Congestion Domain Boundaries in Resilient Packet Rings

February 2005
Revised August 2005

Fredrik Davik   Amund Kvalbein   Stein Gjessing

[simula . research laboratory]

Research Report   Simula 2005-03

Congestion Domain Boundaries in Resilient Packet Rings

February 2005
Revised August 2005

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— **In June 2004, the IEEE approved a new standard for Resilient Packet Rings (RPR). The standard is maintained in the 802 LAN/MAN Standards Committee, and is designated the standard number 802.17. In this paper, we analyze and discuss performance aspects of the Resilient Packet Ring fairness mechanism. We explain that, if the ring is not configured correctly, the fairness mechanism fails to stabilize at a fair division of bandwidth between the active nodes. We present a novel addition to the fairness algorithm, and show that with this modification, RPR reaches a stable state with more optimal parameter settings. We also show that our proposed modification gives shorter convergence time for the Resilient Packet Ring fairness algorithm.**

Keywords: Resilient Packet Ring, Fairness, Simulations

I. Introduction

Resilient Packet Ring (RPR) is a recent IEEE standard (802.17) that uses a dual ring topology to allow for efficient and resilient communication between the nodes connected to the ring [1]. Ring topologies have long been used to connect computers. The first widely known ring network was the Cambridge Ring [2], and we have later seen many different technologies in use, such as IEEE 802.5 Token Ring [3], FDDI [4], SCI [5], MetaRing [6], ATMR [7] and CRMA-II [8], [9]. Depending on the application area of the ring, the choice of medium access control (*MAC*) protocol to use for the ring is crucial. On a ring, where the demand for bandwidth is greater than the supply, and the data-flows from the contending nodes are of equal importance, the challenge becomes to equally or fairly divide the available bandwidth between the contending nodes. The method or algorithm used to solve this problem, is often referred to as a *fairness* algorithm.

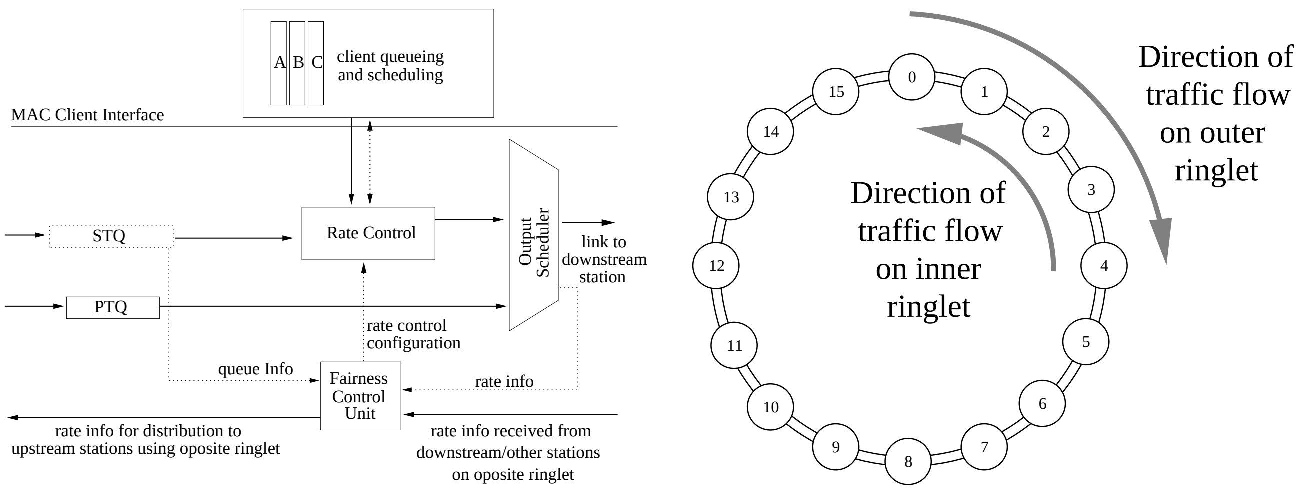
Some ring technologies, like Token Ring and FDDI, use a *token based* access control mechanism. In this class of networks, only the current owner of the token is allowed to transmit data on the ring. Fairness is enforced by limiting the time each node can own the token. Other ring technologies, including SCI, MetaRing, CRMA-II and RPR, use a mechanism known as an *insertion buffer* to control media access [10], [11]. In such insertion buffer rings, the buffer is used to store transit traffic that is held back when a node adds local traffic to the ring. *Slotted* ring technologies, like the Cambridge Ring and ATMR, avoid collisions by only transmitting data in given slots, that circulate the ring. In slotted and buffer insertion rings, various fairness mechanisms are used (see [12] for an overview).

RPR's fairness algorithm has two modes of operation, respectively the *conservative* mode, discussed in [13], and the *aggressive* mode, discussed in [13], [14]. In both modes, a node, with a congested out-link, maintains a local fair rate estimate which it distributes upstream, by use of so called *fairness* messages. Regardless of the mode used, the goal is to arrive at a fair division of sending rates, as defined by the "Ring Ingress Aggregated with Spatial reuse" (RIAS) reference model [15].

In the *conservative* mode of operation, the congested node transmits a fairness message, and then waits to see the change in traffic from upstream nodes. If the observed effect is not the fair division of rates, then the congested node calculates a new fair rate estimate, and distributes it upstream. The time to wait from a fairness message with a new fair rate estimate is issued, until the effect is evaluated, is based on periodic measurements, and is denoted the *Fairness Round Trip Time* (*FRTT*).

For the *aggressive* mode of operation, the congested node also calculates and advertises a fair rate estimate, but does so periodically, without waiting to evaluate the result of the previously transmitted fairness messages. In the aggressive mode, the calculation of the fair rate estimate is based solely on the node's own send rate and preset parameters. The frequent calculation and advertisement of new fair rate estimates gives rise to a more "aggressive" and opportunistic algorithm, that more quickly attempts to adapt to changing load conditions. Thus, the faster response as compared to the conservative version, comes at a cost. Namely the risk of instabilities when rate adjustments are made faster than the system is able to respond.

In this paper we discuss and analyze a performance deficiency of the RPR fairness algorithm, which relates to the handling and distribution of fairness messages, by nodes upstream of the congestion point. We propose a novel method


A
B
C
client queueing and scheduling
MAC Client Interface
Rate Control
Output Scheduler
STQ
PTQ
link to downstream station
rate control configuration
queue Info
Fairness Control Unit
rate info
rate info for distribution to upstream stations using oposite ringlet
rate info received from downstream/other stations on oposite ringlet
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Direction of traffic flow on outer ringlet
Direction of traffic flow on inner ringlet

a) Generic node design for control of data flowing in one direction on an RPR ring

b) Example RPR topology

Figure 1: The generic RPR node design in a), shows a node's attachment to the ring for transferral of data in the downstream direction and transferral of fairness messages (fair rate estimates) in the upstream direction. The solid lines indicate the flow of data through the node. The dotted lines indicate the exchange of control/configuration information between node internal function blocks. To control the sending of data on both ringlets, every node has two instances of the functional blocks and interconnections below the MAC client interface. I.e. in the example topology in b) every node has one instance controlling the data flowing on the outer ringlet and another instance controlling data flowing on the inner ringlet.

that allows a more optimal setting of one of the most important parameters, the so called *lpCoef* parameter. We use performance evaluation by simulations to show that for both modes of the fairness algorithm, our method reduces the risk of network instability as well as speeding up the convergence process. We also show that minimal modifications are required to incorporate the proposed modification into the next version of the standard.

The rest of this paper is organized as follows. In section II, we outline and discuss parts of the RPR fairness mechanism, and identify the deficiencies we address in this paper. In section III, we discuss the propagation of fairness messages in an RPR network, and how this affects the network stability. Then, in section IV, we present our proposed improvement. In section V, we describe simulation scenarios and results used to illustrate and evaluate the performance of our improved algorithm. Finally, in sections VI, VII and VIII we address related work, conclude and point out some directions for future work.

II. Resilient Packet Ring Fairness Characteristics

Traffic in an RPR network is sent in one of three service (priority) classes, named A, B and C. The high priority, class A, traffic and a configured amount of the medium priority, class B, traffic is sent using reserved bandwidth. The rest of the class B and the low-priority, class C, traffic must compete for the remaining unreserved bandwidth. This traffic is known as *fairness eligible*, since the send rate for this traffic is governed by the fairness algorithm.

An RPR node may contain one or two insertion buffers, also known as transit buffers. In this paper, we consider only the dual transit buffer design (*2TB*). In the *2TB* design, transit traffic awaiting transmission of local traffic, is stored in one of the two transit buffers, depending on the service class of the packet. Packets belonging to the high priority class, are stored in the primary transit queue (PTQ), while other traffic is stored in the secondary transit queue (STQ). An overview of a generic RPR node design is given in Fig. 1.

A. Fairness algorithm parameters

In the RPR fairness algorithm, a node with a congested outlink calculates an estimate of the fair division of send rates (for fairness eligible traffic) traversing the congested link. The calculation is done at periodic intervals (every *aging* interval). According to the standard, the aging interval is 100μs (for line rates $\geq$ 622Mbit/s). The fair rate estimate is sent upstream in a fairness messages. Nodes upstream of the head, having received this messages, will limit their sending rate over the congested link to the fair rate estimate contained in the fairness message.

