

Resilient Routing Layers and Optical Burst Switching

Stein Gjessing

Dept. of Informatics, University of Oslo & Simula Research Laboratory,
P-O. Box 1080, N-0316 Oslo, Norway

steing@ifi.uio.no

Abstract – *Two important challenges in Optical Burst Switching (OBS) are to reduce the packet loss and to achieve resilience in case of link or switch failure. In this report we build on a previously published method to increase resilience in IP networks, called Resilient Routing Layers (RRL). Based on RRL, we develop a new method that improves the burst loss probability and the resilience of OBS networks. We show how this new method provides loop free deflection routing in OBS networks in case of congestion. We have made a detailed discrete event simulation model of OBS networks and evaluated the performance of our new mechanism by simulating several realistic network topologies. The simulated traffic load is mainly Poisson distributed bursts, but also self similar IP-traffic is used. The paper also briefly indicates the difference in performance when so called different sets of RRL backup topologies are used.*

Keywords: *Optical networks, Optical burst switching, Network Resilience, Burst loss, Deflection routing*

1 Introduction and motivation

Optical Burst Switching (OBS) is a much research paradigm for the next generation optical Internet [1]. OBS is performed on Wavelength Division Multiplexed (WDM) fibers, is more fine-grained than optical line switching, and more coarse-grained than Optical Packet Switching (OPS). The main motivation behind optical switching is to transport the data with minimal delay by keeping it in the optical domain. Packets (e.g. IP packets) are assembled into bursts in the optical network ingress nodes. Data in one burst is destined for the same optical network egress node. A control packet precedes the data burst in the network and reserves resources on the links and in the switches for the succeeding data burst. Only the control packet is converted from optical to electrical (and back) in each switch.

A major challenge in OBS networks is to reduce the burst loss. Scheduling algorithms have been developed that utilizes the channels on the output link as good as possible, and in this paper we assume such “Just Enough Time” scheduling [2]. This means that the control packet

reserves, on the output fiber, exactly the time slot the burst will later need.

When the time slot that a burst needs on the fiber is not (completely) available, the simplest approach is to discard the burst. By installing wavelength converters, the chance of finding an available time slot increases dramatically [3]. In this paper we assume full wavelength conversion, meaning that the output wavelength is independent of the input wavelength. In addition to wavelength conversion, also fiber delay lines and deflection [4] are used to reduce the burst drop probability. Fiber delay lines are not considered in this report.

The topic of this paper is deflection. Previous deflection methods have two main drawbacks [5,6,7]. Some methods deflect bursts in a random direction, which might be counterproductive when considering the destination of the burst. The burst may also return to the point from which it was originally deflected, which may cause even higher congestion.

Deflection is performed differently according to the underlying routing scheme. If connection oriented routing is used, deflection can be performed by using an alternative connection. However, to calculate and store such connection oriented alternative paths for all switches and all destinations are not an easy task. In this paper we assume bursts are forwarded by a connection less routing mechanism, and also that deflection is performed connectionless. We are not aware of any previous connectionless methods that have been proposed for the deflection of bursts that guarantee that bursts will not loop in the network.

Our new deflection method is based on a network resilience method called Resilient Routing Layers (RRL), developed by the author and several others [8,9]. RRL was originally intended to re-route packets (connectionless) in the case of a network element failure. In OBS-networks it will instead be used to re-route (i.e. deflect) bursts when there is a contention on the primary output link. The resilience properties of RRL are still possible to exploit. Hence we will be able to use RRL in OBS networks both as a deflection mechanism and as a resilience mechanism.

This paper is organized as follows. In the next section we review Resilient Routing Layers (RRL). Then in section 3 a short description of how RRL may be implemented as a deflection mechanism is presented. In section 4 we present our performance evaluation method. In sections 5 and 6 we show the performance of RRL in OBS networks using three different network topologies. In section 7, we briefly discuss how different RRL backup topologies may result in different performance in case of deflection, and in section 8 a self similar IP traffic arrival process is compared to a Poisson distributed burst arrival process. Finally in section 9 we conclude and point to several topics that need more research.

2 Resilient Routing Layers (RRL)

Resilient Routing Layers (RRL) is based on the notion of fully connected sub-graphs. The sub-graphs are called *backup topologies*, and in each backup topology a number of the links in the original topology are removed, while all nodes (switches) are still present. We say that a removed link in a backup topology is *protected* in that topology. In a *complete* set of backup topologies all links in the original topology are protected at least once. Figure 1 shows a full (the original network) topology on top, and a complete set of 3 backup topologies below. In order to identify the topologies, the original (full) topology is numbered 0, and the backup topologies are numbered from 1 and up.

A number of papers on RRL, and a related technology called Multiple Routing Configurations - developed by the same authors - have been written [8,9,10]. We have also devised algorithms to find complete sets of backup topologies for a given network. The sizes of these sets have been shown to be surprisingly small; we have never come across a (normal) network that needs more than 6 backup topologies (Pure ring topologies need the same number of backup topologies as there are links in the ring).

RRL is connectionless, and it is reasonable to use a shortest path algorithm when forwarding tables are created. However, more sophisticated algorithms e.g. based on knowledge about the traffic matrix, may be used as well (e.g. [11]).

When using RRL in an IP-like network, the router that is about to send a packet out on a faulty link, has to find another way to the destination for that packet. Since all nodes are connected in all backup topologies, the switch can send the packet out in any of the backup topologies that does not contain the faulty link. In order to do this, it needs a forwarding table for each backup topology. These are precomputed together with the forwarding tables of the main topology (number 0).

