Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Networks 49 (2005) 161–171
Computer Networks
www.elsevier.com/locate/comnet

Distributed bandwidth allocation for resilient packet ring networks

Fahd Alharbi, Nirwan Ansari *

Advanced Networking Laboratory, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA

Received 12 October 2004; accepted 24 December 2004
Available online 2 March 2005

Responsible Editor: J. Sole-Pareta

Abstract

The Resilient Packet Ring (RPR), defined under IEEE 802.17, has been proposed as a high-speed backbone technology for metropolitan area networks. RPR is introduced to mitigate the underutilization and unfairness problems associated with the current technologies, SONET and Ethernet, respectively. The key performance objectives of RPR are to achieve high bandwidth utilization, optimum spatial reuse on the dual rings, and fairness. The challenge is to design an algorithm that can react dynamically to the traffic flows in achieving these objectives. The RPR fairness algorithm is comparatively simple, but it poses some critical limitations that require further investigation and remedy. One of the major problems is that the amount of bandwidth allocated by the algorithm oscillates severely under unbalanced traffic scenarios. These oscillations are barrier to achieving spatial reuse and high bandwidth utilization. DVSR was another algorithm proposed to solve the fairness issue with no oscillation at the steady state, but at the expense of a high computational complexity O($N\log N$), where N is the number of nodes in the ring.

In this paper, we propose the Distributed Bandwidth Allocation (DBA) algorithm to allocate bandwidth fairly to RPR nodes with a very low computational complexity O(1) that will converge to the exact max–min fairness in a few round trip times with no oscillation at the steady state.

Keywords: RPR; SONET; Ethernet; Spatial reuse; DVSR; DBA

1. Introduction

Rings are the most prevalent metro technologies because of their protection and fault tolerance properties, but the current metropolitan ring

* Corresponding author. Tel./fax: +1 973 596 3670.
E-mail addresses: faa3@njit.edu (F. Alharbi), nirwan.ansari@njit.edu (N. Ansari).

1389-1286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2004.12.004

networking technologies exhibit several limitations. In a SONET ring, each node is granted with the minimum fair share, but it is not possible to reclaim the unused bandwidth; moreover, 50% of the potentially available bandwidth is reserved for protection, thus resulting in poor utilization. On the other hand, Gigabit Ethernet assures full statistical multiplexing at the expense of fairness.

RPR [1,2] shares SONET's ability in providing fast recovery from link and node failures as well as inherits the cost and simplicity of Ethernet.

Like SONET/SDH, RPR is a ring-based architecture consisting of two optical rotating rings: one is referred to as the inner ringlet, and the other the outer ringlet (Fig. 1). RPR defines three service classes of user traffics: Class A with guaranteed rate and jitter, Class B with a committed information rate (CIR) and bounded delay and jitter, and the best effort traffic (Class C). In RPR [3,4], packets are removed from the ring at the destination so that different segments of the ring can be used at the same time for different flows; as a result, the spatial reuse feature is achieved. Enabling the spatial reuse feature (concurrent transfers over the same ring) introduces the challenge of guaranteeing fairness among the nodes sharing the same link.

The remaining of the paper is organized as follows: Section 2 discusses the fairness issue in RPR and the limitation of the RPR fairness algorithm; Section 3 describes our proposed bandwidth allocation algorithm along with mathematical analysis; Section 4 presents simulation results for different configurations and comparisons among existing algorithms; we finally conclude in Section 5 and emphasize that our proposed dynamic bandwidth allocation scheme does not require per-flow information and has a very low complexity of O(1).

2. Fairness and flow control in RPR

The flow control in RPR is achieved by enabling a congested node to send the fairness message according to its measurements to the upstream nodes to throttle ingress data rates in order to eliminate the state of congestion and apply fairness among all the participating nodes.

2.1. RPR node architecture

The RPR node architecture is illustrated in Fig. 2. First, each node uses per-destination rate controller to throttle the traffic entering the ring to support virtual destination queuing and avoid head-of-line blocking. Second, each node uses byte counters to measure transit traffic and node traffic. These measurements are used by the fairness algorithm to compute the fair rate which is fed-back to the upstream nodes in the form of a control message. Nodes that receive the control message will use the control message information with their local information to throttle their rates accordingly.

Third, to ensure hardware simplicity and that the transit path is lossless, the RPR node does not include per-ingress or per-flow queues on the transit path; instead, it supports two scheduling modes. In the single-queue mode (Fig. 3(a)), the

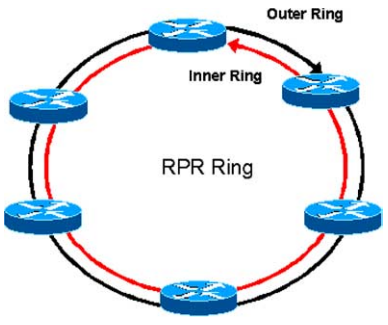

Fig. 1. The Resilient Packet Ring.

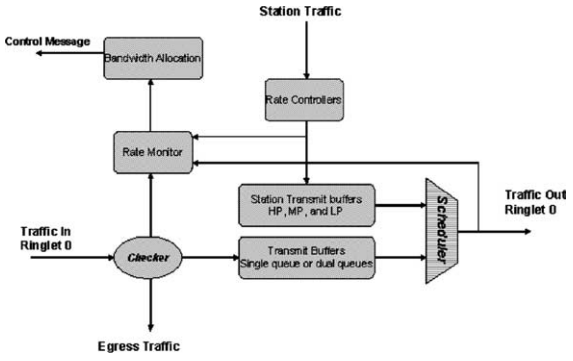

Fig. 2. The RPR node architecture.