1) Aggressive Mode Fair Rate Estimation: For the aggressive mode of the fairness algorithm, the fair rate estimate is the value of a rate counter called *lpAddRate* [14].

lpAddRate is calculated by smoothing the local add rate, (*addRate*), using a two-stage second-order low-pass filter. After aging interval *n*, *lpAddRate* is calculated based on its previous value (that was calculated after aging interval *n-1*), and the number of bytes, *addRate(n)*, added by the node during the current aging interval[1]. This is shown in (1) below:

[1]*addRate* is really already slightly smoothed, but this has no effect on the current discussion.

$$lpAddRate(n) =$$

$$\frac{1}{lpCoef}\left(lpAddRate(n-1)\cdot(lpCoef-1)+addRate(n)\right)$$

$$\text{Where: } lpCoef \in \{16, 32, 64, 128, 256, 512\} \tag{1}$$

Observe that the relative weight of the previous value of *lpAddRate*, compared to the relative weight of the current value of *addRate*, is decided by the setting of the *lpCoef* parameter. Since *lpAddRate* becomes the advertised fair rate estimate, the *lpCoef* parameter decides how fast the congested node tracks and signals its local rate reductions to upstream nodes, when congestion occurs or increases. Correspondingly, *lpCoef* determines how fast the congested node tracks and signals local rate increases to upstream nodes when congestion ceases or decreases.

Intuitively, faster tracking of the local add rate, distributed to upstream nodes using fairness messages, should more rapidly resolve the congestion, resulting in faster convergence to the fair rate. Thus, one may believe that the *lpCoef* parameter should be set as low as possible. However, as discussed in [14], setting the *lpCoef* parameter too low, meaning that rate adjustments are performed too fast, will result in a system that oscillates, and fails to converge to the fair division of bandwidth over the congested link. The optimal setting of the *lpCoef* parameter is discussed in section V-B below.

2) Conservative Mode Fair Rate Estimation: For the conservative mode of the fairness algorithm, the fair rate estimate is the value of a variable termed *allowedRate*. The value of this variable is adjusted every Fairness Round Trip Time (FRTT), based on the occupancy of the *STQ*. The value of FRTT is measured periodically by use of special control messages, and is an estimate of the time it takes from a rate regulation is done by the congested node, until the corresponding effect is observable by the same node. By having an estimate of the time it takes from an adjustment is made, until the effect is observable, we effectively have an estimate of the system time-constant. This is then used, to ensure that rate-regulations are not made too fast, regardless of the setting of the lpCoef parameter.

B. Fairness convergence time

We define the *convergence time*, T_c as the time from congestion occurs, until the fairness algorithm has converged to the RIAS fair sharing of bandwidth for the congested link. A feedback control system is defined to be stable at time t_0, if all values for the controlled variable, sampled in the interval $<t_0,\ t_0+t_s>$, are within $\pm p\%$ of the mean value in the same interval [16]:

$$\begin{aligned} max(X(t)) &< \overline{X(t)} * (1 + p/100), \quad t_0 < t < t_0 + t_s \\ min(X(t)) &> \overline{X(t)} * (1 - p/100), \quad t_0 < t < t_0 + t_s \end{aligned}$$

Later in this paper, we will show that the convergence time, T_c, is dependent on the number of and distance between the nodes contributing to the congestion, and the settings of the parameters used in the fairness mechanism.

C. Congestion Domains

Both modes (*aggressive* and *conservative*) of the RPR fairness algorithm works 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 congestion situation for a given link.

The congestion domain is confined within a region specified by two boundary nodes. 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 of a congestion domain can be considered a two-part problem. The first part of the problem consist of making a node declare itself as the head, responsible for calculating the (RIAS) fair division of unreserved bandwidth among contending nodes sending fairness-eligible traffic over the bottleneck link. The calculated result, the fair rate estimate, is distributed to upstream nodes in the congestion domain using fairness messages.

The second part of the problem consists of making a node declare itself as tail, responsible for stopping the propagation of fairness messages, received from the head. In the RPR standard, the appointment of a tail node can be done 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, this node becomes the tail of the congestion domain that starts at the downstream head. Additionally, it becomes the 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 fairness eligible traffic from upstream nodes, traversing the congested link, does not contribute to the downstream congestion.

The self-appointment of a tail node according to the rule specified in TA1 appears problem-free. The self-appointment of a tail node according to the rule specified in TA2, however, has shown to degrade the network performance for some scenarios (see section V). In the next section, we will discuss how the self-appointment of a tail node according to TA2 affects the stability of the RPR fairness algorithm.

III. Rate Distribution in Fairness Domains

The RPR standard states that when the aggregate of fairness eligible traffic received from upstream nodes does not exceed the fair rate, the propagation of (fair) rate information further upstream is not needed. The rationale for this reasoning, is that since the aggregate of traffic from upstream nodes is less than the fair rate, the distribution of the fair rate information further upstream would have no effect. This can be considered a reasonable assumption as none of the upstream nodes send at a rate exceeding the fair rate.

In this section, we will argue that this assumption is incorrect and furthermore, when adhered to, possible side-effects are unfairness and network instability. Below, we describe the origin of the problem and discuss how it degrades the performance of the RPR fairness algorithm.

The problem may occur when the equilibrium point of the aggregate of traffic received from upstream nodes is close to that of the fair rate estimate. We will illustrate this by use of the simple topology shown in Fig. 2. Assume that nodes B, C and D all send over the same congested link, and are part of the congestion domain spanning from D (head) to B (tail). Next, node A starts sending over the same congested link.

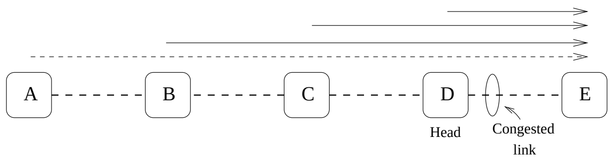


Figure 2: Nodes B, C and D are part of a congestion domain. When node A starts sending across the congested link, the congestion domain is extended to include A.

Initially, since it has no notion in advance of the downstream congestion, node A starts sending at the full capacity of its output link. For a short while, this will cause the sending rates of nodes B, C and D to drop significantly.

Once B's rate measurements of traffic received from A has been affected enough, it does not fulfill any of the tail-appointment conditions. Thus B starts to forward the fairness messages received from the head (D) upstream. By this, B is effectively transferring the role as tail of the congestion domain to A.

Once A receives fairness messages from the head, it will reduce its sending rate to that of the encapsulated fair rate estimate in the fairness message. If at some future time, the *STQ* occupancy of the head falls below a threshold in response to the aggregate rate reductions by its upstream neighbors, the head will gradually start to increase its fair rate estimates. If the delay between the time where B starts receiving increasing fair rate estimates and the time where the furthest upstream node, A, starts to increase is send rate is too large. B will, once again assume tail responsibility according to TA2 and stop the propagation of fairness messages received from the head.

In summary, for the duration of the periods where the propagation of fair rate information to upstream node(s), having a demand that equals or exceeds their fair share, is stopped. This may result in excessive sending and resulting unfairness. Furthermore, this may result in non-convergence of the fairness algorithm. In sections III-A and III-B, we will discuss the effects of this behavior for the aggressive and conservative fairness modes. Then, in section IV, we propose a modification to the fairness algorithm, to resolve these problems.

A. Congestion Domains and Aggressive Mode Fairness

For the *aggressive* mode fairness algorithm, the decrease in the head's amount of added traffic to the ring results in a corresponding decrease in the head's *lpAddRate* counter, which is distributed to the upstream neighbors in the fairness messages. If the duration of the bursts are too long compared to the *lpCoef* setting, the head's *lpAddRate* counter will decrease too much, too fast, and at the end of a cycle, the starting point of the next cycle will be no closer to the theoretical fair division of add rates than the starting point of the previous. Hence there will be no (further) convergence to the fair rate, and the system remains unstable.

Because of the effect described above, there is a connection between the congestion domain size and the minimum setting of the *lpCoef* parameter. In the worst case, a congestion domain spans from node N-1 to node 0 in an N-node ring. To guarantee convergence of the fairness algorithm, the *lpCoef* parameter must be set so that *i)* the head does not make rate adjustments faster than it is able to evaluate the results (at least partially) of the corresponding upstream rate adjustments [14] and *ii)* the effect of tail bursts transiting the head does not affect the value of its *lpAddRate* counter too strongly.

The behavior of the rate-control algorithm executing in the head consists of an infinite series of adjustment cycles. Each cycle consists of two periods. We will illustrate this behavior using the simple scenario shown in Fig. 2 and using two different values of the per link propagation delay.

In the scenario above, let us consider the point shortly after node A has assumed the role as the tail of the congestion domain. In response to the increase in transit traffic, resulting in an increasing *STQ* occupancy, the head will stop the sending of local traffic (as long as the *STQ* occupancy exceeds the high threshold). As a result, the head's fair rate estimate, *lpAddRate*, will decrease monotonically. Further, the upstream nodes will all, upon reception of the fairness messages, reduce their addRate.