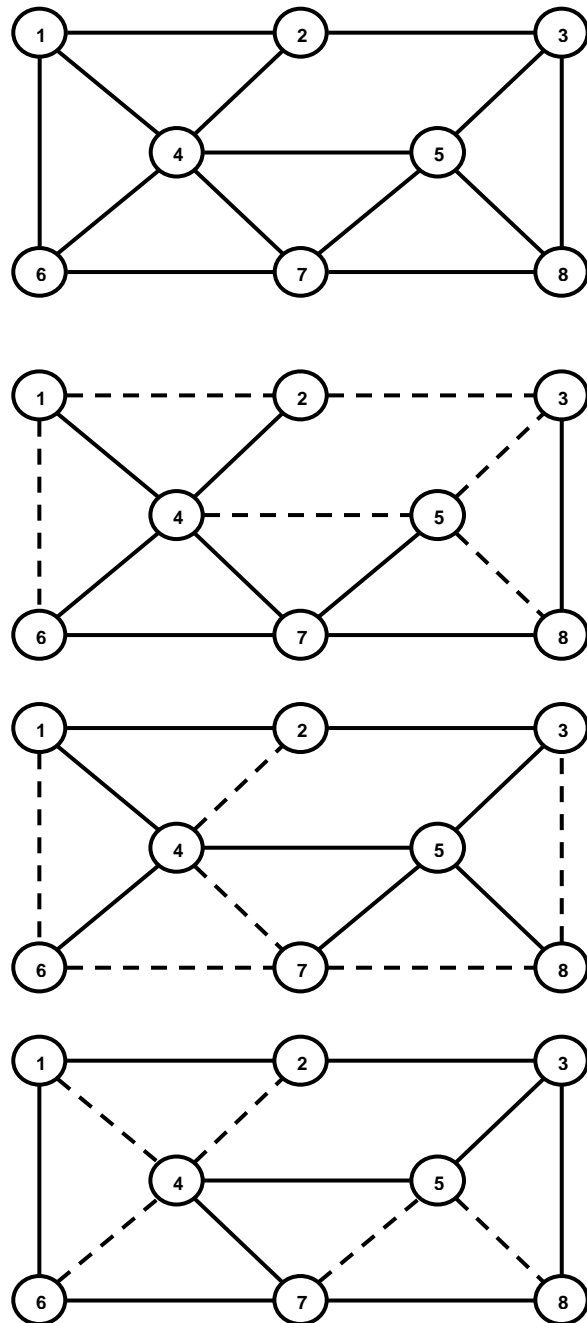


Figure 1. Original network on top, and three backup topologies below. Protected links are dashed. Notice that all links are dashed at least once.

A simple way to prevent a packet that has been sent out in a backup topology to loop in the network, is to not allow it to revert back to the original topology. It must hence be routed in the backup topology all the way to the destination. It may, however, if it encounters a second fault, again change to another backup topology, but this must be performed with care to avoid looping. We propose to use the already indicated ordering of backup topologies,

and when the highest numbered topology has been reached, no more backup conversions are tried (and the packet is discarded instead of routed any further). This guarantees that a packet will never loop in the network.

A number in the packet header tells which topology the packet is currently routed in. Whenever a packet arrives at a router, this topology number is extracted and the corresponding routing table is used. All packets are initially routed in topology 0, unless the ingress node has received an explicit notification of a faulty link, and wants to send the packet all the way to the destination in a backup topology that does not contain this faulty link.

In Figure 1 there are one original and three backup topologies. Hence each router will have four routing tables (that might be optimized to fill less space than four times what is needed for one routing table).

Given the (original) topology of the network, our previously published ([8]) algorithm calculates a set of backup topologies and associated routing tables. In the algorithm it is possible to specify the wanted number of backup topologies. Only if the number is small and the algorithm does not manage to make a complete set with that few topologies, the number of backup topologies is increased. When the algorithm is asked to make a large number of topologies, these topologies may either be dense or sparse. In a dense topology there are many routes to the destination, hence this seems like the best method in order to create short backup routes. However, sparse topologies may also have some advantages when it comes to load distribution. This is not a main topic in this report, but it will be briefly discussed in some more depth in section 7.

RRL in its most basic form only guarantees a viable route to the destination when a packet encounters one faulty link. It is, however, possible to select pairs of links to protect, so that even after two deflections, a loop free path to the destination can be found. Also the chance of surviving even a third fault is quite large [12].

3 RRL in OBS networks

The packet routing and forwarding mechanisms used by RRL as described above, are easily transformed to control packets, bursts and scheduling in an OBS network.

Each switch contains the original routing table, as well as one routing table for each of the backup topologies. When a control packet is transmitted from the ingress node, it (and the burst) is routed in the original topology (topology 0).

When the control packet arrives at a switch, the burst scheduler first tries to schedule the corresponding burst on a wavelength on the outgoing link according to the original topology routing table (assuming this is a 0 topology burst).

If there is no available slot for the burst on the original output fiber, deflection, using one of the backup topologies, is tried. Any topology that does not contain the original link, is a candidate. If there exist more than one backup topology in which the original link is protected, the one with the lowest number is tried first. The routing table of this chosen backup topology is found, and from this routing table a secondary output link is found (which is not the same as the original, because the original link is not present in this backup topology). Then the burst scheduler tries to find a free wavelength on this link. If this succeeds, the scheduler reserves a slot for the burst on this link and sets up the switching so that when the burst arrives it will be deflected to the correct output link. The topology number in the control packet is assigned the corresponding backup topology number, and the control packet is also forwarded on the same secondary link.

