論文 97-22-4-3

버퍼의 문턱치값을 이용한 DQDB망의 공평성 개선 및 혼잡 제어에 관한 연구

正會員 高 成 賢*, 徐 鎭 敎**, 金 峻 年***

A Study on Improving Fairness and Congestion Control of DQDB using Buffer Threshold Value

Seong-Hyeon Koh*, Jin-Kyo Seo**, Joon-Nyun Kim*** *Regular Members*

요 약

MAN(Metropolitan Area Networks)을 위한 IEEE 802.6 표준인 DQDB(Distributed Queue Dual Bus) MAC(Medium Access Control)프로토콜은 이중버스 구조의 용량을 전부 이용하지는 못한다. 대역폭 균형 메카니즘을 사용할 때, 노드들사이의 공평한 대역폭 분산이 보장되지만, 이 프로토콜은 망 부하의 변화에 적응하는데 많은 시간이 필요하게 된다. 게다가, 대역폭 균형 메카니즘은 유용한 대역폭의 일정량을 사용하지못하고 낭비한다. 고속 백본(Backbone)망에서, 각 노드는 호스트뿐 아니라 여러 LAN들을 연결하는 브리지/라우터로서 동작한다. 그러나 고속 LAN들이 존재할 수 있기 때문에 백본망에 접근하는 전송률과 유용한 버퍼 용량의 제약으로 인해 라우터에서 혼잡이 일어날 수 있다. 이러한 혼잡을 방지하기위해 라우터에 혼잡 제어 알고리즘을 구현하는 것이 필요하다.

이 논문에서는 이중버스 고속망을 위한 효과적인 혼잡 제어와 대역폭 손실없는 공평한 MAC(Medium Access Control) 프로토콜을 제안한다. 이 제안한 프로토콜에 근간을 두면 각 노드의 전송률이 그 노드의 부하에 비례하도록 망내의 모든 버퍼들을 완전히 분산시킬 수 있다. 다르게 말하면, 부하가 많은 노드는 혼잡을 예방할 수 있도록 자신의 버퍼안에 있는 세그먼트를 더 빠른 속도로 전송할 수 있도록 한다. 동시에, 혼잡이 일어나지않은 각 노드는 전송을 늦추고 새로 들어온 메시지들을 자신의 버퍼에 저장한다. 이것은 망상의 버퍼들이 동적으로 나누어질 수 있음을 함축한다.

모의실험 결과에 따르면 기존의 DQDB MAC 프로토콜대신 제안된 프로토콜을 이용하면 세그먼트가 큐에서 기다리는 시간이 줄어들고, 부하가 많은 노드에서의 세그먼트 손실율이 감소하며, 백본망에서 높은 처리율을 얻을 수 있음을 알 수 있다. 왜냐하면 제안하는 프로토콜은 세그먼트마다 Request를 신청하는 것이 아니라 자기 노드에 전송할 세그먼트들이 있다는 뜻으로 한 번 Request를 보내므로 표준 DQDB보다 Request를 신청하는 빈도가 적어지기 때문에 그만큼 세그먼트가 큐에서 기다리는 시간이 줄어들게 된다. 그리고 제안하는 프로토콜에서의 각

*현대전자 통신연구소
**양산전문대학 전자통신과
***중앙대학교 전자공학과
論文番號:96226-0726
接受日字:1996年 7月 26日

노드는 부하에 비례하여 대역폭을 차지할 수 있기 때문에 각 노드의 큐의 길이가 제한되었을 경우 부하가 많은 노드에서의 세그먼트 손실율이 감소하게 되며, Bandwidth Balancing DQDB인 경우에는 노드마다 공평한 대역폭을 차지하게하기위해 어느 정도의 대역폭을 손실을 감수하지만 제안하는 DQDB MAC 프로토콜은 전송할 세그먼트를 가지고 있는 노드들끼리 정확히 한 번씩 골고루 대역폭을 차지하게함으로써 대역폭 손실없이 공평한 대역폭을 차지할 수 있다.

ABSTRACT

DQDB(Distributed Queue Dual Bus) protocol, the IEEE 802.6 standard protocol for metropolitan area networks, does not fully take advantage of the capabilities of dual bus architecture. Although fairness in bandwidth distribution among nodes is improved when using so called the bandwidth balancing mechanism, the protocol requires a considerable amount of time to adjust to changes in the network load. Additionally, the bandwidth balancing mechanism leaves a portion of the available bandwidth unused.

In a high-speed backbone network, each node may act as a bridge/router which connects several LANs as well as hosts. However, Because the existence of high speed LANs becomes commonplace, the congestion may occur on a node because of the limitation on access rate to the backbone network and on available buffer spaces. To release the congestion, it is desirable to install some congestion control algorithm in the node.

In this paper, we propose an efficient congestion control mechanism and fair and waste-free MAC protocol for dual bus network. In this protocol, all the buffers in the network can be shared in such a way that the transmission rate of each node can be set proportional to its load. In other words, a heavily loaded node obtains a larger bandwidth to send the segments so that the congestion can be avoided while the uncongested nodes slow down their transmission rate and store the incoming segments into thier buffers. This implies that the buffers on the network can be shared dynamically.

Simulation results show that the proposed protocol significantly reduces the segment queueing delay of a heavily loaded node and segment loss rate when compared with original DQDB. and it enables an attractive high throughput in the backbone network.

Because in the proposed protocol, each node does not send a requet by the segment but send a request one time in the meaning of having segments, the frequency of sending requests is very low in the proposed protocol. so the proposed protocol signigificantly reduces the segment queuing delay. and In the proposed protocol, each node uses bandwidth in proportion to its load. so In case of limitation on available buffer spaces, the proposed protocol reduces segment loss rate of a heavily loaded node. Bandwidth Balancing DQDB requires the wastage of bandwidth to be fair bandwidth allocation. But the proposed DQDB MAC protocol enables fair bandwidth without wasting bandwidth by using bandwidth one after another among active nodes.

Ⅰ. 서 론

급격한 컴퓨터 네트워크의 발전으로 넓은 대역폭을 갖는 고속의 멀티미디어 네트워크 기술에 대한 관심과 기존의 LAN들을 상호 연결하여 사용하고자 하는 요구의 증가로 IEEE는 DQDB(Distributed Queue Dual Bus)[1][2]를 IEEE 802.6 MAN(Metropolitan Area Network) 표준 프로토콜로 채택하였다.

DQDB는 망의 크기나 전송속도의 증가에 관계없이 대역폭 이용도가 거의 100%에 가깝다는 장점을 갖고 있다. 또한 DQDB는 IEEE 802.2 LLC 표준을 지원함으로써 IEEE 802.x(x = 3, 4, 5) LAN들과 상호연결이

가능할 뿐 아니라 53 옥텟의 슬롯을 전송단위로 사용하기 때문에 53 옥텟의 셀을 전송단위로 하는 BISDN (Broadband Integrated Services Digital Network)과의 연동도 쉽게 이루어질 수 있다는 장점을 갖고 있다.

그러나 DQDB 성능 분석[2-8]에서 DQDB의 매체 접근 제어(MAC) 메카니즘이 불공평성 요인을 가지고 있다는 것을 보였다. DQDB의 불공평성은 네트워크의 크기, 제공된 부하, 노드간의 전파지연시간이 클수록 더욱 심각해진다. 이런 이유 때문에 Bandwidth Balancing(BWB)[9] 이라 불리는 메카니즘이 제안되었다. 그러나 이 메카니즘도 여러 문제점들이 있다. 첫째, 공평한 대역폭 할당이 이루어지는데 걸리는 시간이 길다. 둘째, 어느 정도의 대역폭 손실을 가져온다. 셋째, 부하가 적은 노드의 대역폭을 보장하고 나머지 대역폭을 부하가 많은 노드들이 나누어 갖기 때문에 부하가 많은 노드는 부하가 적은 노드에 비해 세그먼트를 전송하기까지 걸리는 평균 시간이 상당히 길다.

위에서 언급한 첫 번째와 두 번째 문제점을 해결하기 위한 메카니즘은 여러 논문[10-12]에서 제안되었다. 하지만 이들 논문에서도 부하가 적은 노드와 부하가 많은 노드가 대역폭을 갖는 권리가 동등하므로 부하가 많은 노드는 세그먼트를 전송하기까지 많은 시간이 걸리기 때문에 부하가 많은 노드의 버퍼 사이즈가 제한된다면 많은 세그먼트가 overflow가 생길 수 있다.

즉, DQDB망이 backbone망이고 DQDB망의 각 노드마다 LAN들이 연결되어있고, DQDB망의 전송용량은 150Mbps이고 DQDB망에 연결된 노드(라우터)의 수가 30개라고 가정했을 때 최악의 경우 각 노드(라우터)는 단지 5Mbps의 전송 스피드로 DQDB망에 접속할 수 있습니다. 그렇기 때문에 DQDB망의 각 노드에 고속의 LAN들이 연결되었을 경우, 제한된 DQDB망의 입력 버퍼에 혼잡이 발생할 수 있다. 따라서 본 논문에서는 노드에 제공된 부하에 비례하여 대역폭을 차지할 수 있고 같은 부하를 가진 노드들 사이에는 대역폭 손실없이 공평하게 대역폭을 차지할 수 있는 새로운 메카니즘을 제시한다.

본 논문은 기존의 DQDB망의 분산 큐 액세스 프로토콜과 Bandwidth Balancing(BWB) 기법에 대한 설명과 그 문제점들을 제 Ⅱ장과 Ⅲ장에서 검토한 후, 제 Ⅳ장에서는 검토된 내용을 기반으로 노드내의 버퍼의 문턱치 값을 이용한 매체 접근의 공평성과 혼잡을 방지하는 MAC 프로토콜을 제시하였다. 또한 제 Ⅴ장에서는 모의 실험을 통해 본 논문에서 제안하는 프로토콜의 성능평가를 수행하였으며, 마지막으로 제 Ⅵ장에 결론을 수록하였다.

Ⅱ. DQDB MAC 프로토콜

DQDB망에 접속되어 있는 각 노드는 그림 1과 같이 각 버스에 해당하는 RQ(Request) 카운터와 CD (Count Down) 카운터를 갖고 있으며, 버스 A로 전송할 데이타가 발생되면 버스 B로 지나가는 슬롯을 조사하여 REQ 비트가 0인 슬롯의 REQ 비트 값을 1로 만들어 상위노드에게 자신이 전송할 것이 있음을 알린다. 반면에 전송할 데이타 세그먼트가 없는 노드는 버스 B로 지나가는 슬롯의 REQ 비트 값이 1이면 RQ 카운터 값을 하나 증가시키고, 버스 A로 지나가는 슬롯의 BUSY 비트가 0이면 RQ 카운터 값을 하나 감소시킨다.

만약 전송할 데이타가 생겨 자신의 전송요구를 신청했다면, 노드는 RQ 카운터 값을 CD 카운터로 복사하고 RQ 카운터를 0으로 초기화함으로써 카운트다운 상태로 들어간다. 카운트다운 상태의 노드는 버스 A로 BUSY 비트가 0인 빈 슬롯이 지나가면 CD 카운터 값을 하나 감소 시킨다. CD 카운터 값이 0이 되면 다음에 도착한 빈 슬롯에 데이타를 실어 전송한다.