As a result, at some future time, the occupancy of the head's *STQ* will change from being higher than the *high-* to being lower than the same threshold. When this happens, the head will start to add fairness eligible traffic over the congested link again. Hence, the value of its *lpAddRate* rate counter starts to increase. Finally, once the *STQ* occupancy, because of the resulting increase in transit traffic, exceeds the high threshold, we are back to the starting point.

Below, in figures 3a, 3b and 3c, this behavior is illustrated, using aggressive mode fairness and plotting rate measurement statistics from node *B* for the scenario. We use a value of 16 for lpCoef and link-lengths of 50 and 250μs. All stations start to send traffic at time 1.1s. The figures plots data for three different data sets. The first is the fair rate estimates, labelled "receivedRate", received by node *B* from the head. The second, labelled “normLpFwRateCongested”, is the aggregate of transit traffic, received by node *B* from node *A*. The third, labelled “rate value sent upstream”, is the rate value of the fairness messages, sent upstream from node B.

As seen in figures 3a and 3b, between times t_1 and t_2, the rate information received by node *B* consist of a sequence of monotonically increasing values, where the value received at t_1 is the minimum value in the sequence and the value received at t_2 is the maximum value in the sequence. The monotonically increasing sequence is ended once the head discovers that its STQ occupancy has exceeded the *high* threshold. In Fig. 3a, the rate value calculated in response to this, is the one received by *B*, at time t_2.

Following this, we have a sequence of monotonically decreasing rate messages received in the interval $\langle t_2, t_3]$. The monotonically decreasing sequence is ended once the head discovers that its STQ occupancy has fallen below the *high* threshold. In Fig. 3a, the rate value calculated in response to this, is the one received by *B*, at time t_3.

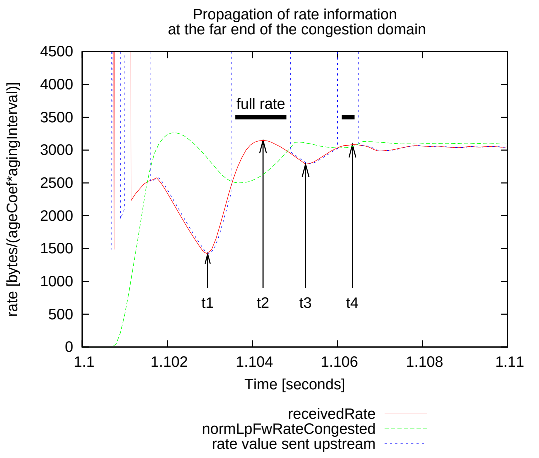
At point t_3 we have reached the end-point of the first cycle and the start point of the next cycle immediately follows. In Fig. 3a, plotting the rate statistics for a link-length of 50μs, we observe that the difference between the max and min value in each cycle decreases towards 0 for each consecutive cycle. I.e. the fairness algorithm converges. In figures 3b and 3c, plotting the rate statistics for a link-length of 250μs, this does not happen. I.e. the fairness algorithm does not converge. During the first few cycles ($t < t_8$) (see Fig. 3c), the difference between the max and min values in each cycle does decrease. After this point however, the difference between the max and min values converges towards a mean value of 1621 $\left[\frac{bytes}{ageCoef \cdot agingIntervals}\right]$ with a standard deviation of 1% of the mean. The statistics properties when observing the magnitude of the oscillations in the dataset, are shown in table I below. The analysis is performed from time t_8 (marked in Fig. 3c). Note that the figure shows only a subset of the dataset analyzed, however the oscillatory behavior for the remainder of the period is the same.

[s]		[#]		$\left[\frac{bytes}{ageCoef \cdot agingInterval}\right]$			
t_{start}	**t_{stop}**	**n**	**max**	**min**	**mean**	**median**	**stdev**
1.11965	1.39855	135	1659	1582	1621	1619	16.90

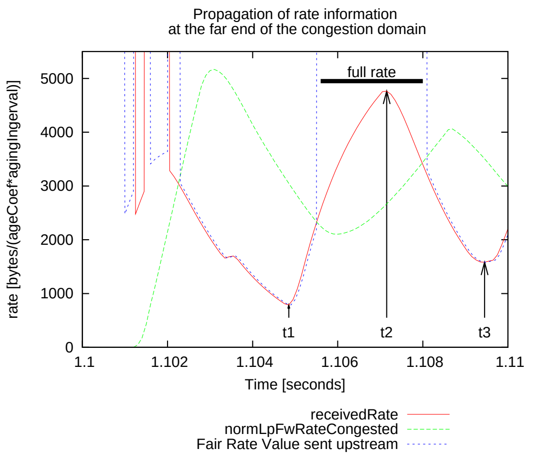
Table I: Magnitude of oscillations for the unstable configuration.

If we investigate the plots closer, starting with Fig. 3a, we can observe the following: At the point between t_1 and t_2, where *receivedRate* increases beyond *normLpFwRateCongested*, node *B* decides that the aggregate of traffic received from upstream nodes no longer contribute to the downstream congestion. Thus, in accordance with TA2, node *B* assumes the role as tail and issues a *full*[2]-rate rate message upstream (the value of rate messages sent upstream is the plot line labelled

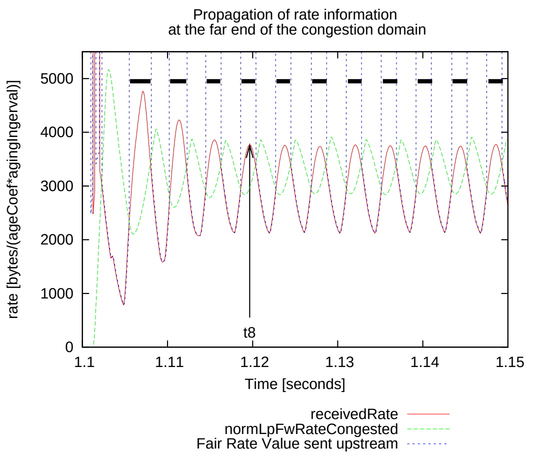
[2]The semantics of the *full*-rate message is that nodes receiving this message no longer have to restrict their local add rates, regardless of the destination of their traffic.



(a) Link-length of 50μs and lpCoef=16, fair rate calculation process converges.



(b) Link-length of 250μs and lpCoef=16. First cycle of fair rate calculation process illustrated for non-converging configuration.



(c) 250 μs and lpCoef=16. Multiple cycles of fair rate calculation process illustrated for non-converging configuration.

Figure 3: Traffic measurements performed at node B. The horizontal lines shows the period where node B issues full-rate fairness messages upstream.

Fair Rate Value sent upstream). The duration of the periods where node B sends *full*-rate messages upstream is indicated by the horizontal lines in figures 3a, 3b and 3c.

The time where node *B* transmits the *full*-rate message upstream, is represented by the vertical line which starts at $t = 1.1035$. One link-propagation time later, when this *full*-rate message is received by node *A*, the effect is that *A* will gradually (every aging interval) increase its amount of added traffic towards the maximum rate. The effect of the increased amount of traffic added from *A* will at the earliest be detected by node *B* two link-propagation delay times after the *full*-rate message was transmitted upstream from *B*.

As a result, *B*'s measurements of traffic from *A*, *normLpFwRateCongested*, will increase beyond the fair rate estimate received from the head, *receivedRate*. At the time between t_2 and t_3, when the value of *normLpFwRateCongested* increases beyond *receivedRate*, node *B* decides, in accordance with TAs 1 and 2, that *A* **does** contribute to the downstream congestion. Thus, it transfers the tail responsibility back to *A*, by passing on the received fairness messages originating from the head. The first of the fairness messages forwarded by *B* at this time, will be received by *A* one link propagation time later. Thus, at the time when *B* detects that its upstream neighbor(s) do indeed contribute to the congestion downstream, they will be allowed to transmit at an excessive rate for an additional period of at least as long as it takes the fairness message to propagate to the upstream active node(s). Thus, the longer the links, the longer the duration of the excessive transmission.

If we investigate Fig. 3c, showing the rate measurements for the unstable configuration, we see that the duration of the periods, where node *A* is allowed to gradually increase its add rate, does decrease slightly initially (while $t < t_8$). Later, the duration of these periods stays relatively constant. As shown in table II, during the observation period, the rise-time (of the fair rate estimate received from the head) from a local minimum- to a local maximum value is on average 1.86 ms. The fall-time from a local maximum- to a local minimum value is on average 2.27 ms.

	[s]		[#]	[ms]			[μs]
flank	$\mathbf{t_{start}}$	$\mathbf{t_{stop}}$	**n**	**max**	**min**	**mean**	**stdev**
Rise	1.11965	1.39855	67	1.90	1.80	1.86	49
Fall	1.11965	1.39855	68	2.30	2.20	2.27	47

Table II: Time between local extremum points at rate plots for unstable configuration.

B. Congestion Domains and Conservative Mode Fairness

The problems described for the aggressive mode fairness algorithm also apply to the conservative mode fairness algorithm. For the conservative mode algorithm however, the rate adjustments are performed every Fairness Round Trip Time as discussed in section II-A.2. Thus as node A in Fig. 2 is alternately included/excluded into/from the congestion domain, the value of FRTT is correspondingly adjusted. However, the adjustment of FRTT takes time, thus when the difference between the system-time constant between the two congestion domain alternatives become too large, this affects the convergence of the fair rate estimation process.

