Fast, Effective and Stable IP Recovery using Resilient Routing Layers

Audun Fosselie Hansen [1] [2], Amund Kvalbein [1] , Stein Gjessing [1], Tarik Čičić [1], Terje Jensen [2], Olav N. Østerbø [2] and Olav Lysne [1]

[1] Simula Research Laboratory
P.O. Box 134 Lysaker, 1325 Lysaker, NORWAY
{audunh, amundk, steing, tarikc, olavly}@simula.no
[2] Telenor R&D
1331 Fornebu, NORWAY
{audun-fosselie.hansen, terje.jensen1, olav-norvald.osterbo}@telenor.com

Abstract. Recovery at the IP layer is hampered by the slow convergence of IP rerouting. Recovery times in the range of seconds do not adhere to the requirements of many Internet applications today. To offer fast, pre-configured and loop-free IP recovery we have proposed a new method named Resilient Routing Layers (RRL). In this paper we demonstrate how RRL also can provide resource-effective recovery in IP networks. We compare the performance of RRL with what intuitively should be the most resource-effective method: Full global rerouting.

1 Introduction

As communications networks become a more integrated part of our critical infrastructure, reliability and availability of such networks become increasingly important. Given the growing size and complexity of networks, the presence of component failures are part of every day maintenance [1]. Hence, much attention has been given to the problem of rapid recovery from (link) failures in IP networks.

IP networks are intrinsically robust, in the respect that a routing protocol like OSPF or IS-IS will ultimately recover from any failure by calculating new routing tables in the routers, based on the changed topology after a failure. This rerouting is global, and based on full distribution of the new link state to all routers in the network. When the new state information is distributed, each router individually calculates new valid routing tables.

Any recovery scheme based on global rerouting has limitations with respect to the minimal recovery time. With IP rerouting, it normally takes several seconds after a failure before packet forwarding is continued. Although this time can be reduced by careful tuning of timeout parameters, such efforts fail to give recovery times below one second without affecting the stability of the network [2].

A number of proposals have been made to create recovery mechanisms for IP networks that can handle failures locally, without global rerouting. A common challenge for such mechanisms is how to avoid loops in the packet forwarding. When two equal cost paths exist toward a destination, traffic can safely be switched from one to the other in case of

a failure. Iselt et al [3] suggest using MPLS tunnels to make sure that there always exists two equal cost paths toward a destination at every node. Some proposals try to handle link failures as locally as possible, by only doing routing updates in a limited number of routers near the failure [4].

Other proposals take a more coherent view on the network, and try to guarantee that there always is a valid routing entry for a given destination even after a failure. With O2-routing [5], the routing tables are set up so that there are always two valid next hops toward a destination. Both routing entries are used in the error-free case, but none of the two is necessarily the shortest path. Another proposal, named Failure Insensitive Routing [6], suggests using interface specific forwarding. The router decides the outgoing interface based on both the IP address and the incoming interface. This way, a router can implicitly avoid forwarding over failed links, without being explicitly notified about the failure.

The authors have previously proposed an alternative solution for fast recovery from link failures [7], called Resilient Routing Layers (RRL). RRL is based on the idea of building spanning sub topologies over the full network topology, and using these sub topologies to forward traffic in the case of a failure. Routing recovered traffic according to a sub topology may cause high concentration of traffic on certain links, and hence loose some of the recovery gain due to congested links.

Nucci and others have studied and improved this effect for IP rerouting using smart link weight assignments [8].

In this paper, we will examine this effect for RRL. We will show that RRL provides acceptable load balancing for the recovered traffic, and thus provides a good alternative for cost-effective and stable IP fast recovery.

The rest of the paper is organized as follows: Section 2 will describe the main features of RRL. In section 3 we discuss the general approach used to evaluate the traffic loads with RRL, while section 4 presents the experiment example and the results. In section 5 we conclude and give some directions for future work.