If the scheduler is not successful in finding an available time slot on the output link according to the first chosen backup topology, another backup topology, with a higher number, may be tried. In this way several output links may be tried, until all output links are tried or the number of backup topologies is exhausted. If no available output link is found, the control packet (and the burst) is discarded.

When a control packet (and its burst) is deflected, it continues with a topology number different from 0. All the way to the egress router it is forwarded according to the routing table of this backup topology. In order not to loop in the network, it may not revert back to the original topology. However, if a burst scheduler finds that the burst may not be forwarded according to the present topology of the burst, (there is no time slot available on this output link), another deflection might take place. To be sure that looping is avoided, the burst must not revert back to a topology that has been used previously. In order to avoid this, only higher number topologies are tried. In this way, the burst may be deflected several times, until all topologies have been tried. Only then is the control packet (and the burst) discarded.

Deflection may in general cause bursts to overtake their control packets. This will also be the case for our method, if the deflected route is too long (passes too many switches). However, one advantage with our technique is that bursts are always forwarded towards the burst's egress router. Hence unnecessary looping will not occur. In the sequel we will report on the performance of our method, and we have never encountered a lost burst because a burst overtook its control packet.

Our deflection method, based on RRL, can also be used to protect traffic in case of a link failure. When a link fails, the burst scheduler will be notified, and it will not be able to schedule any bursts on this link. Then all bursts, having this link as its primary output, will be deflected as if there is a contention on this link. RRL may also be used to protect

OBS traffic in case of a switch failure. Then the backup topologies must be created so that also switches are protected [10]. This will, however, not be treated in more detail in this report.

4 The evaluation environment

We have used the J-sim framework [13], and implemented a full OBS discrete event simulation model on top. The data sources and burst assembly modules, as well as the OBS-switches and schedulers are built from scratch. Topologies, link propagation times and forwarding tables for the specific scenarios are read from files at system start up time.

All traffic is synthetically generated. Each ingress node has as many Poisson processes as the number of destinations (optical network egress nodes), generating fixed sized bursts (a burst size of 50 000 bytes is used in this report). Depending on the traffic matrix, the mean arrival rate of the bursts destined for one and the same egress node is determined.

Our OBS simulator may also generate self similar IP traffic. IP packets are then assembled into variable or fixed sized bursts. A time-out parameter (2 ms. in this report), and, in the case of variable sized bursts, a maximum burst size is used (50 000 bytes in this report). The destination address of each IP packet is set depending on the traffic matrix. We produce a self similar arrival process using a large number of Pareto sources in each ingress node. Whenever a Pareto source starts a new on-period, a destination address is chosen according to the traffic matrix. This address is used for all packets generated by this source for the duration of this on period. In the experiments used in this article an equal all-to-all traffic matrix between ingress and egress nodes is used.

While the bursts are kept in the optical domain, and use very short time through a node, the control packet delay is 10 μ s in each node. The control packet lead time (CPT, i.e. how long ahead of the burst the control packet is sent) is varied from 100 to 250 μ s, depending on the diameter of the network (in number of switches). Hence, if a burst loops in the network, it will overtake the control packet (and they both become discarded) in between 10 to 25 hops.

All experiments reported in this article are set up with equal capacity links. Each link has 10 channels (lambdas) and each channel has a capacity of 1 Gbit/sec.

5 Performance evaluation

When evaluating our new burst deflection method based on RRL, we will compare it to two other well known deflection methods: *Hot Potato* deflection chooses an

alternative output link at random. There are many ways to implement this, but in the experiments reported in the sequel, each time a packet has to be deflected, all of the other output ports are equally likely to be chosen. This means that a packet may also be deflected back to where it came from. *Second Shortest* path deflection tries to output the bursts to the output link where the next switch has the shortest distance to the destination (excluding the primary output link). Hence, this method could more precisely be called the neighbor next-shortest path. Notice that in general we can not guarantee freedom from looping by this, because it might be the case that the shortest path from the chosen neighbor goes back to the deflecting node itself.

For these two deflection methods (Hot Potato and Second Shortest) indefinite looping in the network is only prevented by the fact that when the data burst is overtaking the control packet they are both discarded. In the case that the control packet is not able to reserve the needed resources for the data burst at all (deflection is not possible), the data burst (and the control packet) is discarded by the switch.

In addition to comparing our method to Hot Potato and Second Shortest, we also compare with *Regular* burst dropping, i.e. when a burst may not be scheduled on the primary output link, it is immediately discarded (no deflection).

For each experiment we have only made one run with one seed. However experiments with different parameters are uncorrelated, hence when we see specific patterns in the plots, it is reasonable to believe that those patterns are statistically significant. In future work we will improve our experiments in order to establish results with better statistical confidence.

In this paper the deflection mechanisms are compared based on performances in three networks with different characteristics; the Pan-European COST 239 network [14] and two networks from the Rocketfuel project from Washington University [15]; the Exodus network and the Sprint US network.

All nodes are ingress nodes (generating traffic), egress nodes and internal switching nodes in the network. The traffic matrix is symmetric all-to-all. This may create bottlenecks that would normally not be seen in a well balanced network. However the purpose of the experiments is to evaluate the performances of the methods when burst loss is a problem. We need to observe a varying degree of burst loss, and hence a well balanced network with few or no burst losses is of less interest.

In the three next subsections, the different network topologies that are used in the experiments, are described.

5.1 The Cost network

The COST 239 network is a proposed Pan-European core network topology consisting of 11 nodes (European cities) connected by 26 (bidirectional) links, as illustrated by figure 2. The propagation delays are estimated based on the distances between the cities.

The control packet lead time used by the ingress switch is $200\mu\text{s}$. Since the control packet forwarding time is $10\mu\text{s}$, a burst can be forwarded through 20 switches before it must be discarded because the CPT becomes zero and the burst overtakes the control packet.

