Improvement of Resilient Packet Ring Fairness

Fredrik Davik
Simula Research Laboratory
University of Oslo
Ericsson Research Norway
Email: bjornfd@simula.no

Amund Kvalbein
Simula Research Laboratory
Email: amundk@simula.no

Stein Gjessing
Simula Research Laboratory
Email: steing@simula.no

Abstract— **Resilient Packet Ring (RPR, IEEE std. 802.17-2004) is a recent networking standard developed by the IEEE LAN/MAN working group. RPR is an insertion buffer, dual ring technology, utilizing a back pressure based fairness algorithm to distribute bandwidth when congestion occurs. In its attempt to control a set of nodes' sending behavior over a congested link, the RPR fairness algorithm suffers from some severe performance deficiencies. One concerns how the node closest to a congested link calculates a fair rate estimate, the other deficiency relates to the distribution of this fair rate estimate to nodes upstream from the congested node. In this paper, we analyze these deficiencies and propose improvements to resolve them.**

Keywords: Resilient Packet Ring, Fairness, Performance evaluation.

I. Introduction and motivation

RPR is a ring topology network, developed mainly for MAN environments [1], [8]. By the use of two rings (also called ringlets), resilience is ensured; if one link fails, any two nodes connected to the ring still have a viable communication path between them. When a node wants to send a packet to another node on the ring, it sends the packet onto one of the two ringlets. For bandwidth efficiency, the ringlet that gives the shortest path is used by default. When the packet travels on the ring, it *transits* all nodes between the sender and the receiver. When it reaches the destination, the packet is removed (stripped) from the ring. Hence the bandwidth that would otherwise be consumed by the packet on its way back to the sender (as is the case in a Token Ring), can be used by other communications. Such destination stripping of packets leads to what is commonly known as *spatial reuse*.

RPR uses *insertion buffer(s)* for collision avoidance [2], [3]. When a packet in transit arrives at a node that is currently sending a packet to the ring, the transiting packet is temporarily stored in an insertion buffer, called a *transit queue* in RPR. In a buffer insertion ring like RPR, a *fairness algorithm* is needed to divide the bandwidth fairly between contending nodes, when congestion occurs [4], [5].

The main contribution of this paper is the analysis and improvement of two major performance deficiencies in the RPR fairness algorithm. These deficiencies are caused by the way the rate value of so called fairness messages are calculated and distributed to nodes sending traffic over a congested link. First, we address a deficiency in the method used by the node closest to the congested link to calculate a fair rate estimate. Secondly, we address a deficiency in the method used for the distribution of this calculated rate value to nodes upstream of the congested node.

In this paper, we present only the most important parts of the problem discussion and our proposed modifications, together with some general examples. For an in-depth analysis of the two problems, refer to [6], [7].

The rest of this paper is organized as follows: in section II, we give a brief introduction to the RPR fairness algorithm. Then, in section III, we present the two RPR performance deficiencies mentioned above. Next, in section IV, we present our improvements which resolves these deficiencies. Then in section V, we present our simulation results comparing the performance of our proposed improvements to that of the original fairness algorithm. Finally, in sections VI and VII, we conclude and point out directions for future work.

II. The RPR Fairness Algorithm

Both modes (*aggressive* and *conservative*) of the RPR fairness algorithm work with a concept known as a *congestion domain*. A congestion domain defines a consecutive collection of nodes, of which some or all contribute to a sustained congestion state for a given link. The congestion domain is confined within a region specified by two boundary nodes, see Fig. 1. At one end of the region resides a node, denoted the *head*, which is attached upstream of the most congested link in the region. At the opposite end of the region resides a node, denoted the *tail*. Nodes upstream of the tail are considered as not being contributors to the congestion situation at the head.

The declaration and operation of a congestion domain can be considered a two-part problem. The first part of the problem, consists of making a node assume the role of the head. The head is responsible for, the periodic (every *agingInterval*) calculation and distribution of fair rate estimates upstream (by use of fairness messages). The second part of the problem, consists of making a node assume the role of the tail, responsible for stopping further propagation of fairness messages, received from the head.