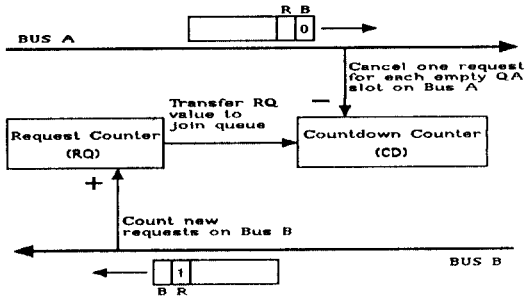


그림 1. 노드에서의 분산큐 동작
Fig. 1 Operation of distributed queue in node

2.1 표준 DQDB의 특징

DQDB의 분산 큐 액세스 프로토콜은 기존의 MAC 프로토콜과는 달리 모든 노드가 망상태에 대한 정보

(Request)를 정확하게 연속적으로 유지하게 함으로써 데이타 전송을 원하는 노드가 분산 큐에서의 자신의 전송 위치를 정확하게 결정하게 해주는 매체 접근 제어(MAC:Medium Access Control) 프로토콜이다. 따라서 전송할 데이타 세그먼트가 발생된 노드는 단지 하위노드에서 전송 대기중인 데이타 세그먼트 갯수를 나타내는 망 상태 정보(Request)만을 확인하여 몇 번째 도착하는 빈슬롯을 자신이 사용할 수 있는가를 정확히 판단함으로써 분산 큐에서의 자신의 전송위치를 결정한다.

하지만 DQDB는 망의 길이가 길고 부하가 많을 때 문제가 발생한다.[5] 만약 상위에 위치한 노드가 대역폭을 사용하고 있다면 하위 노드는 대역폭을 차지하는데 불리하게 된다. 왜냐하면, 하위 노드가 전송할 세그먼트가 발생하여 상위 노드에 Request를 보내는데 걸리는 전파지연시간 때문에 하위 노드는 대역폭을 차지하는데 불리하게 된다. 그리고 상위 노드가 하위 노드보다 먼저 전송할 세그먼트들이 발생했을 때 하위 노드가 상위 노드와 동등한 대역폭을 차지하기까지 걸리는 시간이 매우 긴 문제점들이 있다.

이러한 문제점을 해결하기위해 Bandwidth Balancing(BWB) 기법[9]이 제안되었다. 이 기법을 앞으로 BWB DQDB라고 하겠다.

다음 장에서는 BWB DQDB에 대한 설명과 문제점들을 살펴본다.

Ⅲ. BWB DQDB 프로토콜

DQDB가 안고있는 문제는 노드의 위치에 따라 QA 슬롯 접근에 불공평성을 갖는다는 것이다. 이러한 불공평성을 방지하기위해 IEEE 802.6 표준은 Bandwidth Balancing modulus(BWB_MOD)라고 하는 시스템 파라미터의 적정한 값을 갖는 Bandwidth Balancing(BWB) 메카니즘을 이용한 BWB DQDB를 권고했다.

3.1 Bandwidth Balancing Mechanism

BWB 메카니즘은 노드에서의 빈 QA 슬롯에 대한 접근 기능을 경우에 따라서 접근할 수 없게한다. 이 메카니즘은 BWB_MOD 값에 따라 제어된다.

이 메카니즘을 수행하기 위해서는 각 버스에 대해 독립적으로 Bandwidth Balancing Counter(BWB_CNT)를 동작시킨다.

BWB_CNT는 노드가 QA 세그먼트를 전송할 때마다 하나씩 증가하다가 BWB_MOD 값에 도달하면 BWB_CNT는 0으로 리세트된다. BWB_CNT가 0으로 될 때 보낼 QA 세그먼트가 큐에 대기하고 있지 않을 때는 RQ 카운터의 값을 하나 증가시키고 보낼 QA 세그먼트가 큐에 대기하고 있는 경우에는 CD 카운터의 값을 하나 증가시킨다. 즉, 노드가 BWB_MOD만큼 세그먼트를 전송할 때마다 그 버스를 통과하는 빈 슬롯의 사용권을 하위 노드에 넘겨주는 효과를 갖는다.

3.2 BWB DQDB의 문제점[9]

BWB DQDB는 노드가 자신의 BWB_MOD의 값만큼 전송할 때마다 자신의 RQ 카운터를 하나 증가시키고 빈 슬롯을 그냥 지나치게 한다. 이 지나친 슬롯은 CD 카운터가 0인 첫 번째 active 하위 노드에 의해 사용될 수 있다. 하지만 active된 하위 노드가 없을 경우는 그 지나친 빈 슬롯은 어떤 노드도 쓰지 못하고 낭비되어진다는 것이다. 여기서 active 상태란 전송할 세그먼트가 생겨서 그 세그먼트에 대해 request를 보내고 그 세그먼트가 전송큐에 대기중인 상태를 의미한다.

여기서 노드내의 큐의 개념을 이해하기 쉽게 하기위해 다음과 같이 큐를 구분한다. 일단 노드가 전송할 세그먼트가 발생하면 발생된 세그먼트를 대기큐에 저장하고 그 발생된 세그먼트에 대한 request를 B버스를 통해 전송한 후에 그 세그먼트는 전송큐에 저장되어 CD 카운터의 값이 0이 된 후 버스 A상으로 빈 슬롯이 오면 전송큐에 저장된 세그먼트는 전송되어진다. 그리고 DQDB MAC 프로토콜과 BWB DQDB 프로토콜 모두 전송큐가 비어 있을 때만 대기큐의 첫 번째 세그먼트에 대한 request를 버스 B상으로 보낼 수 있다.[1]

즉, BWB DQDB가 갖는 문제점은 상위 노드들이 지나치게한 빈 슬롯은 낭비되어질 수 있다는 것이다. BWB_MOD의 값이 작을수록 대역폭 손실은 더욱 높아진다. 이런 이유 때문에 대역폭 손실을 줄일 수 있는 최적의 BWB_MOD의 값은 8이나 9이어야 한다는 제안도 있다.

두 번째 문제점은 정상상태에 도달하는 시간이 길

다는 것이다. 여기서 정상상태란 노드들간에 동등한 대역폭을 차지하는 상태를 의미한다. 만약, BWB_MOD의 값이 작으면 정상상태에 도달하는 시간은 짧아진다. 그러나 위에서 말했듯이 BWB_MOD의 값이 작으면 대역폭 손실이 많아진다. 그러므로 대역폭 효율과 정상상태에 도달하는 시간사이에 균형을 맞출 수 있는 BWB_MOD값을 결정해야한다.

세 번째 문제점은 부하가 적은 노드와 부하가 많은 노드 사이에 대역폭을 차지할 수 있는 권리가 동등하기 때문에 부하가 많은 노드에서 발생된 세그먼트들은 부하가 적은 노드보다 전송되기까지 많은 시간을 큐에서 기다려야 되므로 큐 용량이 제한되어 있다면 overflow가 발생될 수 있다. 예를 들어 모든 노드의 부하가 동일하다면 BWB DQDB의 성능은 좋게 나오지만 노드마다 부하를 다르게 주면 부하가 많은 노드의 세그먼트들이 전송되기 까지 많이 기다려야하는 불공평성이 발생한다.

표 1은 BWB DQDB 프로토콜, SIMA 프로토콜[12], 제안하는 DQDB MAC 프로토콜을 적용한 경우, 각 노드에 제공된 부하에 따른 각각의 throughput 결과를 나타내었다. 이 모의실험에서는 BWB_MOD의 값을 8로 하고 노드의 수를 3으로하고, 노드간의 거리를 40slot time으로 가정하고 수행하였다.

표 1은 여섯가지 부하가 적용된 경우에 BWB DQDB가 어느 정도의 대역폭 손실을 가져온다는 것과 부하가 적은 노드는 request를 신청한만큼 대역폭을 쓰고 나머지 대역폭을 부하가 많은 노드끼리 나누어 갖는다는 것을 보여주고 있다. 그리고 BWB DQDB 프로토콜을 적용한 경우, 노드수가 N이고 BWB_MOD 값이 M이라고 할 때 약 $\frac{1}{1+MN}$ 만큼의 대역폭이 손실됨을 알 수 있다.

Ⅳ. 제안하는 DQDB MAC 프로토콜

본 논문에서 제안하는 분산 큐 액세스 프로토콜은 부하가 같은 노드사이에는 대역폭 손실없이 공평하게 대역폭을 차지하게 하고 부하가 많은 노드는 큐의 문턱치 값을 이용해 그 부하에 비례하여 대역폭을 차지할 수 있는 방법으로 노드에서 세그먼트가 발생하여 전송되어 나가기까지 걸리는 세그먼트당 큐에서 기다리는 평균 시간이 부하가 많은 노드나 부하가 적은 노드나 거의 비슷하게 나오게 함으로써 제공된 부하에 관계없이 공평성을 갖도록 할 뿐아니라 부하가 많은 노드에 혼잡을 방지할 수 있다.

4.1 제안하는 DQDB MAC 프로토콜의 기본적인 동작

제안하는 분산 큐 액세스 프로토콜의 기본적인 동작은 SIMA(Slot Interleaved Multiple Access Scheme)[12] 동작에 근거를 하고 있다.

제안하는 분산 큐 액세스 프로토콜의 기본적인 동작은 노드에 전송할 세그먼트가 발생하면 그 세그먼트는 전송큐가 비어있는가 보고 비어있으면 대기큐로 들어가고, 전송큐가 비어있지않으면 대기큐를 거

표 1. 부하를 노드마다 임의로 주었을 때의 throughput 성능 비교
(노드 수 = 3, 노드간의 거리 = 40slot time, BWB_MOD = 8)

Table 1. Throughput performance comparison as loads of each node are different

제공된 부하			THROUGHPUT (BWB DQDB)			THROUGHPUT (SIMA)			THROUGHPUT (제안하는 DQDB)		
station 1	station 2	station 3	station 1	station 2	station 3	station 1	station 2	station 3	station 1	station 2	station 3
0.80	0.20	0.80	0.375	0.20	0.375	0.4	0.2	0.4	0.445	0.11	0.445
0.80	0.25	0.80	0.355	0.25	0.355	0.375	0.25	0.375	0.435	0.13	0.435
0.80	0.30	0.80	0.326	0.30	0.326	0.35	0.30	0.35	0.424	0.152	0.424
0.25	0.80	0.80	0.25	0.35	0.35	0.25	0.375	0.375	0.13	0.435	0.435
0.80	0.25	0.80	0.35	0.25	0.35	0.375	0.25	0.375	0.435	0.13	0.435
0.80	0.80	0.25	0.35	0.35	0.25	0.375	0.375	0.25	0.435	0.435	0.13

치지않고 바로 전송큐로 들어간다. 즉, 전송할 세그먼트가 발생했을 때 전송큐가 비어있지 않다면 그 발생한 세그먼트는 request를 보내지 않고도 바로 전송큐로 들어설 수 있다는 것이다. 그리고, 대기큐로 들어간 세그먼트는 버스 B로 지나가는 슬롯의 REQ 비트값이 0인 슬롯을 보면 REQ 비트값을 1로 만들어 상위 노드에 자신이 전송할 것이 있음을 알린다. 이 때 대기큐에 있는 모든 세그먼트는 모두 전송큐로 들어서게 된다. 그러나, 노드내의 전송큐와 대기큐가 모두 비게되면 버스 B상으로 end request를 보내어야 한다.