2 Resilient Routing Layers (RRL)

RRL is based on the idea of building spanning sub topologies over the full network topology, and using these sub topologies to forward traffic in the case of a failure [7]. In RRL, we denote the sub topologies *layers*. There exists a valid path between any pair of nodes in every layer. In this paper, we focus on applying RRL for protection against link failures, but the RRL scheme can also be applied to protect against node failures [9]. Similar approaches have previously been demonstrated for Up*/Down* routed interconnect networks [10]

We say that a link is *safe* in a layer if the link is not part of that layer. The layers are created by removing links from the full topology, in such a way that every link is safe in at least one layer. Many links can be safe in the same layer. The maximum number of links that can be removed in a layer, is limited by the invariant that all layers must be fully connected. We say that a link can be removed from a topology unless it is an *articulation link*, meaning that removing the link would disconnect the topology.

For an illustration of how a network topology can be covered by layers, consider the example in figure 1, which is the German Telecom core IP network on POP-level [11]. To the left (a), we have the full topology with ten nodes and seventeen links. To make

every link in this topology safe, we must select layers so that for each link, there is a layer where the link is not present. Figure 1b), c) and d) is an example of how all links in this topology can be made safe using three layers. It should be mentioned that using the layers in figure 1 is not the only possible way to make every link in this example topology safe. Algorithms that construct layers have been developed in [7]. In small topologies this task can also be performed manually.

From [7] we have that the number of layers needed are very modest, i.e. between two and five, even for very large topologies (512 nodes).

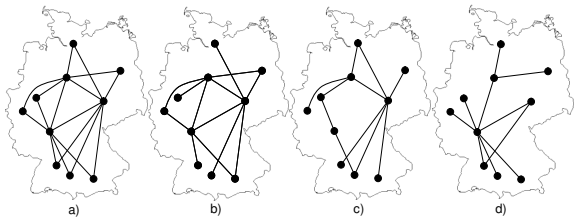

Fig. 1. In this example, three layers (b-d) are used to make every link in the German telecom network (a) safe. Each layer is a fully connected subgraph of the full topology, and every link is removed in (at least) one of the layers.

The constructed layers are used as input to routing or path-finding algorithms that calculate a routing table or path table for each layer. Tables containing routing information for each layer must be kept in every node.

In the fault-free situation RRL does not put any restrictions on the routing. When a failed link is detected, the nodes attached to the link start forwarding traffic that would normally go through that link in the corresponding safe layer. Packets forwarded in a safe layer is marked, so that the other nodes can keep forwarding them in the same layer. This way, recovered traffic is forwarded shortest path to the destination in the safe layer of the failed link. Traffic that did not pass through the failed link is not affected, and is still routed in the original full topology. The decision on when to forward traffic in a safe layer can be taken locally, which allows very fast reaction to a failure situation, without any signaling.

Since RRL restricts the number of links available for routing of recovered traffic, the path lengths for this traffic may increase compared to the optimal path length. We have previously demonstrated that the recovery path lengths for RRL are within acceptable bounds compared to the path lengths for optimal local recovery, i.e. all links but the failed one are available [7].

A candidate implementation strategy for RRL in an IP environment could be the Multi-Topology routing currently under standardization within IETF [12] [13].

3 Overall Approach

RRL provides a network with resilient routing layers, predefined for use during certain failure configurations. These layers can be viewed as a set of topology descriptions where links are isolated and routing tables can be identified. Adding to these factors, traffic

considerations have to be taken into account in order to estimate performance parameters and network resource utilization levels. In the following the overall steps for arriving at these estimates are outlined, presented for a generalized problem formulation.

As a starting point we have the topology with a set of network resources with relevant traffic matrices. Capacities of the relevant network resources must also be given. As a first measure, for example for link capacities, units of Gbit/s could be applied. However, other types of network resource may also be relevant, such as buffer sizes, throughput requirements in nodes, processing capacities, and so forth. Both physical as well as logical resource groups may apply depending on the case at hand.

A set of traffic matrices could be given, for example when several traffic classes are relevant. In a traditional manner, an element in a traffic matrix gives the amount of traffic requested from a given source to a given destination. A traffic class could also be characterized in various ways; separations could be dependability requirements, delay requirements, loss ratios, etc. Broadly, these requirements would likely relate to the Service Level Agreement (SLA) conditions. Given this starting point, traffic handling during normal operation is found. Typically, this can be done by running a routing algorithm such as Shortest Path First (SPF) with Equal Cost Multi-Path (ECMP) in order to find which sequences of resources the traffic should follow from sources to destinations. The completion of this step gives the results applied during failure-free operation.

