Dynamic Max-Min Fairness in Ring Networks*

G. Anastasi[1], L. Lenzini[1], M. La Porta[2], Y. Ofek[3]

[1]University of Pisa, Dept. of Information Engineering
Via Diotisalvi 2 - 56126 Pisa, Italy

[2]Basis Information Technology
viale della Repubblica 141 - 59100 Prato, Italy

[3]Synchrodyne, Inc.
2600 Netherland Ave., Riverdale, NY 10463

Abstract

Ring networks are enjoying renewed interest as Storage Area Networks (SANs), i.e., networks for interconnecting storage devices (e.g., disk, disk arrays and tape drives) and storage data clients. This paper addresses the problem of fairness in ring networks with spatial reuse operating under dynamic traffic scenarios. To this end, in the first part of the paper the Max-Min fairness definition is extended to dynamic traffic scenarios and an algorithm for computing Max-Min fair rates in a dynamic environment is introduced. In the second part of the paper the extended Max-Min fairness definition is used as a measure to compare the performance in dynamic conditions of three fairness algorithms proposed for ring-based SANs. These algorithms are characterized by different fairness cycle sizes (number of links involved in each instance of the fairness algorithm), i.e., different complexity. The results show that the performance increases as the fairness cycle size decreases. In particular, the Global-cycle algorithm (implemented in the Serial Storage Architecture - SSA), whose cycle size is equal to the number N of links in the ring, exhibits the lowest performance, while the One-cycle algorithm, so called because of its cycle size equal to 1, has the best performance. The Variable-cycle algorithm, whose cycle size changes between 1 and N links, performs in between and provides the best tradeoff between performance and complexity.

Keywords: Dynamic Max-Min Fairness, Ring Networks, Fairness algorithms, Cycle size, Storage Area Networks.

1 Introduction

Ring networks are currently enjoying renewed interest as *Storage Area Networks* (SANs). A SAN [Phi98] is a high speed network which provides connectivity between storage devices (e.g., disk, disk arrays and tape drives) and their application clients. The fundamental difference between a LAN and a SAN is that SAN nodes are not peer devices but are either storage resources or hosts running storage data clients. The key emerging SAN technologies are FC-AL (Fiber Channel - Arbitrated Loop, ANSI Standard X3T11) and *SSA* (Serial Storage Architecture, ANSI Standard X3T10) [Du98]. *SSA* is a dual-ring network with spatial bandwidth reuse, in which the underlying network is the MetaRing [Cid93, Che93, and Ofe94].

The main motivation for developing ring networks with spatial bandwidth reuse, such as the MetaRing, Orwell [Fal85], ATMR [Ohn89] and others, is to increase the aggregate ring throughput, in each direction, beyond link capacity, since spatial bandwidth reuse enables concurrent access in each direction of the ring by

*This work has been carried out under the financial support of the Italian Ministero dell'Università e della Ricerca Scientifica e Tecnologica (MURST) in the framework of the MOSAICO (Design Methodologies and Tools of High Performance Systems for Distributed Applications) Project.

more than one node. Increasing the aggregate throughput as much as possible is a key requirement in a SAN environment characterized by the exchange of very large data volumes between storage devices and client hosts.

However, using media access such as buffer insertion and slotted ring, in an unregulated manner, may result in unfair transmission opportunities, and in extreme scenarios it can lead to complete starvation. Starvation can occur if some nodes are constantly being "covered" by upstream ring traffic, and are thus not able to access the ring (see Figure 1). To avoid such situations the SSA implements the Global (SAT) fairness algorithm described in Section 3.1.

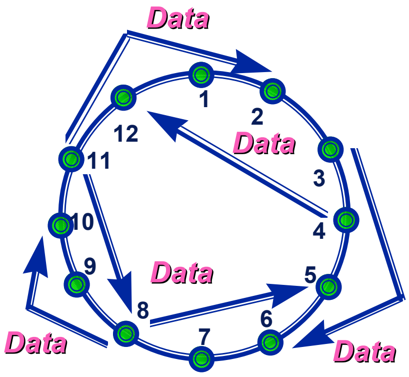


Figure 1: Dual-ring with spatial bandwidth reuse

The fairness of ring networks with spatial bandwidth reuse has been studied extensively in the past decade. However, studying fairness algorithms, using the Max-Min fairness definition as an optimal measure of fairness, have so far been done with reference to static traffic scenarios, i.e., assuming that each node has a unique destination which remains fixed in time. Such scenarios are, of course, not realistic. This paper thus analyzes dynamic traffic patterns, i.e., scenarios where each node is involved in several simultaneous sessions with different destinations.

However, the analysis of dynamic traffic scenarios poses some challenges that need to be faced beforehand. The main problem is the identification of an appropriate performance measure for fairness. The Max-Min Fairness definition [Ber87] [Hay81] [Jaf81] is universally used as an optimal fairness measure in static scenarios. Therefore, in the first part of this paper we extend this definition to dynamic scenarios and introduce an algorithm, hereafter the *Dynamic Max-Min algorithm*, for the computation of Max-Min fair rates in dynamic contexts. This algorithm is a generalization of the algorithm proposed in [Ber87] for static scenarios.

In the remaining part of the paper the Dynamic Max-Min algorithm is used to compare (by simulation) the performance of three fairness algorithms that are suitable for use in SANs based on ring networks with spatial reuse (e.g., SSA). These fairness algorithms are characterized by different *fairness cycle sizes* [Ana99]. The fairness cycle size is defined (see Section 3) as the number of communication links involved in every instance of the fairness algorithm (several instances of the algorithms can be executed concurrently on the same ring) and gives a measure of the algorithm's complexity. The comparison is aimed at investigating which of the algorithms under examination provides the best complexity vs. performance tradeoff, and hence, is the optimal choice for a practical implementation.

The paper is organized as follows. Section 2 extends the Max-Min Fairness definition to dynamic scenarios and introduces the Dynamic Max-Min algorithm for rate computation. Section 3 describes the fairness algorithms analyzed in the paper. Section 4 introduces the simulation environment while Session 5 compares the performance of the fairness algorithms. Finally, conclusions are drawn in Section 6.

2 Max-Min Fairness in Dynamic Scenarios

2.1 Characterization of a dynamic scenario

Throughout we will refer to a data exchange between a source node and a destination node as a *session*. In a dynamic scenario each node usually maintains sessions with several counterparts at the same time which means that consecutive packets transmitted by the same node may have different destinations. Furthermore, sessions originated by the same node are expected to be characterized by different parameters such as message rate, message size distribution, and so on. In order to characterize the dynamic scenario, the above parameters should be explicitly specified for each session. However, things can be made simpler by referring to the *fluid flow* model.

According to this model packets sent through the network are assumed to be infinitely small in size so that the traffic flow generated by a node can be viewed as a fluid flow. Furthermore, if a node is exchanging data with several counterparts, the traffic flow it generates can be imagined as consisting of as many sub-flows as there are sessions originated by that node. Although different sub-flows are directed to different destinations and are characterized by different amounts of traffic, they nevertheless all have a static path.

This means that in the fluid flow model a *dynamic* traffic scenario is completely specified once it is known how the traffic flow generated by each node is distributed among its various sessions (i.e., destinations). A dynamic scenario can thus be represented by the *Flow Matrix* defined below.

Definition 1: (Flow Matrix): *Let N be the number of nodes in the network. The Flow Matrix F related to a dynamic traffic scenario is a square matrix of size N whose generic entry* $F_{ij}\ (i = 1,2,...,N;\ j = 1,2,...,N)$ *specifies the fraction of flow generated by node i and destined to node j. If node i is idle (i.e., it is not transmitting to any other node) then it is* $F_{ij} = 0$ *for any* j .

Note that the elements of the Flow Matrix are real numbers less than or equal to 1 since they represent the fraction of flow related to a session of the total flow generated by a node. This means that the elements of *F* can be thought of as session probabilities. Figure 2 shows an example of a dynamic scenario in a bus network with *N=4* nodes. The corresponding Flow Matrix is reported in Figure 3 .

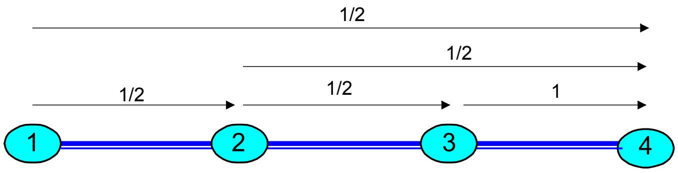


Figure 2: Bus network with N=4 nodes operating in dynamic conditions.

DESTINATION

S O U R C E

	1	**2**	**3**	**4**
1	0	1/2	0	1/2
2	0	0	1/2	1/2
3	0	0	0	1
4	0	0	0	0

Figure 3: Flow Matrix for the traffic scenario depicted in Figure 2.

Remark. In the fluid flow model, sessions of a dynamic scenario (like the one shown in Figure 2) are static and each of them is characterized by a different weight given by the related fraction of flow or session probability. On the other hand, in a static scenario all the sessions would have a unitary weight.