여기서 전송할 세그먼트가 발생했을 때 전송큐가 비어있지 않다는 것은 이미 Request를 신청했지만 전송 못한 세그먼트가 있다는 것으로 표준 DQDB인 경우는 이 때, 그 전송큐에 있는 세그먼트가 전송된 다음에야(즉, 전송큐가 비어있어야만) 그 다음 발생된 세그먼트에 대한 Request를 신청하고 그 발생한 세그먼트는 전송큐에 들어가게 된다. 하지만 제안하는 DQDB MAC 프로토콜의 기본동작은 전송할 세그먼트가 발생했을 때, 이미 Request를 신청한 세그먼트가 있을 경우(즉, 전송큐가 비어있지 않은 경우)에는 그 발생한 세그먼트는 Request를 신청하지 않고 바로 전송큐로 들어갈 수 있기 때문에 표준 DQDB보다 Request를 신청하는 빈도가 적게되어 전송지연이 줄어들게 되는 장점이 있다. 즉, 표준 DQDB의 각 노드의 동작은 세그먼트별로 Request를 신청하지만, 제안하는 DQDB MAC 프로토콜의 각 노드의 동작은 자기 노드에 전송할 세그먼트들이 있다는 뜻으로 한 번 Request를 보내는 셈이 된다. 물론, 어떤 노드에 전송할 세그먼트가 발생했을 때 전송큐가 비어있다면(즉, 이미 Request를 신청한 세그먼트가 없는 경우) 발생한 세그먼트는 대기큐로 들어가서 Request를 신청해야만 전송큐로 들어설 수 있다. 그리고 어떤 시점에서 노드의 대기큐와 전송큐가 모두 비어있으면 그 노드는 버스 B방향으로 end request를 신청하게된다.

제안하는 분산 큐 액세스 프로토콜에서의 노드내의 기본적인 카운터의 동작은 다음과 같다. 여기서는 7개의 카운터 중 기본적인 동작을 수행하는 두가지 카운터(RQ 카운터, CD 카운터)의 동작만 살펴보겠다.

RQ 카운터는 자신을 제외한 하위 노드들 중 active된 노드의 수를 카운트한다. 즉, 버스 B상으로 request를 볼때마다 1씩 증가하고 end request를 볼때마다 1씩 감소한다. 그리고, 표준 DQDB의 RQ 카운터와는 달리 버스 A상으로 빈 슬롯이 지나가도 RQ 카운터의 값은 감소되지 않는다. 그러므로 RQ 카운터는 자신을 제외한 하위 노드들 중 active된 노드의 수만 카운트한다고 생각하면 된다.

어떤 한 노드는 자신의 CD 카운터 값이 0이면 버스 A상의 빈 슬롯에 세그먼트를 전송할 수 있다. 자신의 CD 카운터 값이 0이 되어 버스 A상의 빈슬롯에 세그먼트를 전송할 때마다, RQ 카운터의 값을 CD 카운터의 값에 복사하고 countdown 상태가 된다. countdown 상태가 되면 버스 A상에 빈슬롯이 지나갈 때마다 CD 카운터 값이 1씩 감소한다. 그리고 RQ 카운터의 값을 CD 카운터에 복사할 때, 표준 DQDB와는 달리 RQ 카운터의 값은 0으로 리세트되지 않고 그냥 자신의 값을 갖는다. 즉, 어떤 노드가 세그먼트를 전송할 때마다 무조건 active된 하위 노드의 수만큼 버스 A상의 빈 슬롯을 지나가게 한다.

4.1.1 제안하는 DQDB MAC 프로토콜 기본 동작의 특징

이 프로토콜은 아주 단순하면서도 대역폭 손실없이 대역폭을 노드간에 골고루 나누어 갖는 장점이 있고 세그먼트당 request를 보내는 것이 아니라 전송할 세그먼트가 노드에서 발생했을 때, 대기큐와 전송큐가 비어있을때만 request를 보내므로 request를 보내는 빈도가 매우 적다. 그러므로, 어떤 노드가 대역폭을 얻기위해 상위 노드로 request를 즉시 보낼 수 있다. 그러므로, 어떤 노드에 전송할 세그먼트가 발생하여 전송하는 데 걸리는 시간도 줄일 수 있다.

그리고, 노드의 전송 동작은 자신이 active되고 CD 카운터의 값(초기값은 0)이 0이면 먼저 하나를 전송하고나서 정확히 active된 하위 노드의 수(RQ 카운터의 값)만큼 양보한다. 결국은 active된 노드사이에 골고루 하나씩 전송할 기회를 가질 수 있게 되므로 대역폭 손실없이 대역폭을 골고루 쓸 수 있게 된다.

하지만, 이 기본동작의 문제점은 부하가 많은 노드와 부하가 적은 노드가 똑같은 전송권리를 가지고 대역폭을 쓰기 때문에 부하가 많은 노드는 부하가 적은 노드보다 세그먼트가 발생하여 전송하기까지 많은 시간을 기다려야 하는 문제점이 있다.

이 문제점을 해결하기위해 이 기본 동작을 보완한

DQDB MAC 프로토콜의 전체동작은 뒤에 자세히 설명하겠다.

4.1.2 제안하는 프로토콜 기본동작의 매체 접근 지연 분석

DQDB 망이 무한한 큐 용량을 갖는 N개의 노드들로 구성되어있다고 가정한다. 그 노드들은 버스 A에 관해 1부터 N까지 이름이 붙여진다. 프로토콜의 동작이 두 개의 버스에 대해 같기 때문에 우리는 단지 버스 A방향만을 생각한다. 그리고 노드에서 전송하기 위해 발생하는 세그먼트는 슬롯 크기(53 octets)를 갖는다고 가정한다.

매체 접근 지연은 큐의 head에 도착한 각 세그먼트에 대해 그 세그먼트가 전송되어 나가기까지 걸리는 시간으로 정의하겠다.

매체 접근 지연을 분석하기 전에 변수들에 대한 정의를 내리겠다.

L:두 개의 이웃 노드 사이의 거리(슬롯의 정수로 표현)
U_i:노드 i의 상위에 있는 노드들 중 active된 노드 수
D_i:노드 i-1의 하위에 있는 노드들 중 active된 노드 수
N:전체 노드의 수
M:한 노드가 active일때 망에 존재하는 평균 active 노드 수
λ:각 노드의 세그먼트 발생률
ν:노드 전체의 평균 세그먼트 발생률
δ_i:노드 i의 상위에 위치한 노드들 중 노드 i와 가장 가까운 거리에 있는 active 노 드 사이의 노드 수를 나타내는 random variable
γ_i:δ_i의 평균
τ_i:노드 i의 전송 지연

노드 i에 대해 큐의 첫 번째 위치한 어떤 세그먼트의 매체 접근 지연은 단지 U_i가 0보다 클 때 일어난다. 그러므로 각 세그먼트들이 노드 1에 도착했을 때 큐의 첫 번째 위치한 세그먼트의 매체 접근 지연은 0이 된다. 하지만 노드 1보다 하위에 위치한 노드들의 큐의 첫 번째 위치한 세그먼트의 매체 접근 지연을 분석해보자.

Binomial 분포에 따라 두가지 경우($D_i=0$인 경우와 $D_i>0$인 경우)에서 노드 i ($i\geq 2$)의 확률은 다음과 같다.

$$P\{U_i=a,\ D_i=0\}=C_a^{i-1}\rho^a(1-\rho)^{i-a-1}(1-\rho)^{N-i},\quad 1\leq a\leq i-1 \tag{1}$$

여기서 ρ는 어떤 노드가 active일 확률을 나타내므로 $(1-\rho)^{N-i}$는 자신보다 하위노드가 active가 아닐 확률을 나타낸다.

$$P\{U_i=a,\ D_i=b\}=C_a^{i-1}\rho^a(1-\rho)^{i-a-1}\left[\sum_{b=1}^{N-i+1}C_b^{N-i+1}\rho^b(1-\rho)^{N-i-b+1}\right]\quad 1\leq a\leq i-1,\ b>0 \tag{2}$$

여기서 $\left[\sum_{b=1}^{N-i+1}C_b^{N-i+1}\rho^b(1-\rho)^{N-i-b+1}\right]$는 자신을 포함한 active된 하위노드의 수가 b일 확률을 의미한다.

여기서 a는 노드 i의 상위에 위치한 active된 노드의 수이다.

식 (1)과 (2)는 $M\rho<1$일 때 유효하다. $M\rho>1$이면 그 망은 overload이고 불안정하게 되어 그 망은 세그먼트의 도착률을 따라 잡을 수 없고 각 노드의 큐길이는 무한히 증가한다.

이 때, 식(2)는 다음과 같이 쓰여질 수 있다.

$$P\{U_i=a,\ D_i=b\}=C_a^{i-1}\rho^a(1-\rho)^{i-a-1}[1-(1-\rho)^{N-i+1}],\quad 1\leq a\leq i-1,\ b>0 \tag{3}$$

여기서 $[1-(1-\rho)^{N-i+1}]$는 자신을 포함한 모든 하위노드가 active일 확률을 의미한다.

$U_i>0$이고 $D_i=0$인 경우, 노드 i는 대역폭을 얻기 위해 상위노드로 request를 보내야 한다. 하지만 제안하는 프로토콜의 기본동작은 그런 request를 보내는 빈도가 표준 DQDB 프로토콜 동작보다 매우 적기 때문에 request를 즉시 보낼 수 있다. 노드 i가 빈 슬롯을 얻기 위해 기다리는 시간은 전달 지연 $\tau_i L$의 2배에다가 상위 노드가 현재 차지하고 있는 대역폭을 기다려야 한다.

노드 i보다 상위에 위치한 노드들 중 가장 가까운 active 노드를 노드 j라 하자. 노드 i의 큐에 첫 번째 위치한 세그먼트가 겪는 최악의 매체 접근 지연은 노드 j가 전송하고 있는 동안 request를 수신했을 때 일어나고 노드 i를 위한 빈 슬롯은 2a 슬롯 후로 지나가게 될 것이다. 하지만 노드 j가 빈 슬롯을 사용하기 바로

직전에 request를 받으면 노드 i를 위한 빈 슬롯은 a 슬롯후로 지나가게 될 것이다. 그러므로 전자의 경우와 후자의 경우의 평균 매체 접근 지연은 $2\tau_i L + \frac{3}{2}a$ 가 된다.

