공유 메모리형 패킷 교환기의 QoS 기능 지원을 위한 가중형 동적 임계치를 이용한 버퍼 관리기법에 관한 연구

A New Buffer Management Scheme using Weighted Dynamic Threshold for QoS Support in Fast Packet Switches with Shared Memories

김 창 원*, 김 영 범*

Chang-Won Kim*, Young-Beom Kim*

요 약

공유 메모리 관리를 위한 기존의 방법들은 가상 큐 길이의 정적 제한을 통해 일정 크기의 버퍼 할당을 보장하려는 방식과 전체 버퍼 공간의 할당 측면에서 공유 버퍼의 이용률을 높이고자 하는 방식 등으로 나눌 수 있다. 완전공유 방식의 경우 낮은 트래픽 부하에서 높은 메모리 공유 효과를 보이나 트래픽 부하가 높아지면 특정 가상큐가 공유 메모리를 과다하게 점유하는 것을 방지하는 의미에서의 보호 효과를 거의 기대하기 힘들다. 반대로 정적 임계치 방식의 경우 트래픽 조건 변화에 따른 적절한 임계치 설정이 불가능하다. 본 논문에서는 공유 메모리의 공정 할당이라는 단순한 기능을 가지는 동적 임계치 방식을 확장하여 구현이 용이하고 높은 메모리 이용률과 서비스 품질기능 측면에서 우선 순위에 따른 차등적인 패킷처리 기능을 갖는 가중형 동적 임계치 방식을 제안하고 컴퓨터 시뮬레이션을 통하여 그 성능을 확인하였다.

Abstract

Existing buffer management schemes for shared-memory output queueing switches can be classified into two types: In the first type, some constant amount of memory space is guaranteed to each virtual queue using static queue thresholds. The static threshold method (ST) belongs to this type. On the other hand, the second type of approach tries to maximize the buffer utilization in allocating buffer memories. The complete sharing (CS) method is classified into this type. In the case of CS, it is very hard to protect regular traffic from mis-behaving traffic flows while in the case of ST the thresholds can not be adjusted according to varying traffic conditions. In this paper, we propose a new buffer management method called weighted dynamic thresholds (WDT) which can process packet flows based on loss priorities for quality-of-service (QoS) functionalities with fairly high memory utilization factors. We verified the performance of the proposed scheme through computer simulations.

Keywords : Packet Switches, Buffer Management, Shared Memory, QoS Support, Static Thresholds, Packet Losses

I. 서 론

최근 수년간 인터넷 사용자의 급격한 증가와 광대역 전송 서비스를 필요로 하는 멀티미디어 애플리케이션이 지속적으로 등장함에 따라 광대역 멀티미디어 트래픽을 효율적으로 처리하기 위한 고속 라우터 및 ATM 교환기 구조에 관한 연구가 활발하게 진행되고 있다.

대부분의 패킷 교환기 구조에 있어서 패킷 방식의 특성상 스위치 내 자원에 대한 패킷 간 경쟁이 불가피하며 처리가 지연되는 패킷을 일시적으로 저장하기 위한 메모리 공간이 필요하다. 이러한 버퍼공간의 스위치 내 위치와 버퍼 관리 기법에 따라 패킷교환기의 성능은 크게 좌우된다. 스위치의 입력단에 버퍼를 두는 입력 큐잉, 출력단에 버퍼를 두는 출력 큐잉 등의 스위치 구조가 제안되

*건국대학교 전자공학부
논문 번호 : 2006-2-25 접수 일자 : 2006. 4. 17
심사 완료 : 2006. 7. 24

었으며 처리율 및 처리 지연시간 측면에 있어서 공유 메모리를 갖는 출력 큐잉 교환기가 최적의 성능을 보이는 것으로 알려져 있다 [1]. 공유 메모리형 출력 큐잉 교환기에서는 출력 포트별로 버퍼를 따로 두기보다는 전체 메모리 공간을 출력 포트간, 그리고 트래픽 클래스 간에 공유하도록 함으로써 메모리 공간의 이용률을 높이고 낮은 패킷 유실률과 높은 처리율 (throughput)을 달성할 수 있도록 하고 있다.