2.2 Max-Min Fairness definition

The Max-Min Fairness definition is based on the fluid flow model and assumes that each session has a fixed path. It is given in terms of session rates [Bert87]. Let r_s denote the rate of session *s*. Let C_l be the capacity of link l. Furthermore, let S_l be the set of all the sessions using link l, and T_l denote the sum of rates of all the sessions using link l, i.e., $T_l = \underset{s\in S_l}{\ } r_s$.

Definition 2: *An allocation is said to be feasible if it is ,* $r_s \geq 0$ *for any session s and* $T_l \leq C_l$ *for any link l.*

Definition 3: (Max-Min Fairness): *An allocation of a rate* r_s *for session s is Max-Min fair if it is feasible and for each session s,* r_s *cannot be increased (while maintaining feasibility) without decreasing* $r_{s'}$ *for some session s' for which* $r_{s'} \leq r_s$.

Definition 3 implicitly assumes that all the sessions have the same weight and, hence, must be treated equally. In Section 2.1 it was shown that under the fluid flow model assumption sessions of a dynamic traffic scenario have a static path but they are characterized by different weights. This means that Definition 3 does not immediately apply to dynamic scenarios. However, it can be easily extended to dynamic contexts by introducing a minor modification.

Definition 4: (Dynamic Max-Min Fairness): *An allocation of a rate* r_s *for session s is Max-Min fair if it is feasible and for each session s,* r_s *cannot be increased (while maintaining feasibility) without decreasing* $r_{s'}$ *for some session s' for which* $r_{s'} / w_{s'} \leq r_s / w_s$ *where* w_s *(*$w_{s'}$*) is the session probability of session s (s').*

Note that Definition 4 is reduced to Definition 3 if all the sessions have equal weights.

2.3 Dynamic-Max-Min algorithm

We now give a procedure, hereafter *Dynamic-Max-Min algorithm*, for computing Max-Min fair rates in a dynamic traffic scenario assuming that the corresponding Flow Matrix is known. This algorithm is a generalization of the algorithm proposed in [Ber87] and is reduced to it when all sessions have equal (unitary) session probabilities. The fundamental difference is that in the algorithm reported in [Ber87] all the sessions are considered as equal and, thus, their rates are increased equally step by step. On the other hand, in the Dynamic-Max-Min algorithm session rates are increased proportionally to session probabilities.

The algorithm starts with an all-zero rate vector r and increases the rate of each session by an amount proportional to the session probability until it is T_l=C_l for one or more links l. When the previous condition holds for link l we say that link l is *saturated*. At this point, all the sessions using a saturated link have rates which are proportional to their session probability.

At the next step of the algorithm, the rates of all the sessions not using the saturated links are increased proportionally to the related session probabilities until one or more new links become saturated. The algorithm continues by incrementing, at each step, the rates of sessions that do not pass through any saturated link. It stops when all the sessions pass through at least one saturated link.

The Dynamic-Max-Min algorithm is formally specified in Figure 4 by a pseudo-code. At each step, ***S*** denotes the set of sessions not passing through any saturated link, while ***L*** indicates the set of links not yet saturated. Furthermore, Σ_l denotes the sum of session probabilities of sessions belonging to ***S*** and using link *l*. Finally, Δr indicates the unitary rate increment and $\Delta r \cdot w_s$ is the actual increment of rate for any session *s* belonging to ***S*** (w_s denotes the session probability of session *s*).

If $\mathscr{S}$ denotes the set of all the sessions and $\mathscr{L}$ the set of all the links, then the initial conditions for the algorithm are as follows. Obviously, $r_s = 0$ for any session $s \in \mathscr{S}$ and $T_l = 0$ for any link $l \in \mathscr{L}$ (lines 1-2). Furthermore, since no link is saturated at the initial instant is ***S***= $\mathscr{S}$and ***L***= $\mathscr{L}$ (lines 3-4).

```
1   for each session $s \in \mathscr{S}$ do $r_s = 0$;
2   for each link $l \in \mathscr{L}$ do $T_l = 0$;
3   $\boldsymbol{S} = \mathscr{S}$;
4   $\boldsymbol{L} = \mathscr{L}$;

5   repeat
6     for each link $l \in \boldsymbol{L}$ do
7         $\Sigma_l =$ sum of session probabilities for sessions $s \in \boldsymbol{S}$ sharing link $l$;
8     $\Delta r = \min_{l \in \boldsymbol{L}} \left( \frac{C_l - T_l}{\Sigma_l} \right)$;
9     for each session $s \in \boldsymbol{S}$ do $r_s = r_s + \Delta r \cdot w_s$;
10    for each link $l$ do $T_l = \underset{s\ \text{crossing}\ l}{\quad} r_s$;
11    $\boldsymbol{L} = \{ l \mid C_l - T_l > 0 \}$;
12    $\boldsymbol{S} = \{ s \mid s\ \text{does not cross any saturated link} \}$;
13  until S is empty.
```

Figure 4: Dynamic-Max-Min algorithm

At each step, the algorithm computes for each link the sum of session probabilities of sessions using that particular link and not passing through any saturated links. The unitary rate increment is then determined (line 8) and rates of sessions not passing through any saturated links are increased proportionally to their session probability (line 9). Finally, sets ***L*** and ***S*** are updated (lines 10-11 and 12, respectively). As an example, in the Appendix it is shown how to compute the Max-Min fair rates of sessions in the dynamic scenario shown in Figure 2.

Proving that rates provided by the Dynamic-Max-Min algorithm satisfy Definition 4 is almost immediate. In fact, the rate vector *r* is *feasible* since (**i**) all its elements are clearly greater than or equal to zero; and (**ii**) for each link $l \in \mathscr{L}$ the sum of rates of sessions using *l* is not greater than the link capacity (i.e., $T_l \leq C_l$) because, at each step, the algorithm increases the rates of sessions that do not pass through saturated links in order to saturate one or more new links, and stops when all the sessions pass through at least one saturated link.

Furthermore, at each step, the rate of each session *s* not passing through any saturated links is increased by an amount $\Delta r_s = \Delta r \cdot w_s$ where Δr is chosen in order to saturate one or more new links. This implies that it is not possible to increase the rate r_s of session *s* (while maintaining feasibility) without decreasing $r_{s'}$ for some session s' for which $r_{s'} / w_{s'} \leq r_s / w_s$.

3 Fairness Algorithms for Ring Networks

This section introduces the fairness algorithms which will be compared in Section 5. We assume a dual ring network with *N* nodes. The transmission time around the ring is divided into time slots of equal duration. Each slot starts with a busy-bit; if this bit is 0, the slot is empty, otherwise the slot is full. A node can transmit a packet (or cell) only if it receives an empty slot. The packet is removed from the ring by its destination node and the slot becomes empty. The *shortest path* criterion is used for packet routing, i.e., to choose one of the two possible directions toward the final destination.

Throughout the paper the concepts of Quota and Fairness cycle size will be used extensively, as defined below.

Definition 5: Quota, q, *is used to define the number of transmission permits a node received from the fairness algorithm. Each permit is used to access one empty slot.*

Definition 6: Fairness cycle size *is defined by the number of links used in order to determine when a node can get another transmission Quota.*

Based on the fairness cycle size the following three algorithms are defined:

Global-cycle algorithm, in which the cycle size is equal to the number of links in the ring - *N*. This means that only one instance of the fairness algorithm is executed over the ring at any given time.

Variable-cycle algorithm, in which the cycle size changes between 1 and *N* links. This implies that between *N* and 1 instances of the fairness algorithm are executed, in each direction, at any given time.

One-cycle algorithm, which means that there is one fairness cycle for every link. This implies that *N* instances of the fairness algorithm are executed, in each direction, at any given time - one for each communication link.

The complexity associated with each of the algorithms is summarized in Table 1. As shown in Section 3.1 the Global-cycle algorithm requires the exchange of only one bit of information. The Variable-cycle algorithm requires the use of the node ID, and therefore, in a ring of *N* nodes the complexity is O(lg *N*), however, if it is a bus the complexity is only O(1) (no need for an ID), as shown in Section 3.2. In Section 3.3 it is shown that the One-cycle algorithm requires that every ring interface will maintain a table with an entry to all other nodes, and the control messages are bit-vectors of size *N* (for a ring with *N* links), and therefore, the complexity of this algorithm is O(*N*).

	Communication/Hardware complexity
Global-cycle	O(1)
Variable-cycle	For ring O(lg *N*) For bus O(1)
One-cycle	O(*N*)

Table 1: Complexity Measures of the Fairness Algorithms

3.1 Global-cycle Fairness Algorithm

The Global-cycle algorithm views each direction of the ring as a single shared communication resource, as it is in token ring. The fairness cycle size is thus of *N* links. The aim of such an algorithm is to ensure that all nodes have an equal opportunity to access the ring. In order to achieve global fairness access to each direction of the ring is regulated by a control signal, called SAT (an abbreviation of SATisfied), which circulates in the same or opposite direction to the data traffic it regulates, see Figure 5 and [Cid93, Ofe94].