Running the RRL algorithm provides the set of layers to be used for each of the failure configurations as further detailed in [7]. For each of these layers, the routing of traffic has to be decided upon, e.g. by invoking SPF with ECMP. Then we are set to analyze consequences of each of the network resources failing - fully or partially. The description in section 2 is done for links assumed to be in operation or failed without any partial operational states. Going through the network resource unit by resource unit, the corresponding RRL solution (a certain layer) is examined with corresponding traffic handling. In effect, traffic loads on the network resources are found. If all network resources face load below their capacity limits, the corresponding failure situation can be dealt with without severe degradation of service levels. Depending on the traffic classes several performance requirements may apply for a resource unit.

In case a network resource is found to be in overload, the traffic handling might need to be modified. Here, several options apply depending on which type of resource unit we look at and the requirements related to different traffic classes. For example, for a link, a portion of traffic with lower dependability requirements could be dropped in order to carry traffic with higher requirements. This changes the corresponding low priority traffic flow, which again affects traffic flows on the network resources. Depending on which traffic types we look at, this may only impact the actual resource defined for overload and all subsequent resources on the way to the destination. However, for some cases, as the overall end-to-end throughput of those traffic flows may decrease, also the source could take proper actions. In this context, UDP traffic may represent the former, while TCP with flow control could represent the second case. However, applications may be running over UDP reacting on reduced throughput and thereby also reducing traffic offered by the source. At least, in effect, traffic flows could be reduced on their way toward destinations.

There are different ways of estimating how these traffic reductions should be captured. For example, for UDP like traffic, some similarities with models and results in [14] and [15]

may be alluded to. Correspondingly, the reduce-load approximation may also be applied for these cases.

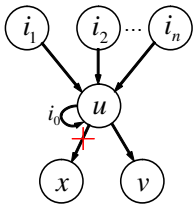

Fig. 2. Shows the relationship between the links and nodes that are represented in the formulas. Neighbors of node u are denoted as neighbor i_x. We model traffic generated in node u as traffic from i_0. Related to equations (1) and (2), T_{0u}, i.e. $i = 0$, is traffic generated in node u.

In the following we present some formulas that we have used to calculate the traffic behavior in a network when using RRL for local pre-configured recovery. RRL complicates the model due to the fact that some traffic is routed on the full topology and some traffic is routed in a layer. In addition, due to local recovery the same traffic flows can be routed on the full topology in one part of the network and according to a safe layer in another part.The fact that we use ECMP routing with original source and destination as input to the path selection, also add some extra complexity.

We assume that all demands are fulfilled to the point of failure or alternatively point of congestion. The traffic load is then adjusted from the point of failure/congestion and all the way to the destination. The adjustment is performed so that all throughputs are reduced according to a common factor. Figure 2 illustrates the connection between links and nodes as presented in the formulas.

Notation:

uv: Link from node u to node v

T_{uv}: Throughput on link uv.

T_{uv}^{lsd}: Throughput on link uv for traffic routed on layer l, from source s to destination d. When $u = 0$ *and* $s = v$ we say that we get the traffic generated in node $v = s$ destined for destination d

D_{uv} : Total demand for link uv.

D_{uv}^{lsd}: Total demand for link uv for traffic routed on layer l, from source s to destination d, includes normally forwarded demand and extra demand due to a failure

F_{uv}^{lsd}: Normally forwarded demand for link uv for traffic routed on layer l, from source s to destination d, disregarding extra demand due to failure on a link ux from node u ($x \neq v$)

E_{uvx}^{ksd}: Extra demand on link uv due to failure on a link ux from node u ($x \neq v$). The value can be other than 0 only when k is the safe layer of link ux, i.e. $k = L(ux)$

$L(ux)$: safe layer of link ux

C_{uv}: Capacity of link uv.

a_{uv}: Adjustment factor on link uv

P_{lsud}: Path (as decides by the routing function) in layer l, between node u and destination d for traffic originating in source s. We need the originating source as input to the ECMP decision in node u

R_u: The amount of traffic that terminates in node u, i.e. the amount of traffic node u receives as destination

R: Total throughput in the network, i.e. the sum of all R_u

Throughput:

Total throughput is the sum of traffic that arrives at all destinations. As a first step to find the total throughput, we calculate the throughput of each link. When we add the amount of traffic obtained from the traffic matrix to the network, the throughput in each link is calculated according to the following rules.

