC with Pentium 4 CPU 2.40GHz and 1.49G Memory. All the results shown are the average values of 10 random experiments.

	DFN	AT&T	other topo
Topo. Type	BOTTOM-UP	BOTTOM-UP	Router Only
BOTTOM-UP Topology Parameters			
Grouping Model	Random Pick	Random Pick	
NumAS	17	31	
AS Assignment	Constant	Constant	
Inter BWdist	Heavy Tail	Heavy Tail	
BW Range	100–1024	100–1024	
Router Parameters			
N	30	154	40–200
Model	GLP	GLP	GLP
α	0.45	0.45	0.45
β	0.64	0.64	0.64
m	3	2	2
Pref. Conn	none	none	none
BWdist	Heavy Tail	Heavy Tail	Heavy Tail
BW Range	100–1024	100–1024	100–1024

Table 3. Parameters for BRITE generator.

	Num of nodes	Num of edges
ABILENE	11	14
DFN	30	150+
AT&T	154	550+
40-topo	40	100+
80-topo	80	200+
100-topo	100	300+
120-topo	120	400+
160-topo	160	500+
200-topo	200	700+

Table 4. Topology size for simulation.

We have also conducted experiments with real trace on CERNET2 topology, and similar results are observed. Due to space limitation, a full version can be found in [10].

5.2. Simulation Results

Figure 6 shows the comparison of our CP algorithm and the full protection scheme on ABILENE, DFN, and AT&T topologies. In Figure 6 (a), we see that when the *availability bound* is 98%, i.e., the availability requirement to the network availability of the full protection scheme, Ω, is 98%, our CP algorithm consumes negligible cost as compared to the full protection schemes in DFN and AT&T topology. For ABILENE topology, the cost gain of CP is not as significant. That is because the DFN (30 nodes) and the AT&T (154 nodes) topologies are much denser than

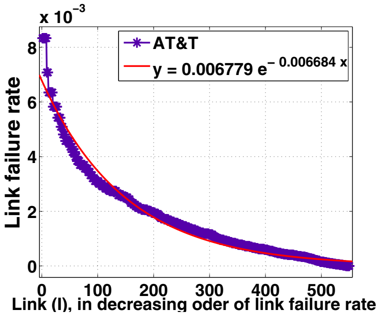


Figure 5. Failure rate for AT&T topology.

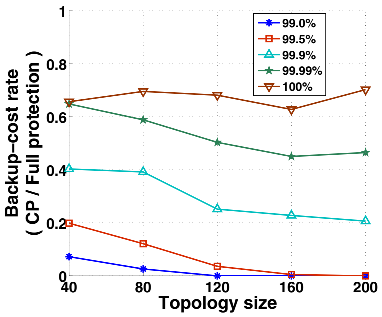


Figure 7. Results on 40 - 200 - node topologies with different availability bound ($C/MRate = 1$).

ABILENE topology (11 nodes and 14 edges). Note that in our algorithm, when we select a link (which should be protected), we need to verify whether a backup path exists. If there is no backup path for this link, our algorithm will need to select another one. It is possible that backup path cannot be found, especially when the network is sparse; note that when a link fails, our algorithm need to compute the backup paths for all the links in a sub-graph without this link. As such, if the topology is too sparse, our algorithm may compute backup paths for more links that cannot be finally protected; and increase the computational cost contributing nothing for the network availability increase. On the contrary, the full protection scheme always compute backup paths for all the links; no matter whether the backup paths exist or not. Thus the total cost will not be affected.

We can also see that even when the CP algorithm reaches the same $Availability(G)$ with the full protection scheme (availability bound = 100%), in DFN and AT&T, the cost of CP algorithm is less than 50% and 70% to that of the full protection scheme. By looking into the details of our simulation log files, we find that in the generated topologies, there are some *absolutely not* critical links, i.e., the links that undertakes no shortest paths, or the failure rate is zero. As such, these links do not need to be protected at all.

Comparing Figure 6 (a) and (b), we put different weights on the computational cost and the memory cost. We see that the difference is small. This indicates that the absolute number of links we over compute is small.

In Figure 7 we evaluate our scheme under different topology sizes. We can see that, except for the case that *availability bound* = 100%, our CP works better for larger topologies. This is also because larger topologies are denser than the smaller ones; thus, if we protect the same proportion of links, the sub-topologies consisted of the protected links in larger topologies is denser and can reach higher *availability bound*. In other words, with the same *availability bound*, larger topologies can protect a smaller ratio of links. This results in the lower backup-cost rate.