다음은 τ_i를 계산하기 위해 우리는 맨 처음 노드 i에 대해 $U_i = a$이고 $D_i = 0$인 경우 노드 i와 그것의 가장 가까운 상위 active 노드 사이에 k개의 노드가 있을 조건부 확률을 생각해보자.

$$P\{\delta_i = k \mid U_i = a,\ D_i = 0\} = C_{a-1}^{i-k-1}\rho^{a-1}(1-\rho)^{i-k-a}\rho(1-\rho)^{k-1}(1-\rho)^{N-i}$$
$$a = 1, 2, \cdots, i-k-1 \qquad (4)$$

노드 i와 가장 가까운 상위 active 노드를 j라고 하면 여기서 i−k−1이 의미하는 것은 노드 j를 포함한 노드 j의 상위에 위치한 노드의 수를 의미한다.

$$\begin{aligned} P\{\delta_i = k\} &= \sum_{a=1}^{i-k-1} P\{\delta_i = k \mid U_i = a,\ D_i = 0\} \\ &= \sum_{a=1}^{i-k-1} C_{a-1}^{i-k-1}\rho^{a-1}(1-\rho)^{i-k-a}\rho(1-\rho)^{N-i+k-1} \\ &= \rho(1-\rho)^{N-i+k-1}\left[\sum_{a=1}^{i-k-1} C_{a-1}^{i-k-1}\rho^{a-1}(1-\rho)^{i-k-a}\right] \\ &= \rho(1-\rho)^{N-i+k-1} \end{aligned}$$

여기서 $\left[\sum_{a=1}^{i-k-1} C_{a-1}^{i-k-1}\rho^{a-1}(1-\rho)^{i-k-a}\right]$는 모든 경우를 의미하므로 1이 된다.

$$\begin{aligned} \tau_i = E[\delta_i] &= \sum_{k=1}^{i-1} k\rho(1-\rho)^{N-i+k-1} \\ &= \rho(1-\rho)^{N-i}\sum_{k=1}^{i-1} k(1-\rho)^{k-1} \\ &= \rho(1-\rho)^{N-i}\frac{\partial}{\partial\rho}\left[\sum_{k=1}^{i-1}(1-\rho)^k\right] \\ &= \rho(1-\rho)^{N-i}\frac{\partial}{\partial\rho}\left[\frac{(1-\rho)[1-(1-\rho)^{i-1}]}{\rho}\right] \\ &= (1-\rho)^{N-i}\left[\frac{1-(1-\rho)^{i-1}}{\rho} - (i-1)(1-\rho)^{i-1}\right] \end{aligned} \qquad (5)$$

안정한 M/M/1 queueing system에 따라 한 노드가 active일 때 망에서의 평균 active 노드의 수 M은 다음과 같다.

$$M = \frac{(N-1)\rho}{1-(N-1)\rho} + 1 \qquad (6)$$

여기서 $(N-1)\rho$는 그 밖의 다른 노드에서의 utilization factors의 합을 나타내며, $1-(N-1)\rho$은 한 노드를 제외한 모든 노드가 active가 아닐 경우를 나타낸다. 제안한 프로토콜의 기본동작에서 어떤 노드에서 하나의 세그먼트가 도착할 때 자신의 큐가 이미 비어있지 않다면 request의 전송을 할 필요가 없다. 다르게 말하면 한 노드의 큐가 비게되면 그 노드는 가능한 한 빨리 end request를 보낸다. 그러므로 세그먼트가 도착할 때 그 버퍼가 비어있다면 첫 번째 세그먼트의 매체 접근 지연은 $(2\tau_i L + \frac{3}{2}a)$와 같게 된다.

세그먼트가 도착했을 때 노드가 nonactive 일 확률은 $1-\frac{M}{N}$ 임을 알 수 있다.

식 (1)부터 식 (6)을 조합하면 큐의 첫 번째 위치한 세그먼트의 평균 매체 접근 지연 Φ를 얻을 수 있다.

$$\Phi = \sum_{a=1}^{i-1}\{C_a^{i-1}\rho^a(1-\rho)^{i-a-1}(1-\rho)^{N-i}(2\tau_i L + \frac{3}{2}a)(1-\frac{M}{N}) + C_a^{i-1}\rho^a(1-\rho)^{i-a-1}[1-(1-\rho)^{N-i+1}](\frac{a}{2})\} \qquad (7)$$

$$\frac{3}{2}\left[\sum_{a=1}^{i-1} C_a^{i-1}a\rho^a(1-\rho)^{i-a-1}\right](1-\rho)^{N-i} = \frac{3}{2}(i-1)\rho(1-\rho)^{N-i} \qquad (8)$$

여기서 $\left[\sum_{a=1}^{i-1} C_a^{i-1}a\rho^a(1-\rho)^{i-a-1}\right]$는 노드 i보다 상위에 위치한 노드의 평균 active 수를 의미하므로 $(i-1)\rho$와 같은 의미가 된다.

$$2\tau_i L\left[\sum_{a=1}^{i-1} C_a^{i-1}\rho^a(1-\rho)^{i-a-1}\right](1-\rho)^{N-i} = 2\tau_i L[1-(1-\rho)^{i-1}](1-\rho)^{N-i} \qquad (9)$$

여기서 $[\sum_{a=1}^{i-1} C_a^{i-1}\rho^a(1-\rho)^{i-a-1}]$는 노드 i 보다 상위에 위치한 노드들 중 적어도 한 개의 노드라도 active 일 확률이므로 $1-(1-\rho)^{i-1}$과 같다.

$$\frac{1}{2}\sum_{a=1}^{i-1} a C_a^{i-1}\rho^a(1-\rho)^{i-a-1}[1-(1-\rho)^{N+i-1}]$$

$$=\frac{1}{2}(i-1)\rho[1-(1-\rho)^{N-i+1}] \quad (10)$$

식 (8), (9), (10)을 식 (7)에 대입하면,

$$\Phi=\{\frac{3}{2}(i-1)\rho(1-\rho)^{N-i}+2\tau_i L[1-(1-\rho)^{i-1}](1-\rho)^{N-i}\}$$

$$(1-\frac{M}{N})+\{\frac{1}{2}(i-1)\rho[1-(1-\rho)^{N-i+1}]\} \quad (11)$$

4.2 제안하는 DQDB MAC 프로토콜의 전체 동작

앞에서 설명한 제안하는 DQDB MAC 프로토콜의 기본동작은 모든 노드에 제공된 부하가 같을 때 뛰어난 성능을 가지지만, 노드마다 제공된 부하를 달리했을 때는 부하가 많은 노드가 부하가 적은 노드보다 세그먼트 당 큐에서 기다리는 평균시간이 상당히 길게 되어 노드마다 큐 용량이 제한되어 있다면 부하가 많은 노드는 overflow가 발생될 수 있다. 왜냐하면 부하가 많은 노드나 부하가 적은 노드나 대역폭을 차지하는 권리가 동등하기 때문이다.

이런 문제점을 보완하고자 제안하는 DQDB의 MAC 프로토콜의 기본동작에다가 몇가지 기능을 추가하게 되었다.

4.2.1 접근 제어 필드(Access Control Field)의 정의

제안된 프로토콜을 구현하기 위해 슬롯의 ACF 필드를 확장할 필요는 없으나, 노드가 큐의 일정한 문턱치 값을 넘어서면 자신의 부하가 많다는 것을 알리기위해 high load request를 보내야 되고 또한 큐의 일정한 문턱치 값을 넘어섰다가 다시 문턱치 값 이내로 돌아오면 end high load request를 보내야하므로 이들 request를 구분해서 우선 순위등급과 함께 전송요구 필드에 나타내도록 2비트의 예약비트와 PSR 비트를 추가로 이용해야 한다. ACF의 각 비트에 대한 정의는 그림 2와 같다.

ACF 필드 (1 octet)	Segment (52 octets)

그림 2. QA 슬롯 양식
Fig. 2 Format of QA slot

여기서 PSR비트는 원래 표준 DQDB에서는 목적지 해제 방식을 수행하기 위해 만들어진 비트인데 제안하는 DQDB MAC 프로토콜에서는 각 노드의 맨 처음 보내는 request(high load request)를 위해 사용했다. 하지만 이 처음 보내는 request는 버스 A와 버스 B 양방향으로 보내기 때문에 각 노드의 RQ 카운터와 High load counter(HLC)의 합이 처음 N(전체 노드의 수)-1과 같을 때까지만 사용된다. 그러므로 각 노드의 RQ 카운터와 HLC의 합이 N-1이 된 순간부터는 PSR비트는 원래의 목적지 해제 기능을 위해 사용될 수 있다. 그림 3에서 high load request이면서 맨 처음 보내는 request가 의미하는 것은 각 노드가 처음으로 보내는 request가 high load request일 수가 있기 때문이다.

4.2.2 카운터의 동작

RQ 카운터의 동작은 제안하는 프로토콜의 기본동작에서와 같다.

각 노드의 큐가 일정한 문턱치 값을 넘어서면 high load request를 버스 B 방향으로 전송한다. 이 high load request를 카운트하는 것이 HLC(High Load Counter)이다. 그리고 각 노드의 큐가 일정한 문턱치 값을 넘어서 high load request를 보낸 후에 다시 문턱치 값 이내로 큐가 쌓이면 end high load request를 버스 B 방향으로 보낸다. 버스 B방향에서 end high load request를 볼 때마다 각 노드는 HLC의 값을 하나씩 감소시킨다. 그리고 각 노드는 자신이 high load request를 버스 B방향으로 전송할 때마다 SHLC(Self High Load Counter)의 값을 하나씩 증가시키고 버스 B방향으로 end high load request를 전송할 때마다 SHLC의 값을 하나씩 감소시킨다.

NNQ는 단지 자신의 큐에 남아있는 세그먼트의 수를 카운트한다. 즉 세그먼트를 버스 A를 통해 전송할 때마다 하나씩 감소하고 전송할 세그먼트가 자신의 큐에 도착할 때마다 하나씩 증가한다.

TCC(Threshold Control Counter)의 동작은 세그먼

트를 버스 A를 통해 전송할 때마다 하나씩 감소하고 전송할 세그먼트가 자신의 큐에 도착할 때마다 하나씩 증가시키는 것은 NNQ의 동작과 같고 이 TCC의 값이 일정한 큐의 문턱치 값을 넘어서면 high load request를 버스 B방향으로 전송하면서 TCC의 값을 문턱치 값만큼 감소시킨다. 결국 이 카운터의 동작은 어떤 노드가 자신의 큐에 일정한 문턱치 값의 배수가 쌓이게 되는 경우가 생기면 그 배수에 해당하는 만큼 high load request를 신청할 수 있다는 것이다.

나머지 카운터의 동작은 노드의 부하가 많은 경우(SHLC의 값이 0보다 큰 경우)와 노드의 부하가 적은 경우(SHLC의 값이 0인 경우)에 따라 동작이 다르기 때문에 뒤에 설명하겠다.

4.2.3 노드의 SHLC가 0보다 큰 경우의 동작

노드의 대기큐에 세그먼트가 있어서 request나 high load request를 보냈으면 버스 A방향에서 빈 슬롯이 올 때 한 번 전송하고 그리고 나서 HLC의 값을 HCD (High Load Countdown) 카운터에 복사시킨다. 이 때도 HLC의 값은 0으로 리세트되지 않고 자신의 값을 그대로 간직한다.