$$(1) \quad F_{uv}^{lsd} = \begin{cases} 0 & \text{if } uv \notin P_{lsud} \textit{ or } uv \textit{ has } failed \\ \sum_i T_{iu}^{lsd} & \text{if } uv \in P_{lsud} \text{ and } uv \textit{ has not } failed \end{cases}$$

$$(2) \quad E_{uvx}^{ksd} = \begin{cases} \sum_l (\sum_i T_{iu}^{lsd} & \text{if } ux \textit{ has } failed \textit{ and } ux \in P_{lsud} \textit{ and } uv \in P_{ksud} \text{ and } k = L(ux)) \\ 0 & Otherwise \end{cases}$$

$$(3) \quad D_{uv}^{lsd} = F_{uv}^{lsd} + \sum_x E_{uvx}^{lsd}$$

$$(4) \quad D_{uv} = \sum_l \sum_s \sum_d D_{uv}^{lsd}$$

$$(5) \quad a_{uv} = \begin{cases} 1 & \text{if } D_{uv} \leq C_{uv} \\ \frac{D_{uv}}{C_{uv}} & \text{if } D_{uv} > C_{uv} \end{cases}$$

$$(6) \quad T_{uv}^{lsd} = \frac{D_{uv}^{lsd}}{a_{uv}}$$

$$(7) \quad T_{uv} = \sum_l \sum_s \sum_d T_{uv}^{lsd}$$

In (1), (2), (3) and (4) we say that the demand for a link uv is defined by the throughput in the predecessor links (iu links from figure 2) that is routed over link uv. This includes normally forwarded traffic and traffic that is rerouted over uv in safe layer k due to a failure in another link ux from node u. In (1) and (2), when $i = 0$, we look at traffic generated in node u. (6) and (7) say that the throughput of a link is defined by the demand and the adjustment factor (4).

When equation (6) and (7) holds for all links uv, i.e. the set of algorithms has terminated and reached steady state, we say that the total throughput in the network is defined by the following:

$$(8) \quad R_u = \sum_i \sum_l \sum_s T_{iu}^{ls\mathbf{u}}$$

$$(9) \quad R = \sum_u R_u$$

(8) defines that the amount of traffic that node u receives as final destination is the sum of the throughput of all links iu with node u as destination. And (9) defines the total over all nodes.

Link load distribution

We also measure the load distribution for all links in the network. These results are obtained by letting:

$$(10) \quad a_{uv} = 1 \textit{ and } T_{uv} = D_{uv}$$

4 Experiment and Results

As an example we will present results obtained from the German Telecom core IP network on the POP-level [11]. This network, as illustrated in figure 1a) has 10 nodes and 17 bi-directional links. The capacity of each link is 100 and all link weights are 1. To generate layers we use the *Rich* algorithm from [7], and we run the experiment with three (figure 1) and six layers. We compare the performance of RRL with the normal failure-free situation, with the no recovery situation, and global end-to-end recovery routed on the full topology except the failed link. The latter should intuitively be the most optimal reference point for recovery with respect to load balancing since all links but one are available for recovery routing.

For the purpose of this analysis we have built a java code base that generates layers and calculates traffic loads according to rules presented above. As routing function we use shortest path routing with ECMP. The ECMP path is chosen based on both originating source and final destination. Input to the routing function will then be what topology or layer to route according to, the originating source, the current source point and the final destination.

For traffic demands we use uniform traffic matrices. The total and source-destination demands are presented in table 1. In addition, the table gives the average link loads in the normal case and the maximum link load for all methods.

We measure the loads for all uni-directional links. When links fail, both directions of a bi-directional link fail. We present average results from all link failures and results for the worst case link failure.