We then evaluate the total computing time for backup path in the CP algorithm and the total computing time for all other parts (including program initialization and computing $Availability(G)$, etc). This shows the real overhead of the CP algorithm (Step 1). From Figure 8 (a) we can see that, the total amount of time for computing the backup paths increases fast when *availability bound* is higher. This is because more backup paths need to be computed. The computational overhead of CP itself increases moderately. This clearly indicates that our CP algorithm has successfully achieved one of its objective, i.e., the Step 1 must be fast itself. In Figure 8 (b), we change the topology size from 40 to 200. We can also see that the computational cost of the CP algorithm is negligible as compared to the computation for backup paths. We thus conclude that focusing on the cost defined in Section 3 is valid.

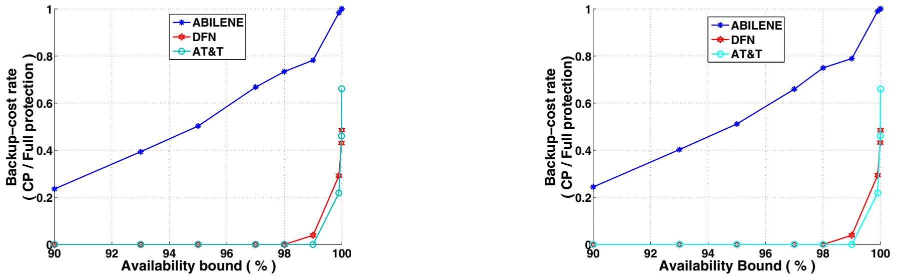
5.3. Limitation of the Study

We would like to comment on one limitation of our work. In our study, we apply the history information of the link failures to develop our algorithm as such that we know these data in advance. We admit that this is invalid. Our argument is that we have seen in many different situations (e.g., web caching [12]) such that the more unstable a link (or an object/file) is in the history, the more unstable this link is in the future. As such using history data might be a reasonable approximation. We leave a more comprehensive autonomous self-learning on the network failures to our future work.

6. Conclusion and Future Work

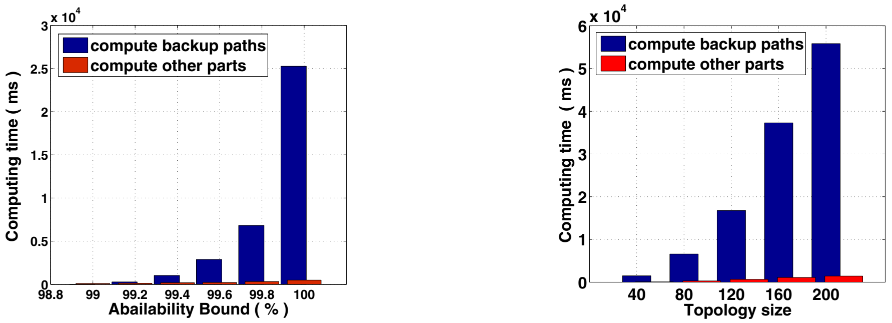
In this paper, we proposed a selective protection scheme for handling failures in link state routing. By careful study on the failure characteristics of CERNET2, we observed that a substantial number of failures in the Internet are caused by a small set of links; this conforms to previous measurement studies on Sprint ASes. Consequently, we proposed a selective protection scheme where only a subset of links were protected. We formulated an optimization problem where the cost should be reduced and the network performance should be guaranteed. The challenge for the algorithm design was that the system cost highly depends the computation overhead for the backup paths. Therefore, the link selection algorithm should be fast itself. We thus proposed a novel Critical-Protection Algorithm which identified critical links to be selected early. We evaluated our scheme comprehensively with topologies generated from BRITE and other real world topologies.

To the best of our knowledge, our study is the first to propose a selective protection scheme. Though Internet today is looking forward for correct delivery service that is 99.99%, we have shown that even in this scale, our selective protection may still be cost efficient. In the future, we would like to conduct a larger scale of failure analysis. We believe more precise prediction of failure behavior and identification of vulnerable links can further improve the performance. We also believe that our selective protection scheme can be used



(a) $C/MRate = 1$ (b) $C/MRate = 2$

Figure 6. Results on three sub-real topologies.



(a) Topology size = 100. (b) Availability bound = 99.9%.

Figure 8. Time for backup paths computation and other parts in CP algorithm ($C/MRate = 1$).

as a building block or a starting phase for more sophisticated protection schemes.