In the case of aggressive mode fairness, the fair rate estimate is the head's own send rate. In the case of conservative mode fairness, the fair rate estimate is calculated by the head, independent of its send-rate. Regardless of the fairness mode used, the goal is to arrive at the RIAS (Ring Ingress Aggregated with Spatial Reuse) fair sharing of bandwidth over the congested link [8]. That is, when there is a congestion on a link, the capacity of that link is shared equally[1] between the nodes sending traffic over this link. The behavior of the RPR conservative and aggressive fairness modes has been studied by several [1], [8]–[12].

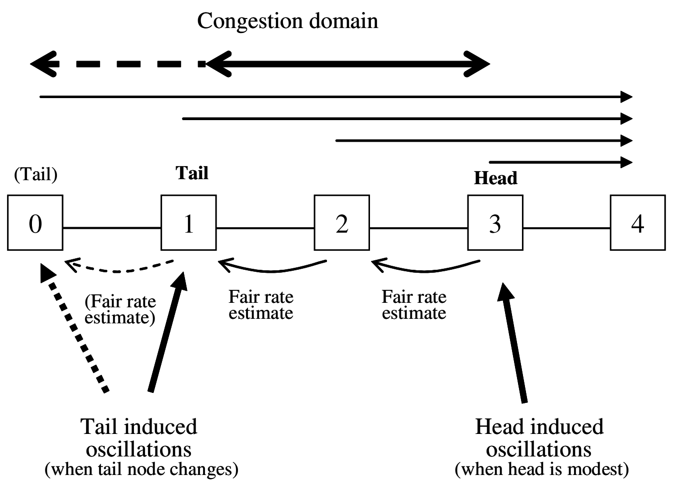


Fig. 1. Nodes 0, 1, 2 and 3 are all sending so much data to node 4 that the link between nodes 3 and 4 becomes congested. Then node 3 becomes the head of a congestion domain and calculates and distrubutes a fair rate estimate. The node furthest away and upstream from the head, contributing to the congestion, is called the tail of the congestion domain. Because of unstability in the control system, the possition of the tail node may vary between nodes 0 and 1.

III. Fairness Algorithm Performance Deficiencies

A. Problem 1 - Congestion Domain Fair Rate Calculation

In a congestion scenario, the sending pattern of the individual nodes may vary from modest (the node's demand is less than its fair share) to greedy (the node sends as much as the available bandwidth allows). Regardless of the demand of the individual nodes, the fairness algorithm should perform equally well. For aggressive mode fairness, several papers have reported severe performance deficiencies for so called unbalanced traffic scenarios [13]. The unbalanced traffic scenario is characterized by a congestion domain, where the head is a modest sender, while its upstream neighbors have a greedy sending behavior.

Fig. 1 may be used to illustrate an unbalanced traffic scenario. Assume all nodes start sending traffic to node 4, and that nodes 0, 1 and 2 have a greedy sending behavior, while node 3, using the aggressive mode fairness algorithm, has a modest sending behavior. As long as node 3 is uncongested (its transit buffer (*STQ*) occupancy is below a threshold termed *low*), it sends *full-rate* messages (i.e. fairness messages containing a rate value of *full-rate*) upstream, allowing upstream nodes to increase their send rate. As a result, at some future time node 3 becomes congested (the *STQ* occupancy exceeds the *low* threshold). When this happens, node 3 starts sending fairness messages containing its fair rate estimate (which for the aggressive mode is the node's own send rate). As node 3 sends less than its fair share of traffic, the send rate of node 3 will **always** be lower than the (RIAS) fair rate. Thus, the resulting aggregate rate reduction by the upstream nodes will be too high (i.e. the *STQ* occupancy falls below the *low* threshold before becoming empty). Once this happens, the head is by (aggressive mode) definition no longer congested and it starts to send *full-rate* messages upstream. By this, we are back to the starting point. Thus we have a cyclic behavior, leading to large oscillations in the throughput of traffic from nodes upstream of the head, as well as reduced link-utilization. We call this *Head induced oscillations* (Fig. 1).

Other groups, having looked into the problems described for the unbalanced traffic scenario, have mostly solved the problem by proposing alternate fairness algorithms that would have to replace the current RPR fairness algorithm [9], [12], [17]. Some work has presented results that seems to fit within the framework given by the RPR standard [10]. In that work however, the effect of the proposed modifications is to reduce the symptoms (oscillations) rather than removing them.