HCD 카운터의 값은 버스 A방향에서 빈 슬롯이 올 때마다 하나씩 감소시키다가 HCD 값이 0이 되면 다시 active 상태가 되면서 자신의 SHLC의 값을 STC (Successive Transmission Counter)에 복사하게 되고 버스 A방향에서 빈 슬롯이 오면 STC의 값이 0이 될 때까지 전송한다. 이 때 STC의 값은 전송할 때마다 하나씩 감소된다. STC의 값이 0이 될 때까지 전송하고 나서 RQ 카운터의 값을 CD 카운터에 복사한다. 이 때도 RQ 카운터의 값은 0으로 리세트되지 않고 자신의 값을 그대로 가진다. 그리고 나서 버스 A방향에서 빈 슬롯이 올 때마다 CD 카운터의 값은 하나씩 감소되다가 CD 카운터의 값이 0이 되면 다시 active 상태가 되어 위의 동작을 반복한다.

4.2.4 노드의 SHLC의 값이 0인 경우의 동작

노드의 대기큐에 세그먼트가 있어서 request를 보냈으면 버스 A방향에서 빈 슬롯이 올 때 한 번 전송하고나서 RQ 카운터의 값과 HLC의 값의 합을 CD 카운터에 복사한다. 이 때 RQ 카운터의 값과 HLC의 값은 0으로 리세트되지않고 자신의 값을 그대로 가진다.

이 때, CD 카운터의 값은 버스 A방향으로 빈 슬롯이 올 때마다 하나씩 감소되다가 0이 되면 이 노드는 다시 active상태가 되어 위의 동작을 반복한다.

4.2.5 제안하는 DQDB MAC 프로토콜의 문제점과 해결 방안

제안하는 DQDB MAC 프로토콜의 전체적인 동작에서의 문제점은 각 노드에 제공된 부하가 아주 많을 때 어떤 한 노드가 다른 노드에 비해 늦게 active되었을 경우, 먼저 active된 노드는 이미 제공된 부하가 많기 때문에 부하가 많을 때의 노드 동작을 하고 있게 된다. 즉, 먼저 active된 노드는 제공된 부하만큼 대역폭을 쓰고 있게된다. 이런 경우, 늦게 active된 노드는 초기에는 부하가 적을 때의 노드 동작을 하게 되므로 정상상태 즉, 모든 노드가 대역폭을 공평하게 쓰는 상태에 도달하기가 힘들게 된다. 이런 문제점을 해결하기위해 제안하는 DQDB MAC 프로토콜에 다음의 동작을 추가한다.

어떤 노드가 맨 처음 active될 때까지는 그 노드의 HLC(High Load Counter)는 작동해서는 안된다. 왜냐하면, 자신보다 하위에 위치한 노드가 자신보다 먼저 active되고, 제공된 부하가 많고 자신은 아직 active되지 않았다면, 그 하위에 위치한 노드는 아직 active되지 않은 상위 노드에 high load request를 버스 B상으로 보냈을 것이다.

그렇게 되면 active되지 않은 상위 노드가 active 되었을 경우 초기에는 부하가 적을 때의 노드동작을 하게 되고, 이미 그 노드의 HLC는 0보다 크게 되므로 늦게 active된 상위 노드는 먼저 active된 하위 노드에 비해 대역폭을 차지할 수 있는 기회가 적어진다. 그러므로 어떤 노드가 맨 처음 active될 때까지는 그 노드의 HLC는 작동해서는 안된다.

그리고, 각 노드는 자신의 맨 처음 보내는 request에 대해서는 버스 A방향과 버스 B방향으로 동시에 전송한다. 게다가 다른 노드가 보낸 맨 처음 request를 본 노드는 자신의 SHLC(Self High Load Counter), TCC(Threshold Control Counter), 그리고 HLC(High Load Counter)를 0으로 리세트시킨다. 그리고 버스 B방향에서 온 맨 처음 request에 대해서는 자신의 RQ 카운터를 하나 증가시킨다.

그리고, 각 노드의 맨 처음 보내는 request가 high

⇐ ACF 필드 ⇒

B	S_T	PSR	예약	예약	Req 2	Req 1	Req 0	정 의
0	0	0	0	0	0	0	0	빈 슬롯
0	0	0	0	0	0	0	1	request
0	0	0	0	1	0	0	1	end request
0	0	0	1	0	0	0	1	end high load request
0	0	0	1	1	0	0	1	high load request
0	0	1	1	1	0	0	1	high load request이면서 맨 처음request
0	0	1	0	0	0	0	1	맨 처음 request

그림 3. ACF(Access Control Field)의 정의
Fig. 3 Definition of ACF(Access Control Field)

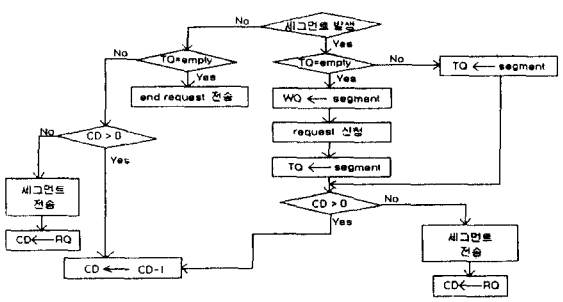


그림 4. 제안하는 DQDB MAC 프로토콜의 기본동작 순서도
Fig. 4 Basic operation flow-chart of proposed DQDB MAC protocol

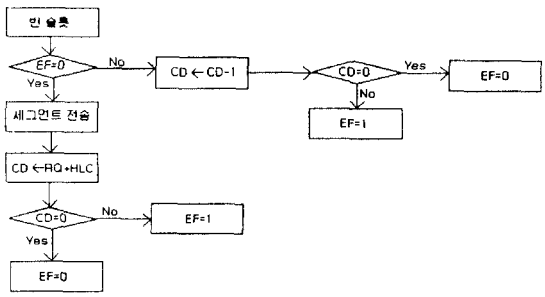


그림 5. 제안하는 DQDB MAC 프로토콜의 전송동작(SHLC = 0)
Fig. 5 Transmission operation of proposed DQDB MAC protocol in case of SHLC = 0

load request일 수도 있다. 이런 경우에도 high load request를 버스 A와 버스 B로 동시에 전송한다. 이 high load request를 본 다른 노드는 자신의 HLC (High Load Counter)를 1로 만들고 이 때, 자신의 SHLC(Self High Load Counter)가 0보다 크면 이것 또한 1로 만든다. 즉, 이런 동작도 늦게 active된 노드가 먼저 active된 노드와 동등하게 대역폭을 차지할 수 있게 하기위함이다.

4.2.6 버퍼의 적정 문턱치 값(TH)을 구하는 방법

제안하는 DQDB MAC 프로토콜에서 어떤 방식으로 버퍼의 적정 문턱치 값(TH)을 정하느냐는 매우 중요하다. 우선 버퍼의 적정 문턱치 값에 영향을 미치는 파라미터에 대해 알아보자.

N : DQDB 망의 전체 노드의 수
D : DQDB 망의 왕복 전파지연 시간 (단위 : slot time)
d : DQDB 망의 노드간의 평균 전파지연 시간 (단위 : slot time)

만약 버퍼의 문턱치 값(TH)을 너무 크게 잡으면 부하가 많은 노드는 부하가 적은 노드보다 전송하는데 걸리는 시간이 그 만큼 많아지고 버퍼의 문턱치 값(TH)을 너무 적게 잡으면 부하가 적은 노드가 부하가

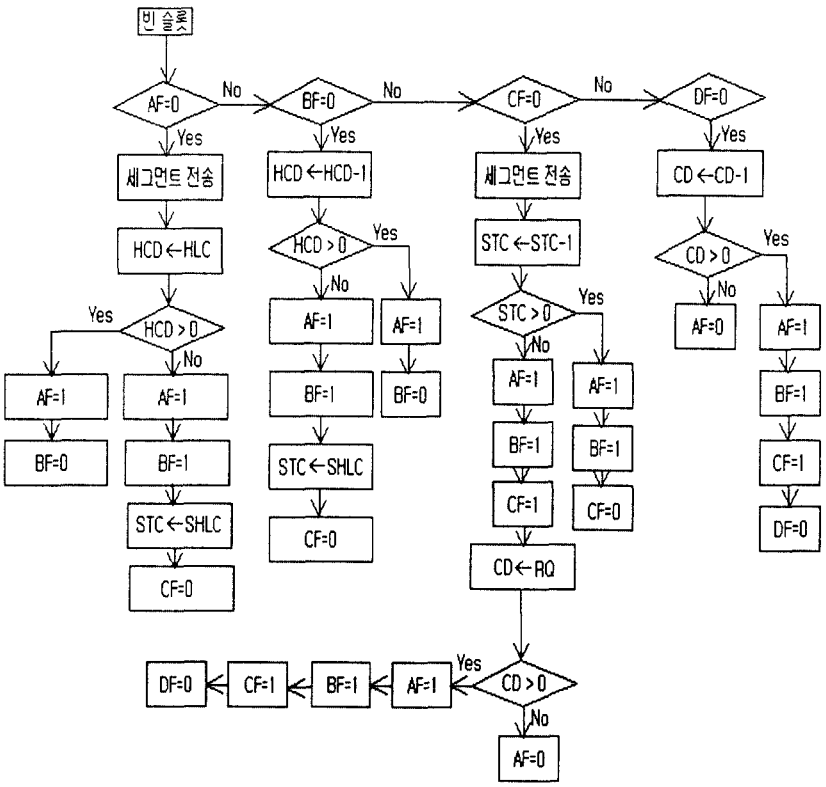


그림 6. 제안하는 DQDB MAC 프로토콜의 전송동작(SHLC > 0)

Fig. 6 Transmission operation of proposed DQDB MAC protocol in case of SHLC > 0

많은 노드보다 전송하는데 걸리는 시간이 더 많아진다.

그러므로 버퍼의 적정 문턱치 값(TH)을 정하는 것은 중요하다. 버퍼의 문턱치 값은 D가 클수록 커져야 한다. 왜냐하면 D가 클수록 버퍼에 많이 쌓이기 때문이다.

