A Fair and Fast Protocol for DQDB Metropolitan Area Networks

Maria C. Yuang and Hao-Tsan Chang

Department of Computer Science and Information Engineering
and Center for Telecommunications Research
National Chiao Tung University, Taiwan, R.O.C.

Abstract

Distributed Queue Dual Bus (DQDB) was shown to exhibit throughput unfairness problems in which bandwidth allocation is dependent on the physical locations of nodes, as the size and load of the network increase. To eliminate the unfairness problems, various mechanisms have been proposed. This paper first analyzes the unfairness problem and presents a survey of these mechanisms, including an analysis of their strengths and weaknesses. This paper then proposes a fair and fast protocol, called Multiple Segment Control (MSC). MSC grants a node multiple slots on continuation basis for any single request issued in an attempt to minimize the propagation delay. The node keeps being granted until it has issued an "end" request. Consequently, MSC offers fair and high throughput to all nodes for both light and heavy load conditions. Finally, the performance of MSC is evaluated and compared with that of DQDB via simulation results.

1. Introduction

The main goals of MANs [2,7,8,9,10] are to provide high speed transmissions and the integration of various services, such as data, voice, and images, over larger geographic areas. Distributed Queue Dual Bus (DQDB) [1], accepted as a new IEEE 802.6 standard Medium Access Control (MAC) protocol for MANs, was shown to successfully meet these goals. Unfortunately, it exists the unfairness problems [3] resulting from the long propagation delay under heavy load conditions of the network. To eliminate the problems, various relief mechanisms, such as BWB [12,13], RRC [6], APFC [4], Erasure Node [11], and Slot Reuse [5], have been proposed. These mechanisms were shown to be able to successfully eliminate the unfairness problems, however, with the price of the bandwidth wastage paid.

In this paper, we propose a fair and fast MAC protocol, called Multiple Segment Control (MSC). The MSC mechanism grants a node multiple slots on continuation basis for any single request issued in attempt to minimize the propagation delay. The node keeps being granted until it has issued an "end" request. Consequently, MSC not only assures fair access but also offers high throughput under both heavy and light load conditions of a network.

The paper is organized as follows. Section 2 overviews the DQDB Distributed Queue (DQ) protocol. Its unfairness problems are also described and analyzed in this section. Section 3 surveys four existing relief mechanisms. The MSC mechanism is then proposed in Section 4. Section 5 shows simulation results to demonstrate the superiority of the performance of MSC over DQDB. Finally, Section 6 concludes the paper.

2. DQDB Overview

2.1 Distributed Queueing Algorithm

The network configuration of DQDB (Figure 1) consists of a pair of slotted unidirectional buses, called bus A and bus B. These two buses support full duplex communications in opposite directions between any pair of nodes in the network. The headend node of each bus generates fixed length slots passing downstream. Every node is assumed to be able to select the appropriate bus to transmit or receive data. When the specific bus is selected, say bus A, a reservation for the slot access of bus A is required and made on bus B via a request. Since the operations for both buses are symmetric, we only consider the data transmissions on bus A and the corresponding requests on bus B.

DQDB supports three different types of traffic (synchronous, asynchronous, and isochronous), four priority levels, and two types of slots (Pre Arbitrated (PA) and Queued Arbitrated (QA) slots). The PA and QA slots are used to transfer isochronous and asynchronous segments, respectively. It was shown that the PA access method is free from the unfairness problem. The discussion of the PA method is beyond the scope of the paper. Thus, this paper deals solely with the QA access method. The transmissions of QA segments are based on the Distributed Queueing (DQ) algorithm.
The QA slot consists of a 52-byte segment field and a one-byte Access Control Field (ACF). The ACF includes a Busy (B) bit and four Request (R) bits for four priority levels, respectively. Since the operations for different priority levels are identical, we herein discuss one priority level only. The B bit of a slot passing toward downstream indicates whether or not the slot is busy. The R bit passing toward upstream indicates whether or not a request for an empty slot is carried.