그러나 공유 메모리형 패킷 교환기의 경우 적절한 버퍼 관리 기법이 적용되지 않으면 높은 부하의 트래픽 조건하에서 상당한 성능저하가 초래될 수 있다. 특정 출력포트로 향하는 패킷들이 공유 메모리의 대부분을 차지하게 되는 경우 상대적으로 이용률이 낮은 출력포트로 향하는 패킷들은 메모리를 이용할 기회가 줄어들게 되어 전체적인 패킷 처리율이 떨어지게 된다. 따라서 특정 큐가 전체 버퍼를 과도하게 독점하지 않도록 통제하기 위한 적절한 버퍼관리 기법이 필요하다.

공유 메모리형 패킷교환기의 출력버퍼 관리 기법으로서 대표적인 것으로서는 정적 임계치 (Static Threshold) [2], Pushout (PO)[3], 그리고 동적 임계치 (Dynamic Threshold)[4] 등을 들 수 있다. 정적 임계치 방식에서는 메모리 공유에 있어서 개별 큐(queue)가 점유할 수 있는 메모리 공간이 기설정된 임계치에 의해 정적으로 제한된다. 정적 임계치 방식은 구현이 매우 간단하다는 장점이 있으나 트래픽 조건이 변하는 경우 임계치 값이 적절하게 새로운 값으로 조정되지 못한다는 단점이 있다.

정적 임계치 방식에서 임계치가 하나의 값으로 설정되어 변동된 트래픽 환경에서도 그대로 쓰이는 데 반하여 동적 임계치 방식에서는 개별 큐의 길이를 제어하기 위한 임계치가 변동하는 트래픽 상태에 따라 적절하게 바뀜으로서 뛰어난 적응력을 갖고 있다. 뿐만 아니라 항상 일정량의 버퍼를 사용하지 않고 남겨두는데 이는 비활성화 상태였던 큐가 갑자기 활성상태로 됐을 때 최소한의 버퍼 공간을 제공하는 기능을 갖는다. 이에 따라 모든 큐는 어떠한 상태에서도 최소한의 스위치 자원을 제공받을 수 있다[2].

동적 임계치를 이용한 기존의 버퍼관리 방식은 교환기 내 버퍼 점유상태를 동적으로 반영하여 변하는 트래픽 조건하에서도 전체공간을 출력 포트 간에 공정하게 분배되도록 하는 효율적인 통제 기능을 갖고 있지만 서비스 품질 (Quality-of-service: QoS) 기능 측면에서 서로 다른 우선 순위를 갖는 트래픽 플로우간의 버퍼 공유를 위한 차등적인 패킷처리 기능이 마련되어 있지 않다.

본 논문에서는 출력 포트간의 공정한 버퍼 공유를 위한 효율적인 버퍼관리 기법인 동적 임계치 방식을 확장하여 서로 다른 손실율 요구조건을 갖는 트래픽 클래스간의 차등적인 버퍼 공유를 위한 버퍼관리 기법을 제안하고 컴퓨터 시뮬레이션을 통하여 그 성능을 검증하였다. 트래픽 클래스 별로 상이한 손실율 요구조건에 따라 차등적으로 설정된 임계치가 사용되며 이러한 임계치는 변화하는 트래픽 조건에 따라 동적으로 적절히 조정되도록 하였다. 이에 따라 낮은 우선순위의 트래픽에 대해서는 동일한 전체 버퍼 점유 상태 하에서도 더 작은 버퍼할당 임계치가 적용되어 버퍼 점유량이 더 낮은 값으로 제한됨으로써 그만큼의 여유 버퍼공간이 높은 우선순위의 트래픽에 할당되도록 함으로써 보다 낮은 패킷 유실률을 달성하도록 하였다. 반면에 전체 버퍼 점유량이 많지 않은 경우에는 기존의 동적 임계치 방식이 갖는 이점인 높은 버퍼 이용률과 패킷 처리율을 얻을 수 있도록 하였다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 공유 메모리형 출력 큐잉 패킷 교환기 구조에 대해 간략히 기술하고 이러한 구조의 교환기를 위한 기존의 버퍼관리 기법과 장단점에 대해 자세히 설명한다. Ⅲ장에서는 동적 임계치를 이용한 버퍼관리 기법과 우선순위가 다른 트래픽 클래스별로 차등적인 버퍼할당을 할 수 있는 가중형 동적 임계치를 이용한 버퍼관리 기법에 대해 기술한다. Ⅳ장에서는 컴퓨터 시뮬레이션을 통하여 제안된 방식의 성능을 검증하며 마지막으로 Ⅴ장에서 본 논문의 결론을 맺는다.