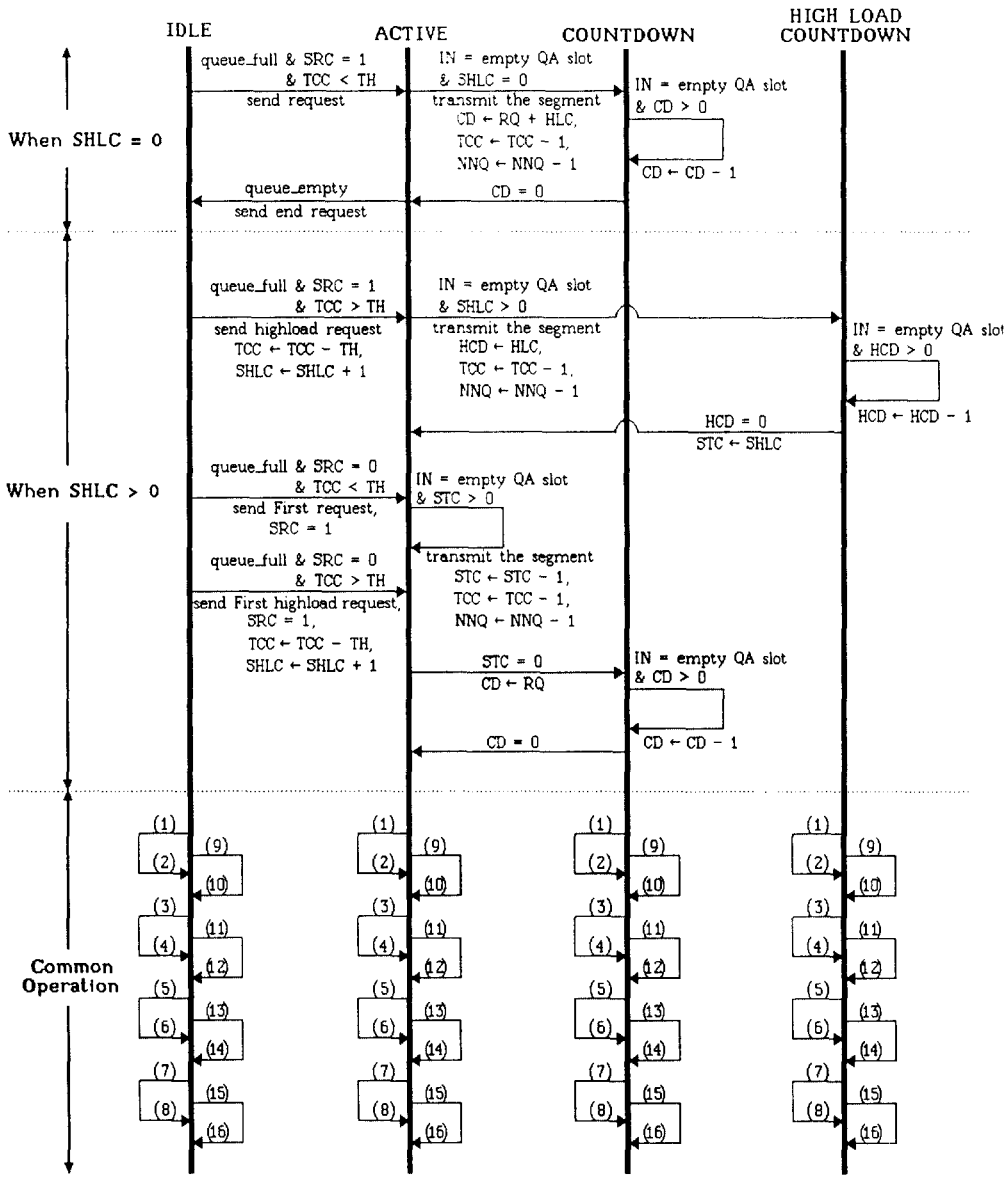
DQDB망의 왕복 전파지연 시간에 따라 적정 문턱치값을 달리 하면서 시뮬레이션을 한 결과 다음의 결과를 얻었다.

$d \le 2$이면 $\quad TH = \frac{D}{2}$, (소수점이하 내림) (12)

$2 < d < 30$이면 $\quad TH = \frac{N(d+10)}{7}$, (소수점이하 내림) (13)

$d \ge 30$이면 $\quad TH = \frac{N(d+20)}{7}$, (소수점이하 내림) (14)

위의 문턱치 값을 구하는 식은 노드의 수를 달리하면서 여러번의 시뮬레이션을 통한 최적의 문턱치 값을 구하는 식으로 최적의 문턱치 값들은 DQDB망의 길이와 노드간의 평균 길이에 따라 최적의 값들이 변화되는 것을 볼 수 있었다. 그래서 DQDB망의 전체 부하가 1일때를 기준으로해서 망의 길이를 달리하고 노드간의 평균 길이를 달리 했을 때의 모든 경우에 적용할 수 있게 만든 식입니다. 그리고 최악의 경우 각 노드의 버퍼는 Request를 신청해서 상위노드에 알리는 데 걸리는 전파지연시간 (즉, DQDB망의 왕복



(1) IN = Req_B
(2) RQ ← RQ + 1
(3) IN = end Req_B
(4) RQ ← RQ - 1
(5) IN = highload Req_B
(6) HLC ← HLC + 1
(7) IN = end highload Req_B
(8) HLC ← HLC - 1
(9) NNQ < SHLC X TH
(10) send end highload Req
SHLC ← SHLC - 1
(11) IN = First highload Req_B
SHLC > 0
(12) HLC ← 1, SHLC ← 1
(13) IN = First Req_B
(14) RQ ← RQ + 1
SHLC = TCC = HLC = 0
(15) IN = First highload Req_B
SHLC = 0
(16) HLC ← 1

그림 7. 제안하는 DQDB MAC 프로토콜의 state machine
Fig. 7 State machine of proposed DQDB MAC protocol

길이)만큼 쌓일 수 있다는 생각에서 문턱치 값을 구하는 식의 파라미터로 DQDB망의 전체길이와 노드간의 평균 길이를 사용했다.

지금까지 설명한 제안하는 DQDB MAC 프로토콜의 전체적인 동작을 state machine으로 그리면 그림 7과 같다.

V. 모의 실험 및 고찰

본 장에서는 본 논문에서 제안하는 새로운 분산 큐 액세스 프로토콜을 이용한 DQDB망의 성능분석을 위해 SLAM Ⅱ를 이용하여 모의실험을 수행하였으며, 제안된 프로토콜의 성능 평가를 위하여 기존의 DQDB 프로토콜, BWB DQDB 프로토콜, SIMA 프로토콜[12]과 모의실험을 통해 동일 조건하에서 비교하였다.

우선순위를 고려할 경우에도 본 논문에서 제안하는 새로운 분산 큐 액세스 프로토콜은 우선순위별로 똑같은 카운터들이 존재하기 때문에 우선순위 하나만을 생각하는 저의 제안하는 DQDB MAC 프로토콜이 각 우선순위별로 똑같이 동작하기 때문에 우선순위를 고려했을 때도 무리없이 동작할 수 있다고 보여진다. 그래서 우선순위 하나만을 고려해서 모의실험을 수행하였다.

그리고 그림 7은 제안하는 DQDB MAC 프로토콜의 전체 알고리즘을 state machine으로 표현한 것이다. 여기서 IDLE 상태란 노드의 큐에 전송할 세그먼트가 없어서 아직 request를 신청하지 않은 상태를 의미하며, ACTIVE 상태란 노드의 큐에 전송할 세그먼트가 있어서 request를 이미 신청했고 버스 A방향에서 오는 빈 슬롯을 차지해서 전송할 수 있는 상태를 의미한다. 그리고 COUNTDOWN 상태란 버스 A 방향에서 오는 빈 슬롯을 request를 신청한 하위 노드에게 양보하는 상태를 의미하며, HIGH LOAD COUNTDOWN 상태란 버스 A 방향에서 오는 빈 슬롯을 high load request를 신청한 하위 노드에게 양보하는 상태를 의미한다.

성능 비교를 위해 망 성능에 가장 큰 영향을 미치는 노드에서의 전송지연 시간을 성능측정요소로 설정하였으며, 노드 사이의 거리를 달리하면서 각 노드의 대기큐와 전송큐에서 발생되는 지연시간의 합을 측정하였다.

모의 실험에 사용된 기본적인 가정은 다음과 같다.

- 전송 속도는 155.520 Mbps로 하였다.
- 노드의 수는 3개일 때와 7개일 때를 비교하였고 각 노드는 등간격으로 분포되었다.
- 데이타 세그먼트는 각각의 노드에서 포아송 분포로 발생되며 다른 노드의 발생과 독립적이다.

5.1 모의실험 결과검토

본 장에서는 제안하는 프로토콜과 기존 프로토콜의 성능 비교를 위하여 노드에서 발생되는 전송지연 시간을 측정하였다.

표 1은 BWB_MOD의 값을 8로 했을때의 BWB DQDB 프로토콜과 SIMA 프로토콜 그리고, 제안하는 DQDB MAC 프로토콜을 모의실험을 통해 비교한 결과를 나타내었다. 이 모의실험에서는 노드의 수를 3으로 하고 노드간의 거리가 40slot time으로 할 때의 각 노드에 제공된 부하에 따른 throughput을 나타내었다.

표 1에서 보면 BWB DQDB는 부하가 적은 노드는 대역폭을 다 차지하고 부하가 많은 노드는 부하가 적은 노드가 쓰다 남은 대역폭을 똑같이 나누어서 차지함을 보여주여주고 있으며 전체 throughput이 1 이하임을 알 수 있다. 즉, 대역폭 손실이 발생한다. 그리고 SIMA 프로토콜[12]은 대역폭 손실없이 부하가 많은

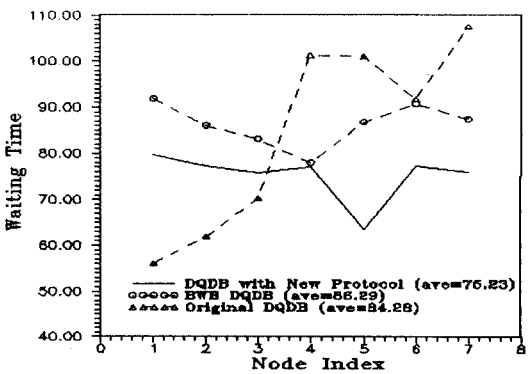


그림 8. $\rho=1$, d=40slot, 노드사이의부하가 같을 때의 DQDB망의 노드에서 겪는 큐에서의 전송지연

Fig. 8 Transmission delay in queue of node($\rho=1$, d=40slot, uniform load)

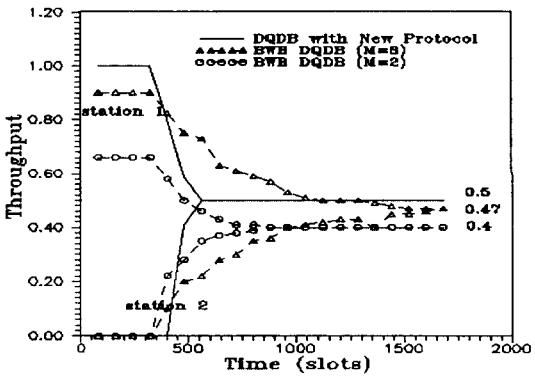


그림 9. N = 2, d = 40slot, 2번 노드가 1번 노드보다 늦게 active될 때, Throughput 성능 분석

Fig. 9 Throughput performance analysis (N = 2, d = 40slot)

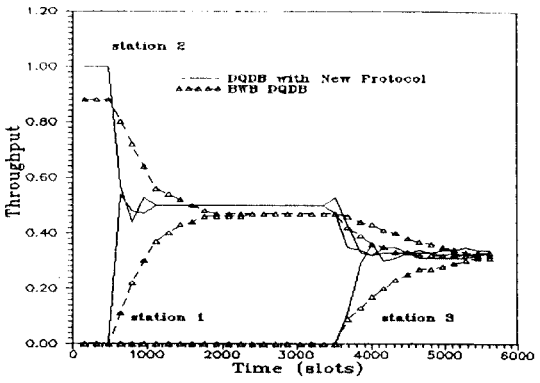


그림 10. N = 3, d = 40slot, active 순서가 2, 1, 3번 노드일 때의 Throughput 성능 분석

Fig. 10 Throughput performance analysis(N = 3, d = 40slot)

노드는 부하가 적은 노드가 쓰다 남은 대역폭을 똑같이 나누어서 차지함을 보여준다. 반면에 제안하는 DQDB는 대역폭 손실이 없고 노드에 제공된 부하에 비례하여 대역폭을 차지하게 됨을 알 수 있다.