Generally, the DQ algorithm requires each node to maintain an implicit distributed queue through keeping track of two counters at the node:

- RQ (ReQuEst) counter, and
- CD (CountDown) counter.

The RQ counter keeps track of the number of outstanding requests generated downstream before the node attempts to transmit. The CD counter value indicates the position of the node in the distributed queue. By counting the number of requests a node has received and empty slots which has been passed, the node can determine the order of its transmission. For instance, if the number of requests a node has received is 8, and it has passed 5 empty slots to downstream nodes, the node can then access the fourth empty slot. In other words, in this example: RQ = 8, CD = 3, and the node can transmit when CD reaches zero. For any node wishing to transmit, if there is no request queued ahead of it, access is immediate; otherwise deference is given to those requests already queued. The DQ algorithm is detailed in the following. The state transition diagram is shown in Figure 2.

**Distributed Queueing Algorithm**

A node on the bus can be in one of two states: IDLE and COUNTDOWN.

A node is in the IDLE state if it has no segment to transmit. The procedure for this state is described as follows:

1. The RQ counter is incremented by one whenever a slot with R bit set ON is sensed on bus B.
2. The RQ counter is decremented by one whenever an empty slot (B=0) is passed downstream on bus A (provided that RQ value was greater than zero).

A node switches from the IDLE to the COUNTDOWN state when the node has data segment to transmit. The procedure for this state is described as follows:

1. The node issues a request on the reverse bus by setting the first unmarked (OFF) R bit of a slot ON.
2. The current RQ counter value is transferred to the CD counter.
3. The current RQ counter is reset to be zero. Then, the RQ counter keeps track of requests made thereafter.
4. The CD counter value is decremented by one whenever a free slot (B=0) is passed through the node.
5. As soon as the CD counter value reaches zero, the node is allowed to transmit using the next free slot passing by.

Upon finishing the transmissions, the node returns to the IDLE state.

### 2.2 Unfairness Problems

DQDB exhibits unfair behavior when the system is overloaded. There are three types of unfairness problems caused by three major factors. The three types of unfairness problems are: Delay Unfairness Type 1 (DU1), Delay Unfairness Type 2 (DU2), and Location Unfairness (LU). The three major causes are: propagation delay (Dij) for transmitting requests, locations of nodes (Li), and four traffic conditions. Each of them is described in detail in the following. Assume
that, \( N \) is the total number of nodes, \( i, j, k, \) and \( m \) are node indexes (where \( 1 \leq i, j, k, m < N \)), all nodes are equally spaced \( (= d) \), and \( v \) is the propagation speed.

**Three causes of unfairness:**

- **Propagation delay** \( (D_{ij}) \) \( D_{ij} \) is the propagation delay of a request sent from a downstream node \( j \) to an upstream node \( i \); and thus \( D_{ij} = (j-i) \frac{d}{v} \). Due to the existence of \( D_{ij} \), assuming that nodes \( i \) and \( j \) are simultaneously making requests, the upstream node \( i \) may occupy most of the empty slots within the time period \( D_{ij} \). In other words, the access bandwidth allocation of node \( i \) is larger than that of node \( j \) proportionally to the values of parameters \((j-i)\) and \( d \).

- **Location** \( (L_i) \) The location of node \( i \) \( (L_i) \) determines the access priority for node \( i \). The (upstream) nodes located closer to the slot generator of bus \( A \) can access the empty slots of bus \( A \) as quickly as possible. This results in starving of downstream nodes. By the same token, the (downstream) nodes located closer to the slot generator of bus \( B \) can dominate the access of slots with free request bits. This results in complete blocking of upstream nodes.