7. Acknowledgment

This work is supported by HKPU/ICRG under Grant G-YG78, A-PB0R, RGC/GRF PolyU under Grant 5305/08E, the Natural Science Foundation of China under Grant 90604024, HI-Tech Research and Development Program of China (863) under Grants 2006AA01Z209, 2007AA01Z2A2, 2008AA01A303 and 2009AA01Z205, the National Science and Technology Support Plan of China under Grant 2008BAH37B05, and the National Basic Research Program of China (973) under Grant 2009CB320505.

References

[1] A. Atlas and A. Zinin, “Basic specification for IP fast-reroute: Loop-free alternates”, Internet draft, draft-ietf-rtgwg-ipfrr-spec-base-06.txt, Feb 2006.

[2] BRITE: Boston university Representative Internet Topology gEnerator, http://www.cs.bu.edu/brite/.

[3] S. Bryant, M. Shand, and S. Previdi, “IP fast reroute using notvia addresses”, Internet draft, draft-ietf-rtgwg-ipfrr-notvia-addresses-00.txt, Dec 2006.

[4] The China Education and Research NETwork 2 (CERNET2), http://www.cernet2.net/.

[5] N. Feamster and H. Balakrishnan, “Packet loss recovery for streaming video”, in *Proc. International Packet Video Workshop*, Pittsburg, PA, Apr. 2002.

[6] P. Francois,C. Filsfils, J. Evans, and O. Bonaventure, “Achieving subsecond IGP convergence in large IP networks”, in *ACM SIGCOMM Computer Communications Review*, vol. 35, no. 3, pp. 35-44, 2005.

[7] P. Francois, “Improving the convergence of IP routing protocols”, in *PhD thesis*, Universit catholique de Louvain, Oct. 2007.

[8] M. Gjoka, V. Ram, X. Yang, “Evaluation of IP fast reroute proposals”, in *Proc. IEEE COMSWARE’07*, Bangalore, India, Jan. 2007.

[9] O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmetz, “Generating realistic ISP-level network topologies”, in *IEEE COMMUNICATIONS LETTERS*, vol. 7, no. 7, pp. 335-336, 2003.

[10] M. Hou, D. Wang, M. Xu, and J. Yang, “Selective Protection: A Cost-Efficient Backup Scheme for Link State Routing”, Technical Report, Department of Computer Science & Technology, Tsinghua University, 2009.

[11] M. Kodialam and T. V. Lakshman, “Dynamic routing of bandwidth guaranteed tunnels with restoration”, in *Proc. IEEE INFOCOM’00*, Tel-Aviv, Israel, Mar. 2000.

[12] K. Krishnamurthy and J. Rexford, Web Protocols and Practices, Addison-Wesley Professional, May 2001.

[13] N. Kushman, S. Kandula, D. Katabi, and B. Maggs “R-BGP: Staying connected in a connected world”, in *Proc. USENIX NSDI’07*, Cambridge, MA. Apr. 2007.

[14] A. Li, P. Francois, and X. Yang, “On improving the efficiency and manageability of NotVia”, in *Proc. ACM CoNext’07*, New York, NY, Dec. 2007.

[15] K. Levchenko, G. Voelker, R. Paturi, and S. Savage, “XL: An efficient network routing algorithm”, in *Proc. SIGCOMM’08*, Seattle, USA, Agu. 2008.

[16] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, Y. Ganjali, and C. Diot, “Characterization of failures in an operational IP backbone network”, in *IEEE/ACM Transactions on Networking*, vol. 16, no. 4. pp. 749-762, Aug. 2008.

[17] Personal communication with Dr. Wenlong Chen, Bitway Networks, Sept. 2008.

[18] S. Nelakuditi, S. Lee, Y. Yu, Z. Zhang, and C. Chuah, “Fast local rerouting for handling transient link failures”, in *IEEE/ACM Transactions on Networking*, vol. 15, no. 2, pp. 359-372, Apr. 2007.

[19] D. Pei, M. Azuma, D. Massey, and L. Zhang, “BGP-RCN: improving BGP convergence through root cause notification”, in *Computer Networks*, vol. 48, no. 2, pp. 175-194, 2005.

[20] M. Shand and S. Bryant, “IP fast reroute framework”, Internet Draft, draft-ietf-rtgwg-ipfrr-framework-06.txt, 2006.

[21] J. Suurballe, “Disjoint paths in a network”, in *Networks*, vol. 4, pp. 125-145, 1974.