그림 8은 노드 사이의 거리가 긴 경우, 표준 DQDB는 하위 노드로 갈수록 세그먼트 당 평균 기다리는 시간이 길다는 것을 보여준다. 결국 표준 DQDB는 노드 사이의 거리가 길거나 제공된 부하가 많을 경우, 불공평성이 생김을 알 수 있다. 표준 DQDB는 세그먼트별로 상위노드에게 자신이 전송할 것임을 알리므로 노드 사이의 거리가 길수록 request를 상위노드에게 알리는데 전파지연 시간이 있으므로 상위 노드보다 대역폭을 차지하는데 불리하게 된다.

그림 9는 노드의 수가 2일 때 어느 한 노드가 다른 노드보다 먼저 active되었을 경우, 늦게 active된 노드가 먼저 active된 노드와 똑같은 throughput을 얻는데 걸리는 시간을 보여준다. 여기서 보면, BWB DQDB는 BWB_MOD의 값이 8일 경우는 throughput이 0.47에서 두 노드가 똑같은 대역폭을 얻었고 BWB_MOD의 값이 2일 경우는 throughput이 0.4에서 두 노드가 똑같은 대역폭을 얻었다. 이것은 BWB_MOD의 값이 작을수록 대역폭 손실이 많다는 것을 보여준다. 그리고, BWB_MOD의 값이 2인 경우가 BWB_MOD의 값이 8인 경우보다 두 노드가 똑같은 대역폭을 얻는데 걸리는 시간이 짧다는 것을 보여주고 있다. 즉, BWB_MOD의 값이 작을수록 정상 상태에 도달하는 시간이 짧음을 알 수 있다.

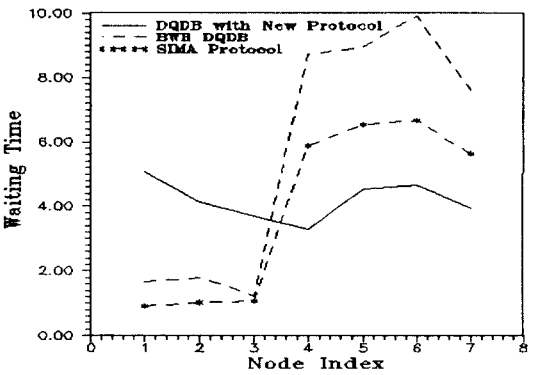


그림 11. $\rho = 1$, d = 2slot, $\rho_1 = \rho_2 = \rho_3 = 0.1$, $\rho_4 = \rho_5 = \rho_6 = \rho_7 = 0.175$일 때의 DQDB 망의 큐에서의 전송 지연

Fig. 11 Transmission delay analysis($\rho = 1$, d = 2slot, $\rho_1 = \rho_2 = \rho_3 = 0.1$, $\rho_4 = \rho_5 = \rho_6 = \rho_7 = 0.175$)

결론적으로, BWB DQDB의 경우, BWB_MOD의 값이 작을수록 정상 상태에 도달하는 시간은 짧으나 대역폭 손실이 많고, BWB_MOD의 값이 클수록 대역폭 손실은 적으나 정상 상태에 도달하는 시간이 길게 된다. 하지만 그림 9에서 보듯이 제안하는 DQDB는 노드마다 제공된 부하가 같은 경우, active된 노드들은 똑같은 대역폭을 차지하게 되므로 빠른 시간에 정상 상태에 도달할뿐아니라 대역폭 손실도 없음을 알 수 있다. 왜냐하면 제안하는 DQDB MAC 프로토

콜은 active된 노드들사이에 서로 골고루 대역폭을 차지할 뿐아니라 BWB DQDB처럼 BWB_MOD값만큼 전송한 다음 무조건 하나의 빈 슬롯을 하위 노드에게 양보하지 않으므로 대역폭 손실도 생기지 않기 때문이다. 그리고 2번 노드가 늦게 active되더라도 request를 1번 노드에 알리는데 걸리는 왕복 전파지연시간내에 바로 정상상태에 도달함을 알 수 있다.

그림 10은 노드의 수가 3이고 2, 1, 3번 노드 순서로 active될 때 각 노드가 정상 상태에 도달하는 시간을 보여주고 있다. 이 경우에도 그림 9에서 설명했듯이 제안하는 DQDB MAC 프로토콜은 대역폭 손실없이 빠른 시간안에 정상상태에 도달함을 알 수 있다.

그림 11은 어떤 노드는 부하를 많이 주고, 다른 노드는 부하를 적게 주었을 경우의 전송지연시간을 보여주고 있다. 여기서 보면 BWB DQDB와 SIMA 프로토콜[12]은 부하가 많은 노드가 부하가 적은 노드보다 세그먼트당 큐에서 기다리는 시간이 많음을 알 수 있다. 왜냐하면, BWB DQDB와 SIMA 프로토콜[12]은 부하가 많은 노드나 적은 노드나 대역폭을 차지하는 권리가 동등하기 때문이다. 하지만 제안하는 DQDB의 경우는 노드에 제공된 부하에 비례하여 대역폭을 차지할 수 있기 때문에 부하가 적은 노드나 부하가 많은 노드나 세그먼트당 큐에서 기다리는 시간이 비슷하게 나옴을 알 수 있다.

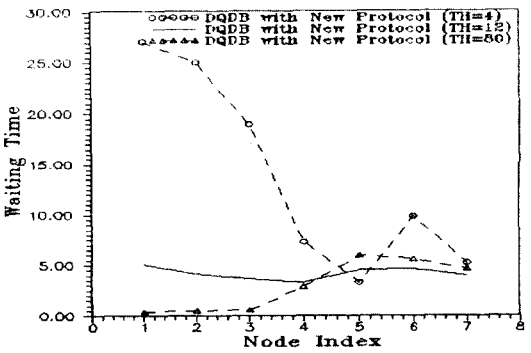


그림 12. N = 7, d = 2slot, $\rho_1 = \rho_2 = \rho_3 = 0.1$, $\rho_4 = \rho_5 = \rho_6 = \rho_7 = 0.175$일 때의 TH값에 따른 큐에서의 전송지연 분석

Fig. 12 Transmission delay analysis in queue as TH value is changed(N = 7, d = 2slot, $\rho_1 = \rho_2 = \rho_3 = 0.1$, $\rho_4 = \rho_5 = \rho_6 = \rho_7 = 0.175$)

그림 12는 어떤 노드는 부하를 많이 주고, 다른 노드는 부하를 적게 주었을 경우, 제안하는 DQDB의 TH값에 따른 전송 지연 시간을 분석하였다. 이 그림에서 보면, TH값을 너무 크게 잡으면, 부하가 적은 노드가 부하가 많은 노드보다 전송 지연 시간이 짧고, TH값을 너무 작게 잡으면, 부하가 적은 노드가 부하가 많은 노드보다 전송 지연 시간이 길게 됨을 알 수 있다. 즉, TH값이 클수록 부하가 많은 노드는 highload re-

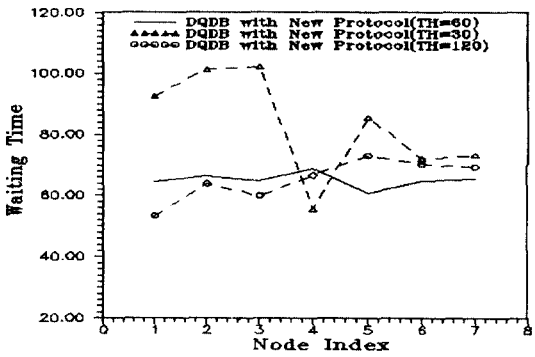


그림 13. N = 7, d = 40slot, $\rho_1 = \rho_2 = \rho_3 = 0.1$, $\rho_4 = \rho_5 = \rho_6 = \rho_7 = 0.175$일 때의 TH값에 따른 큐에서의 전송지연 분석

Fig. 13 Transmission delay analysis in queue as TH value is changed.(N = 7, d = 40slot, $\rho_1 = \rho_2 = \rho_3 = 0.1$, $\rho_4 = \rho_5 = \rho_6 = \rho_7 = 0.175$)

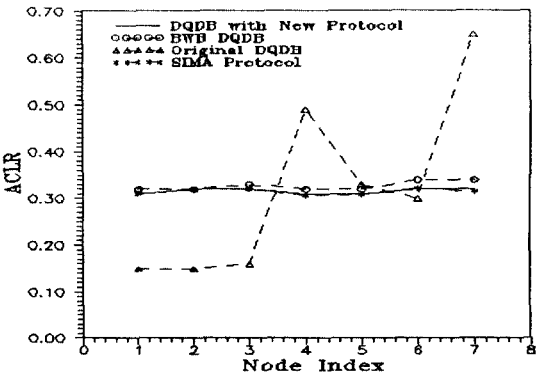


그림 14. $\rho = 1.5$, d = 20slot, B = 500, 노드사이의 부하가 모두 같은 경우, 노드마다의 ACLR 분석

Fig. 14 ACLR analysis in each station($\rho = 1.5$, d = 20slot, B = 500, uniform load)

quest를 전송할 기회가 적어지므로 부하가 적은 노드의 전송 지연 시간이 짧게되고, TH값이 작을수록 부하가 많은 노드는 highload request를 전송할 기회가 많아지게 되므로 부하가 적은 노드의 전송 지연 시간이 길어지게 되는 것이다. 그렇기 때문에 최적의 TH값을 정하는 것이 중요하다.

그림 12의 경우, 노드의 수(N)가 7이고 노드간의 평균 거리(d)가 2slot이고, 전체 망의 왕복 거리(D)가 24 slot이므로 앞에서 설명한 식 (12)에 대입하면 최적의 TH값은 12가 된다. 그림 12에서도 TH값이 12일 때 가장 좋은 성능을 보여주고 있다.

그림 13은 N=7, d=40slot, D=480slot인 경우, DQDB망이 가장 좋은 성능을 갖는 최적의 TH값을 보여주고 있다. d가 40 슬롯이므로 앞에서 설명한 식 (14)에 대입하면 최적의 TH값은 60이 된다. 여기서는 그림 12의 경우보다 노드간의 평균 길이가 길기 때문에 최악의 경우 노드 사이의 전파지연시간 때문에 각 노드의 큐에 많이 쌓이게되므로 최적의 TH값도 커지게된다.