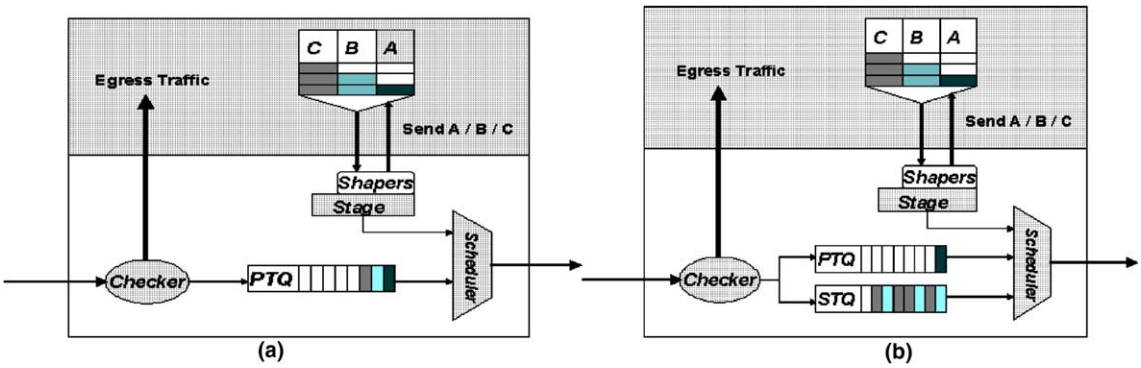

Fig. 3. The RPR scheduling: (a) single queue; (b) dual queue.

transit path is a single FIFO and the transit traffic has strict priority over the station traffic. On the other hand, in the dual-queue mode (Fig. 3(b)), the transit path consists of two queues: Primary Transit Queue (PTQ) for Class A traffic, and Secondary Transit Queue (STQ) for Class B and Class C traffic; in this mode, PTQ will be served first. When PTQ is empty, STQ has strict priority over the station traffic when the queue length exceeds the STQ threshold; otherwise, the station traffic is served in the following order: Class A, then Class B. If the station (node) has no Class A or B traffic, then Class C traffic will be served.

Throughout the rest of the paper, we consider the best effort (Class C) traffic in which flows share the available bandwidth at the congested link in a weighted manner.

2.2. Fairness in RPR

The RPR fairness algorithm [2–5] operates in two modes: the aggressive mode and the conservative mode. In both modes, each node measures at the output of the scheduler the byte count of all serviced transit traffic named *forward-rate* and the byte count of all serviced station traffic named *my-rate*. These measurements will be taken over a fixed *aging-interval*. Both modes have the same measurements, but use them differently in detecting congestion and computing fair rates.

In the aggressive mode (RPR-AM), the transit path has a dual-queue [4,6]. Node k is considered to be congested when either *STQ-depth*[k] > low-threshold, where the low-threshold is equal to 1/8 of the STQ size, or *my-rate[k]* + *forward-rate[k]* > unreserved-rate, where the unreserved-rate is equal to the link capacity minus the reserved rate for the high priority class traffic.

When node k is congested, it calculates its local-fair-rate as the normalized value of its own *my-rate value*, and then sends a fairness control message to upstream nodes containing *my-rate*. On the other hand, if the node is not congested, it sends a *NULL* value as the fairness control message to inform the upstream nodes to increase their rates.

In the conservative mode (RPR-CM), the transit path has a single queue [4,6] and each node has an access timer to measure the time between its transmitted packets. Here, the node is considered to be congested when either the access time expires, or *my-rate[k]*+*forward-rate[k]* > low-threshold, where the low threshold is equal to 0.8 of the link capacity.

In the conservative mode, each node not only measures *my-rate* and *forward-rate*, but also measures the number of active nodes where a node i will be counted active if at least a single packet was received from node i during the aging interval. If node k is congested in this aging interval, but was not congested in the previous interval, it will send a fairness control message containing the fair rate equal to the unreserved bandwidth divided by the number of active nodes.

If node k continues to be congested, then it sends a normalized local-fair-rate depending on the value of the sum of *my-rate[k]* and *forward-rate[k]*.

If this value is less than the low threshold, the local-fair-rate will ramp up. On the other hand, the local-fair rate will ramp down when the sum is greater than the high threshold, which is 0.95 of the unreserved-rate.

When node $k-1$ receives the control message from node k, it will set its rate limiter value, namely, the allowed-rate based on the control message value, and then send a control message to the other upstream nodes with the value according to the following:

(a) Minimum of $my\text{-}rate[k-1]$ and the received control message value *if* node $k-1$ is congested.
(b) A *Null* value *if* node $k-1$ is not congested.
(c) A *Null* value *if* node $k-1$ is congested, but $my\text{-}rate[k-1] > forward\text{-}rate[k-1]$ because node $k-1$ is the cause for congestion.

When a node receives a control message with a *NULL* value, it will increase its allowed-rate to reclaim the unused bandwidth.