In principle, the node forwards the SAT signal upstream without any delay, unless it is not SATisfied or “starved.” By “starved” we mean that the node has not been able to use the permitted number of slots since the last time it forwarded the SAT signal. More specifically, the node is satisfied if between two consecutive visits of the SAT signal, the node has used *q* slots or if its output buffer is empty. If the node is not satisfied, it will hold the SAT until it is satisfied and then forward the SAT upstream. After a node forwards the SAT, it can use up to q more slots, before receiving and forwarding the SAT signal again.

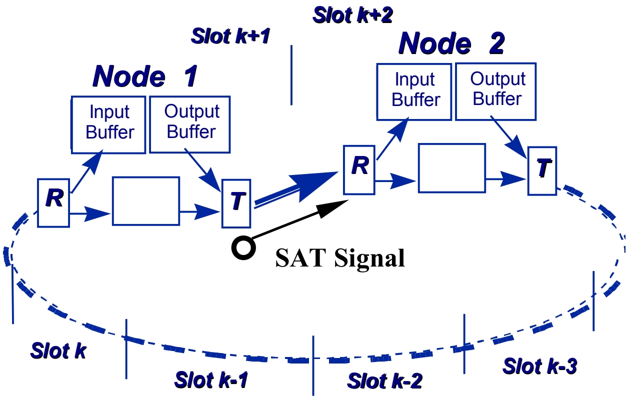


Figure 5: Global-cycle fairness with the SAT signal

3.2 Variable-cycle Fairness Algorithm

In the Variable-cycle algorithm, each of the N links is viewed as a communication resource. The cycle is created dynamically and its size is a function of the instant traffic interference of nodes with one another. This algorithm has two modes of operation: (1) *unrestricted* and (2) *restricted*, see [Che93] for details.

Initially a node is unrestricted and can transmit whenever it encounters an empty slot. This mode is identified as the *Free Access* (FA) state, shown in Figure 6. A node will enter the restricted mode if it cannot transmit (does not encounter empty slots) or if it receives a signal from a downstream node that cannot transmit. The restricted mode has three states: *Tail* (T), *Body* (B) and *Head* (H), shown in Figure 6. In the restricted mode a node can transmit only a predefined Quota, q, before it transits back to the non-restricted mode.

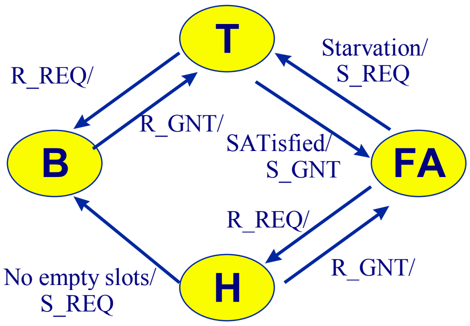


Figure 6: Variable-cycle fairness state transition diagram

The algorithm uses two control signals to facilitate the transition between the states and operation modes, as shown in Figure 7:

REQ: This signal initiates the restricted period of operation and is forwarded upstream over the congested ring segment.

GNT: This signal is used, when the node is satisfied, to terminate the local fairness cycle.

The two control signals create fairness cycles over the congested segments of the ring. Note that if the ring is congested, the time interval a node is in the non-restricted mode can be zero. In this case, a node will transmit one Quota in every fairness cycle. The size of a fairness cycle depends on the number of nodes this node is interfering with, directly and indirectly. For example, if node A interferes with node B, and node B interferes with node C, then node A also interferes with node C (i.e., the interference is a transitive closure operation).

A starved node triggers a fairness cycle by sending the REQ signal upstream and by entering the tail (T) State. Upon reception of the REQ signal, a node enters the restricted mode of operation, and if its upstream is idle, it will enter the head (H) State. If this node cannot provide empty slots downstream, it will forward the REQ upstream, and will enter the body (B) state. Upon satisfaction, i.e., transmission of a certain predefined Quota of *q* slots, the tail node sends a GNT signal upstream and transits back to the non-restricted free access (FA) state. Upon receiving this GNT, the node upstream follows similar rules: If it is in the body node, it transits to a tail (T) state and will similarly forward GNT upon satisfaction. If it is in the Head State, the local fairness cycle on this segment of the ring is terminated.

In this scenario, the algorithm has created a REQUEST PATH, which contains unique and distinct head and tail nodes. Each node of the REQUEST PATH is able to transmit one Quota including the tail initiator, as shown in Figure 7.

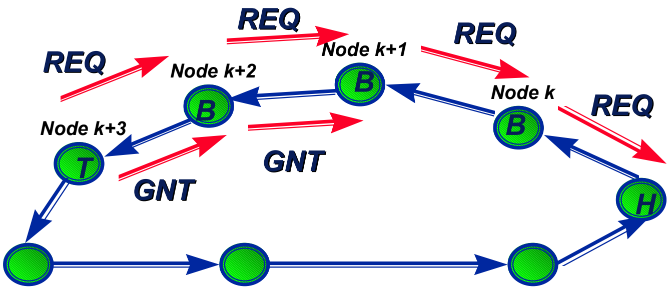


Figure 7: Variable-cycle REQUEST PATH (from Node k+3 to Node k+1)

Properties:

1. The size of the fairness cycle, which is the size of the REQUEST PATH, varies between 1 and N.
2. In every cycle each node in the REQUEST PATH can transmit one Quota.
3. However, nodes on different REQUEST PATHs can transmit different numbers of Quotas.

3.3 One-cycle Fairness Algorithm

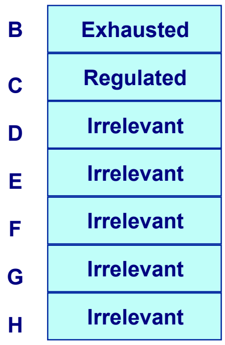
In the One-cycle algorithm, an instance of the algorithm is executed in each direction of each link [May96]. The transmission Quota, *q*, is defined as the number of time slots a node can use in each fairness cycle. Every node *i* maintains a table with an entry for every other node. Each entry *j* reflects the status of node *i* with respect to node *j*, if node *j* is the source of a conflicting downstream session. Hence, a source which sends packets through node *i* can be uniquely labeled "upstream" with respect to node *i*, thus "upstream" and "downstream" are always well defined.

The table is denoted "mode" and each entry mode(*j*) has a value in the set:

Unregulated - implies that node *i* can transmit through node *j* "freely".

Regulated - implies that node *i* can only transmit one more Quota before it becomes exhausted with respect to node *j*.

Exhausted - implies that node *i* can no longer send any packets through node *j*. Consequently, if a node wants to transmit a new Quota, it has to check that all entries of conflicting downstream nodes are not exhausted.



B	Exhausted
C	Regulated
D	Irrelevant
E	Irrelevant
F	Irrelevant
G	Irrelevant
H	Irrelevant

Figure 8: Table "Mode" for Node A in Figure 10

For the example shown in Figure 10, a possible value of the table of node A is shown in Figure 8. If node A wants to transmit a new Quota, it has to check the entries of nodes B and C, since these are the conflicting downstream sessions. As it turns out the entry of node B is exhausted and hence node A cannot transmit right away.

The table mode is updated by the following two control-signals:

- R(*j*) - which indicates that node *j* wants to start transmitting another Quota, and
- U(*j*) - which indicates that node *j* just finished transmitting a Quota.

If an active node *i* receives an R-signal from a conflicting node *j* (session) downstream, *i* knows that a new conflict has arisen with *j* and hence sets mode(*j*) to regulated. Symmetrically, if *i* receives a U-signal from node *j*, *i* knows that the current conflict has ended and it sets mode(*j*) to unregulated. If node *i* is conflicting with downstream node *j* and mode(*j*) is set to regulated, then after transmitting one Quota, mode(*j*) is set to exhausted. These simple state transitions are shown in Figure 9.

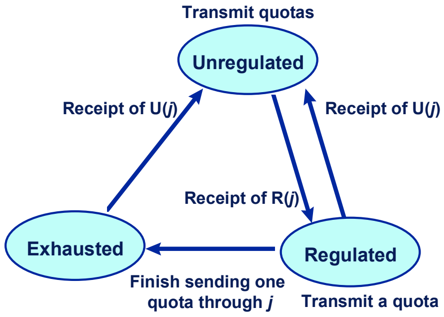


Figure 9: State diagram for Mode(j) at Node i

When node *i* tries to transmit a new Quota, it checks its table and if it is positive (i.e., all conflicting downstream nodes are not exhausted), then node *i* sends an R(*i*)-signal upstream to indicate to conflicting upstream nodes that a new cycle is about to start. Subsequently, node *i* transmits *q* packets in *q* empty slots. After finishing one Quota (or after ceasing to be active), node *i* sends an U(*i*)-signal upstream to indicate to upstream nodes that the current cycle of conflict has ended. It then updates its table according to the state-transition diagram in Figure 9 and the signals it received from downstream nodes.