B. Problem 2 - Congestion Domain Fair Rate Propagation

In an RPR network, a node will assume the responsibility of the tail for two reasons (denoted TA, as a shorthand for Tail Appointment Condition). For both cases, we assume the node is aware of the presence of a downstream head:

TA 1: When a node finds itself more congested than the downstream head. In this case, the node becomes tail of the congestion domain that starts at the downstream head. Additionally, it becomes head of a new congestion domain that extends from this node and upstream.

TA 2: When a node, based on measurements of traffic rates from upstream nodes, decides that the aggregate of traffic from upstream nodes, traversing the congested link, does not contribute to the downstream congestion.

The self-appointment of the tail responsibility, according to the rule specified in TA1, appears problem-free. The self-appointment of the tail responsibility according to the rule specified in TA2, however, has shown to degrade the network performance for some scenarios.

The normal operation of the aggressive fairness mode has a behavior, where, to maintain the *STQ* occupancy constant at the *high* threshold, rate adjustments are done in an monotonically increasing/decreasing fashion [14]. This behavior, in combination with the enforcement of TA2 introduces an unwanted side-effect. Consider the scenario shown in Fig. 1,

[1]The RIAS reference model does not include the use of different weights for the individual nodes as supported by RPR. However, for the general fairness discussion in this paper, we assume equal node weights. Thus the RIAS fairness definition can be used.

having all (nodes 0-3) greedy senders. In this scenario, the amount (rate) of transit traffic for all nodes downstream of node 1, will always be larger than the fair rate. Node 1 only transits traffic from one upstream node. Thus, because of the oscillatory behavior of the fair rate estimate, there may be periods where the amount of transit traffic falls below the fair rate estimate.

For the duration of these periods, node 1 assumes tail responsibility according to TA2. When this happens, node 1 sends *full*-rate messages upstream. Upon reception of these rate-messages, node 0 increases its send-rate, restricted upwards only by the link-rate. Thus node 0 will have an excessive sending of traffic, which will persist until node 1, once again discover, according to TA2, that node 0 indeed is a contributor to the downstream congestion.

If the delay between node 1 and node 0 is too large, this excessive sending behavior will happen every time we have a transition from a series of monotonically decreasing to a series of monotonically increasing fair rate estimates. The result is that the fair rate calculation process does not converge and there will be sustained large oscillations in throughput for the nodes in the congestion domain. We call this *Tail induced oscillations*. In addition, node 0 will, on average, be sending more than its downstream neighbors, hence we do not achieve RIAS fairness.

For the conservative mode fairness, a similar behavior may be observed, but in this case, the side-effect is not as serious, as rate adjustments at the head usually are performed much slower than for aggressive mode fairness.

IV. Improving the Fairness Algorithm

A. Rate Calculation Improvement

In this chapter, we propose a novel fairness mode for use in Resilient Packet Rings, which we have called the *moderate* fairness mode. The goals we seek to meet in designing the moderate fairness mode are shown below.

1) Remove the oscillatory behavior in the presence of a head with a modest sending behavior.
2) Retain the behavior of the original algorithm in the presence of a head with a greedy sending behavior.
3) Minimize the changes (state information, code and processing requirements) to the current algorithm.
4) Fit our modifications into the framework given by the RPR standard.

Let us assume we have an established congestion domain on the ring. When the head calculates a fair rate estimate to be distributed to its upstream neighbors, we know that at some future time, the resulting aggregate traffic from the upstream nodes is either too high (the *STQ* occupancy in the head will increase), too low (the *STQ* occupancy will decrease) or correct[2] (the *STQ* occupancy remains constant).

[2]A *STQ* occupancy that remains constant at 0 or at the *high*-threshold is not an indication of a correct fair rate estimate. But these are special cases that requires special handling.

Just as for the aggressive and conservative fairness modes, in the moderate mode, a node (that was previously not congested) becomes congested when the *STQ*-occupancy exceeds the *low*-threshold.