그림 14에서 보면 표준 DQDB는 하위 노드일수록 ACLR(Average Cell Loss Rate)이 높지만 BWB DQDB와 SIMA 프로토콜[12]은 제안하는 DQDB에서 처럼 노드의 위치에 관계없이 ACLR이 비슷함을 알 수 있다. 즉, BWB DQDB와 SIMA 프로토콜[12]은 노드마다 똑같은 부하를 제공할 경우는 좋은 성능을 보여주고 있음을 알 수 있다. 왜냐하면 BWB DQDB와 SIMA 프로토콜은 노드마다 부하가 같을 때는 공평하게 대역폭을 차지하게되므로 제안하는 프로토콜과 거의 차이가 나지않고 좋은 성능을 나타낸다. 그렇지만 표준 DQDB는 전파지연 때문에 하위노드가 상위노드보다 대역폭을 차지하는데 불리하기 때문에 하위노드일수록 ACLR이 높게 나타난다.

그림 15에서 보면 버퍼 크기가 300일 경우, BWB DQDB, 표준 DQDB 그리고 SIMA 프로토콜[12]은 부하가 많은 노드가 셀 탈락율이 높지만 제안하는 DQDB는 탈락된 셀이 없음을 알 수 있다. 제안하는 DQDB는 부하에 비례하여 대역폭을 차지할 수 있기 때문에 부하가 많은 노드는 그만큼 대역폭을 차지할 수 있는 기회가 많아지므로 부하가 많은 노드에서 셀 탈락이 일어나지않음을 알 수 있다. 그리고 노드마다 부하를 다르게 주었을 경우, BWB DQDB와 SIMA

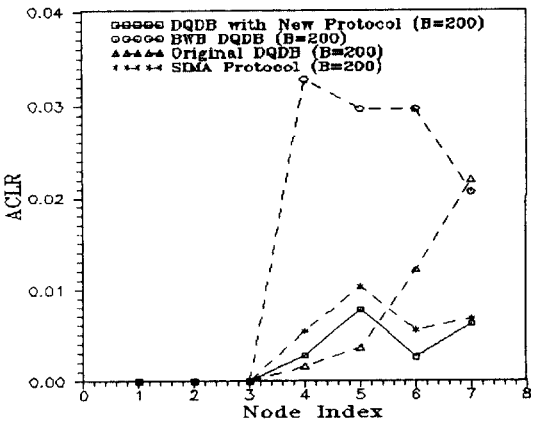


그림 15. d=2slot, B=300, $\rho_1=\rho_2=\rho_3=0.1$, $\rho_4=\rho_5=\rho_6=\rho_7=0.175$일 때 각 노드의 ACLR 분석

Fig. 15 ACLR analysis in each node(d=20slot, B=300, $\rho_1=\rho_2=\rho_3=0.1$, $\rho_4=\rho_5=\rho_6=\rho_7=0.175$)

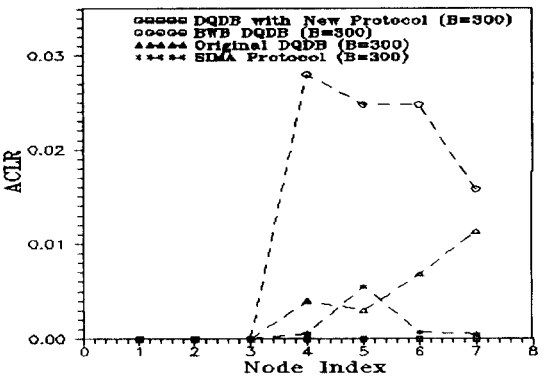


그림 16. d=20slot, B=300, $\rho_1=\rho_2=\rho_3=0.1$, $\rho_4=\rho_5=\rho_6=\rho_7=0.175$일 때 노드마다의 ACLR 분석

Fig. 16 ACLR analysis in each node(d=20slot, B=300, $\rho_1=\rho_2=\rho_3=0.1$, $\rho_4=\rho_5=\rho_6=\rho_7=0.175$)

프로토콜[12]의 성능은 안 좋게 나타남을 알 수 있다.

그림 16에서는 버퍼 크기가 200일 때의 ACLR(Average Cell Loss Rate)을 보여주고 있다. 그림에서 보듯이 제안하는 DQDB의 셀 탈락율은 표준 DQDB, BWB DQDB, SIMA 프로토콜[12]에 비해 적다는 것을 알 수 있다. 여기서도 제안하는 DQDB는 부하에 비례하여 대역폭을 차지할 수 있기 때문에 부하가 많은 노드에서의 셀 탈락율이 적어지게 됨을 알 수 있다.

Ⅵ. 결 론

본 논문에서는 DQDB 망의 공평성 개선을 위한 새로운 DQDB MAC 프로토콜을 제시하였다. 지금까지의 공평성 개선을 위한 논문들을 보면 대부분 노드마다 부하를 같게 주었을 때 대역폭을 공평하게 차지하는데 초점을 맞추었다. 하지만 실제 DQDB 망이 backbone 망으로 쓰였을 경우, DQDB 망의 각 노드에는 여러 개의 LAN(Local Area Network)이 연결되어있게 되므로 각 노드의 부하는 예측할 수 없다. 즉, 어떤 노드는 부하가 많을 수 있고 또 어떤 노드는 부하가 적을 수 있다. 그러므로, 본 논문에서는 부하가 많은 노드이든지 부하가 적은 노드이든지간에, 각 노드에 들어오는 세그먼트가 전송되기까지 큐에서 기다리는 평균 시간을 노드마다 어떻게 비슷하게 만드느냐에 촛점을 맞추었다.

본 논문에서 제안한 DQDB MAC 프로토콜에서는 노드마다 세그먼트당 큐에서 기다리는 시간을 비슷하게 하기위해서 노드에 제공된 부하에 비례하여 대역폭을 차지할 수 있게 하였다. 그렇게 함으로써 어떤 노드에 혼잡(congestion)이 발생했을 때 그 노드는 혼잡을 막거나 최소한으로 줄이기위해 더 많은 대역폭을 얻게 된다.

기존의 표준 DQDB와 BWB DQDB, 그리고 SIMA 프로토콜[12]에 비해 본 논문에서 제안한 DQDB MAC 프로토콜은 세그먼트 당 큐에서 기다리는 시간, ACLR (Average Cell Loss Rate)과 같은 성능에서 더 우수할 뿐 아니라 부하가 같은 노드들 사이에는 대역폭 손실 없이 빠른 시간안에 정상상태에 도달하기도 한다.

향후에는 우선 순위를 갖는 DQDB MAC 프로토콜에서의 대역폭 점유문제와 우선권 점유 문제에 대해 연구를 지속해야 할것이다.

참 고 문 헌

1. IEEE 802.6 Working Group, "Proposed standard: Distributed queue dual bus(DQDB) metropolitan area network," Unapproved Draft D9, Aug. 1989.
2. M. Conti et al., "A methodological approach to an extensive analysis of DQDB performance and fairness," IEEE J. Select. Areas Commun., vol. SAC-9, no. 1, pp. 76-87, Jan. 1991.
3. M. Conti et al., "DQDB under heavy load: Performance evaluation and unfairness analysis," in Proc. INFOCOM'90, San Francisco, CA, June 5-7, 1990, pp. 313-320.
4. D. Davids and T. Welzel, "Performance analysis of DQDB based on simulation," in Proc, 3rd IEEE Workshop Metropol. Area Netw., San Diego, CA, Mar. 28-30, 1989, pp. 7.6.1-7.6.15.
5. J. W. Wong, "Throughput of DQDB networks under heavy load," in Proc. 7th Europ. Fibre Optic Commun. Local Area Netw. Expo., Amsterdam, The Netherlands, June 1989, pp. 146-151.
6. M. Zukerman, "On packet switching capacity in QPSX," in Proc. IEEE/IEICE Global Telecommun. Conf., Tokyo, Japan, Nov. 15-18, 1987, pp. 45.2. 1-45.2.5.
7. D. Karvelas and M. Papamichail, "A simulation model for DQDB," in Proc. Twenty-Second Ann. Pittsburgh Conf. Modeling and simulat., Pittsburgh, PA, May 1991, pp. 2265-2272.
8. C.C. Bisdikian, "Waiting time analysis in a single buffer DQDB(802.6) network," IEEE J. Select. Areas Commun., vol. SAC-8, no. 8, pp. 1565-1573, Oct. 1990.
9. E. L. Hahne et al., "Improving the fairness of distributed queue dual bus networks," in Proc. INFOCOM'90, San Francisco, CA, June 1990, pp. 175-184.
10. D. Karvelas and M. Papamichail, "No Slot Wasting Bandwidth Balancing with Immediate use of TAR bit", ICC'94, pp. 968-972.
11. D. Karvelas and M. Papamichail, "The No Slot Wasting Bandwidth Balancing Mechanism for Dual bus Architectures", IEEE Journal on selected areas in commun., vol. 11, no. 8, Oct. 1993, pp. 1214-1228.
12. Nen-Fu Huang and Shiann-Tsong Sheu, "A slot Interleaved Multiple Access Scheme for DQDB Metropolitan Area Networks", Procceedings of IEEE INFOCOM'93, pp. 1075-1082.

高 成 賢(Sung-Hyun Ko) 정회원

1969년 8월 20일생

1989년 3월~1995년 2월 : 중앙대학교 공과대학 전자공학과 졸업(공학사)

1995년 3월~1997년 2월 : 중앙대학교 대학원 전자공학과 졸업(공학석사)

1997년 1월~현재 : 현대전자 통신연구소 연구원

※주관심분야 : 통신망 성능분석, 망 연동, 멀티미디어 통신 등

徐 鎭 敎(Jin-Kyo Seo) 정회원

1963년 11월 14일생

1983년 3월~1989년 2월 : 중앙대학교 공과대학 전자공학과 졸업(공학사)

1989년 3월~1991년 2월 : 중앙대학교 공과대학 전자공학과 졸업(공학석사)

1992년 3월~1997년 2월 : 중앙대학교 공과대학 전자공학과 졸업(공학박사)

1997년 3월~현재 : 양산전문대학 전자통신과 전임강사

※주관심분야 : 통신망 성능분석, 망 연동, 멀티미디어 통신 등

金 峻 年(Joon-Nyun Kim) 정회원

1954년 10월 14일생

1974년 3월~1978년 2월 : 서울대학교 공과대학 전자공학과 졸업(공학사)

1978년 10월~1980년 10월 : 대영전자(주) 연구원

1981년 2월~1986년 7월 : 아이오와 주립대학 컴퓨터공학과 졸업(공학석사)

1986년 7월~1987년 12월 : 아이오와 주립대학 컴퓨터공학과 졸업(공학박사)

1988년 3월~현재 : 중앙대학교 전자공학과 교수

1993년 1월~1996년 12월 : 한국통신학회 데이타 통신망 연구회 전문위원장

1997년 1월~현재 : 대한전자공학회 통신연구회 전문위원장

1993년 11월~현재 : ISO/IEC JTC1/SC6 WG1(Data Link 계층) 위원장(Convener)

※주관심분야 : 통신망 성능분석, 멀티미디어 통신, 고속 근거리망 등