IV. Propagation of Fairness Messages Beyond Tail

At least two different approaches can be imagined to avoid the instabilities discussed above. One possibility would be to create a mechanism that prevents node B in the above example from taking on the role as congestion tail before the traffic from upstream nodes has been below the fair rate for a given amount of time. This would demand a timer to be set once the conditions for being a tail are met. Only when this timer expires, would the new tail stop propagating fairness messages received from the head to upstream nodes. Such a timer would, however, probably contribute to a longer convergence time for the fairness mechanism in many cases.

Instead, we propose a mechanism that alters the responsibility of the tail of a congestion domain. With our modification, the propagation of fairness messages is not stopped by the tail. The rationale behind this is that nodes that do not send traffic over the congested link, are not affected by the received rate limitations. However, nodes that *do* send across the congested link will always receive the fairness messages, containing information on the closest downstream congestion, even if they temporarily send below the allowed rate. By this, nodes upstream of the tail will limit their sending-rate over the congested link, to that of the received fair rate estimates. This way, the oscillations otherwise experienced are avoided.

One way of implementing this, could be to propagate the fair rate estimates unconditionally beyond the congestion tail. This would cause the fair rate estimate of the most congested link to be propagated all the way around the ring, effectively allowing only one congestion domain. This solution however, appears to be in conflict with the RPR standard, as the RPR standard allows the existence of several independent congestion domains on an RPR ring.

What we propose, is to use one bit in the fairness frames to mark the frame before being forwarded upstream by a congestion domain tail. The RPR fairness frame format contains 13 reserved and currently unused bits. We propose using one of these as a *passedTail* bit, indicating that the fairness frame contains a fair rate estimate that has been propagated beyond a congestion tail. With our modification, the role of a congestion tail is no longer to stop the propagation of the fairness message received from the head, but instead to set the *passedTail* bit of forwarded fairness messages to one. With our modification, the fair rate estimate calculated at one congestion head is not terminated before it reaches the head of the next congestion domain. If there is only one congested link on the ring, the fair rate estimate calculated by the node immediately upstream of the congested link will be propagated all the way around the ring. If there are several congested links on the ring, the fairness message will propagate upstream from the head of one congestion domain, until it reaches the head of the next congestion domain. Thus effectively allowing the existence of several congestion domains on a ring. For the remainder of this article, we will refer to this mechanism as the *tail-fix*.

Fig. 4 shows the difference between the original and our modified mechanism. In the figure, we have two congestion domains A and B, spanning from node 3 to 1 and node N to N-3 respectively. With the original RPR fairness algorithm, the tails of the respective congestion domains (nodes 1 and N-3), will terminate the propagation of the fairness message received from the downstream head. Thus, they will advertise *full*-rate to the upstream nodes. With our proposed mechanism, the tail nodes instead propagate the fairness messages received from their respective heads, after setting the *passedTail* bit to one. The fair rate estimate from node N is not terminated before it reaches the head of the next congestion domain (node 3).

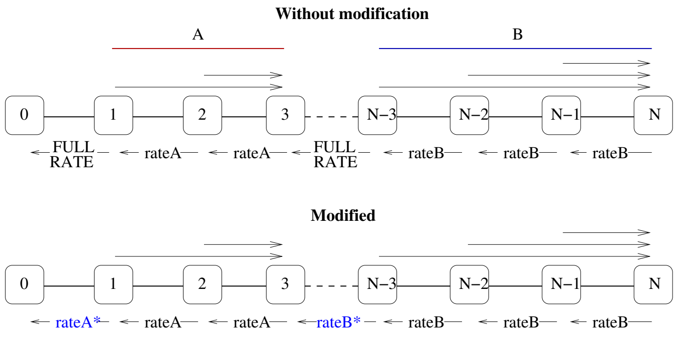


Figure 4: With our proposed solution, fairness messages from one congestion domain are not terminated before they reach the head of the next domain.

V. Simulation Scenarios and Results

In this section, we describe simulations made to evaluate our modified method. The experiment described in section V-B was run on our simulator written in the Java programming language, within the J-Sim [17] simulation framework. For the remaining experiments, we have used our simulator model implemented within the OPNET [18] simulation framework.

A. Fair Rate Propagation Beyond Congestion Tail

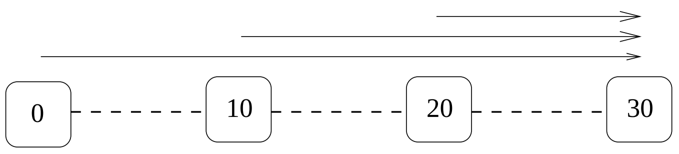
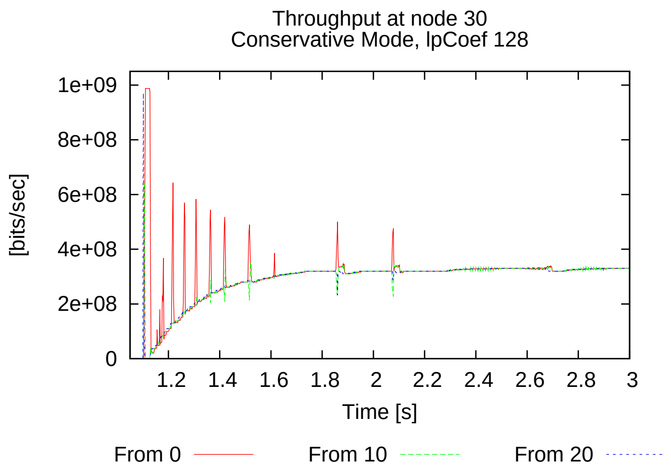


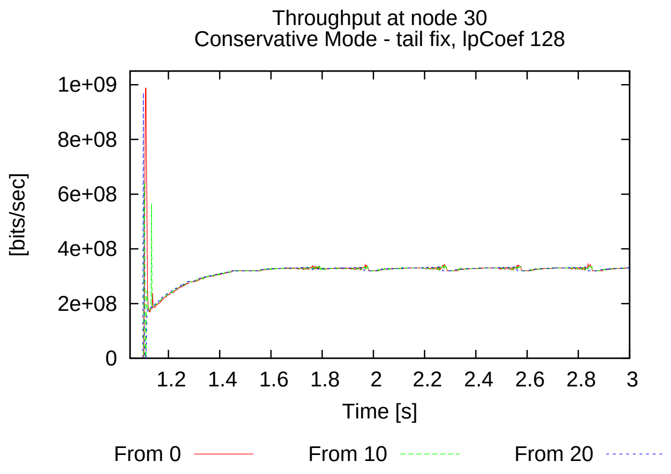
Figure 5: In this scenario, nodes 0, 10 and 20 send at their maximum allowed rate to node 30.

As discussed above, the Resilient Packet Ring fairness algorithm does not converge if the ring size is too large compared to the parameter settings, notably the *lpCoef* parameter. In this section a simulation scenario illustrating this behavior is presented. It shows that 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.

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 class C



(a) With the original conservative mode fairness and lpCoef=128 for the scenario shown in Fig. 5, the fair rate estimation process does converge. However, there are some brief periods of unfairness, where the furthest upstream node (node 0) gets to send excessive amounts of traffic.



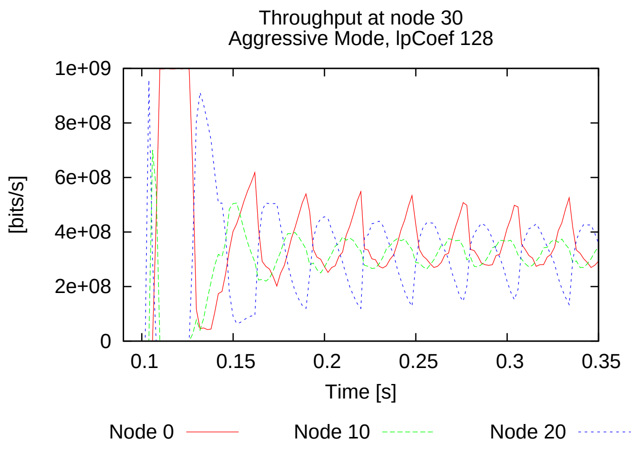
(b) With the conservative mode fairness with the tail-fix implemented and lpCoef=128 for the scenario shown in Fig. 5, the fair rate estimation process converges faster and without any unfairness.

Figure 6: Fairness convergence for the conservative fairness mode with and without the tail-fix.

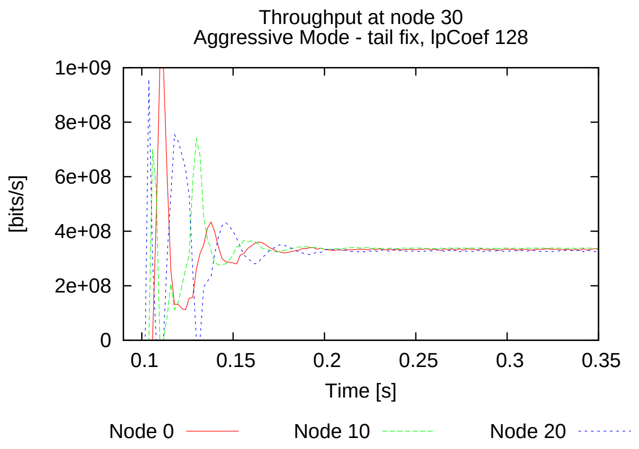
traffic at their maximum allowed rate to node 30, making node 20 the congestion head and node 0 the congestion tail. They all start sending simultaneously, at time 0.1 s. The scenario is illustrated in Fig. 5.