The routing is performed according to a shortest path algorithm, taking into account the propagation delay on the links according to figure 2.

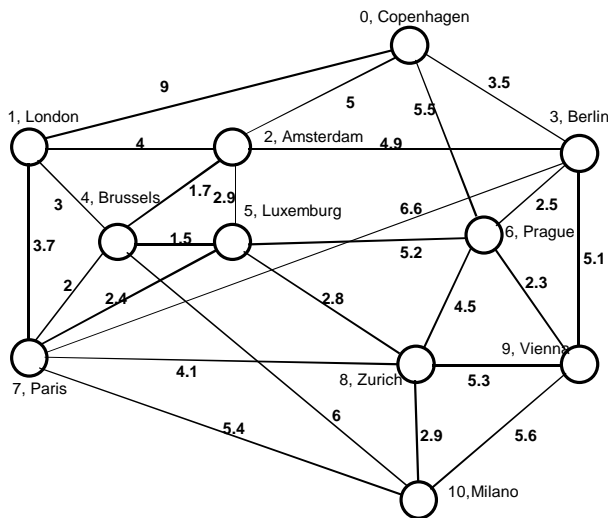


Figure 2. The COST 239 network between 11 European cities. Propagation delay in milliseconds on each link.

5.2 The Exodus network

The Exodus network is described by the Rocketfuel project and is AS number 3896. By collapsing switches in the same cities, and also collapsing parallel links, we have reduced the network to 17 nodes connected by 29 links. The link latencies vary from 2 to 15 ms. Initial control packet lead time is set to $150\mu\text{s}$.

5.3 The Sprint network

The second network we use from the Rocketfuel project is the Sprint US network (AS 1239). Also this network we have reduced, this time to 45 switches and 95 links. Link latencies vary from 2 to 64 ms. Initial CPT is set to $250\mu\text{s}$.

6 Simulation results

In this section we report the simulation results from running the three deflection methods, RRL, Hot Potato and Next Shortest as well as no deflection (Regular). The burst arrival process is Poisson.

6.1 The Cost network

Figures 3 and 4 show the results from the Cost network. We have increased the load up to an unrealistically high value in order to also see how the methods behave under extreme loads.

Notice how the Regular method is better than all the others when the load increases above approximately 1.1 million bursts per second. Above this value the network is so congested that any deflection will only increase the traffic in the network and make things worse.

Notice also how Hot Potato deflection performs best for low loads, but how this method performs terrible when the load is high. At about 4% burst loss probability, Hot Potato is no longer the best method.

Finally observe that RRL and Next Shortest perform almost identical for all load values, although Next Shortest seems to always perform slightly better than RRL. These methods are the most stable ones, meaning that they perform quite well for all loads.

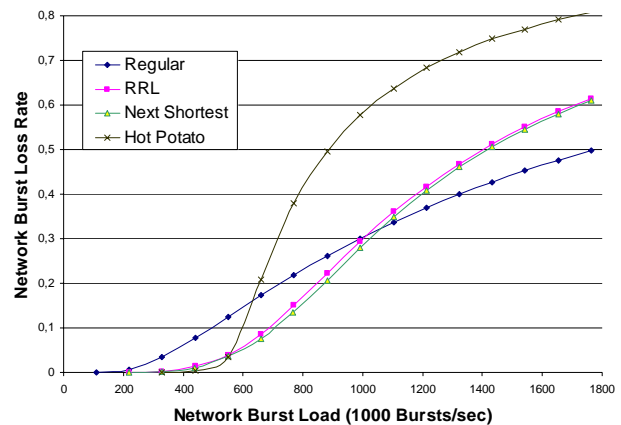


Figure 3. Burst loss probability in the Cost network with increasing Poisson distributed burst load.

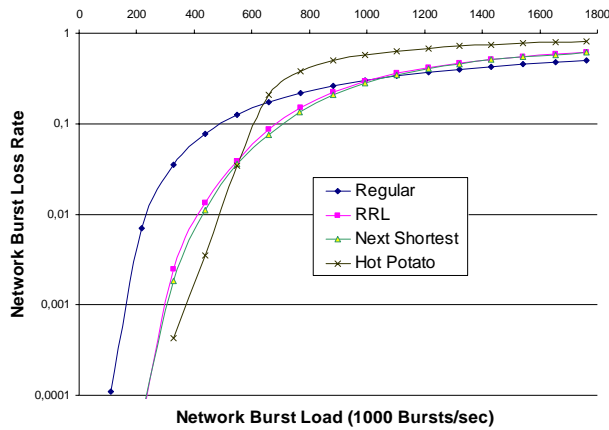


Figure 4. Burst loss probability in the Cost network with increasing Poisson distributed load (logarithmic y-axis)

6.2 The Exodus network

We simulated the performance of the Exodus network only for reasonably small loads. The results are depicted in figures 5 and 6. Again, notice that Hot Potato deflection performs best.

Also in this network Next Shortest and RRL perform about the same. For very load loads, Next Shortest is best, but a little above 1% loss rate, RRL is the better of the two.

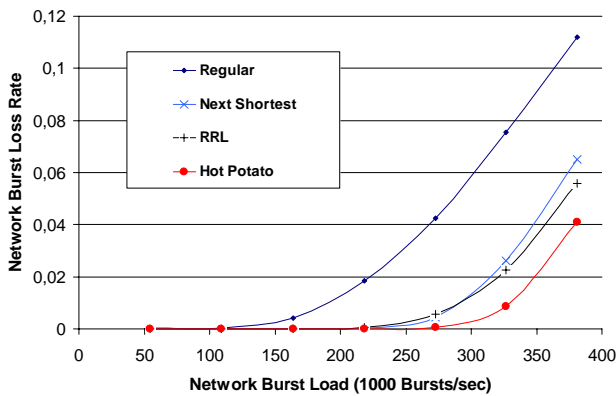


Figure 5. Burst loss probability in the Exodus network with increasing Poisson distributed burst load.