- **Traffic conditions** There are four traffic conditions directly causing unfairness to occur. They are:

  1. **Globally Saturated** \( (GS) \) All nodes \((1 \leq i < N)\) start with heavy load simultaneously.
Three types of unfairness:

- **Delay Unfairness type 1 (DU1)**: DU1 is defined as the problem of which upstream nodes dominate most of the bandwidth, due to the propagation delay ($D_{ij}$), under the $DS$ traffic condition. Upstream nodes could continuously transmit segments on bus A prior to the arrivals of the requests sent on bus B from downstream nodes. Consequently, downstream nodes starve from the lack of non-busy slots.

- **Delay Unfairness type 2 (DU2)**: DU2 is defined as the problem of which the nodes located downstream to a group of active nodes (say, from node $i$ to $k$) starve due to the propagation delay, under the $LS$ traffic condition. For every data segment transmitted, any downstream node $m$ ($\forall m > k$) suffers from bandwidth losses for at least twice of the propagation delay ($2D_{km}$). One is the propagation delay of the request slot on bus B; and the other one is the propagation delay of the access slot on bus A.

- **Location Unfairness (LU)**: LU is defined as the problem of which downstream nodes exhibit better performance owing to the location factor ($Li$) under the $US$ traffic condition. In other words, downstream nodes can make requests on bus B as quickly as possible. Consequently, upstream nodes starve from the lack of non-requested slots on bus B.

In the following section, we analyze existing mechanisms in terms of the ability of coping with the three types of unfairness discussed above.

3. Survey of Existing Mechanisms

3.1 BWB Mechanism

The BandWidth Balance (BWB) mechanism [12,13] was proposed to prevent the bandwidth from being dominated by a single user. Generally, BWB requires each node to keep track of the number of transmissions. Each time, when a node detects that the number reaches a predefined constant $m$, called the BWB_MOD value (the recommended maximum is 7), the node is forced to let a free slot through. This results in using of only a fraction $m/(m+1)$ of the slots the node could otherwise capture.

Basically, the BWB mechanism successfully overcomes the DU1 and DU2 problems over a long-term period of time. However, the LU problem is left unsolved. That is, heavy-loaded downstream nodes could still occupy most of the slots with free request bits. In addition, the short-term unfairness behavior of the network can still not be guaranteed. Finally, different BWB_MOD values are required for different loads and sizes of the network. Setting these values becomes nontrivial.

3.2 APPC Mechanism

To prevent bandwidth from being dominated by a heavy user, Access Protection and Priority Control (APPC) [4] imposes two control parameters: an upper limit ($p$) and a lower limit ($\lambda$) on the CD counter; and a buffer threshold value on the request rate. The protection limits are applied when the RQ counter value is transferred to the CD counter. Applying the upper limit to CD ($\lambda = \min\{RQ, p\}$) can assure bounded delay access; whereas applying the lower limit to CD ($\lambda = \max\{RQ, \lambda\}$) can prevent bandwidth from being hogged by the node itself. The buffer threshold value of each node restricts the request rate of the node to prevent bus B from being saturated.

On the whole, the APPC mechanism can effectively alleviate all three types of unfairness problems. However, the settings of the control parameters and request threshold could be complicated tasks. On the one hand, static setting may result in bandwidth waste for different traffic conditions. On the other hand, dynamical setting requires the network to periodically collect the status information. This may introduce more delay and bandwidth overhead to the network.

3.3 RRC mechanism

To alleviate throughput unfairness, the Reservation Request Control mechanism [6] made two modifications from DQDB. First, RRC allows each active node to send more than one outstanding requests. Each node then has to keep track of the number of requests from downstream nodes between any two successive requests. A data structure, called the request vector, is used to store these request numbers. Second, in addition to issue a R bit sent toward upstream like DQDB, a node wishing to transmit is required to send a Start (S) bit downstream to inform downstream nodes about its active status. Notice that, the S bit must be carried and sent through bus A. Moreover, a node is also required to send a End (E) bit downstream upon finishing transmissions. Two new counters are added. One is the CD2 counter (described later), and the other one is the active-node counter. Each detected S bit causes the active-node counter to increm-
resented by one; whereas each detected E bit causes the counter to be decremented by one.