Figures 6a and 6b shows the convergence towards the fair sharing of bandwidth, by measuring the throughput of traffic received by node 30. We use a value for *lpCoef* of 128 and conservative mode fairness. As shown, the algorithm converges with or without the tail-fix. With the tail-fix however, the convergence time is reduced and excessive sending by the most upstream node is prevented.

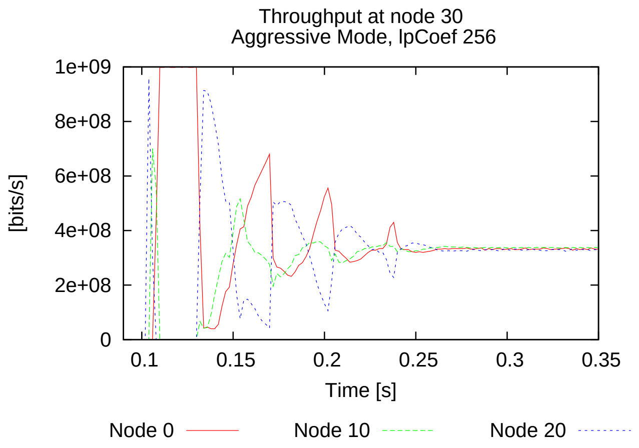
In figures 7a-7c, we have the same set of measurements for the aggressive fairness mode. Fig. 7a shows the results for the original RPR standard implementation, while Fig. 7b shows the result when the fair rate estimate calculated at node 20 is propagated beyond the congestion tail with the *passedTail* bit set. We see that with the original RPR implementation, the sending rate of each active node does not converge to the fair division of bandwidth ($fairRate = \frac{1Gbit/s}{3} = 333Mbit/s$), while with our modified method, the system converges after about 0.2s of simulated time. Fig. 7c shows that if the value of the *lpCoef* parameter is doubled (256), the original RPR fairness algorithm will also converge to a stable state, but the convergence time will be longer.



(a) With the original aggressive mode fairness and lpCoef=128, the fair rate estimation process fails to converge to the fair rate for the scenario described in Fig. 5 above.



(b) With the aggressive mode fairness with the tail-fix implemented and lpCoef=128, the fair rate estimation process converges to the fair rate for the scenario described in Fig. 5 above.



(c) With the original aggressive mode fairness and lpCoef=256, the fair rate estimation process converges to the fair rate, but as seen, the convergence time increases with approximately 30% when compared to aggressive mode with tail-fix implemented and lpCoef=128.

Figure 7: Fairness convergence for the aggressive fairness mode with and without the tail-fix.

B. *Fairness Convergence*

The simulation results described above, showed that our new mechanism improved the convergence time for the fairness algorithm. In the scenario described in this section, we investigate the relation between the size of a congestion domain, the setting of the *lpCoef* parameter, and the convergence time T_c as defined in section II-B. We use $t_s = 50ms$, a sampling interval of $2ms$, and $p = 8\%$. This relatively high setting of p is needed to allow small oscillations in a visually stable system.

We use a ring with 40 km links, with a capacity of 1 Gbit/s. There are three active nodes, node 0, node *i/2* and node *i*, that all send traffic to node 30 at their maximum allowed rate, as shown in Fig. 8. This makes node *i* the head of a congestion domain spanning from node *i* to node 0. *i* is varied from 2 to 28 with step 2, to adjust the size of the congestion domain. Note that the topology described in section V-A is a special case of this scenario, with i=20.

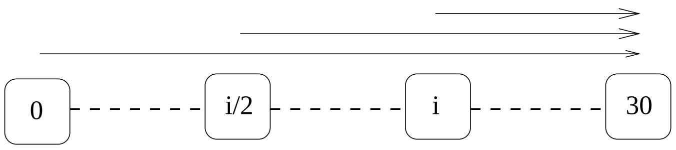


Figure 8: In this scenario, nodes 0, i/2 and i send at their maximum allowed rate to node 30.

Fig. 9 shows the convergence time T_c for different congestion domain sizes and *lpCoef* settings. Results are shown for the original RPR standard (Fig. 9a) and the modified version propagating the fair rate estimate beyond the congestion tail (Fig. 9b).

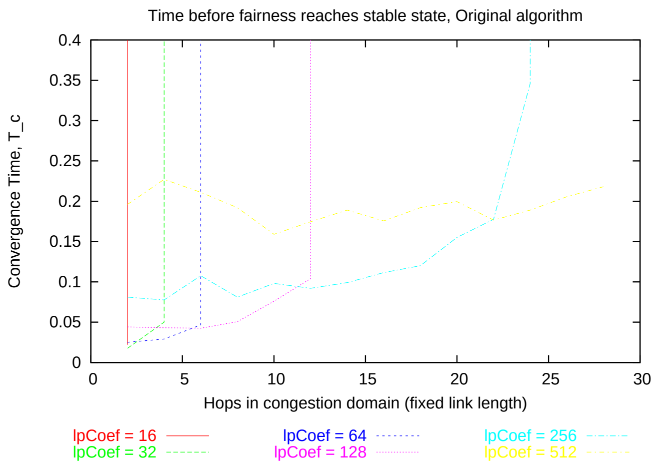
The plots in Fig. 9 show that the propagation of fairness messages beyond the congestion tail allows for a lower setting of the *lpCoef* parameter for a given congestion domain size. The simulations indicate that the value of the *lpCoef* parameter can be at least halved for a given ring size, while maintaining stability/convergence of the fairness algorithm.

We observe that for a given congestion domain size and *lpCoef* setting, our modified algorithm gives a shorter convergence time T_c than the original RPR fairness algorithm.

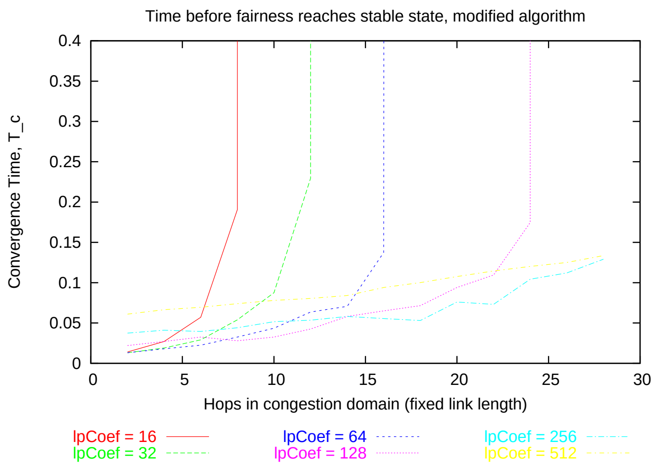
Finally, we see that when the congestion domain size approaches the maximum size for a given *lpCoef*, the increase in stabilization time T_c increases rapidly. Consequently, the optimal setting of the *lpCoef* parameter with respect to T_c, is not always the lowest possible value resulting in a stable system. However, in a dynamic network the traffic patterns and congestion domains will vary. In such an environment, we believe that a low value of the *lpCoef* parameter gives the best overall performance.

C. *A General Congestion Scenario*

In this section, we want to illustrate the behavior of our proposed modification for a general congestion scenario, where some senders are modest and some are greedy and there are some local flows that do not interfere for others. An example of a such scenario is shown in Fig. 10.



(a) Aggressive mode.



(b) Aggressive Mode with tail-fix.

Figure 9: The figure shows fairness convergence as a function of congestion domain size and lpCoef setting. Propagating the fair rate estimate beyond the congestion tail allows for a lower setting of the lpCoef parameter, giving reduced fairness convergence time. The link length is kept constant, so the propagation delay increases linearly with the hop count.

In the figure, the most congested link, is the outgoing link from node 40. There are 7 flows traversing this link, thus the RIAS fair rate (with all greedy senders) is 14.29% of the line-rate. However, as some of the nodes are modest, the RIAS fair rate will be $\frac{100-1-5-15}{4} = 19,75\%$ (the 1, 5 and 15% flow should all get their full demand, while the remaining senders must share the remaining available bandwidth).

Further, the local flows (8,12) and (21,24) should take whatever spare capacity is left on the respective outgoing links. Thus these flows should not interfere with the flows traversing the congested link. That is, other than causing packets transiting nodes 8 or 12 having to await the transmission of one packet, if the transit packet arrives once the transmission of a local packet has started.