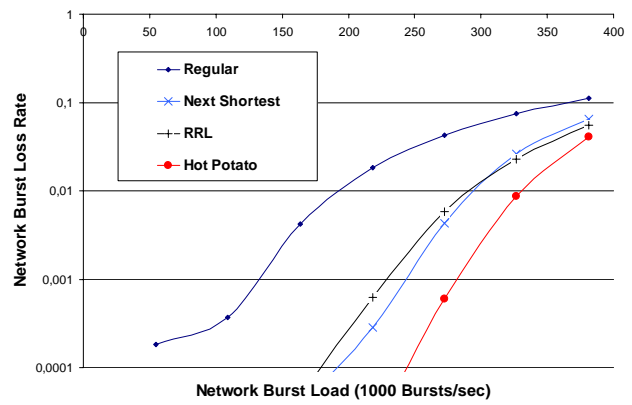


Figure 6. Burst drop rate in the Exodus network with increasing Poisson distributed load (logarithmic y-axis)

6.3 The Sprint network

The sprint network was, as the Exodus network, run with a wide range of loads, and, as seen in figures 7, 8 and 9. The overall picture is the same, although not as distinct. Also here the Hot Potato deflection method performs worst for high loads. However, this time Hot Potato deflection is not best for low loads. Also notice that at the very end of the x axis in figure 7 and 8, Regular is slightly better than the others. Again RRL and Next Shortest are the closest competitors. However, for burst loss up to 10%, this time RRL is clearly best, as can best be seen from figure 9.

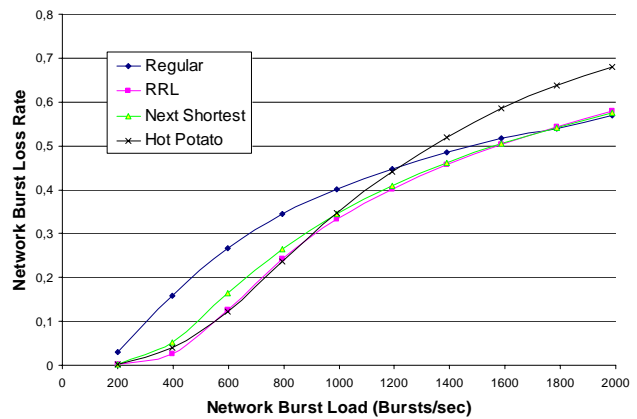


Figure 7. Burst loss probability in the Sprint network with increasing Poisson distributed burst load.

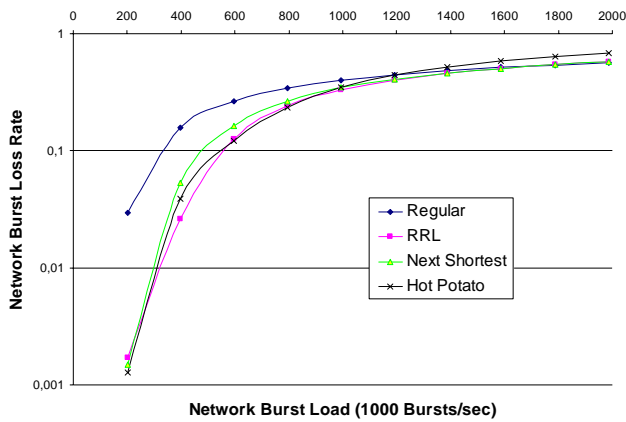


Figure 8. Burst loss probability in the Sprint network with increasing Poisson distributed load (logarithmic y-axis)

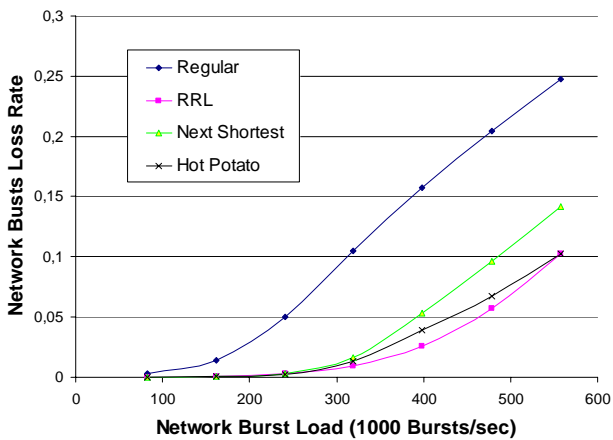


Figure 9. Burst loss probability in the Sprint network with increasing Poisson distributed burst load, low load values.

7 Different sets of backup topologies

So far we have not been concerned with the properties of the RRL backup topologies as long as we have a complete set. In this section we conclude that we should be concerned, i.e. different sets have different performance characteristics.

The algorithm that creates a complete set starts with low numbered links (i.e. between low numbered nodes) and is greedy in the way that it grabs as many links as possible and tries to protect them in the same topology [10]. For the Exodus network, we created one more complete set of backup topologies by renumbering the nodes and then re-run the algorithm. Both times we asked the algorithm to produce 7 topologies where each link is only protected once. When we simulated the network using this new set of topologies we got the result denoted “RRL 7-2” in figures

10 and 11. The set already used in section 6 is denoted “RRL 7-1”. Notice that the new set performs much worse than the previous set.