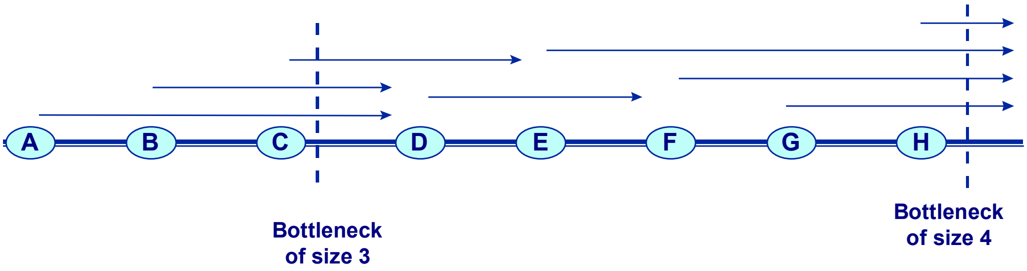


Figure 10: Multiple sessions on a bus segment

4 Simulation Environment

4.1 Selection of the Dynamic Scenarios

A complete analysis of the above three fairness algorithms would require taking into consideration all the possible dynamic scenarios related to a given network configuration. Since this approach is impractical due to the very large number of (dynamic) scenarios to consider, in this paper we focus on some specific scenarios of special interest which have been selected according to some general principles which are discussed below.

We distinguish dynamic scenarios on the basis of the workload distribution along the ring. To give a measure of how much a ring link is loaded we consider the number of sessions using that link. However, since in a dynamic scenario different sessions can be characterized by different session probabilities, i.e., different amounts of traffic, we also consider the total amount of traffic on the link. Based on these measures, dynamic scenarios can be classified as follows:

1. **EN-EA**. Scenarios characterized by the same number of sessions per link and the same amount of traffic (sum of session probabilities) per link. Throughout, this class of scenarios will be referred to as the EN-EA (Equal Number of sessions, Equal Amount of traffic) class.
2. **EN-DA**. Scenarios having an equal number of sessions but a different amount of traffic per link. Throughout, this class of scenarios will be referred to as the EN-DA (Equal Number of sessions, Different Amount of traffic) class.
3. **DN-DA**. Scenarios where both the number of sessions and the amount of traffic vary from link to link. Throughout, this class of scenarios will be referred to as the DN-DA (Different Number of sessions, Different Amount of traffic) class.

In the following sections we will analyze (by simulations) the performance of the fairness algorithms introduced in Section 3 by considering only one representative scenario for each of the above classes. In addition, we will discuss the possible impact of different choices on the algorithms' performance.

4.2 Performance Metrics

In this section we define some indices on which the performance comparison reported in the next section will be based. Throughout we will refer to γ_i as the throughput achieved by node *i* normalized to the ring capacity. Specifically, γ_i is defined as:

$$\gamma_i = \frac{N_i(T)}{T} t_s$$

where $N_i(T)$ indicates the number of packets transmitted by node *i* in a time interval of duration equal to *T* time slots and t_s is the time needed to transmit a packet (i.e., the slot duration). Hence, t_s is the inverse of the link capacity expressed in slots/second.

For a given fairness algorithm, if *N* is the number of nodes in the ring, the *Fairness Deviation* δ_F is defined as:

$$\delta_F = \sqrt{\sum_{i=1}^{N} \left(\frac{\gamma_i^{fa} - \gamma_i^{o}}{\gamma_i^{o}} \right)^2} \qquad [6]$$

In [6], γ_i^{fa} is the throughput achieved by node *i* when a given fairness algorithm is implemented while γ_i^{o} is the Max-Min rate of node i calculated according to the Dynamic Max-Min algorithm specified in Section 2.3. For a given fairness algorithm, the Fairness Deviation provides a measure of the algorithm's effectiveness, i.e., how much the node throughputs provided by the algorithm deviate from the Max-Min fair rates.

However, another important issue to take into consideration is the algorithm's efficiency, i.e., how close the aggregate (normalized) throughput provided by the algorithm is to the optimal aggregate (normalized) throughput resulting from the Max-Min fair rates. To this end the *Throughput Deviation* will be used. For a given fairness algorithm, if *N* is the number of nodes in the ring, the Throughput Deviation δ_T is defined as:

$$\delta_T = \frac{\sum_{i=1}^{N} \gamma_i^{fa} - \sum_{i=1}^{N} \gamma_i^{o}}{\sum_{i=1}^{N} \gamma_i^{o}} \qquad [7]$$

As can be seen, δ_T is the difference between the aggregate (normalized) throughput provided by a specific fairness algorithm and the optimal aggregate (normalized) throughput given by the sum of the Max-Min rates, and normalized to the optimal aggregate (normalized) throughput. By definition, δ_T is positive if the fairness algorithm provides, for a specific traffic scenario, an aggregate throughput greater than the optimal aggregate throughput, and negative otherwise.

We will refer to an ideal fairness algorithm as the algorithm for which both Fairness and Throughput Deviations are equal to zero. By definition, $\delta_F = 0 \Rightarrow \delta_T = 0$. On the other hand, the reverse implication does not generally hold.

4.3 Simulation Parameters

Table 2 reports the common set of parameter values used in the simulation experiments. In all the experiments we assumed that each node has always a message ready for transmission. The destination of each message is selected randomly by using session probabilities in the Flow Matrix characterizing the dynamic scenario under consideration[1]. The (fixed) message size was set to 10 Kbytes. However, in Section 5.5 in order to investigate the influence of the message size on the algorithm's performance we consider additional sizes. Similarly, in Section 5.6, in order to analyze the influence of the ring size we also consider the distance between nodes equal to 5 and 10 time slots.

Parameter	Value
Number of Nodes (*N*)	12
Distance between nodes (*d*)	1 slot
Network Packet Size (packet)	48 bytes
Average Message Size (*msg*)	10 Kbytes = 210 packets
Message Size Distribution	Constant

Table 2: Common set of simulation parameters.

5 Results

The analysis[2] can be divided into two parts. In the first part (Sections 5.1 - 5.3) we investigate the influence of the (dynamic) traffic scenario on the performance of the fairness algorithms. To this end, in Sections 5.1 - 5.3 we consider three traffic scenarios, one for each class defined in Section 4.1, and compare the performance of the three algorithms in these scenarios. In the second part (Sections 5.4 - 5.6) we focus on only one of these scenarios, the most realistic one, and analyze the sensitivity of the algorithms to the number of network nodes, the message size and the distance between nodes, respectively.

5.1 Scenario A (EN-EA)

We begin our analysis by considering a traffic scenario belonging to the EN-EA class. Among the set of scenarios belonging to this class we selected the conceptually simplest one. In this scenario, hereafter Scenario A, each node sends messages to all its possible destinations with the same probability. The resulting *Flow Matrix* is shown in Figure 11, while Table 3 reports the number of sessions and the amount of traffic

[1] Recall that the sum of session probabilities for each (active) node is equal to 1.

[2] The results presented in this section have been estimated by using the independent replication method and assuming a confidence level of 90% [Law82]. The duration of each simulation experiment was fixed in such a way as to achieve confidence intervals less than 5 %.

(i.e., the sum of session probabilities) on each link. Obviously, the latter quantities are constant since Scenario A belongs to the EN-EA class.

DESTINATION

		1	2	3	4	5	6	7	8	9	10	11	12
	1		.1666	.1666	.1666	.1666	.1666	.1666					
	2			.1666	.1666	.1666	.1666	.1666	.1666				
S	**3**				.1666	.1666	.1666	.1666	.1666	.1666			
O	**4**					.1666	.1666	.1666	.1666	.1666	.1666		
U	**5**						.1666	.1666	.1666	.1666	.1666	.1666	
R	**6**							.1666	.1666	.1666	.1666	.1666	.1666
C	**7**	.1666							.1666	.1666	.1666	.1666	.1666
E	**8**	.1666	.1666							.1666	.1666	.1666	.1666
	9	.1666	.1666	.1666							.1666	.1666	.1666
	10	.1666	.1666	.1666	.1666							.1666	.1666
	11	.1666	.1666	.1666	.1666	.1666							.1666
	12	.1666	.1666	.1666	.1666	.1666	.1666						

Figure 11: Flow Matrix for Scenario A

Link	Number of Sessions	Amount of Traffic
1-2	21	3.5
2-3	21	3.5
3-4	21	3.5
4-5	21	3.5
5-6	21	3.5
6-7	21	3.5
7-8	21	3.5
8-9	21	3.5
9-10	21	3.5
10-11	21	3.5
11-12	21	3.5
12-1	21	3.5

Table 3: Number of sessions and sum of session probabilities per link for Scenario A.

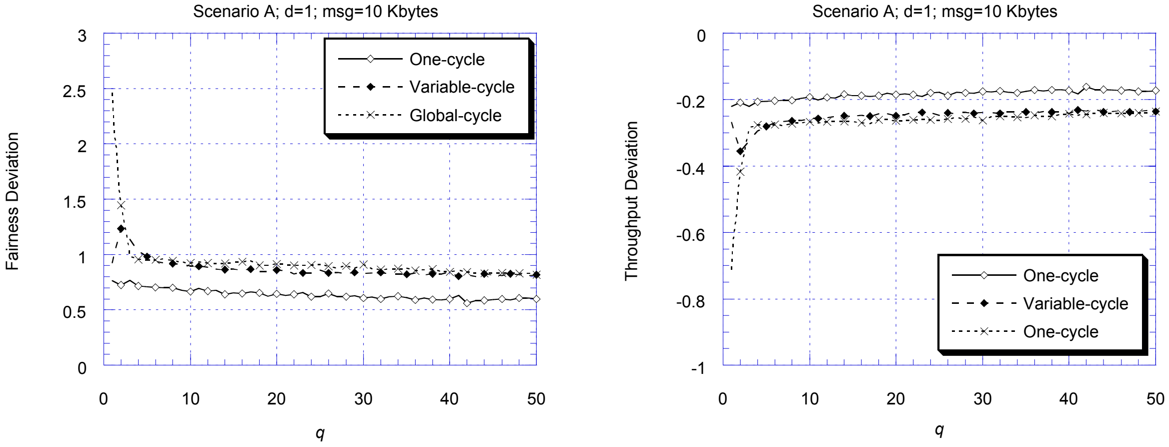


Figure 12: Fairness and Throughput Deviation in Scenario A.