Now consider the simple parking lot scenario [4] in Fig. 4(a), where the flow from node 1 to node 3 is greedy while the flow from node 2 to node 3 is a low rate 50 Mbps, and both links have a capacity of 622 Mbps. In the case of using the aggressive mode Fig. 4(b), node 2 will be congested when the sum of its rate and the rate of flow (1, 3) is greater than the link capacity; then, it sends the fairness control message with its *my-rate* of 50 Mbps to node 1; accordingly, node 1 throttles its allowed-rate to 50 Mbps. When the congestion is resolved, node 2 sends the fairness control message with a *NULL* value, and so node 1 can increase its rate to claim the unused bandwidth until congestion occurs again starting a new cycle of oscillation.

On the other hand, using the conservative mode Fig. 4(c), node 2 will send the fairness control message with the fair rate equal to the link capacity divided by the number of active nodes (in this case, 2). When the congestion is resolved, node 2 sends the fairness control message with a *NULL* value so that node 1 can increase its rate to claim the unused bandwidth until the congestion occurs again.

Both modes, RPR-AM and RPR-CM, incur oscillations in the allocated bandwidth, resulting in a bandwidth loss.

Knightly and co-workers [6] proposed an algorithm called "Distributed Virtual Time Scheduling in Ring" (DVSR) to overcome the problems encountered in the RPR fairness algorithm. Unlike the RPR fairness algorithm, in DVSR, each node

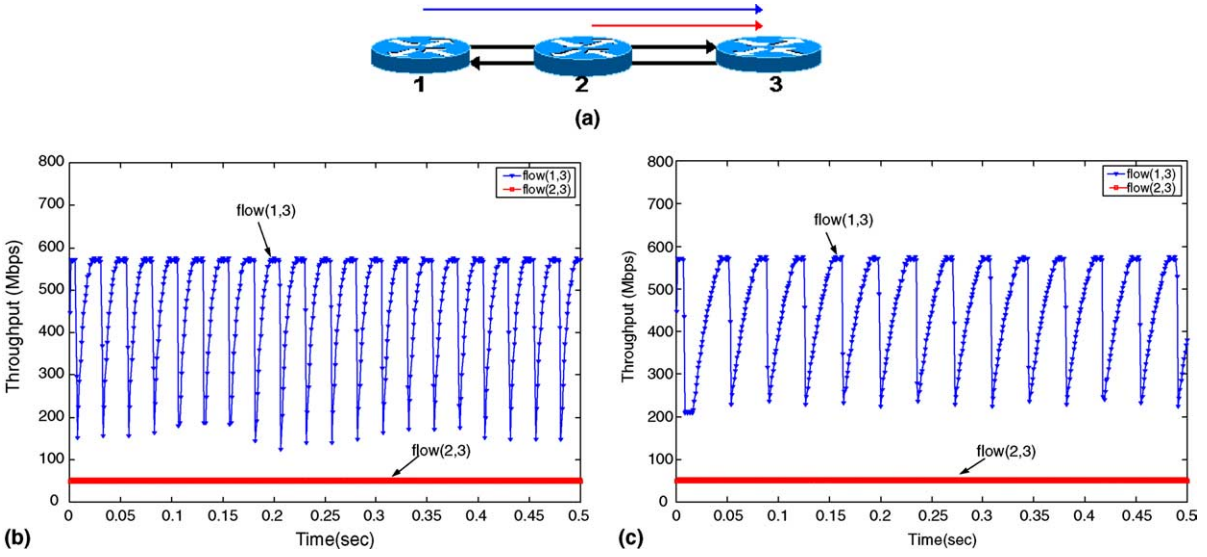

Fig. 4. Simple parking lot. (a) Scenario setup, (b) aggressive mode and (c) conservative mode.

uses per ingress byte counters (measurement modules) to estimate the demand from every ingress node including itself during the measurement interval T. These measurements are used by the fairness algorithm to compute the fair rate. DVSR has a computational complexity of $O(N\log N)$ where N stands for the number of nodes.

In this paper, we propose a fairness algorithm, which does not require per-source information as in RPR-CM [2] and DVSR [6], that can achieve a performance compatible with DVSR but with a much lower computational complexity of O(1).

3. The DBA fairness algorithm

In this section, we introduce a new bandwidth allocation algorithm referred to as the Distributed Bandwidth Allocation (DBA) algorithm. This algorithm adopts the Ring Ingress Aggregated with Spatial Reuse (RIAS) fairness concept[4,6], where the level of traffic granularity at a link is defined as an ingress-aggregated (IA) flow, i.e., the aggregate of all flows originated from the same node but destined to different nodes. At the state of congestion, all nodes should be able to send the same amount of data on the congested link relative to the other nodes.

Without loss of generality, throughout the analysis we consider only one of the two rings, the outer ring with N nodes numbered from *0 to* $N-1$ along the ring direction.

Definition 1. The available bandwidth for the best effort traffic (Class C) is defined as

$$C = Link\_Capacity - reserved\_BW, \tag{1}$$

where *reserved_BW* is the bandwidth reserved for the higher priority class traffic.

Definition 2. At node k, the ingress aggregated traffic demand of node i during a measurement interval T is defined as follows:

$$R_i = \sum_{j=(k+1)\bmod N}^{(i-1)\bmod N} r_{i,j}. \tag{2}$$

That is, R_i is equal to the sum of all flows $r_{i,j}$ originated from node i, traversing through node k, and destined to node j.

Definition 3. According to the (RIAS) fairness concept, the fair share at link k is

$$F_k(n) = \frac{C - B(n)}{w(n)}, \tag{3}$$

where $B(n)$ is the sum of the arrival rates of flows bottlenecked elsewhere or at their ingress points at time n and $w(n)$ is the number of flows bottlenecked at link k. Here, we have adopted the same index to a particular node and link, i.e., link k refers to the link in the direction of the traffic flow of node k.