We also made a complete set of backup topologies for the Exodus network by hand. This time a set of size 4 was created. Making this set we tried to be “clever” in the sense that when we protected a link in a topology, we tried to make sure that reasonable backup paths existed. The performance using this hand made set is denoted “RRL 4 - Hand made” in figures 10 and 11. Notice that, for the simulated load values, this set performs about the same as “RRL 7-2”, although one should expect that a larger set should perform better than a smaller one. It is hard to tell if “RRL 7 - 2” is a particularly bad set, or “RRL 4 - Hand made” is good because it was created with some intelligence.

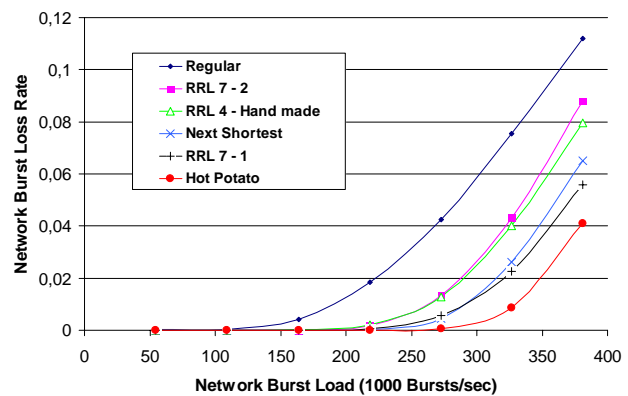


Figure 10. Burst loss probability in the Exodus network with two additional sets of topologies.

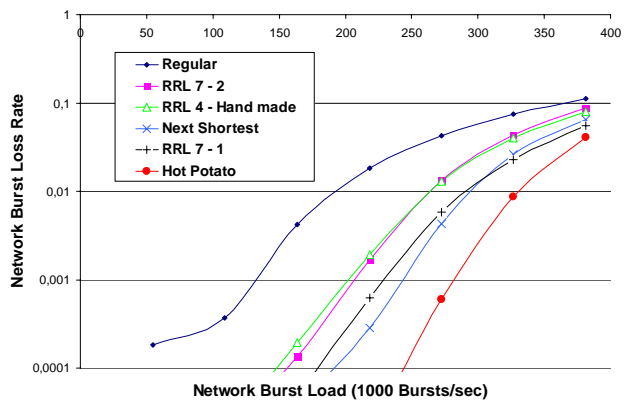


Figure 11. Burst loss probability in the Exodus network with two additional sets of topologies. Logarithmic y-axis.

8 Self-similar IP traffic load

The traffic load onto an OBS core network may come from IP-subnets and Ethernets. It is well known that Ethernet and IP traffic exhibit self similar properties. The arrival process of IP packets arriving to the ingress nodes of an OBS core network is self similar. In our simulator we implement a self similar arrival process using 100 Pareto sources in each ingress node [16]. Each source draws on and off periods according to a Pareto distribution with Hurst parameter 0.9. Whenever the source is in the on mode, IP packets are generated with constant intervals of 10.7 μ s. The size of the IP packets is varied from 80 to 1600 bytes, with a mean of 500 bytes. If a burst is not completely filled when the time out timer (that is set to 2 ms.) expires, the burst is sent with its current size, up to a maximum of 50 000 bytes.

We then recreated the experiment in section 6.1 (the Cost network), using self similar IP-traffic load instead of Poisson distributed burst load.

The results are depicted in figures 12 and 13. With 100 Pareto sources per ingress station, the maximum load it is possible to send into the network from the 11 ingress nodes is about 800 000 bursts/sec. As long as the total traffic generated is below 150 000 burst/sec (i.e. each of the 11 ingress nodes generates about 5.5 Gbit/sec), there is no packet loss anywhere in the network. When the load has increased to 300 000 bursts/sec, the burst loss probability for the regular method is above 1%, while all the deflection methods still yield very good results. Notice from figure 12 that the Regular burst loss probability increases almost linearly above 3%, while there is a very distinct change in the increase for the Hot Potato deflection method, also at about 3% loss probability. Here the loss probability of this deflection method starts to increase steeply, while the loss probability of RRL and Next Shortest continue with a much smaller increase in the loss rate.

Running the experiments, we noticed that only for very low loads, the burst were sent with a smaller size than the maximum of 50 000 bytes. Hence the result of running with fixed sized bursts (of the maximum size) will not give significantly different results in the burst loss probability (except for very small loss rates at very light loads).

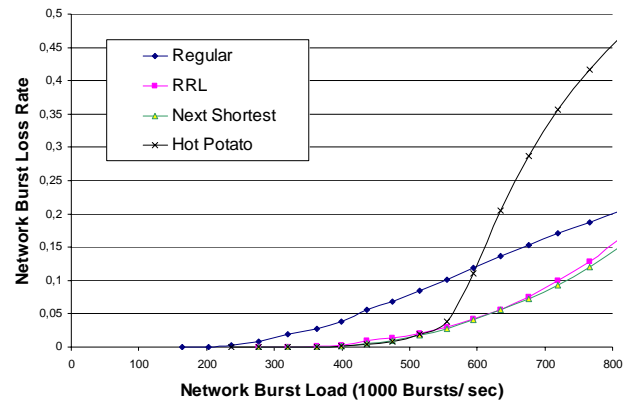


Figure 12. Burst loss probability in the Cost network with increasing self similar IP traffic