As previously discussed, the use of the head's own send-rate as a fair rate estimate, does not work for unbalanced traffic scenarios. Thus, in the moderate fairness mode, just as in the conservative mode fairness, we introduce a new rate estimate. In the moderate mode, we call this rate estimate *mRate* (moderate Rate estimate). We also maintain aged and low-pass filtered versions of this variable, denoted respectively *ageMRate* and *lpMRate*. The value of *lpMRate* is the value (i.e. the **fair** rate estimate) that is distributed to the upstream neighbors during congestion.

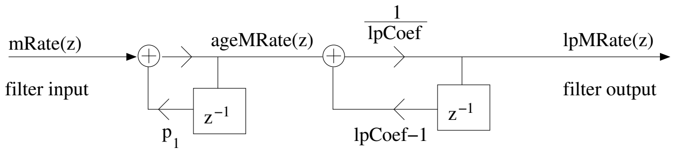


Fig. 2. Z-domain representation of the 2-stage second-order low pass filter used by the fairness algorithm.

We use the same two-stage second-order low-pass filter construct, used by both the aggressive and the conservative modes as shown in Fig. 2. We have previously shown that for the values used for the configuration settings $ageCoef$ and $lpCoef$ ($p_1 = \frac{ageCoef-1}{ageCoef}$), this filter can be approximated by a first-order low-pass filter with a time-constant, $\tau \approx lpCoef \cdot agingInterval\ [s]$ [14]. In our moderate mode, the input to this filter is the variable, *mRate*, while the output of the filter, $lpMRate$, is the fair rate estimate.

To aid the convergence process, we use a rate interval, limited by a maximum and a minimum value, denoted respectively $mRateMin$ and $mRateMax$. Upon transition from the uncongested state, we initialize $mRateMin$ to the value of the node's own send rate, *lpAddRate*. The value of $mRateMax$ is set to half of the maximum allowed rate. We also set the fair rate estimate, *lpMRate*, to $mRateMin$, while the low-pass filter input, *mRate*, is set to *mRateMax*.

Note that the size and location of the rate interval, as given by the values of $mRateMin$ and $mRateMax$, will change during the convergence process as well as in the case of a change in the applied load.

The basic idea of our fairness mode is simple. If the *STQ* occupancy of the congested node increases, the fair rate estimate is too high and must be decreased. Correspondingly, If the *STQ* occupancy of the congested node decreases, the fair rate estimate is too low and must be decreased. Further, the rate adjustments are restricted within the previously mentioned rate interval $\langle mRateMin, mRateMax\rangle$.

Below follows a description of one iteration of the cyclic convergence process, starting as we have an increasing *STQ* occupancy at the head.

When the *STQ* occupancy increases, we decrease the fair rate estimate towards the minimum value of the rate interval

($mRateMin$). Once the fair rate estimate has been sufficiently reduced, the *STQ* occupancy will start to decrease. At this point, for the given load, we know that in the future, the fair rate estimate should not be set lower than its current value. Thus, we set $mRateMin$ to the current fair rate estimate.

Next, to oppose the decreasing *STQ* occupancy, we increase the fair rate estimate towards the maximum value of the rate interval ($mRateMax$). Once the fair rate estimate has been sufficiently increased, the *STQ* occupancy will start to increase. At this point, for the given load, we know that in the future, the fair rate estimate should not be set higher than its current value. Thus, we set $mRateMax$ to the current fair rate estimate. By this, the cycle is concluded and we are back to the starting point. For each iteration of this cycle, the size of the rate interval is improved (reduced).

The actual increase/decrease behavior follows an exponential ramp function, given by the properties of the second-order low-pass filter. During the periods of increasing *STQ* occupancy, the filter-input is set to $mRateMin$, thus ensuring a monotonically decreasing (towards $mRateMin$) fair-rate estimate. Correspondingly, during the periods of decreasing *STQ* occupancy, the filter-input is set to $mRateMax$, thus ensuring a monotonically increasing (towards $mRateMax$) fair-rate estimate.

The exponential ramp function ensures that the time, used to adjust the fair-rate estimate between the max/min values, remains constant, regardless of the size of the rate-interval. Thus, during the convergence process, the narrower the rate interval gets around the RIAS fair rate, the slower the fair rate estimate is adjusted. This way, the variations in throughput during steady-state are minimized.