The Fairness and Throughput Deviations exhibited by the three algorithms in Scenario A are reported in Figure 12. We can observe that, at least for Quota values not too small, all the algorithms perform similarly both in terms of Fairness and Throughput Deviation. However, a more detailed observation reveals that the

One-cycle algorithm performs slightly better than the others, while the Variable-cycle and Global-cycle algorithms do not show meaningful differences. For example, the absolute values of the Throughput Deviation (i.e., the relative deviation of the aggregate throughput from the optimal aggregate throughput) are, 25-30% for the Global-cycle and Variable-cycle, and 15-20% for the One-cycle.

In the range of Quota values [1, 3] the Global-cycle algorithm performs very poorly and the curves of Fairness and Throughput Deviations are linear. This phenomenon (already observed in static conditions [Ana99]) is due to the value of the Quota which is small with respect to the ring size. As a consequence, it may occur that nodes observe empty slots but cannot use them since their Quota has been exhausted. In fact, as shown in Table 4, the average SAT rotation time (i.e., the average time between two consecutive SAT releases by the same node) is exactly equal to the ring size (12) when the value of the Quota is less than or equal to 2. This means that all nodes are already satisfied when they receive the SAT and, in accordance with the Global-cycle algorithm, they immediately release it. When the Quota is equal to 3, nodes are satisfied most of the time but not always. This justifies an average SAT rotation time slightly greater than the ring size. Finally, the probability that a node receiving the SAT is already satisfied becomes lower and lower as the Quota increases.

Value of Quota	Average SAT Rotation Time (slots)	Confidence Interval (slots)
1	12.00	± 0.00
2	12.00	± 0.00
3	14.49	± 0.14
4	18.96	± 0.18
5	23.70	± 0.34

Table 4: Average SAT rotation time for several values of Quota in Scenario A

For values of the Quota which are not too small, the similarity in performance can be explained by observing that, due to the presence of sessions with a large source-destination distance, all the algorithms behave as global algorithms. In particular, the Request Path of the Variable-cycle algorithm spans the entire ring and this is why the Variable-cycle algorithm performs exactly like the Global-cycle algorithm. However, the One-cycle algorithm, thanks to its cycle size equal to 1, takes advantage of the presence of sessions with short source-destination distances and this is the reason for the slight difference in favor of this algorithm.

The results presented above are clearly dependent on the particular scenario taken into consideration. To investigate this dependency, we considered other EN-EA scenarios. Although the results of the related experiments change with the scenario, they nevertheless allow us to draw the conclusion that, as expected, in scenarios belonging to the EN-EA class the three algorithms perform in a similar way.

5.2 Scenario B (EN-DA)

Scenario A is characterized by a uniform distribution of the workload along the ring. To "break" this uniformity we modified the traffic pattern by considering different probability values for the different destinations of each node. Specifically, we assumed that the session probabilities of nodes from 1 to 6 *increase* linearly as the source-destination distance increases, while the session probability values of nodes from 7 to 12 *decrease* linearly as the same distance increases. In other words, nodes from 1 to 6 transmit above all to distant destinations, while nodes from 7 to 12 transmit above all to close destinations. The session probability distributions for Scenario A and for the new scenario, hereafter Scenario B, are shown in Figure 13, while Figure 14 reports the Flow Matrix of Scenario B. As shown in Table 5, in this scenario the number of sessions per link is constant (as in Scenario A) but the amount of traffic varies from link to link.

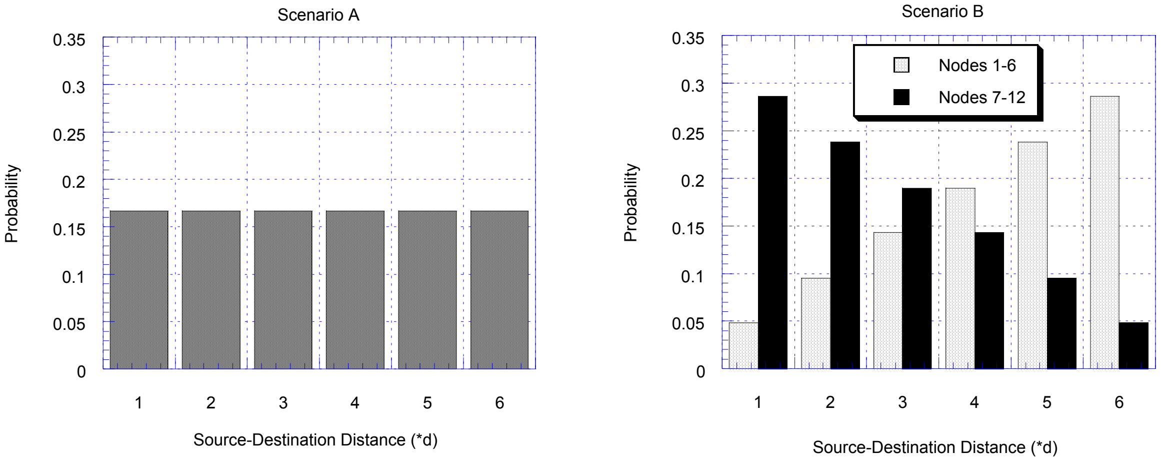


Figure 13: Destination probability values for scenarios A (left) and B (right)

DESTINATION

SOURCE	1	2	3	4	5	6	7	8	9	10	11	12
1		.048	.095	.143	.190	.238	.286					
2			.048	.095	.143	.190	.238	.286				
3				.048	.095	.143	.190	.238	.286			
4					.048	.095	.143	.190	.238	.286		
5						.048	.095	.143	.190	.238	.286	
6							.048	.095	.143	.190	.238	.286
7	.048							.286	.238	.190	.143	.095
8	.095	.048							.286	.238	.190	.143
9	.143	.095	.048							.286	.238	.190
10	.190	.143	.095	.048							.286	.238
11	.238	.190	.143	.095	.048							.286
12	.286	.238	.190	.143	.095	.048						

Figure 14: Flow Matrix for scenario B.

The results obtained for the Scenario B are shown in Figure 15. As in Scenario A, the One-cycle algorithm exhibits the best performance but now the curves of the Global-cycle and the Variable-cycle algorithms are clearly separated to the advantage of the Variable-cycle algorithm. Furthermore, the distance between the corresponding curves of the One-cycle and the Global-cycle algorithms is greater than in Scenario A. In fact, the absolute values of the Throughput Deviation are approximately 35% for the Global-cycle, 30% for Variable-cycle, and 20% for the One-cycle algorithm.

Link	Number of Sessions	Amount of Traffic
1-2	21	2.671
2-3	21	2.912
3-4	21	3.288
4-5	21	3.714
5-6	21	4.095
6-7	21	4.333
7-8	21	4.333
8-9	21	4.095
9-10	21	3.714
10-11	21	3.288
11-12	21	2.912
12-1	21	2.672

Table 5: Number of sessions and amount of traffic per link in Scenario B.

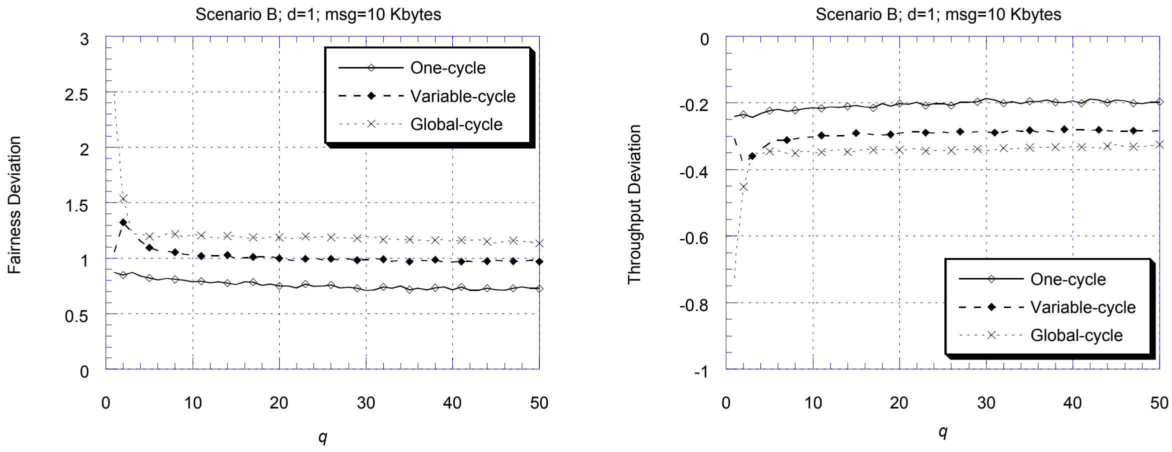


Figure 15: Fairness and Throughput Deviation in Scenario B.