Total demand	per src-dest	avg Normal	max Normal	max Global full	max RRL3	max RRL6
540	6	25.8	36	54	60	54
630	7	30.1	36	63	70	63
720	8	34.4	48	72	80	72
810	9	38.7	54	81	90	81
900	10	42.9	60	90	100	90
990	11	47.2	66	99	110	99
1080	12	51.5	72	108	120	108
1170	13	55.8	78	117	130	117
1260	14	60.1	84	126	140	126
1350	15	64.4	90	135	150	135
1440	16	68.7	96	144	160	144
1530	17	73.0	102	153	170	153
1620	18	77,3	108	162	180	162
1710	19	81.6	114	171	190	171
1800	20	85.9	120	180	200	180

Table 1. Shows the total traffic demands which are the sum of all src-dest demands. In addition, the table shows the average link load in the normal (failure-free) case and the maximum load on any link for all methods (including normal). It should be noted that the last demand with no overload in the normal case is 1440.

Figure 3a shows the average total network throughput over all link failures. It should be noted that total demand of 1440 is the last point where the normal case is not overloaded. It should also be mentioned that a network operator would hardly run the network with this high load without an upgrade.

We observe that all methods recover all traffic until total demand is 990 and the average link load is 47.2. From that point we observe that RRL performs close to the full global recovery reference. The difference between RRL with 3 layers and RRL with 6

layers is negligible. Figure 3b shows the throughput for the different link failures for RRL with 3 layers. This figure serves to illustrate that the throughput will be dependent on what link is failing and the original load on that link.

Figure 4a shows the same as figure 3a but with no recovery as reference point, i.e. it shows the amount of recovered traffic. Figure 4b shows throughput for the worst case link failure scenario. The same tendency is observed here, however with a slightly increased difference between RRL with 3 and 6 layers. Indications of this difference can be found in table 1 for max load of each method.

Also note that no recovery can perform better than the recovery methods in cases where the failure-free case has overloaded links.

Fig. 5 shows the average distribution of link loads over all link failures. Three different levels of demand are represented (900, 1260 and 1620). The figures give the number of links within each interval of loads, the interval size being 20.

We observe that all methods, but the normal have two links with load 0, which are the two directions of the failed bi-directional link. Otherwise we observe that the recovery methods give almost the same distribution as the normal (failure-free) case for modest demand. When the demand increases, the recovery methods have more links with higher load, including overload. However, mutually the recovery methods seem to give close to equal distribution.

Fig. 6 shows the link load distributions for the worst case link failure scenario. The figures show that the difference between the normal (failure-free) case and the recovery methods has increased compared to the average figures. In addition, we see that RRL with 3 layers gives more links with higher loads than full global recovery and RRL with 6 layers. These differences are also indicated in table 1.

Concerning RRL and the number of layers, we have observed that there are minor performance differences between 3 or 6 layers, respectively. One difference is that RRL with 6 layers gives the same performance as full global recovery when it comes to worst overloaded link (table 1). There exists a dependency between the number of layers and the amount of state that has to be stored in a router. Since there are minor performance differences between 3 layers and 6 layers, advising 3 layers is natural.

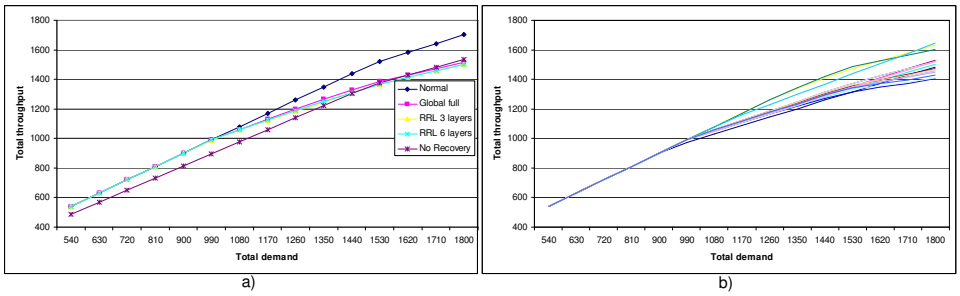

Fig. 3. a) shows average (over all link failures) throughput for the normal case (failure-free) and the different recovery methods in absolute values. b) shows the throughput for RRL3, where each line represents the throughput during a certain link failure.