Based on these modifications, RRC requires each node to count down twice. For making a request, the first count-down procedure takes place. The CD2 counter is first loaded with the value of the active-node counter. The node then starts to countdown. If there are \( n \) active upstream nodes (i.e., active-node counter value = \( n \)), a node will defer to the \( n \) upstream nodes by allowing \( n \) slots available for sending request to pass upstream. As soon as the CD2 counter reaches zero, the node then sends the request. To prepare for subsequent requests, the node copies the current RQ counter value to the request vector, resets the RQ counter, and copies the current active-node counter to the CD2 counter. As a result, the node sends request periodically \( n \) slots apart.

For accessing slots on bus A, the second count-down procedure takes place. The CD1 counter (the original counter in DQDB) is first loaded with the next queued entry from the request vector. The rest of access procedures are the same as that in DQDB.

Generally, the RRC mechanism can effectively overcome the DU1 and LU problems by minimizing the effects of large propagation delay. However, RRC still suffers from the DU2 problem under the LS traffic condition. Moreover, RRC increases implementation complexity and introduces more throughput overhead to the network.

### 3.4 Destination Release Mechanism

The Destination Release (DR) Mechanism [5,11], originally presented as a contribution for DQDB, allows slots to be reused once they have reached their destinations. In DQDB, a slot could only be used once to transmit data between any two nodes in the network. By the Destination Release scheme, slots could be used from source to destination and then released for subsequent uses by other downstream sources. To implement this, a class of specialized nodes, called erasure nodes, is added to recognize and release received slots.

The chief advantage of the DR mechanism is that it can effectively improve the network utilization. However, it has three disadvantages. First, DR violates the OR-WRITE implementation of DQDB in attempt to release slots. Thus, DR introduces a great deal of hardware and software complexity to the network. Second, none of three types of unfairness can be completely overcome. Finally, supporting multicast and broadcast transmissions becomes another minus for the implementation of the DR mechanism.

### 4. MSC Mechanism

Generally, MSC grants any node multiple slots on continuation basis for each single request issued. A node wishing to transmit first sends one request (R) bit upstream in order to join an implicit Round-Robin (RO) or RO queue. MSC then allocates bandwidth to all nodes registered in this RO queue in a round-robin manner. Upon finishing transmissions, the node withdraws itself from the RO queue by sending an end (E) bit toward upstream. The goals and achievements of the MSC mechanism are two-fold. First, MSC offers fair throughput with least overhead introduced. Second, MSC provides high throughput for both stream and bursty transmissions. In this section, the detailed algorithm of MSC is first given, followed by the discussions for handling bursty transmissions.

#### 4.1 Detailed algorithm

In MSC, the RQ counter of a node keeps track of the number of registered nodes downstream. The CD counter of a node indicates the position of the node within the RO queue. The ACF of the QA slot for MSC is shown in Figure 3. The state transition diagram is depicted in Figure 4.

A node on the bus can be in one of two states: IDLE and COUNTDOWN. A node is said to be in the IDLE state if it has no segment to transmit. The procedure for this state is described as follows:

1. The RQ counter is incremented by one whenever a slot with R bit set ON is sensed on bus B.
2. The RQ counter is decremented by one whenever a slot with E bit set ON (E = 1) is sensed on bus B.

If a node has data to transmit, it first sends a request on bus B and transfers the current RQ counter value to the CD counter. Notice that, different from DQDB, the RQ counter is not reset. The node then switches from the IDLE to the COUNTDOWN state. The RQ counter operations at this state remain the same as those at the IDLE state. Thus, the description for RQ counter operations is ignored here. The procedure for the COUNTDOWN state is described as follows.