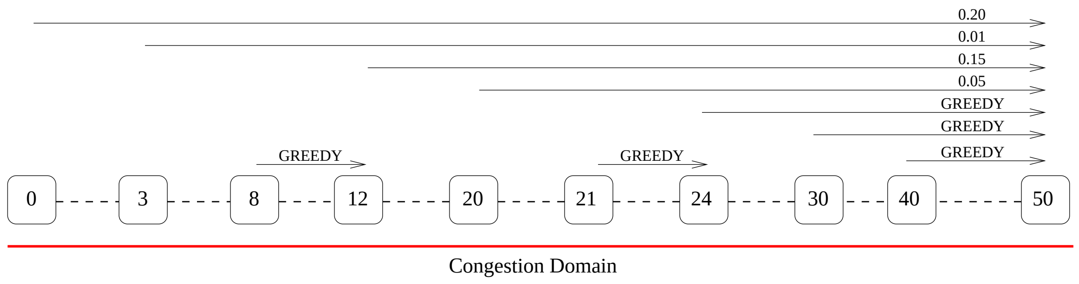
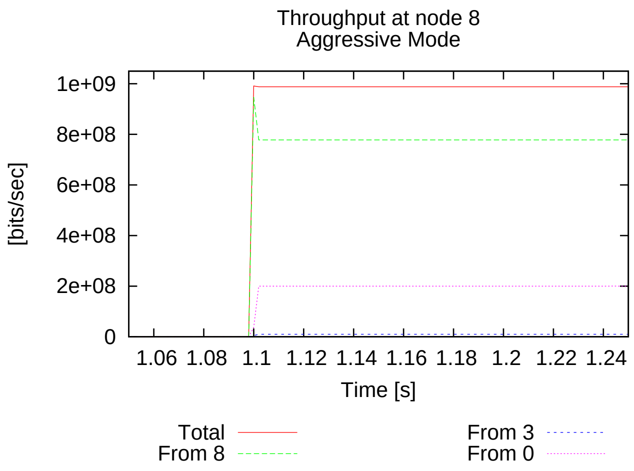
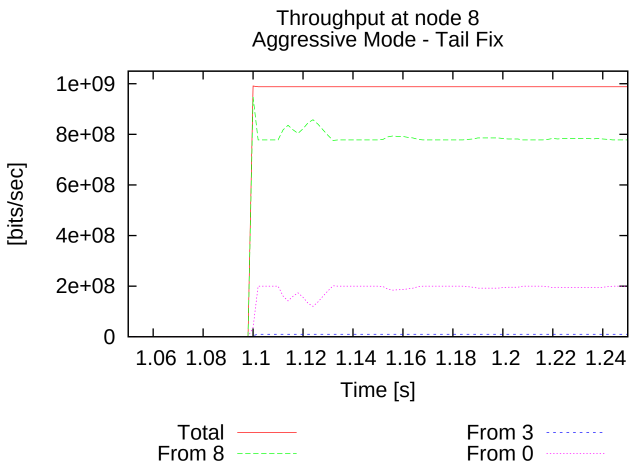


Figure 10: A general congestion scenario. Some senders are modest while others are greedy. Additionally, there are some local flows that do not interfere with any of the other flows.

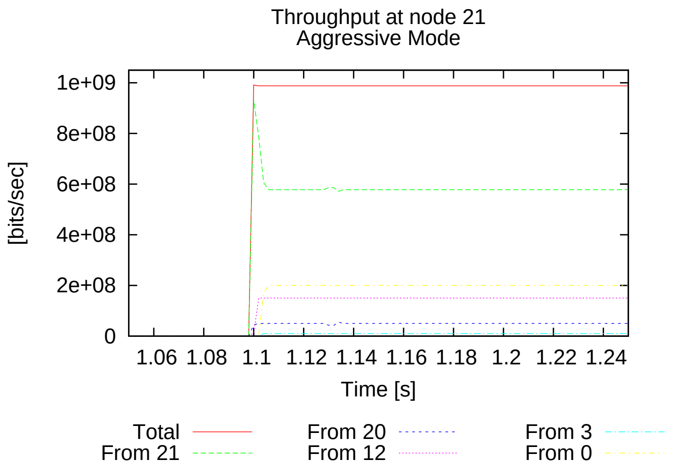


(a) Original algorithm.

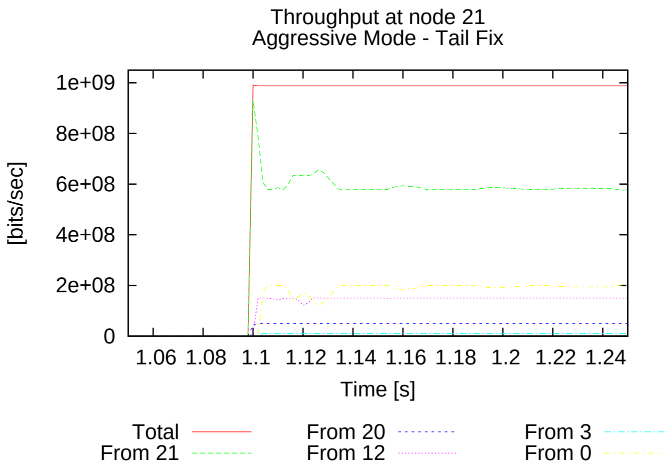


(b) Tail-fix.

Figure 11: Throughput measured at node 8 for aggressive mode fairness. With the tail-fix, the throughput of traffic from node 0 is throttled by the head during convergence to the fair rate, thus node 8 can send more local traffic during this period.



(a) Original algorithm.

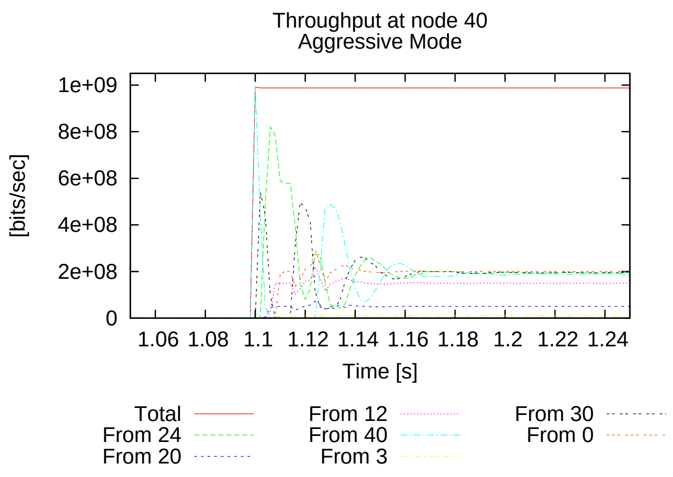


(b) Tail-fix.

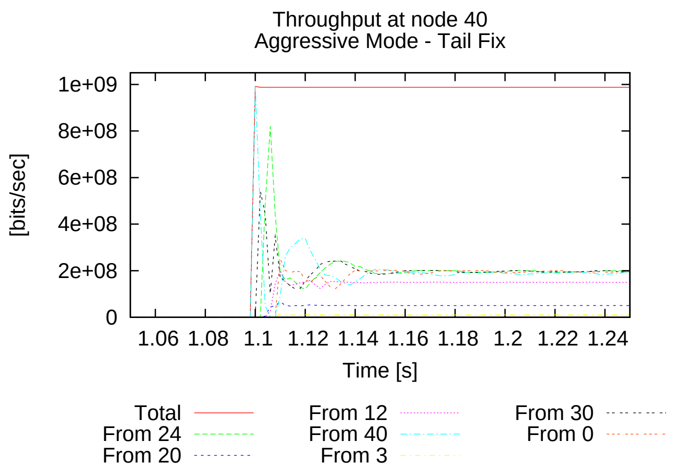
Figure 12: Throughput measured at node 21 with and without the tail-fix.With the tail-fix, the throughput of traffic from nodes 0 and 12 is throttled by the head during convergence to the fair rate, thus node 21 can send more local traffic during this period

We have collected throughput statistics for the scenario illustrated in Fig. 10 for both the original aggressive fairness mode as well as for the aggressive mode with the tail-fix. In figures 11a-13b, we show the throughput measured at nodes 8, 21 and 40 respectively, since, given the applied load, we should expect full link-utilization at these links.

As seen from the figures, the total link-utilization for these links is close to 100% both with and without the tail-fix. Thus our proposed modification does not degrade the link-utilization. The convergence towards the fair rate however



(a) Original algorithm.



(b) With the tail-fix, the fair rate estimation processes converges somewhat faster and with a lesser degree of unfairness.

Figure 13: Throughput measured at node 40 for aggressive mode fairness with and without tail-fix.

differs slightly for aggressive mode fairness and aggressive mode fairness using our proposed tail-fix.

As seen from figures 13a and 13b, the throughput convergence for flows transiting the head is somewhat faster when using our proposed tail-fix. The behavior at the head however has some impact on the convergence at the upstream nodes 8 and 21 which transmit local greedy flows. The fact that the head gradually allows upstream flows with a demand greater than the fair rate to increase their send rate, enables the local flows to utilize a greater share of the available bandwidth for local traffic (during the transient period). While for the original algorithm, the upstream greedy nodes send at a rate greater than their fair share (during the convergence period), thus the local traffic flows suffers from this and have to decrease their send-rate somewhat.

D. Multiple Congestion Domains

In sections V-A, V-B and V-C, we showed how our proposed modification improves the performance when we have (ingress aggregated) flows that are confined within one congestion domain. For multiple and independent congestion domains (i.e. no flows traverses multiple congestion points), we can expect the performance improvement to increase.