Unfortunately, this simple calculation requires dynamic per-source information as in DVSR [6].

3.1. Derivation of the algorithm

Recall that each node i will send through link k at a rate according to the received fair rate from node k. Thus, the rate of source i through link k at time n is

$$R_i(n) = \rho_i F_k(n), \tag{4}$$

where ρ_i is the activity level of source i with respect to the fair rate, $F_k(n)$, of the current interval [7,8]. The activity level is equal to one for flows bottlenecked at link k, and less than one for flows bottlenecked elsewhere.

Define M as the number of flows traversing link k, and the arrival rate $\widetilde{A}(n)$ at link k can be expressed as a function of the link fair rate $F_k(n)$ as follows:

$$\begin{aligned}\widetilde{A}(n) &= \sum\nolimits_{i=1}^{M} R_i \\ &= \sum\nolimits_{i=1}^{M} \rho_i F_k(n) \\ &= F_k(n)\sum\nolimits_{i=1}^{M} \rho_i.\end{aligned} \tag{5}$$

From Eq. (5), we see that the arrival rate $\widetilde{A}(n)$ is a continuous, non-decreasing and concave function of the link fair rate $F_k(n)$.

In [7], we exploited the relationship between the arrival rate $\widetilde{A}(n)$ and the fair rate $F_k(n)$ to estimate

the next fair rate $F_k(n+1)$ at the end of every time interval T.

As shown in Fig. 5, a line connects the current point $(F_k(n), \widetilde{A}(n))$ with the origin and intersects with the line representing the available bandwidth C at the new estimated fair rate $F_k(n+1)$.

Define $\widetilde{M}$ as the effective number of flows traversing link k

$$\widetilde{M} = \sum_{i=1}^{M} \rho_i. \tag{6}$$

The effective number of flows is the sum of the activity levels of flows traversing link k and is less than or equal to the number of flows M.

The effective number of flows can be estimated by a linear function that connects the origin and $(F_k(n), \widetilde{A}(n))$ as shown in Fig. 5.

$$\widetilde{M} = \frac{\widetilde{A}(n)}{F_k(n)}. \tag{7}$$

The goal of the fairness algorithm is to maximize the link fair rate $F_k(n)$ subject to the constraint:

$$\widetilde{A}(n) \leqslant C. \tag{8}$$

Now, we propose the following formula to estimate the link fair rate:

$$F_k(n+1) = F_k(n) + \frac{1}{\widetilde{M}}(C - \widetilde{A}(n)). \tag{9}$$

DBA first estimates the effective number of flows, which can be estimated by the slope of the line connecting the origin and the current point $(F_k(n), \widetilde{A}(n))$. Then, it uses Eq. (9) to estimate the fair rate of the next interval. The goal is to adjust $F_k(n)$ so that the total arrival rate $\widetilde{A}(n)$ matches the available bandwidth and $F_k(n)$ converges to the optimal fair rate F_k^*.

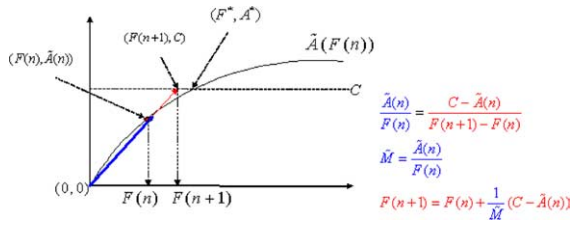

Fig. 5. DBA estimation process.

Note that one of the important features of the DBA algorithm is its low computation complexity of O(1), thus making DBA scalable for a ring network with any number of nodes, i.e., independent of N. Moreover, DBA does not require per-source information as in DVSR.

Fig. 6 shows the pseudo code of DBA, where the total arrival rate $\widetilde{A}(n)$ is updated at the arrival of every packet traversing link k. At the end of the measurement interval T, Eq. (7) and Eq. (9) are used to calculate the next advertised fair rate $F_k(n+1)$.

3.2. Proof of convergence

Let k be the bottlenecked link. The number of flows traversing link k is M, where M' is the number of flows bottlenecked elsewhere or at their ingress points, and $M'' = M - M'$ is the number of flows bottlenecked at link k. Let $R_{b1}, R_{b2}, \ldots, R_{bM'}$ be the flows bottlenecked elsewhere, and $R_1, R_2, \ldots, R_{M''}$ be the flows bottlenecked at link k.

At the end of the nth measurement interval ($t = nT$), the effective number of flows is estimated as

$$\widetilde{M} = \frac{\widetilde{A}(n)}{F_k(n)}, \tag{10}$$

where $\widetilde{A}(n) = \sum_{i=1}^{M} R_i$ is the arrival rate at node k, and $F_k(n)$ is the advertised fair rate of the previous interval.

The next advertised fair rate is

$$F_k(n+1) = F_k(n) + \frac{1}{\widetilde{M}}(C - \widetilde{A}(n)). \tag{11}$$

Initialization: at the beginning of every measurement interval T, reset $\tilde{A}(n)$
During the measurement interval
At the arrival of a packet from node i with length L *bytes*
$\tilde{A}(n) = \tilde{A}(n) + L$
At the end of the measurement interval:
The effective number of active flows: $\tilde{M} = \max\left(1, \frac{\tilde{A}(n)}{F_k(n)}\right)$
The advertised fair rate: $F_k(n+1) = F_k(n) + \frac{1}{\tilde{M}}(C - \tilde{A}(n))$

Fig. 6. The DBA pseudo-code.