For the simple scenario shown in Fig. 1, with 3 greedy and one modest sender, we show the convergence process of our moderate fairness mode. In the scenario, we assume a stable demand by the individual nodes. The example illustrates, without loss of generality, the convergence for the transition between a no-load situation, and a max-load situation (i.e. a worst-case situation). For a dynamic scenario, the load-change typically is much smaller. Thus in this case, the task of the fairness mode is to shift the established rate region higher or lower, so that the new RIAS fair rate is included in the interval. This is done by expanding the rate interval on one side before continuing the convergence cycle.

Fig. 3a shows the convergence of the head's fair rate estimate, $lpMRate$, and the rate interval $\langle mRateMin, mRateMax\rangle$. The corresponding *STQ* occupancy for the head during the same period is shown in Fig. 3b.

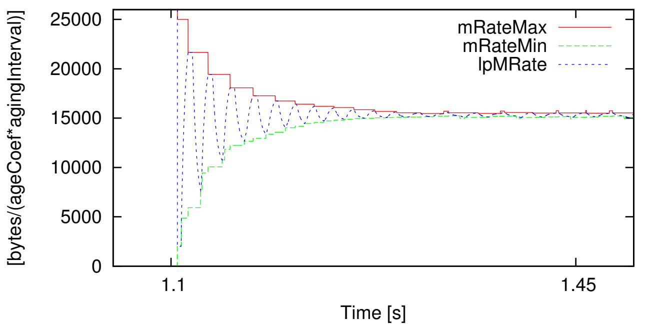
B. Rate Distribution Improvement

From section III-B, it is clear that the exclusion of nodes from a congestion domain, having a demand larger than their fair share, has a negative influence on the convergence process. Further, as long as nodes upstream of the tail are sending traffic over the (nearest) congested link, there is no reason to conceal rate restrictions in effect over the same link. Thus, we propose to alter the responsibility of the tail, so that it is no longer its responsibility to stop the propagation of fair rate estimates received from the head. However, to allow for the establishment of more than one congestion domain on the ring (which is the case for the original fairness algorithm), we remove the fairness message once it reaches a (new) congestion domain head.

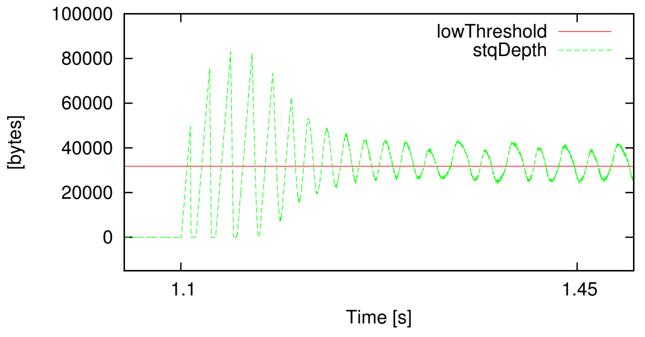
Thus, the challenge becomes to allow for a congested node to decide whether a fairness message has passed through a downstream tail or not. If the fairness message has passed through a downstream tail, the node will remove the message from the ring and become head of a new fairness domain. Otherwise, the node will assume head (and tail) responsibility as defined by the rules of the original fairness algorithm.

The RPR fairness frame format contains 13 reserved and currently unused bits. We propose using one of these as a *passedTail* bit. When the fairness message reaches the tail, the tail sets this bit to one to indicate that the fairness frame contains a fair rate that has been propagated beyond a congestion tail. Fairness messages with the *passedTail* bit set, are terminated once they reach a head node.

By this simple improvement, we avoid the problem described in section III-B where the active node, furthest away from the head, gets to send excessive amounts of traffic during the periods when its downstream neighbor stops the propagation of fairness messages from the head.



a) Fair Rate Estimate



b) STQ Occupancy

Fig. 3. Illustration of the fair rate calculation convergence process for our moderate fairness mode for the scenario shown in Fig. 1 with 3 greedy and one modest sender. lpCoef=32, link-length=410μs (82km).