In this experiment, we want to evaluate the performance of the system when we have flows, that are not confined to one congestion domain. Fig. 14 shows such a scenario consisting of two congestion domains. The downstream congestion domain consist of the node set: $n \in [3..16]$ (node 3 is tail, while node 16 is head), while the upstream congestion domain consist of the node set: $n \in [0..3]$ (node 0 is tail, while node 3 is head).

The normalized RIAS fair shares for this scenario are shown in the figure. As seen, the downstream head is the most severely congested, having infinite demand flows traversing its outgoing link, each getting 20% of the available bandwidth. Two of these infinite demand flows are crossing the upstream head as well, thus the remaining available bandwidth available for other flows in the upstream congestion domain is limited to $100 - 2 \cdot 20 = 60\%$. Thus, each of the two remaining flows gets $60/2 = 30\%$.

However, the bandwidth shares as given by the RIAS reference model are theoretical values. For this general type of scenarios, where we have a downstream head, more severely congested than the upstream one, there will be a constant alternation between the existence of one and two congestion domains on the ring. A brief explanation of this behavior, in context of the given scenario, is given below.

Let us assume both congestion domains are active on the ring. The downstream one covering nodes 3-16 and the upstream one covering nodes 0-3.

As long as there are two congestion domains in effect, the flows from nodes 0-3 are rate restricted by the upstream head (node 3), thus the upstream head gradually allows the upstream active nodes to increase their send rate towards 25% of the line rate (the fair rate over the upstream congestion point). This will lead to an increasingly severe congestion situation at the downstream head (node 16). Thus the fair rate estimate from node 16 will be gradually lowered, until the point where node 3 no longer fulfills TA1 or TA2. At this point node 3 will no longer maintain its tail responsibility, thus the downstream congestion domain is extended to cover the whole region, covering nodes 0-16.

When this happens, only flows traversing node 16 are rate limited at their ingress point. Thus the aggregate of traffic from nodes 0 and 3 will increase towards the link-rate until the point where node 3 becomes more congested than node 16. At this point, node 3 fulfills TA1, and thus we are back to the starting point, having two congestion domains as shown in Fig. 14.

Thus we have an endless series of cycles, where the sending rates of the nodes are controlled by two different nodes, having two different theoretical target rates. Thus the scenario never converges to the RIAS fair division of rates.

This behavior is illustrated in figure 15. Plotting respectively the throughput and cumulative throughput of the different

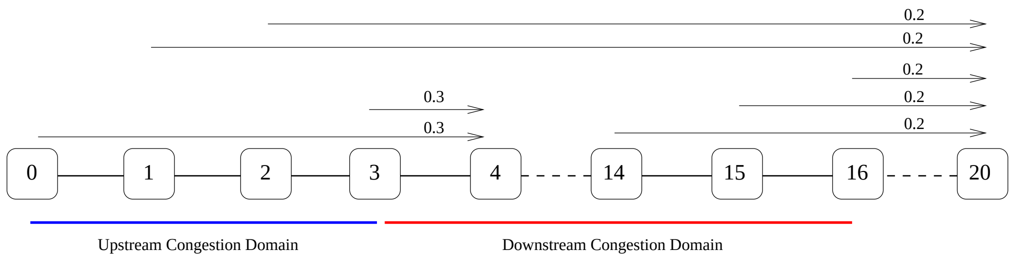


Figure 14: A scenario with (ingress aggregated) flows between two different congestion domains. All flows have infinite demand. The normalized bandwidth shares are the RIAS fair shares.

flows, measured at the upstream congestion point and the downstream receiver. We show the performance using aggressive mode fairness with and without our proposed tail-fix.

When looking at the throughout for the flows measured at both node 3 and node 20, as expected, the fairness algorithm does not converge to the fair rates. If we look at the throughput of the flows measured at node 20, shown in Fig. 15 a) and b). It is clear that for the original algorithm (15 a)), we have an initial period of unfairness, where the upstream nodes (1 and 2) are allowed to send more than their downstream neighbors. This is caused by TA2, where node 3 assumes tail responsibility and prevents nodes 1 and 2 from receiving rate information from the downstream head. With our proposed modification (15 b)), node 1 and 2 will still receive rate information from the downstream head (node 16). Thus the initial period of unfairness is prevented.

In the long term, all flows traversing the downstream head (node 16), should ideally receive 20% of the bandwidth each. However, as the flows from nodes 1 and 2 have to traverse two bottleneck links we should expect the long term throughput of these flows to be somewhat lower than that of the flows only traversing the downstream bottleneck. Fig. 15 c) (original algorithm) and d) (with our proposed improvement) illustrates this. In the figure, we show the cumulative throughput, aggregated over 4s of simulated time. In Tab. III, we show the actual numbers (represented by each flow's share of the total). As seen from the table, the cumulative throughput of flows 1 and 2 are on average approximately 2% lower than those of the downstream ones. For both cases, the total cumulative throughput is the same.

When looking at the cumulative throughput performance of the flows confined within the upstream congestion domain shown in Fig. 15 g) and h), (i.e. the flows from nodes 0 and 3 both going to node 4), we observe that we do not achieve RIAS fair sharing of the 60% bandwidth portion not used by the flows from nodes 2 and 3. In fact, as shown in figures and Table IV, the flow from node 0 receive more than the double amount of bandwidth than the flow from node 3. This is caused by the periods where there is only one congestion domain on the ring. In these periods, node 0 will transmit as much as possible, while node 3 must take whatever portion is left. Thus node 3 is suffering from an excessive sending behavior of its upstream neighbor.

This behavior, as can be expected, will remain regardless of whether we use our proposed improvement or not. To achieve RIAS fairness in this kind of scenario, where we have flows crossing two congestion domains, it is not sufficient for the individual nodes to have knowledge about the nearest congestion point only.

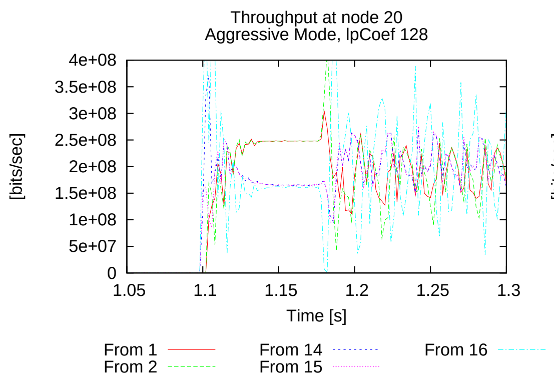
After studying the obtained throughput results, we conclude that the performance for our proposed improvement for this type of scenario is marginally better than that of the original fairness algorithm. It is only for the initial period of unfairness shown in Fig. 15 there are some clear differences.

	From 1	**From 2**	**From 14**	**From 15**	**From 16**
Original	18.8%	18.9%	20.9%	20.9%	20.5%
Modified	19.0%	18.9%	20.8%	20.8%	20.4%

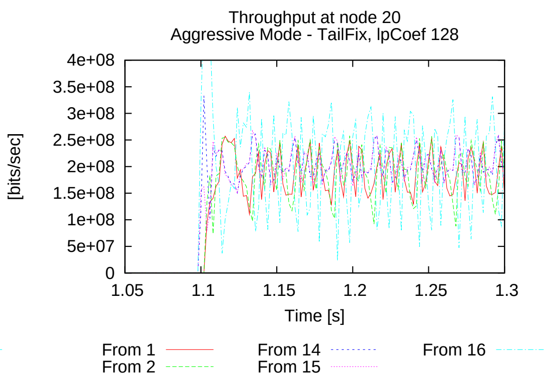
Table III: Cumulative bandwidth shares for flows crossing the downstream congestion head.

	From 0	**From 1**	**From 2**	**From 3**
Original	42.9%	18.9%	18.9%	19.4%
Modified	42.6%	19.0%	19.0%	19.5%

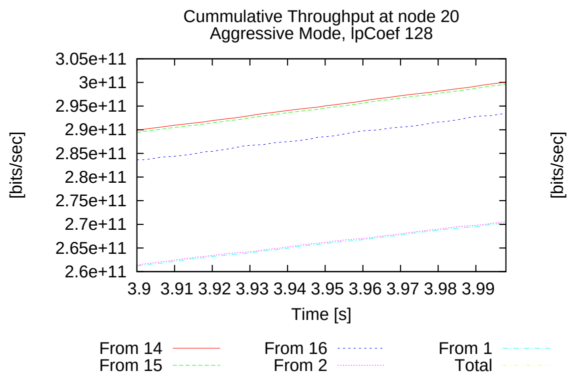
Table IV: Cumulative bandwidth shares for flows crossing the upstream congestion head.