Substituting Eq. (10) into Eq. (11) yields

$$F_k(n+1) = F_k(n)\frac{C}{\widetilde{A}(n)}. \tag{12}$$

Define $\alpha(n) = \widetilde{A}(n)/C$ as the load factor, and rewrite Eq. (12) as

$$F_k(n+1) = \frac{F_k(n)}{\alpha(n)}. \tag{13}$$

According to the load factor value, two cases are considered. First, consider the case where the load factor $\alpha(n)$ is less than one. In this case, the arrival rate is less than the available bandwidth and the link is under-loaded. According to Eq. (13), the advertised fair rate will increase. If all flows are bottlenecked elsewhere ($M'' = 0$), the fair rate has been achieved. On the other hand, if there are some flows bottlenecked at link k ($M'' > 0$), the bottlenecked flows will continue to increase their rates until the load factor becomes greater than or equal to one.

Second, consider the case where the load factor $\alpha(n)$ is greater than one. In this case, the arrival rate is greater than the available bandwidth and the link is over-loaded. According to Eq. (13), the advertised fair rate will decrease and the participating flows will decrease their rates. This will continue until the load factor becomes less than or equal to one.

It is obvious from the above two cases that the load factor oscillates around one and converges to one. Thus, in the following analysis, we assume that the load factor is close to one.

Next, we shall show that the iterative algorithm Eq. (11) will generate a sequence of $F_k(n)$ that will converge to the optimal value of the advertised fair rate $F_k^* = C - \sum_{i=1}^{M'} R_{bi}/M''$.

Note that the iterative equation Eq. (11) is in the form of

$$F_k(n+1) = F_k(n) + \lambda[\nabla^2 D(F_k(n))]^{-1}\nabla D(F_k(n)). \tag{14}$$

That is, the link fair rate is adjusted in the direction of the gradient, where

$$\nabla D(F_k(n)) = C - \widetilde{A}(F_k(n)). \tag{15}$$

Here, λ is a positive step size, and in our case is equal to one, and $[\nabla^2 D(F_k(n))]^{-1}$ is the inverse of the Hessian.

It is well known that the Newton method Eq. (11), where the gradient is scaled by the inverse of the Hessian typically converges faster than the gradient projection; see [9, pp. 201].

The Hessian $\nabla^2 D(F_k(n)) = \widetilde{M}$ is approximated by using two points, the current point of $(\widetilde{A}(n), F_k(n))$ and the origin (0, 0).

Hence, the above iterative equation converges, and the stable value of the link advertised fair rate is detailed as follows:

First, assume that all the flows are bottlenecked at link k. In this case, $M' = 0$ and $M'' = M$. All flows are running at the fair rate $R_i(n) = F_k(n)$, and the total arrival rate at node k is

$$\widetilde{A}(n) = F_k(n)\sum_{i=1}^{M}\rho_i. \tag{16}$$

Since all flows are bottlenecked at link k, the effective number of flows is $\sum_{i=1}^{M}\rho_i = M$.

Substituting the value of $\widetilde{A}(n)$ into Eq. (12) with a load factor $\alpha(n)$ of one at the steady state yields

$$F_k(n+1) = \frac{C}{M}, \tag{17}$$

which is the desired value for F_k.

Finally, assume that some flows are bottlenecked elsewhere. These flows will have their rates $R_{b1}, R_{b2}, \ldots, R_{bM'}$ stabilized, and the allocated bandwidth for these flows is $B = \sum_{i=1}^{M'} R_{bi}$.

Since we have a load factor $\alpha(n)$ of one at the steady state, we have

$$\sum_{i=1}^{M''} R_i = C - \sum_{i=1}^{M'} R_{bi}, \tag{18}$$

and

$$\sum_{i=1}^{M''} R_i = M'' F_k(n). \tag{19}$$

Substituting Eq. (19) into Eq. (18) yields

$$F_k(n) = \frac{C - \sum_{i=1}^{M'} R_{bi}}{M''}. \tag{20}$$

Substituting the value of $F_k(n)$ into Eq. (13) yields

$$F_k(n+1) = \frac{C - \sum_{i=1}^{M'} R_{bi}}{M''}, \quad (21)$$

which is indeed the desired value for F_k and the proof is complete.

3.3. Fair rate advertisement

At the end of every measurement interval T, every node k will broadcast a control message containing the value of the last computed fair rate F_k. Thus, every node is aware of the supported fair rates at all links.

3.4. Rate limiting and per flow sub-allocation

Upon receiving all the last computed fair rates, the node itself will do sub-allocation for all the flows that are sharing the same link and are destined to different egress nodes to support virtual destination queuing in order to achieve per flow fairness and avoid head of line blocking (HLB).