1. The CD counter value is decremented by one whenever a free slot (B = 0) is passed through the node on bus A.
2. As soon as the CD counter value reaches zero, the node is allowed to transmit using the next free slot.
3. For every subsequent data segment (if any), the node transfers the RQ counter value to the CD counter, and repeats the operations of steps 1 and 2.
4. Upon finishing transmissions, the node withdraws itself from the RORO queue by passing an E bit toward upstream on bus B.

Figure 4. State transition diagram of the MSC mechanism (for one priority level only)
4.2 Bursty transmissions

MSC handles bursty transmissions (defined by single-segment transmissions) differently. A node, with only one data segment to transmit, is allowed to go to the COUNTDOWN state directly without making a request. A timer is used to count the waiting time during the COUNTDOWN state to prevent bandwidth from being dominated by active upstream nodes. When the timer expires, the node then treats bursty transmissions the same as stream transmissions.

In summary, during the light load condition of a network, MSC allows light-load nodes to access the bus as quickly as possible without involving with the control-bit delay overhead (R and E bits). During the heavy load condition, MSC time-division-multiplexes bandwidth to all active nodes fairly and efficiency.

5. Simulation Results

We assume that the bit rate of each bus is 155.5 Mbps, the length of each bus is 98 km, a slot is 53-byte long, the propagation speed is $5\mu$s/km. In addition, there are 31 equally spaced (= 6 slot) nodes in the network. Moreover, the priority mechanism is excluded from simulation. The behavior of MSC in terms of average access delay under four different saturated traffic conditions (GS, DS, US, and LS) are compared with that of DQDB in Figures 6, 7, 8 and 9, respectively.

As shown in Figure 6, because of the Li factor, non-upstream and non-downstream nodes experience with higher access delay in DQDB. This unfairness is eliminated in MSC. Under the DS traffic condition (Figure 7), for DQDB, downstream nodes experience much higher access delay than upstream nodes do. Therefore, DQDB exhibits a DU1 problem; whereas MSC is free from this problem. Under the US traffic condition (Figure 8), upstream nodes experience with high access delay in DQDB. This shows that DQDB exhibits an LU problem. Figure 9 demonstrates a DU2 phenomenon of DQDB and DU2-free of MSC under the LS traffic condition. For example, in DQDB, the nodes located downstream from an active group of nodes (nodes 14 and 15) suffer from the request propagation delay. Although MSC cannot completely be immune from this delay, it keeps the effect to a tolerable degree. Finally, Figure 10 illustrates fair and high throughput performance of MSC under various loads of the network.

6. Conclusions and Future Research
In this paper, we first described DQDB and analyzed its unfairness problems under different types of traffic conditions. In summary, there are three types of unfairness problems: Delay Unfairness type 1 (DU1), Delay Unfairness type 2 (DU2), and Location Unfairness (LU). They are caused by the request propagation delay, locations of nodes, and four traffic conditions (Globally Saturated (GS), Downstream Saturated (DS), Upstream Saturated (US), and Locally Saturated (LS)).

To approach the unfairness problems, we surveyed four existing relief mechanisms, i.e. BWB, RRC, APPC, and DR, and discussed their trade-offs. We then proposed a fair and fast protocol, called Multiple Segment Control (MSC).
The MSC mechanism grants a node multiple slots on continuation basis for any single request issued. Generally, a node wishing to transmit first sends one request (R) bit upstream in order to join a Round-Robin (RORO) queue. MSC then time-division-multiplexes necessary bandwidth to all nodes registered in the queue in a round-robin manner. Upon finishing transmissions, the node withdraws itself from the RORO queue by sending an end (E) bit toward upstream. We demonstrated the performance superiority of MSC over DQDB in terms of fair and high throughput via simulation results. Finally, the design of recovery capability, such as how to remove broken nodes from the RORO queue, is one of our current on-going research.

7. Acknowledgments

The authors would like to especially thank Dr. K. Khalil for his providing us the most up-to-date DQDB document. In addition, we would also like to thank Dr. A. L. P. Chen and Dr. J. H. Huang for their giving valuable comments.

References