This difference in performance is obviously due to the imbalance in the distribution of the workload between links (see Table 5) and can easily be explained. In fact, when using the Global-cycle algorithm the throughput achieved by each node is determined by the most congested link and is the same for all the nodes (the fairness cycle size is equal to *N*). Hence, in scenarios like Scenario B some nodes are penalized by the workload imbalance. On the other hand, the fairness cycle size is 1 in the One-cycle and generally less than *N* in the Variable-cycle and, hence, the throughput of a node is only determined by the number of nodes directly or indirectly interfering with it. Thanks to this property the non-global algorithms can manage unbalanced traffic scenarios more efficiently than the Global-cycle algorithm.

When considering EN-DA scenarios different from Scenario B, the results may be generally different. However, we performed additional simulation experiments by considering probability shapes different from those shown in Figure 13 (right) and we found that, as expected, the difference between the performance curves of the algorithms increases as the imbalance in the workload distribution grows.

5.3 Scenario C (DN-DA)

Scenario B is slightly more realistic than Scenario A since it is characterized by a non-uniform distribution of the workload between links. It highlights that the workload imbalance along the ring can significantly affect the algorithms' performance. However, even Scenario B is far from reality. Since SAN nodes are not peer devices it is not realistic to think that each node transmits to all the other nodes. Real traffic scenarios are normally characterized by the following features

- destination nodes are addressed with different probabilities;
- nodes usually have few active sessions at a given time;
- the number of active sessions generally varies from node to node;
- some nodes are more frequently addressed than others;
- both the number of sessions and the amount of traffic over links are different from link to link;
- etc.

DESTINATION

SOURCE

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1		0.5		0.5								
2				1.0								
3						1.0						
4					0.5	0.5						
5						0.5		0.5				
6							0.5	0.5				
7								0.5		0.5		
8										0.1		0.9
9										0.1		0.9
10		0.1										0.9
11		0.1										0.9
12		1.0										

Figure 16: Destination Probability Matrix for scenario C.

Of the above aspects, only the first one is exhibited by Scenario B (Scenario A does not even exhibit that one). Therefore, to make the analysis more realistic, in this section we consider a traffic scenario, hereafter

Scenario C, which like the previous ones is artificial, but includes all the above characteristics. In particular, we assumed that node 12 is addressed more frequently than the other ones: nodes having node 12 as their destination address node 12 itself with a probability of 0.9 and the alternate destination with probability 0.1. The Flow Matrix for Scenario C is reported in Figure 16. This scenario clearly belongs to the class of DN-DA scenarios as highlighted by Table 6.

Link	Number of Sessions	Amount of Traffic
1-2	5	2.2
2-3	2	1.5
3-4	3	2.5
4-5	3	2
5-6	4	2.5
6-7	3	1.5
7-8	4	2
8-9	3	1.5
9-10	5	2.5
10-11	4	2.8
11-12	6	3.8
12-1	3	1.2

Table 6: Number of sessions and amount of traffic per link in Scenario C.

Figure 17 shows the Fairness and Throughput Deviation measured in Scenario C. The trend observed with EN-DA scenarios is exhibited by the present scenario as well: the One-cycle algorithm has the best performance, the Global-cycle is in the worst position and the Variable-cycle is in between. Furthermore, in Scenario C the distance between the performance curves of the One-cycle and the Global-cycle algorithms, respectively, is accentuated with respect to Scenario B, and the One-cycle algorithm now performs close to the optimal level. The Throughput Deviations are approximately 40% for the Global-cycle, 30-35% for the Variable-cycle and only 10-15% for the One-cycle algorithm.

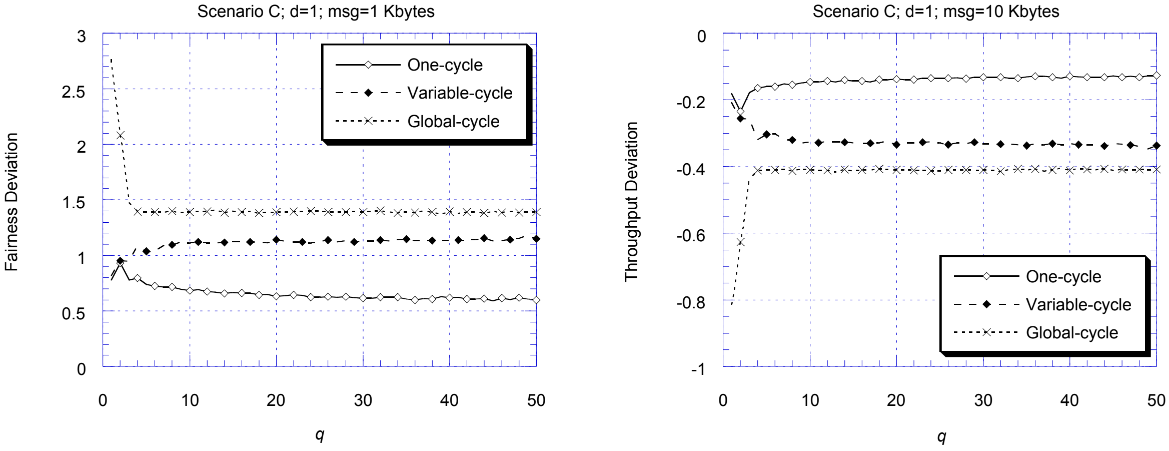


Figure 17: Fairness and Throughput Deviations in Scenario C.

When analyzing Scenario B, the difference in performance was explained in terms of the non-homogeneous distribution of the workload between links. The same reasoning applies to Scenario C.

The results obtained with Scenarios A, B and C allow us to make some remarks. First, in dynamic conditions the performance of the algorithms is significantly influenced by the traffic scenario. In particular, the workload distribution along the ring plays an important role in this regard. Furthermore, based on the above results, the difference in performance increases in favor of the non-global algorithms and, specifically, in favor of the One-cycle algorithm, as soon as the traffic pattern tends to become realistic.

5.4 Influence of the number of nodes

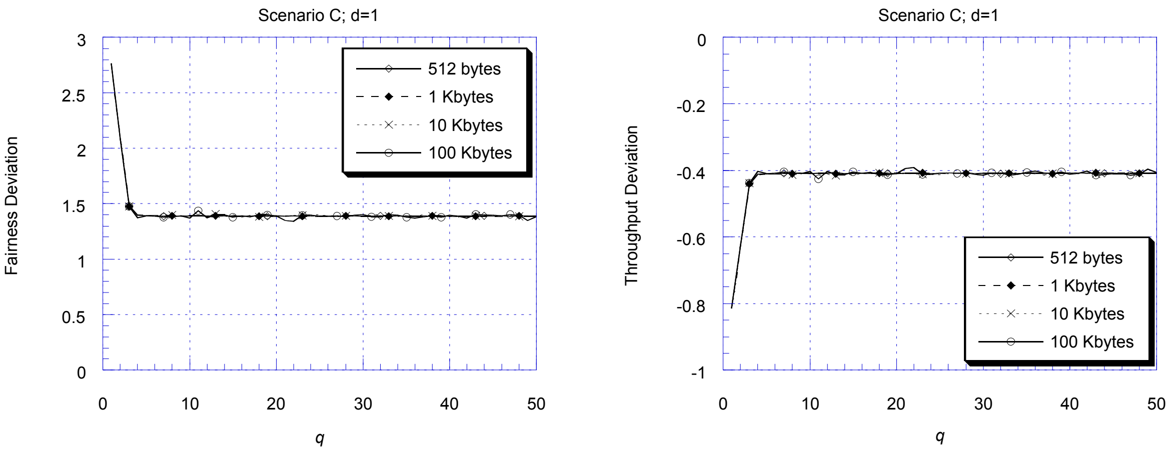
The difference in performance observed in Scenario C is compressed by the limited number of nodes taken into consideration. In fact, when the number of nodes increases it is more likely that some links are very loaded while other links are slightly loaded or even idle and, hence, the non-homogeneity in the workload distribution should increase. As a consequence, we can expect that the difference in performance becomes larger as the number of nodes increases.

This means that the Variable-cycle and the One-cycle algorithms should scale well, in performance, with the number of nodes, while the same property is not expected to hold for the Global-cycle algorithm. However, for completeness, recall that the number of nodes also affects the algorithm's complexity. In particular, in Section 3 (see Table 1) it was observed that the complexity is O(1) for the Global-cycle, O(log N) for the Variable-cycle, and O(N) for the One-cycle.