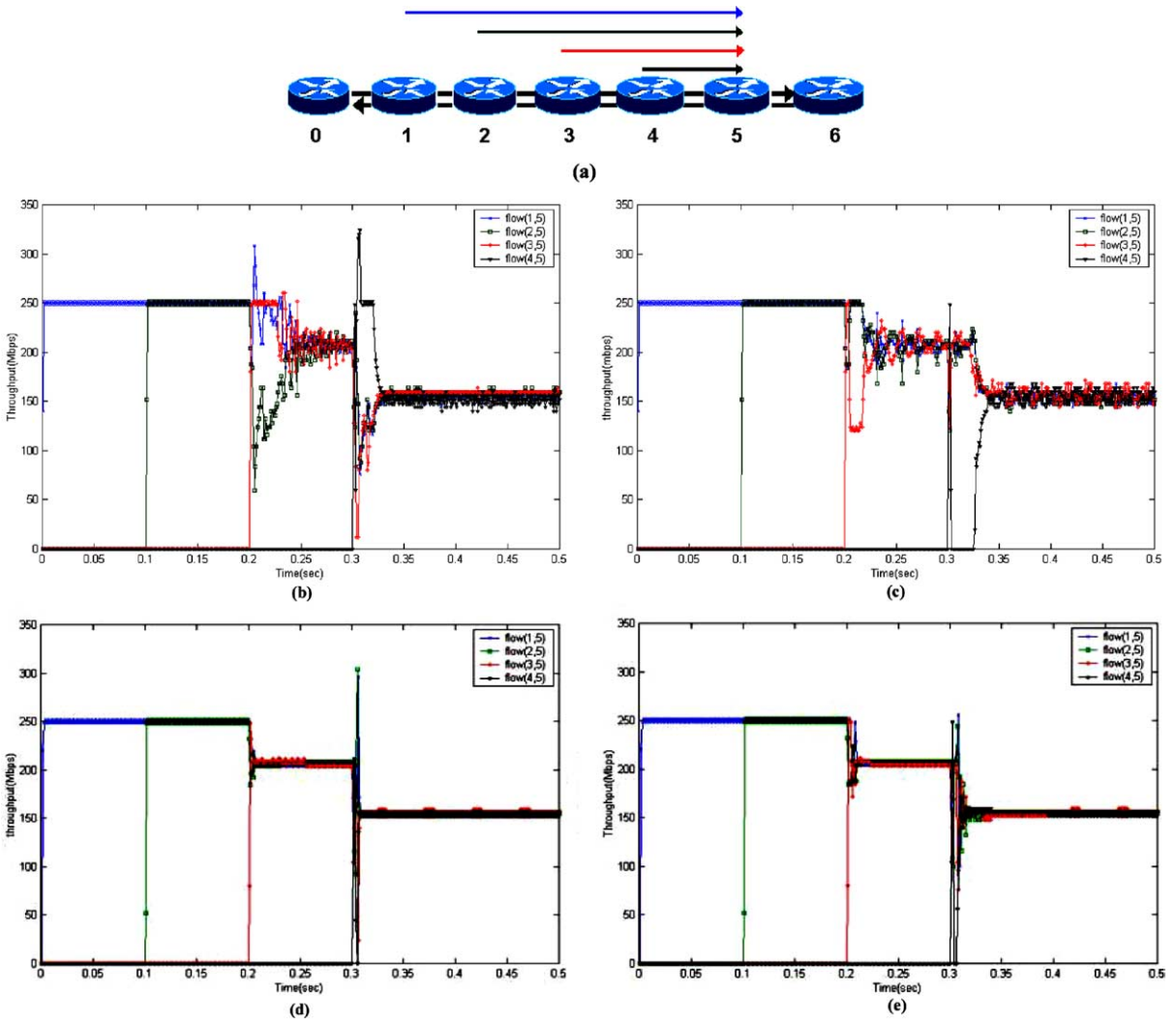

Fig. 7. Parking Lot Scenario. (a) Scenario setup, (b) RPR-AM, (c) RPR-CM, (d) DVSR and (e) DBA.

4. Simulation results

Here, we have considered three scenarios. First, we consider the parking lot scenario to show the performance of our algorithm in achieving fairness. Second, we demonstrate convergence of our algorithm even in the unbalanced traffic scenario. Finally, we study the performance of the DBA algorithm in the presence of different traffic models. All simulation results are obtained by using the RPR simulator [10].

4.1. Parking lot scenario

Fig. 7(a) shows the parking lot scenario [4,6]. In this experiment, we compare the convergence time of the fairness algorithms. The links have the same capacity of 622 Mbps, and each link has a propagation delay of 0.1 ms. All flows are UDP flows, with a rate equal to 250 Mbps. The flows, flow (1,5), flow (2,5), flow (3,5) and flow (4,5) start at time 0, 0.1, 0.2 and 0.3 s, respectively. The measurement time interval is set to $T = 1$ ms.