V. Performance Evaluation

In this section, we have picked some particular scenarios we believe represent the general behavior of our proposed improvements. The scenarios picked, are a subset of those contained in [6], [7]. First, in section V-A, we show the improved properties from the modified behavior of congestion domain tails applied to the aggressive mode fairness algorithm. Secondly, in section V-B, we present the properties of our moderate fairness mode, which includes both the improved rate calculation method for the head as well as the modified behavior of the tail. The simulation scenarios are run using our RPR discrete event simulator models, implemented within the J-Sim and OPNET modeler frameworks [15], [16].

A. Congestion Domain Tail Scenario

In this section, we present a simulation scenario illustrating the convergence of the aggressive mode algorithm as a function of network size and setting of the lpCoef parameter, with and without the modified tail behavior. It shows that the propagation of the fair rate estimate beyond the congestion tail allows the fairness algorithm to converge with a lower value of the *lpCoef* parameter than would otherwise be needed, thus providing a shorter convergence time for the fairness algorithm.

In this scenario, we have a 64 node ring with 40 km links. The link capacity is 1 Gbit/s. Nodes 0, 10 and 20 send traffic at their maximum allowed rate to node 30, making node 20 the head and node 0 the tail. They all start sending simultaneously, at time 0.1 s. The value of *lpCoef* is set to 128.

Fig. 5 shows the throughput of traffic, received at node 30 from each source node. Fig. 5a shows the results for the aggressive fairness mode, while Fig. 5b shows the result when the fair rate estimate calculated at node 20 (the head) is propagated beyond the congestion tail with the *passedTail* bit set. We see that with the original RPR implementation, the sending rate of each active node fails to stabilize at the RIAS fair rate ($fairRate = \frac{1Gbit/s}{3} = 333Mbit/s$), while with our modified method, the system stabilizes after about 0.2s of simulated time. Fig. 5c shows that if the value of the *lpCoef* parameter is doubled (256), the original RPR fairness algorithm will also converge to the fair rate, but the convergence time is longer.

B. Improved Fair Rate Calculation and Distribution

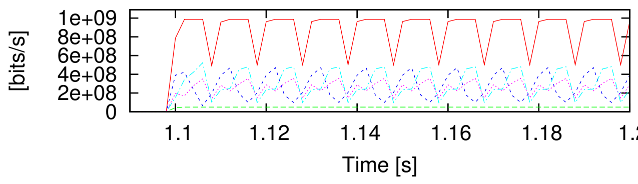
Due to space constraints, we are only able to show the most essential operation of our moderate fairness mode. In this section, we compare the behavior of the three fairness modes for the unbalanced traffic scenario, presented in section III-A. However, we have also run a series of of simulations where we test the stability of our algorithm for the various allowed values of lpCoef. The results indicate that the convergence time of the algorithm is proportional to the aggregate propagation delay of the congestion domain. We have also successfully tested the algorithm on a network with nodes where we have a mix of nodes using the aggressive and moderate modes. Finally, our results indicate that the algorithm performs equally well, regardless of the sending behavior of the head [6], [7].

1) Comparison to the Original RPR Fairness Algorithm: On a 64 node ring, using 1Gbit/s links, we use the same scenario as illustrated in Fig. 1, where nodes 0, 1, and 2 have a greedy sending behavior, while node 3 sends CBR traffic at 5% of the line rate. The parameter settings for the three fairness modes are identical for all common configuration parameters.

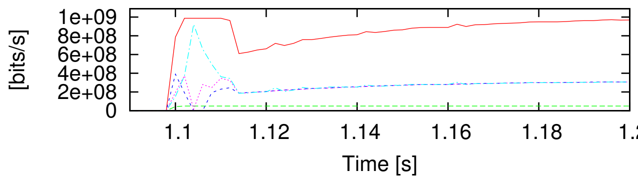
For the aggressive mode, shown in Fig. 4a, convergence to the fair rate is clearly not achieved. Furthermore, the link utilization is reduced to 89%, i.e. an average aggregate throughput loss of 110Mbit/s.

For the conservative mode, shown in Fig. 4b, it takes approximately 62ms just to reach a level of 90% of the available throughput on the congested link. What is not shown in the plot, is that once the conservative mode reaches steady-state, the *STQ* occupancy will fall to 0 at periodic intervals (just as for the aggressive mode). This leads to a reduction in link-utilization for the congested link. For this scenario, the reduction in link-utilization is approximately 2% when compared to our moderate mode.