5.5 Influence of the message size

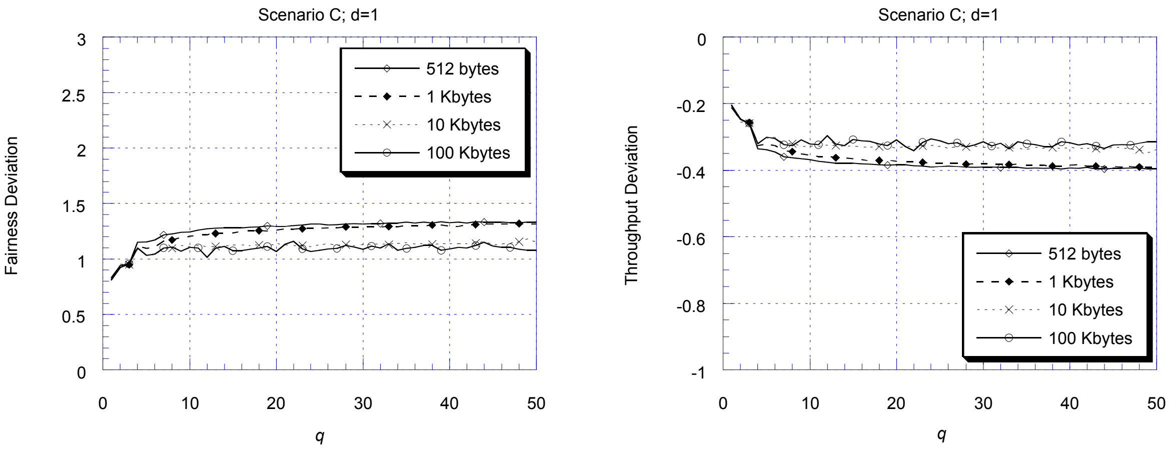
This section is aimed at investigating the sensitivity of the algorithms to the message size. To this end, we focused on Scenario C (since it is the most realistic one) and considered a set of message sizes ranging from 512 bytes (the size of disk blocks) up to 100 Kbytes. Large message sizes arise in transfers of large blocks of data, e.g., large file transfers, images transfers, etc.

Figure 18 shows that the Global-cycle algorithm is insensitive to the message size for the set of message sizes taken into consideration. Figure 19 and Figure 20 report the Fairness and Throughput Deviations for the Variable-cycle and the One-cycle algorithms, respectively. Although the curves do not overlap completely as in the Global-cycle algorithm, the two algorithms do not show a significant sensitivity to the message size, at least in the range of values of the Quota explored and in terms of the message sizes taken into consideration.


Scenario C; d=1

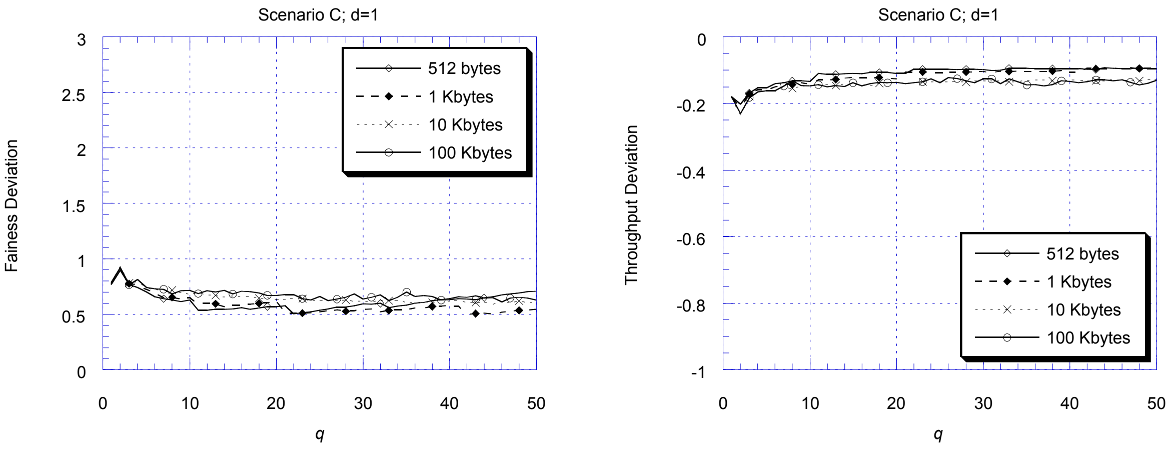
Scenario C; d=1

Figure 18: Influence of message size in the performance of the Global-cycle algorithm


Scenario C; d=1

Scenario C; d=1

Figure 19: Influence of message size in the performance of the Variable-cycle algorithm.


Scenario C; d=1

Scenario C; d=1

Figure 20: Influence of message size in the performance of the One-cycle algorithm.

Careful observation highlights a slight performance degradation in the Variable-cycle algorithm when the message size decreases. However, even for message sizes of 512 bytes the performance figures of the Variable-cycle algorithm are never worse than the corresponding performance figures of the Global-cycle algorithm.

Obviously, in reality messages are not of fixed size and, furthermore, messages generated by different nodes are generally characterized by different size distributions. Since traffic in a SAN is expected to consist above all in large messages, based on the previous results we can reasonably expect that in a real environment the One-cycle algorithm would perform much better than the Global-cycle algorithm, and that the Variable-cycle algorithm would be, as usual, in an intermediate position.

5.6 Influence of the ring size

This section is devoted to the analysis of sensitivity to the ring size. We assume that the distance, *d*, between consecutive nodes is constant, and we consider three different values for *d*: 1, 5 and 10 slots. Since there are 12 nodes in the ring, this implies considering rings of size 12, 60 and 120 slots, respectively. Fairness and Throughput Deviations for the case *d*=1 have been already shown in Figure 17 and are not reported here. On the other hand, Fairness and Throughput Deviations for *d*=5 and *d*=10 are shown in Figure 21 and Figure 22, respectively.

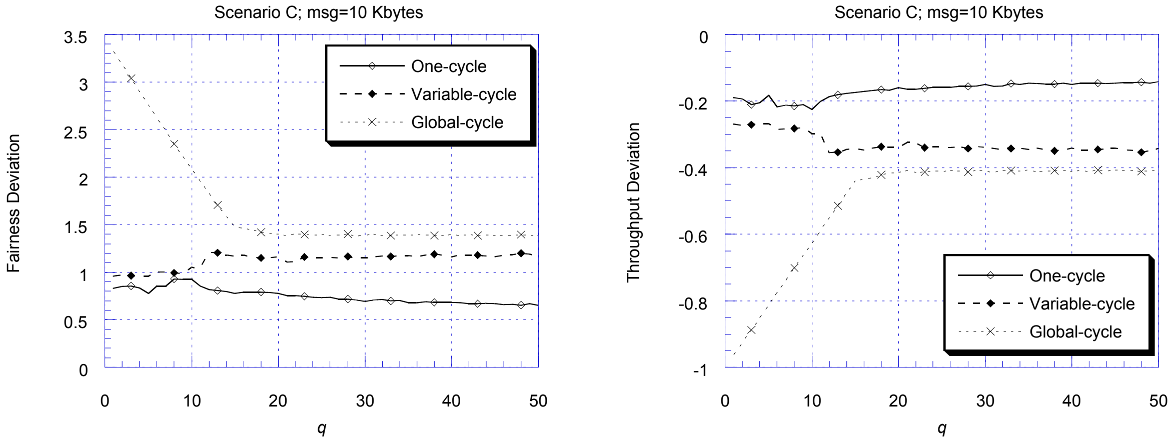


Figure 21: Fairness and Throughput Deviations for *d*=5.

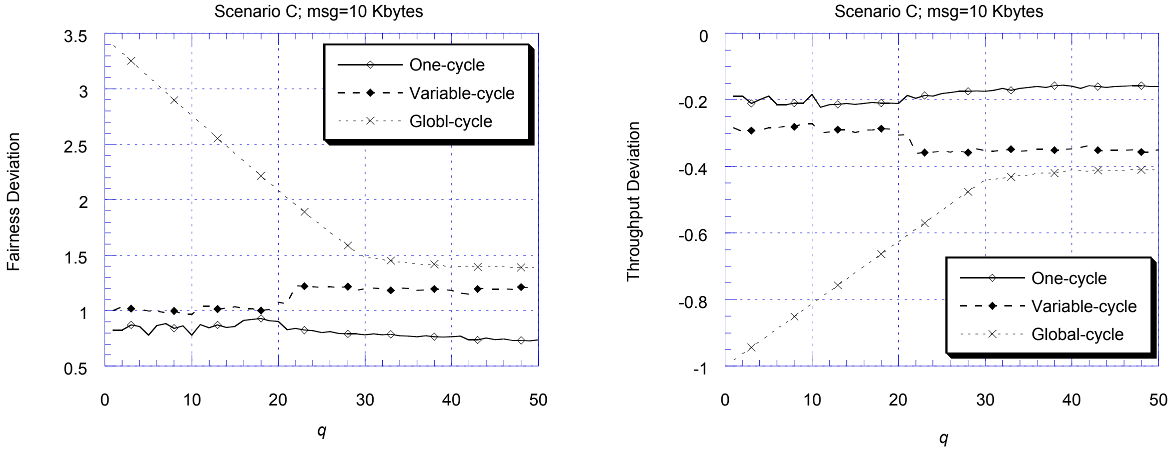


Figure 22: Fairness and Throughput Deviations for *d*=10.