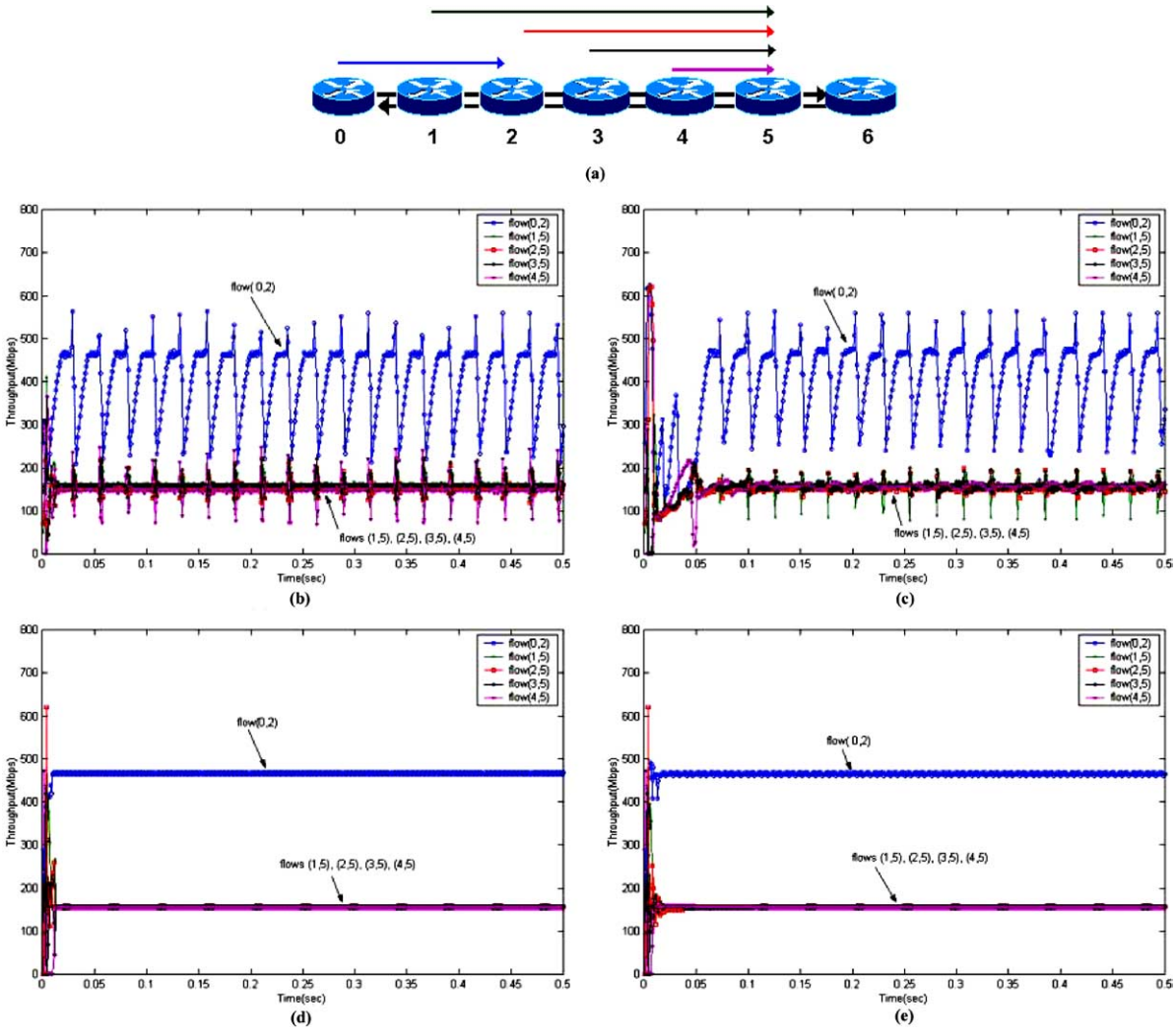

Fig. 8. Available Bandwidth Re-claim Scenario. (a) Scenario setup, (b) RPR-AM, (c) RPR-CM, (d) DVSR and (e) DBA.

The performance of the RPR fairness algorithms RPR-AM and RPR-CM are shown in Fig. 7(b) and (c), respectively. In both algorithms, flows oscillate for a significant period of time before converging to the fair rate. Moreover, the range of oscillation is large.

The results shown in Fig. 7(d) exhibit that DVSR needs only a few milliseconds to converge to the RIAS faire rate, however, at the expense of O(Nlog$N$) computational complexity; it also requires per-source information.

Simulations shown in Fig. 7(e) have verified that DBA converges to the RIAS fair rates in a few measurement intervals with a very low computational complexity of O(1), and it does not require per-source information as compared to DVSR [6]. The oscillation has also been significantly reduced.

4.2. Available bandwidth re-claim scenario

In this experiment, we consider the scenario [6] illustrated in Fig. 8(a), where all flows are greedy and start at time $t = 0$.

Fig. 8(b) shows the RPR-AM algorithm where all the flows (0,2), (1,5), (2,5), (3,5) and (4,5) start at time 0 s. After some time, due to the congestion experienced at link 4, flows (1,5), (2,5), (3,5) and (4,5) will converge to the fair rate (155.5 Mbps), meanwhile node 0 starts to reclaim the unused bandwidth at link 1. When node 1 becomes congested, it sends *my-rate* value of 155.5 Mbps to node 0, thus throttling flow (0,2) to 155.5 Mbps. When the congestion at node 1 is cleared, node 0 starts to increase its rate again starting another cycle of oscillation.

On the other hand, using the RPR-CM algorithm(Fig. 8(c)), node 1 will send *my-rate* value equal to the available bandwidth divided by the number of sources using link 1 (two in this case). Thus, flow (0,2) will be throttled to 311 Mbps. When the congestion at node 1 is cleared, node 0 starts to increase its rate again starting another cycle of oscillation.

Fig. 8(d) shows that DVSR converges very fast to the RIAS faire rates at the expense of a high computational complexity and the need for per-source information.