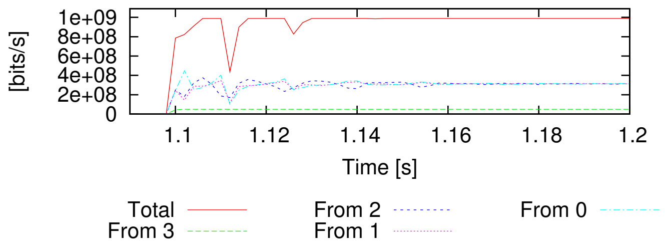
Finally, for the moderate mode, shown in Fig. 4c, the convergence time (the time required to reach a throughput within ±5% of the steady-state value) is 56ms. For this mode, once in steady-state, the *STQ* never becomes empty, thus we do not incur any loss in link-utilization.



a) Aggressive Mode Throughput

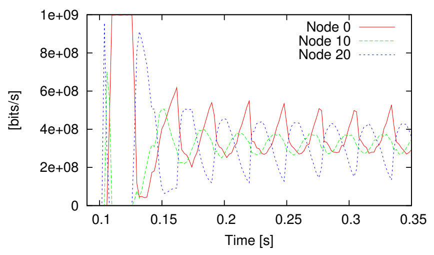


b) Conservative Mode Throughput.

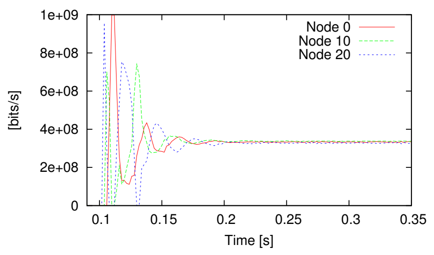


c) Moderate Mode Throughput.

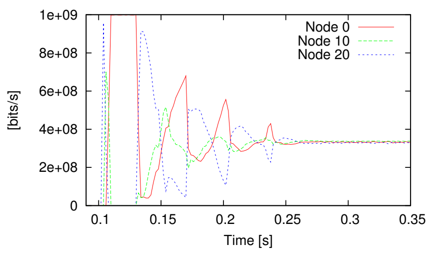
Fig. 4. Throughput convergence in the presence of a head sending a rate lower than the fair rate (fairRate=1G/4=250Mbit/s. Head sends CBR traffic at 50Mbit/s.



Aggressive mode fairness, lpCoef=128



Tail modification, lpCoef=128



Aggressive mode fairness, lpCoef=256

Fig. 5. Throughput of traffic received at node 30. Using a *lpCoef* of 128, the traffic from the different sources does not stabilize without propagation of fair rate beyond the congestion tail. Increasing the *lpCoef* to 256 gives a stable system with the original RPR algorithm, but with a longer convergence time.

VI. Conclusion

In this paper we have analyzed and proposed improvements for two major deficiencies in the RPR fairness algorithm. The first deficiency relates to the calculation of the fair rate estimate by the head of a congestion domain. For some so-called unbalanced traffic scenarios, the operation of the aggressive mode algorithm results in non-convergence of the fairness algorithm, leading to sustained, large oscillations and reduced link utilization. With our moderate mode algorithm, this behavior is avoided and we do not incur reduced link-utilization. Additionally, the algorithm convergences faster and provides better link-utilization than the conservative mode. Although not shown here (but in the in-depth analysis in [6], [7]), our algorithm meets the 4 design goals specified in section IV-A.

The second deficiency relates to the propagation of the fair rate estimates calculated by the head, to upstream nodes in a congestion domain. For the method used by the standard, the stability of the fairness algorithms is seriously affected. With our simple proposed modification, we can, for a given configuration of the lpCoef parameter, provide stable operation of the fairness algorithm for networks twice (aggregate propagation delay) the size of that possible with the original algorithm.

VII. Further Work

The configuration of the RPR fairness algorithm is complicated and requires a thorough knowledge by the network operator to be configured in a safe and cost-efficient manner. The use of dynamic measurement and configuration methods (not discussed in this paper) in the conservative fairness mode eases the task of the operator somewhat. The introduction of additional functionality in the RPR fairness algorithm to perform self configuration during run-time, is one area of improvement which would benefit both the owners and the user of an RPR network.