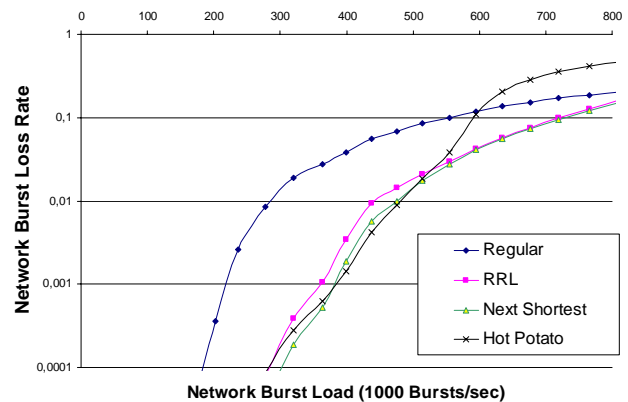


Figure 13. Burst loss probability in the COST network with increasing self similar IP traffic (logarithmic y-axis)

In order to compare the burst loss probability of aggregated self similar IP traffic and Poisson distributed burst traffic, we also used the Cost network. Figures 14 and 15 show the results from the Regular method and the RRL method for these two arrival processes in the same plot. For these experiments, all bursts are of equal and maximum size. Notice that the self similar and the Poisson curves follow the same trends, but that the performance of the Poisson traffic is a little worse than the aggregated IP-traffic. I.e. for the same load, the Poisson traffic has greater loss rate than the aggregated IP-traffic. This result is only a side issue in this report, but never the less very interesting. At this point in time we can only speculate that the reason is that the arrival rates of bursts into the network, caused by the aggregation of self similar IP traffic is smoother than Poisson distributed burst arrivals.

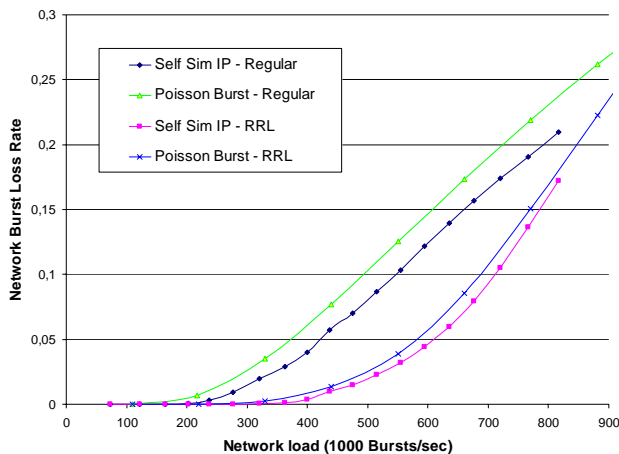


Figure 14. Comparing aggregated self similar IP traffic and Poisson distributed bursts in the Cost network.

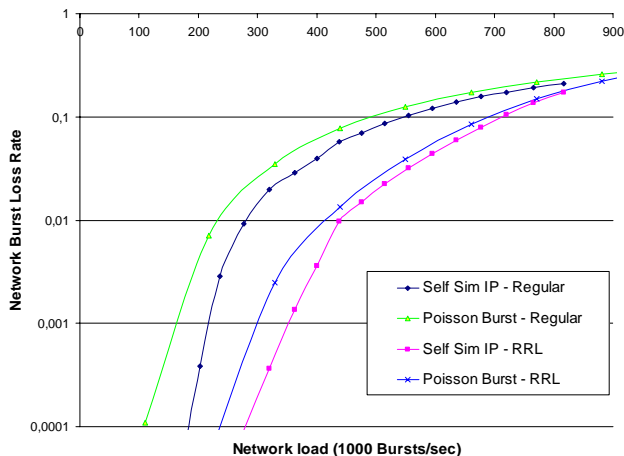


Figure 15. Comparing aggregated self similar IP traffic and Poisson distributed bursts in the Cost network (logarithmic y-axis)

9 Conclusions and Further Work

This paper has discussed how to reduce the burst loss rate in Optical Burst Switched (OBS) networks, using Resilient Routing Layers (RRL) as a deflection technique. Packets re-routed in a backup topology are guaranteed not to loop in the network. Hence, by using RRL, looping, that can be caused by other deflection techniques, is avoided. When RRL is used in OBS networks in this way, RRL can also be used as originally intended, i.e. to protect and re-route traffic in a loop less manner in the case of a link failure.

Our method is compared to “Next Shortest” path and “Hot Potato” deflection, as well as with no deflection (just discarding bursts that can not be scheduled on the output link). Three topologies with different characteristics have

been used, with number of switches varying from 11 to 45 and ratio between links and switches varying between 1.7 and 2.3. The traffic matrix used has been symmetric, all-to-all, and all links have the same capacity. The load onto the network has been all synthetic. The main arrival process has been Poisson distributed, but also self similar IP traffic has been used. In order to experience burst loss, the load has been varied, in some cases so much that the burst loss rate has reached unrealistic and extreme values (80%).

For the purpose of this article we have not performed any explicit resilience experiments. We rely on previous results ([9]) to support our claim that RRL also increases resilience in OBS networks.

First of all, our results show that RRL is a viable deflection method for OBS networks. By precalculating shortest paths in the backup topologies, a deflected burst is routed towards the destination along an alternative path. Our method does only guarantee that deflection can be performed once, but as long as not all topologies are tried, deflection may be successfully tried several times.

We have compared RRL deflection to Next Shortest deflection and Hot Potato deflection. Both of these latter methods may result in indefinite looping even when the rest of the network is not congested. For higher loads our experiments confirms to previous results [5], that deflection routing, and in particular Hot Potato routing, creates more network traffic, and hence makes the burst drop rate higher than Regular routing with immediate drop of packets when the primary output link is congested. RRL deflection seems to overall be comparable in performance to Next Shortest deflection. The latter has the advantage that less state is needed in the switches, since only one alternative output port is needed for each destination (egress node). RRL, on the other hand, needs as many alternative forwarding tables as the number of backup topologies. On the other hand, Next Shortest deflection may (and will in some cases) loop the packet immediately back to the point of congestion, and as long as the original output link is congested, the burst will continue to loop in the network (until discarded when overtaking the control packet). RRL makes sure that such looping never happens.