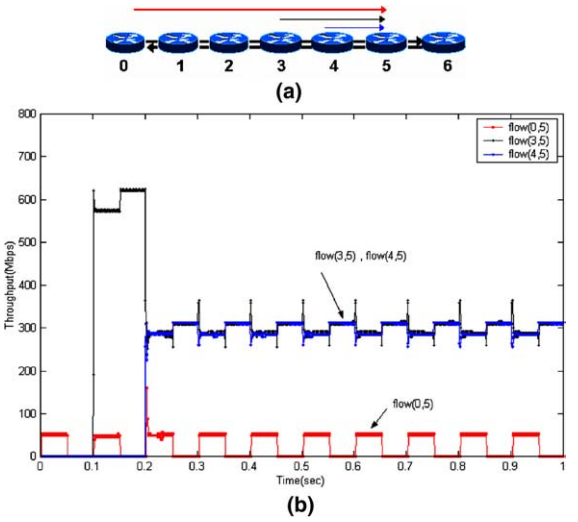

Fig. 9. Different traffic models. (a) Scenario setup and (b) DBA.

Using the DBA algorithm, nodes converge very fast to the RIAS fair rates. Moreover, the oscillation is significantly damped, as shown in Fig. 8(e).

4.3. Different traffic models

In this experiment (Fig. 9(a)), the congested link is shared by different traffic models.

Flows (3,5) and (4,5) are greedy UDP flows and start at time 0.1 and 0.2 s, respectively.

Flow (0,5) is an ON/OFF flow. During the ON period, flow (0,5) sends at a rate equal to 50 Mbps.

Fig. 9(b) shows that our proposed algorithm reacts responsively to the presence of the ON/OFF flow, and converges very fast to the RIAS fair rates.

5. Conclusions

In this paper, we have proposed the Distributed Bandwidth Allocation (DBA) algorithm for Resilient Packet Ring networks to achieve spatial reuse, high bandwidth utilization, and fairness. Unlike DVSR [6] and RPR-CM [2], DBA does not require per-source information and converges to the RIAS fair rates in a few measurement intervals with a very low computational complexity, i.e. O(1), and is thus scalable.

References

[1] IEEE Standard 802.17: Resilient Packet Ring. http://ieee802.org/17.

[2] IEEE Draft P802.17, draft 3.0, Resilient Packet Ring, November 2003.

[3] F. Davik, M. Yilmaz, S. Gjessing, N. Uzun, IEEE802.17 resilient packet ring tutorial, IEEE Communications Magazine 42 (3) (2004) 112–118.

[4] V. Gambiroza, P. Yuan, E. Knightly, The IEEE802.17 media access protocol for high speed metropolitan-area resilient packet rings, IEEE Network 18 (3) (2004) 815.

[5] F. Davik, S. Gjessing, The stability of the Resilient Packet Ring Aggressive Fairness Algorithm, Proceedings of 13th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN2004), Mill Valley, CA, USA, 24–27 April 2004, pp. 17–22.

[6] V. Gambiroza, P. Yuan, L. Balzano, Y. Liu, S. Sheafor, E. Knightly, Design, analysis, and implementation of DVSR: A fair, high performance protocol for Packet Rings, IEEE/ACM Transactions on Networking 12 (1) (2004) 85102.

[7] F. Alharbi, N. Ansari, Low complexity distributed bandwidth allocation for Resilient Packet Ring Networks, Proceedings of IEEE Workshop on High Performance Switching and Routing HPSR2004, April 2004, pp. 277–281.

[8] S. Fahmy, R. Jain, S. Kalyanaraman, R. Goyal, B.Vandalore, On determining the fair bandwidth share for ABR connections in ATM networks, Proceedings of the IEEE International Conference on Communications (ICC98), vol. 3, June 1998, pp. 1485–1491.

[9] D.P. Bertsekas, J.N. Tsitsiklis, Parallel and Distributed Computation, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[10] S. Gjessing, The Simula RPR Simulator implemented in Java, Simula Research Laboratory Technical Report 2003-12, December 2003.

Fahd Alharbi received the B.S.E.E. from College of Technology at Riyadh in 1995, and M.S.E.E. from Fairleigh Dickinson University in 2000. Between 1995 and 1998, he was an instructor at the College of Technology at Jeddah. He is currently pursuing his doctoral degree at the New Jersey Institute of Technology.

Nirwan Ansari received the B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. from NJIT, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively.

He joined the Department of Electrical and Computer Engineering, NJIT, as an assistant professor in 1988, and has been promoted to a full professor since 1997. His current research focuses on various aspects of multimedia communications and high-speed networks. He is a technical editor of the IEEE Communications Magazine, Computer Communications, the ETRI Journal, and the Journal of Computing and Information Technology. He authored with E.S.H. Hou Computational Intelligence for Optimization (1997, and translated into Chinese in 2000), and edited with B. Yuhas Neural Networks in Telecommunications (1994), both published by Kluwer Academic Publishers. He has frequently been invited to give talks and tutorials. He was a distinguished speaker at the 2004 Sendai International Workshop on Internet Security and Management, and a keynote speaker at the IEEE/ACM co-sponsored International Conference on E-Business and Telecommunication Networks (ICETE2004). He has also contributed over 200 publications in journals, and conferences.

He initiated (as the General Chair) the First IEEE International Conference on Information Technology: Research and Education (ITRE2003), was instrumental, while serving as its Chapter Chair, in rejuvenating the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award and a 2003 Chapter Achievement Award, served as the Chair of the IEEE North Jersey Section and in the IEEE Region 1 Board of Governors during 2001–2002, and currently serves in various IEEE committees. He was the 1998 recipient of the NJIT Excellence Teaching Award in Graduate Instruction, and a 1999 IEEE Region 1 Award.