As highlighted by the corresponding graphs in Figure 17, Figure 21 and Figure 22 each fairness algorithm exhibits the same behavior for the different ring sizes considered. This allows us to make some general remarks. Firstly, all the algorithms reach a level of performance that can be assumed as stationary for values of the Quota that are larger than a given threshold which varies from algorithm to algorithm. Specifically, for the Global-cycle algorithm the Fairness and Throughput Deviations remain approximately constant for values of the Quota greater than *3d.* For smaller values both the Fairness Deviation and (the absolute value of the) Throughput Deviation decrease linearly as the Quota decreases. The reasons for this linear behavior were outlined in Section 5.1.

From Figure 17, Figure 21 and Figure 22 it can be argued that the performance of a specific algorithm does not depend on the ring size (i.e., the distance between nodes) provided that the value of the Quota remains fixed. This is better clarified by Figure 23 where the Fairness and Throughput Deviations are reported as a function of the ring size for values of the Quota equal to *4d.* Similar shapes are achieved by plotting the Fairness (Throughput) Deviation vs. the ring size for different fixed values of the Quota, e.g., *q=d, 2d, 3d, 4d* and so on.

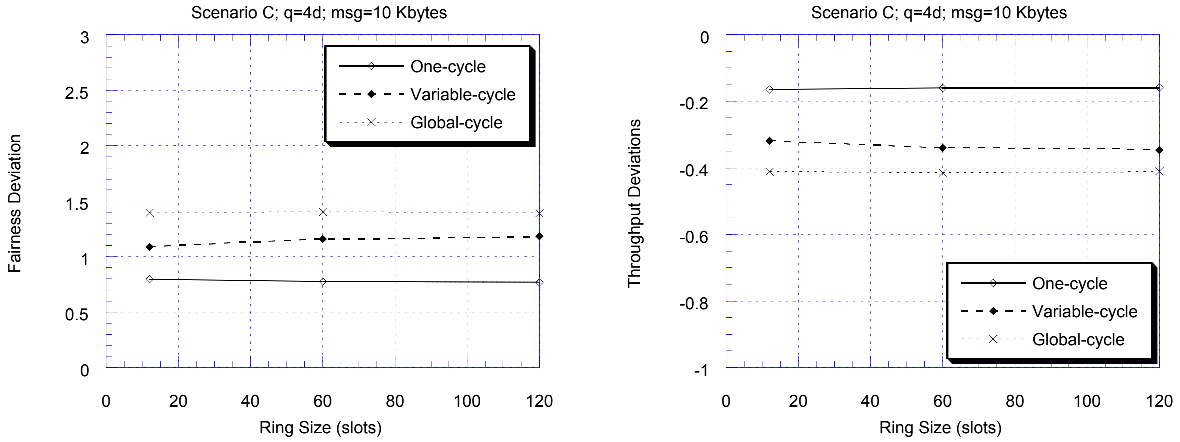


Figure 23: Throughput and Fairness Deviations for different ring sizes and for *q=4d.*

6 Conclusions

In this paper we have addressed the problem of studying fairness in ring networks with spatial bandwidth reuse operating in dynamic traffic scenarios. A major contribution of this paper is the generalization of the Max-Min fairness definition to dynamic traffic scenarios. A procedure for computing Max-Min fair rates in dynamic conditions has also been introduced. This procedure generalizes the one proposed in [Ber87] for static scenarios.

In the second part of the paper the above procedure has been used to compare the performance of three fairness algorithms proposed for ring-based SANs like the SSA (Serial Storage Architecture) and characterized by different fairness cycle sizes. From the comparison it has emerged that the One-cycle algorithm performs better than the Global-cycle and the Variable cycle algorithms is in an intermediate position. Furthermore, the disparity in performance becomes more significant as the scenario tend to become realistic.

The key question is which fairness algorithm is the best for practical implementation. This is not a simple question to answer since the apparently higher performance algorithm, One-cycle, is significantly more complex than the Variable-cycle and Global-cycle algorithms; and the Variable-cycle algorithm is more complex than the Global-cycle algorithm, which has a low complexity but also the worst performance. The Global-cycle algorithm has the advantage that it has already been implemented in several systems and is currently being used in various SSA products, which are implemented by IBM following the ANSI Standard X3T10. However, based on the analysis in static conditions reported in [Ana99] and on the results outlined in the present paper, we believe that the Variable-cycle algorithm is the best tradeoff when optimizing under performance and complexity measures.

References

[Ana99] G. Anastasi, L. Lenzini, Y. Ofek. Tradeoff between the Cycle Complexity and the Fairness of Ring Networks, *Submitted for publication.*

[Ber87] D. Bertsekas, R. Gallager. Data Networks. Prentice Hall, 1987.

[Cid93] I. Cidon, Y. Ofek. MetaRing, a Full Duplex Ring with Fairness and Spatial Reuse. *IEEE Transaction on Communications*, COM-41(1):110--120, January 1993 (also IEEE INFOCOM, 1990).

[Che93] J. Chen, I. Cidon, Y. Ofek. A local fairness algorithm for gigabit LANs/MANs with spatial reuse. *IEEE J. on Selected Areas in Comm.*, 11(8):1183--1192, October 1993.

[Du98] D. H. C. Du, J. Hsieh, T. S. Chang, Y. Wang, and S. Shim. Interface Comparisons: SSA versus FC-AL. *IEEE Concurrency*, Vol.6, No.2, April-June 1998.

[Hay81] H. Hayden. Voice flow control in integrated packet networks. MIT Laboratory for Information and Decision Systems - LIDS Report TH-1152, 1981.

[Jaf81] J. M. Jaffe. Bottleneck flow control. *IEEE Transactions on Communications*, COM-29(7):954-962, July 1981.

[Law82] A. M. Law, W. D. Kelton. "Simulation Modeling and Analysis". McGraw-Hill Book Company, 1982.

[May96] A. Mayer, Y. Ofek, M. Yung. Approximating Max-Min Fair Rates via Distributed Local Scheduling with Partial Information. *IEEE INFOCOM*, 1996.

[Phi98] B. Phillips. "Have Storage Area Networks Come of Age?". *IEEE Computer*, Vol.31, No. 7, July 1998, pp. 10-12.

[Ofe94] Y. Ofek. Overview of the MetaRing Architecture. *Computer Networks and ISDN Systems*, Vol. 26, Nos. 6-8, March 1994, pp. 817-830.

Appendix: Max-Min rates computation in a dynamic scenario

In this appendix the Dynamic Max-Min algorithm is used to compute the Max-Min fair rates of sessions in the dynamic scenario depicted in Figure 2 and whose Flow Matrix is reported in Figure 3.

session	source node	destination node	session probability
1	1	2	1/2
2	1	4	1/2
3	2	3	1/2
4	2	4	1/2
5	3	4	1

Table 7: Sessions e related session probabilities

Assuming that sessions are numbered as shown in Table 7 and that each link is identified by means of the identifier of its upstream node (i.e., $\mathcal{S}$={1,2,3,4,5} and $\mathcal{L}$={1,2,3}) the initial conditions are as follows

L= $\mathcal{L}$={1,2,3};

S= $\mathcal{S}$={1,2,3,4,5};

C_1=1.0; C_2=1.0; C_3=1.0;

r_1=0.0; r_2=0.0; r_3=0.0; r_4=0.0; r_5=0.0;

T_1=0.0; T_2=0.0; T_3=0.0;

At the first step the algorithm saturates link 3 for which the ratio between between the residual capacity ($C_3 - T_3 = 1.0$) and the sum of session probabilities (Σ_3=2) is minimum and equal to 1/2. Session rates are thus increased by an amount proportional to their weight (session probability) and assume the following values

r_1=1/4; **r_2=1/4;** r_3=1/4; **r_4=1/4;** **r_5=1/2;**

Link 3 is now saturated (T_3=1.0, see below) and the rates of sessions 2, 4 and 5 which use this link are fixed at the current value. At this point is

L= $\mathcal{L}$={1,2};

S= $\mathcal{S}$={1,3};

C_1=1.0; C_2=1.0; C_3=1.0;

T_1=1/2; T_2=3/4; **T_3=1.0**

In the second step link 2 is the link to be saturated. In fact, $\frac{C_1 - T_1}{\Sigma_1} = \frac{1 - 1/2}{1/2} = 1$ while $\frac{C_2 - T_2}{\Sigma_2} = \frac{1 - 3/4}{1/2} = 1/2$. Since both session 1 and session 3 have the same weight (1/2) their rates are increased by an equal amount (1/4) and the vector rate assumes the following value

r_1=1/2; **r_2=1/4;** r_3=**1/2**; **r_4=1/4;** **r_5=1/2;**

Since session 3 shares link 2 which is now saturated its rate is fixed at the current value. At this point is also

L= $\mathscr{L}$={1};

S= $\mathscr{S}$={1};

C_1=1.0; C_2=1.0; C_3=1.0;

T_1=3/4; ***T_2*=1.0;** ***T_3*=1.0**

The only link not yet saturated is link 1 which is thus saturated in the third step. The rate of session 1, the only session not passing through any saturated link, is increased by an amount 1/4 and, hence, the final rate vector is the following

r_1=3/4; **r_2=1/4;** r_3=**1/2**; **r_4=1/4;** **r_5=1/2;**