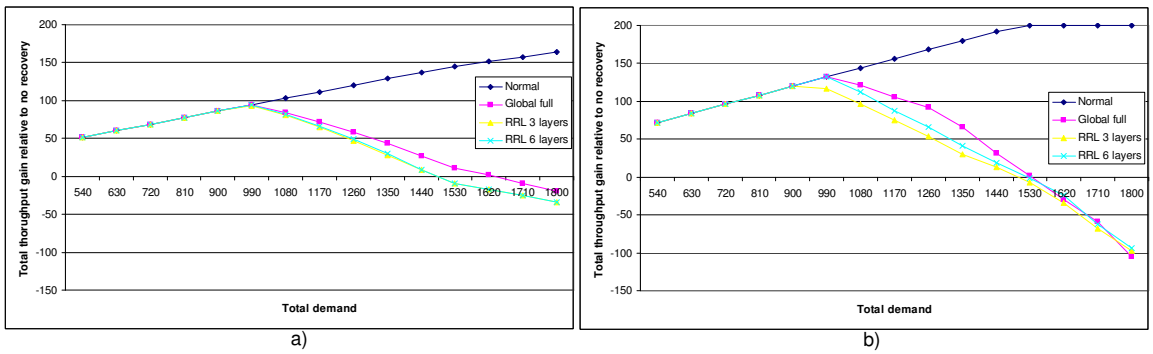

Fig. 4. a) shows average (over all link failures) added throughput for the normal case (failure-free) and the different recovery methods with no recovery as reference point. b) shows worst case for each method

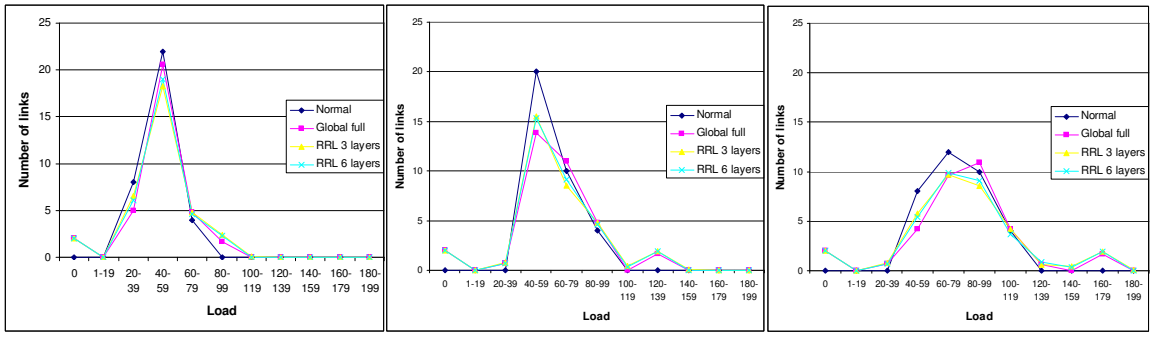

Fig. 5. Shows the average (over all link failures) distribution of link loads for the normal case (failure-free) and the different recovery methods, and for demands 900, 1260 and 1620 (left, center, right). The measure interval is 20

5 Conclusion and Future Work

RRL is proposed as a recovery method for handling transient failures and also for recovery of traffic before IP rerouting has completed and converged. We have seen that RRL performs close to what intuitively should be the most optimal method with respect to throughput; global full recovery. Pre-configured global recovery is however not a viable option for connectionless networks like IP networks. RRL on the other hand provides pre-configured milliseconds recovery with fully acceptable performance on both throughput and link load distributions. Based on these observations one may propose extending the transient period, i.e. suppressing IP rerouting for a time, and hence providing a more stable IP behavior.

Many network providers run their network with very moderate loads in the failure-free situation to make sure that a failure does not cause uncontrolled and severe traffic loss and overloads. In addition, recovery is often performed at a lower level and with coarser granularity than IP due to the slow converging of global IP rerouting. RRL and also other methods seem to elude this slow recovery of IP, and hence enable differentiation at finer granularities. If only certain amount of the traffic need time-guaranteed recovery, the normal loads could be tuned higher than what is often the situation today. Table 1