In the experiments presented in this report we have seen that the performance of RRL deflection is dependent on the specific sets of backup topologies created, i.e. both the number of topologies, and their appearance. When creating more than a minimal number of backup topologies in a set, there is a trade off between rich topologies with few links protected in each topology, and sparse topologies with many links protected in each topology. In general, with rich topologies, there are shorter paths in the network, but fewer topologies to choose from when a burst is to be deflected. On the other hand with sparse topologies, the backup paths are longer, but a burst may live longer in the

network since there are more topologies to choose from. As we have seen, this is a good property when the network load is light. By creating backup topologies that deflects traffic to lightly loaded links, RRL deflection may spread traffic more evenly in the network. In future work we will investigate how to make such good sets of backup topologies.

In section 8 we injected self similar IP-traffic into our network, and re-ran some of the experiments that had been conducted with Poisson distributed burst arrivals. The performance results from the previous runs were confirmed. However, the self similar load produced a slightly better network performance with regards to burst drop rates. This effect will also be studied closer in future work.

Acknowledgements

Thanks to Audun Fossellie Hansen who took part in the initial design of the OBS simulation model, has provided that backup topologies and programmed most of the algorithm that finds complete sets of backup topologies used in the reported experiments. The author would also like to thank Amund Kvalbein for making the Pareto source module. Our OBS simulation model is based upon another network simulator developed by him and the author. Thanks also to the rest of the team who developed RRL.

References

- [1] Y. Chen, C. Qiao and X. Yu, "Optical Burst Switching (OBS): A New Area in Optical Networking Research," IEEE Network Magazine 18, 16-23, May/June 2004.
- [2] Myungsik Yoo, Chunming Qiao, "Just-Enough-Time (JET): A high speed protocol for bursty traffic in optical networks", In Vertical Cavity Lasers, 1997 Digest of the IEEE/LEOS Meetings, pp. 26-27, Aug. 1997.
- [3] J. Ramamirtham, J. Turner, J. Friedman, "Design of Wavelength Converting Switches for Optical Burst Switching", IEEE Journal on Selected Areas in Communications, Vol. 21, No. 7, Sept. 2003.
- [4] X. Wang, H. Morikawa, T. Aoyama, "Burst optical deflection routing protocol for wavelength routing WDM networks", Proc. SPIE, 2000.
- [5] C.-F. Hsu, T.-L. Liu, N.-F. Huang, "Performance Analysis of Deflection Routing in Optical Burst-Switched Networks", Proceedings IEEE INFOCOM, pp. 66-73, 2003
- [6] Zalesky, A.; Hai Le Vu; Rosberg, Z.; Wong, E.W.M.; Zukerman, M.; "Modelling and performance evaluation of optical burst switched networks with deflection routing and wavelength reservation", INFOCOM 2004, Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 3, 2004, pp:1864 – 1871.
- [7] SuKyoung Lee; Sriram, K.; HyunSook Kim; JooSeok Song, "Contention-Based Limited Deflection Routing Protocol in Optical Burst-Switched Networks", Selected Areas in Communications, IEEE Journal on, Volume 23, Issue 8, Aug. 2005 pp:1596 – 1611
- [8] A. Kvalbein, A. F. Hansen, T. Cicic, S. Gjessing and O. Lysne, "Fast Recovery from Link Failures using Resilient Routing Layers", In 10th IEEE Symposium on Computers and Communications (ISCC 2005), pp 554-560, IEEE Computer Society, 2005.
- [9] A. Kvalbein, A.F. Hansen, T. Cicic, S. Gjessing, O. Lysne, "Fast IP Network Recovery using Multiple Routing Configurations". In proceedings IEEE 25th Annual Conf. on Computer Communications (INFOCOM) May 2006.
- [10] A. F. Hansen, A. Kvalbein, T. Cicic, S. Gjessing, O. Lysne, "Resilient Routing Layers for Recovery in Packet Networks", In Proceedings 2005 Int. Conference on Dependable Systems and Networks, IEEE Comp Soc, 2005.
- [11] Jing Teng, Rouskas, G.N., "Routing path optimization in optical burst switched networks", Optical Network Design and Modeling, pp: 1- 10, Feb. 7-9, 2005
- [12] T. Cicic, A. F. Hansen, S. Gjessing, O. Lysne, "Applicability of Resilient Routing Layers for k-Fault Network Recovery", Proceedings 4th International Conference on Networking, Lecture Notes in Computer Science (LNCS), Vol. 3421, pp 173-183, 2005.
- [13] John A. Miller, Andrew F. Seila and Xuwei Xiang, "The JSIM Web-Based Simulation Environment," Future Generation Computer Systems, Vol. 17, No. 2, pp. 119-133. Oct. 2000.
- [14] O'Mahony, M.J., "Results from the COST 239 project. Ultra-High Capacity Optical Transmission Networks", 22nd European Conference on Optical Communication, pp: 15-19, Sept. 1996
- [15] <http://www.cs.washington.edu/research/networking/rocketfuel/>
- [16] Willinger, W., Taqqu, M.S., Sherman, R., Wilson, D.V., "Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level", IEEE/ACM Transactions on Networking, Vol. 5, No. 1, pp: 71 – 86 , Feb. 1997.