References

[1] IEEE Computer Society, "IEEE Std 802.17-2004 Resilient packet ring (RPR) access method and physical layer specifications," September 24 2004.

[2] E. Hafner, Z. Nendal, and M. Tschanz, "A digital loop communication system," *IEEE Trans. Commun.*, vol. 22, no. 6, pp. 877–881, June 1974.

[3] C. C. Reames and M. T. Liu, "A loop network for simultaneous transmission of variable-length messages," in *Proceedings of the 2nd Annual Symposium on Computer Architecture*, vol. 3, December 1974.

[4] H. van As, W. Lemppenau, H. Schindler, and P. Zafiropulo, "CRMA-II: A MAC protocol for ring-based Gb/s LANs and MANs," *Computer Networks and ISDN Systems*, vol. 26, no. 6-8, pp. 831–840, March 1994.

[5] I. Cidon and Y. Ofek, "MetaRing - a full duplex ring with fairness and spatial reuse," *IEEE Trans. Commun.*, vol. 41, no. 1, pp. 110–120, January 1993.

[6] F. Davik and S. Gjessing, "Improvement of Resilient Packet Ring Fairness," Simula Research Laboratory, Tech. Rep. 2005-02, February 2005. [Online]. Available: http://www.simula.no

[7] F. Davik, A. Kvalbein, and S. Gjessing, "Congestion domain boundaries in Resilient Packet Rings," Simula Research Laboratory, Tech. Rep. 2005-03, February 2005. [Online]. Available: http://www.simula.no

[8] F. Davik, M. Yilmaz, S. Gjessing, and N. Uzun, "IEEE 802.17 Resilient Packet Ring tutorial," *IEEE Commun. Mag.*, vol. 42, no. 3, pp. 112–118, March 2004.

[9] V. Gambiroza, P. Yuan, L. Balzano, Y. Liu, S. Sheafor, and E. Knightly, "Design, analysis, and implementation of DVSR: a fair high-performance protocol for packet rings," *IEEE/ACM Trans. Networking*, vol. 12, no. 1, pp. 85–102, 2004.

[10] X. Zhou, G. Shi, H. Fang, and L. Zeng, "Fairness algorithm analysis in Resilient Packet Ring," in *Proceedings of the 2003 International Conference on Communication Technology (ICCT 2003)*, vol. 1, April 9-11 2003, pp. 622–624.

[11] J. Schuringa, G. Remšak, and H. R. van As, "Cyclic Queuing Multiple Access (CQMA) for RPR networks," in *Proceedings of the 7th European Conference on Networks & Optical Communications (NOC2002)*, Darmstadt, Germany, June 2002, pp. 285–292.

[12] H. Kong, N. Ge, F. Ruan, and C. Feng, "Congestion control algorithm for Resilient Packet Ring," *Tsinghua Science and Technology*, vol. 8, no. 2, April 2003.

[13] E. Knightly, L. Balzano, V. Gambiroza, Y. Liu, P. Yuan, S. Sheafor, and H. Zhang, "Achieving high performance with Darwin's fairness algorithm," July 2002, presentation at IEEE 802.17 Meeting. [Online]. Available: http://grouper.ieee.org/groups/802/17/documents/presentations/mar2002

[14] F. Davik, A. Kvalbein, and S. Gjessing, "An analytical bound for convergence of the Resilient Packet Ring Aggressive mode fairness algorithm," in *Proceedings of the 40th annual IEEE International Conference on Communications*, Seoul, Korea, May 16-20 2005.

[15] H. Tyan, "Design, realization and evaluation of a component-based compositional software architecture for network simulation," Ph.D. dissertation, Ohio State University, 2002.

[16] OPNET Technologies, Inc, "OPNET Modeler." [Online]. Available: http://www.opnet.com/

[17] F. Alharbi and N. Ansari, "Low complexity distributed bandwidth allocation for Resilient Packet Ring networks," in *Proceedings of 2004 Workshop on High Performance Switching and Routing, 2004. HPSR*, April 18-21 2004, pp. 277–281.