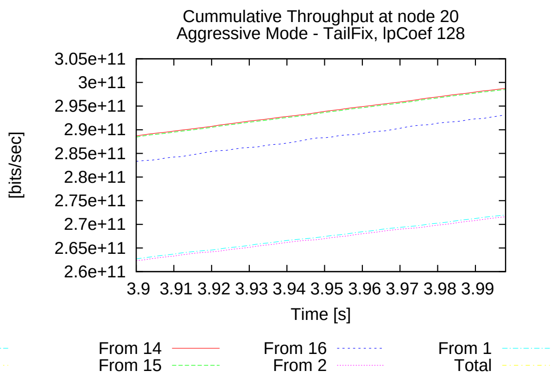
a)



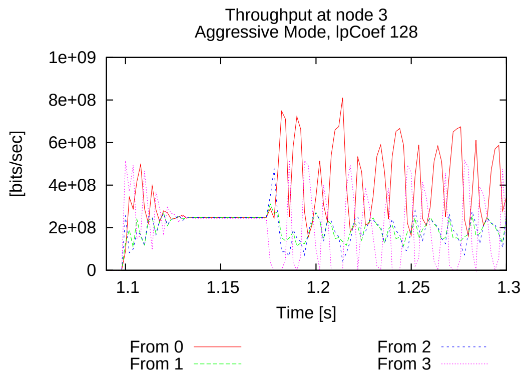
b)



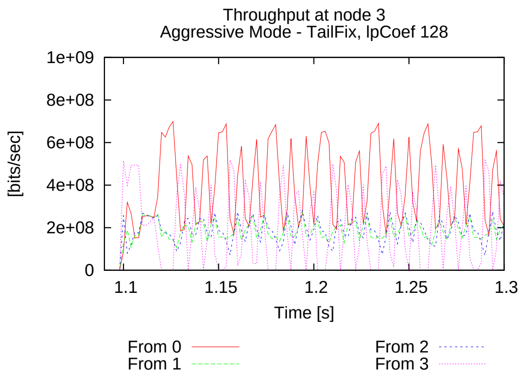
c)



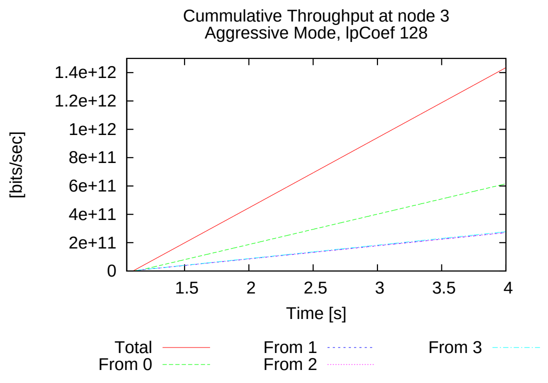
d)



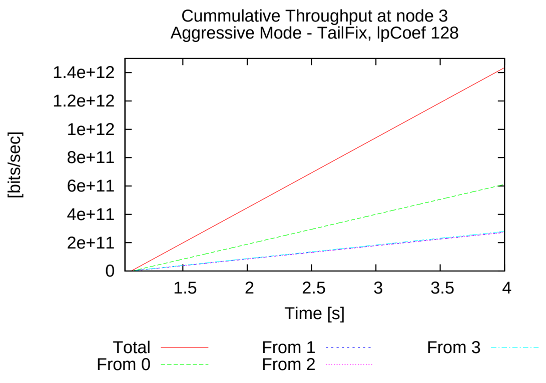
e)



f)



g)



h)

Figure 15: Multiple Congestion Domains with two congestion points. a)-d) shows throughput measurements for the furthest downstream congestion point (measured at the receiver node (node 20)), while e)-h) shows throughput measurements at the furthest upstream (node 3) congestion point. We show performance for the aggressive mode fairness with and without our proposed tail-fix.

VI. Related Work

In an insertion-buffer ring, where the demand for link bandwidth is larger than the available capacity, a fairness algorithm is required to provide fair sharing of bandwidth resources between contending nodes. Many groups have studied the performance and implementation of different algorithms for various insertion-ring architectures [15], [19]–[22]. Several papers have been published studying different RPR performance aspects, both for hardware implementations [15], [23] and simulator models [13], [15], [24]–[26]. Huang et al. presents a thorough analysis of ring access delays for nodes using only one transit queue [24]. Robichaud et al presents ring access delays for class B traffic for both one- and two transit queue designs [25]. Gambiroza et al. focus on the operation of the RPR fairness algorithm and their alternative proposal, DVSR, and their ability, for some given load scenarios to converge to the fair division of rates according to their RIAS fairness reference model [15].

We are not aware of work by others addressing the problem discussed in this paper.

VII. Conclusion

In this paper, we have analyzed a problem, where the the tail of a congestion domain stops the propagation of fair rate estimates (fairness messages) received from a downstream head. The negative side-effect of this is that a node upstream of the tail may send excessive amounts of traffic, thus preventing the convergence of the fairness algorithm. Thus we incur both unfairness as well as non-convergence of the fairness algorithm.

We have proposed a modification, termed the *tail-fix*, that solves these problems by propagating the fair rate estimate beyond the tail of a congestion domain. This modification is easily implemented using one of the currently unused bits in the fairness message as our *passedTail* bit. Our simulations indicate that the modification allows for a substantially lower setting of the *lpCoef* parameter for a given ring size. Further, it gives improved performance as compared to the current RPR standard, by shortening the convergence time for a given ring configuration.

VIII. Further Work

The Resilient Packet Ring is a complex technology, with many operational settings to be configured by the operator. Wrong setting of these parameters can, as illustrated in this paper, lead to unwanted behavior. One area of improvement, that would benefit both the owner and the user of an RPR network, would be the implementation of functionality to ease the task of configuring a Resilient Packet Ring. One opportunity could be the use of protocols that automatically discover the optimal settings for a given network. Specifically, we believe that the *lpCoef* parameter can be more optimally set by dynamically monitoring the conditions in the system.

References

[1] IEEE Computer Society, "IEEE Std 802.17-2004," September 24 2004.

[2] R. Needham and A. Herbert, *The Cambridge Distributed Computing System*. London: Addison-Wesley, 1982.

[3] "IEEE Standard 802.5-1989," IEEE Standard for Token Ring.

[4] F. E. Ross, "An Overview of FDDI: the Fiber Distributed Data Interface," *IEEE J. Select. Areas Commun.*, vol. 7, no. 7, pp. 1043 – 1051, September 1989.

[5] "IEEE Standard 1596-1990," IEEE Standard for Scalable Coherent Interface (SCI).

[6] 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.

[7] "ISO/IECJTC1SC6 N7873," January 1993, specification of the ATMR Protocol (V.2.0).

[8] H. R. van As, W. W. Lemppenau, P. Zafiropulo, and E. Zurfluh, "CRMA-II: A Gbit/s MAC Protocol for Ring and Bus Networks with Immediate Access Capability," in *Proceedings of the Ninth Annual European Fibre Optic and Local Area Networks Conference (EFOC/LAN'91)*, London, June 1991, pp. 262 – 277.

[9] W. W. Lemppenau, H. R. van As, and H. R. Schindler, "Prototyping a 2.4 Gbit/s CRMA-II dual-ring ATM LAN and MAN," in *Proceedings of the 6th IEEE Workshop on Local and Metropolitan Area Networks*, 1993, pp. 17 – 18.

[10] E. Hafner, Z. Nendal, and M. Tschanz, "A Digital Loop Communication System," *IEEE Trans. Commun.*, vol. 22, no. 6, pp. 877 – 881, June 1974.

[11] 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.

[12] H. R. van As, "Media Access Techniques: The Evolution Towards Terabit/s LANs and MANs," *Computer Networks and ISDN Systems*, vol. 26, no. 6-8, pp. 603 – 656, March 1994.

[13] 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.

[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, To appear in.

[15] 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.

[16] C. L. Phillips and R. D. Harbor, *Feedback Control Systems*, 2nd ed. Prentice-Hall, 1991.

[17] H. Tyan, "Design, Realization and Evaluation of a Component-Based Compositional Software Architecture for Network Simulation," Ph.D. dissertation, Ohio State University, 2002.

[18] "OPNET Modeler. http://www.opnet.com." [Online]. Available: http://www.opnet.com/

[19] I. Cidon, L. Georgiadis, R. Guerin, and Y. Shavitt, "Improved fairness algorithms for rings with spatial reuse," *IEEE/ACM Trans. Networking*, vol. 5, no. 2, pp. 190–204, 1997.

[20] I. Kessler and A. Krishna, "On the cost of fairness in ring networks," *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 306–313, 1993.

[21] D. Picker and R. Fellman, "Enhancing SCI's fairness protocol for increased throughput," in *IEEE Int. Conf. On Network Protocols*, October 1993.

[22] J. Schuringa, G. Remsak, 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.

[23] A. Kirstadter, A. Hof, W. Meyer, and E. Wolf, "Bandwidth-efficient resilience in metro networks - a fast network-processor-based RPR implementation," in *Proceedings of the 2004 Workshop on High Performance Switching and Routing, 2004. HPSR*, 2004, pp. 355 – 359.

[24] C. Huang, H. Peng, F. Yuan, and J. Hawkins, "A steady state bound for resilient packet rings," in *Global Telecommunications Conference, (GLOBECOM '03)*, vol. 7. IEEE, December 2003, pp. 4054–4058.

[25] Y. Robichaud, C. Huang, J. Yang, and H. Peng, "Access delay performance of resilient packet ring under bursty periodic class B traffic load," in *Proceedings of the 2004 IEEE International Conference on Communications*, vol. 2, June 20-24 2004, pp. 1217 – 1221.

[26] 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.