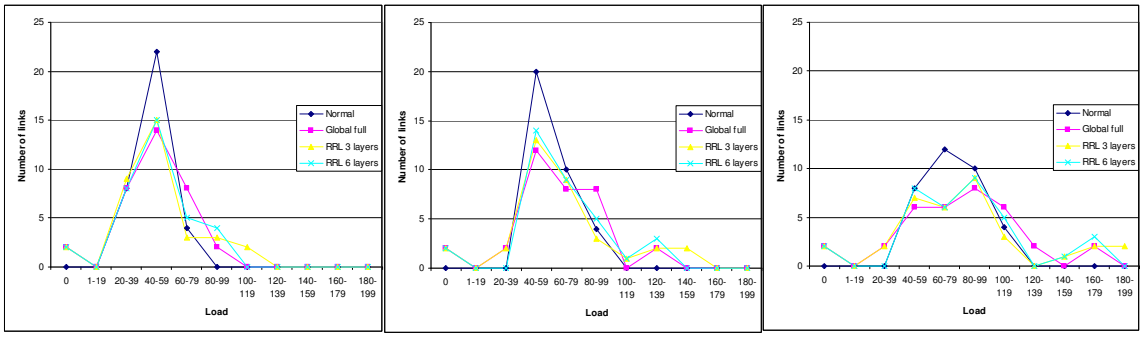

Fig. 6. Shows the worst case (failure of worst link) distribution of link loads for the normal case (failure-free) and the different recovery methods, and for demands 900, 1260 and 1620 (left, center, right). The measure interval is 20

shows the maximum load for the different methods. The amount of traffic that can be guaranteed is then determined by link capacity minus overload.

Neither RRL layer generation nor RRL layer routing have yet been optimized for load balancing of traffic. This will be future work. In addition, we will evaluate RRL with respect to other traffic assumptions than have been elaborated in this paper. This includes different traffic classes and different traffic types. In addition, it could be interesting to study other performance parameters like packet loss and delay/jitter.

References

1. Iannaccone, G., Chuah, C.N., Mortier, R., Bhattacharyya, S., Diot, C.: Analysis of link failures in an ip backbone. In: 2nd ACM SIGCOMM Workshop on Internet Measurement. (2002) 237–242
2. Basu, A., Riecke, J.G.: Stability issues in OSPF routing. In: SIGCOMM. (2001)
3. Iselt, A., Kirstadter, A., Pardigon, A., Schwabe, T.: Resilient routing using ECMP and MPLS. In: Proceedings of HPSR 2004, Phoenix, Arizona, USA (2004)
4. Liu, Y., Reddy, A.L.N.: A fast rerouting scheme for ospf/is-is networks. In: Proc. of ICCCN. (2004)
5. Schollmeier, G., et al.: Improving the resilience in ip networks. In: IEEE HPSR, Torino, Italy (2003) 91–96
6. Nelakuditi, S., Lee, S., Yu, Y., Zhang, Z.L., , Chuah, C.N.: Fast local rerouting for handling transient link failures. Submitted to IEEE/ACM Transactions on Networking (2005)
7. Kvalbein, A., Hansen, A.F., Cicic, T., Gjessing, S., Lysne, O.: Fast recovery from link failures using resilient routing layers. In: to appear at the 10th IEEE Symposium on Computers and Communications (ISCC 2005), La Manga, Spain (2005)
8. Nucci, A., Schroeder, B., Bhattacharyya, S., Taft, N., Diot, C.: IGP Link Weight Assignment for Transient Link Failures. In: International Teletraffic Congress (ITC 18), Berlin, Germany (2003) 321–330
9. Hansen, A.F., Cicic, T., Gjessing, S., Kvalbein, A., Lysne, O.: Resilient routing layers for recovery in packet networks. In: to appear at The International Conference on Dependable Systems and Networks, Yokohama, Japan (2005)
10. Theiss, I., Lysne, O.: FROOTS - fault handling in up*/down* routed networks with multiple roots. In: Proceedings of the International Conference on High Performance Computing (HiPC 2003). (2003)
11. Heckman, O.: Topologies for ISP Level Network Simulation. (http://dmz02.kom.e-technik.tu-darmstadt.de/~heckmann/index.php3?content=topology)
12. Psenak, P., et al.: MT-OSPF: Multi Topology (MT) Routing in OSPF. In: IETF, Internet Draft (2004)
13. Menth, M., Martin, R.: Network resilience through multi-topology routing. Technical Report 335, University of Wurzburg, Institute of Computer Science (2004)
14. Kelly, F.P.: Blocking probabilities in large circuit-switched networks. Statistical Laboratory, University of Cambridge, England (1985)
15. Whitt, W.: Blocking when service is required from several facilities simultaneously. AT&T Technical Journal **64** (1985) 1807–1856