Ⅱ. 공유 메모리형 패킷 스위치를 위한 기존의 버퍼관리기법

출력 버퍼형 패킷 스위치는 전통적인 회선 스위치와는 달리 서로 다른 입력 포트를 통해 도착한 여러 패킷들이 동시에 동일한 출력 포트로 향하게 되는 경우 출력 전송라인을 두고 패킷 간에 충돌이 발생할 수 있다. 한 번에 한 개의 패킷만이 출력포트를 통해 전송될 수 있으므로 즉시적으로 전송될 수 없는 패킷들을 저장하기 위한 버퍼가 출력측에 놓이게 된다. 그림 1은 일반적인 패킷 스위치 구조를 나타내고 있다. 그림에서 보는 바와 같이 패킷 스위치는 일반적으로 LI(Line Interface), CP(Call Processor), 스위치 패브릭(Switching Fabric) 등으로 구성된다.

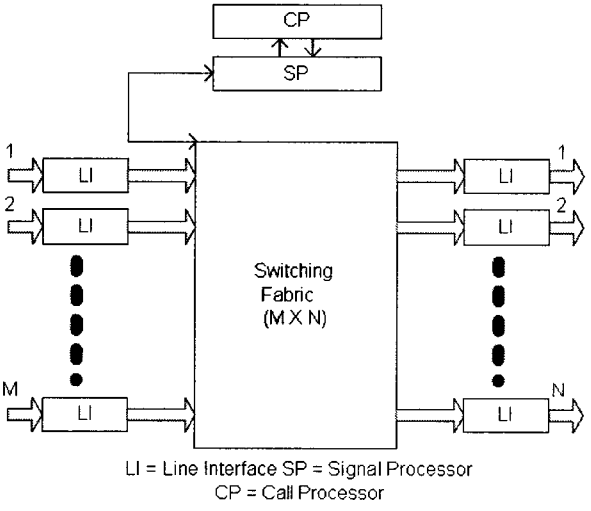

그림 146. 일반적인 패킷 스위치 구조
Fig. 1. A general packet switch architecture.

공유 메모리형 스위치에서는 모든 입/출력 포트들이 하나의 버퍼 메모리를 공유하게 되며 이를 교환의 방법으로 이용한다. 입력 포트를 통해 들어오는 패킷들은 각각의 출력 포트에 따라 차례로 공유 메모리에 write되며 출력하는 경우 출력 포트별로 차례로 read되어 각각의 출력 링크를 통해 전송되어 진다. 따라서 입출력 링크의 전송속도를 기준으로 각 시간 슬롯 동안에 순차적으로 M개의 입력 패킷의 memory writie와 N개의 출력 패킷 read를 처리할 수 있는 고속 메모리 제어기(Memory Controller)가 필요하다. 따라서, 이 구조를 갖는 스위치는 용량 확대가 제한적이지만 버퍼 메모리의 효율적인 이용이 가능하고 다중점 연결(multipoint connection)을 위한 패킷 복사와 우선 순위에 따른 버퍼 관리가 용이하다는 장점이 있다. 공유 버퍼형 스위치에 있어서 공유 메모리는 정확한 스위칭을 위해 논리적으로 N개의 버퍼처럼 동작을 해야 한다. 이를 위해 메모리 제어기는 같은 출력단자로 전달되는 패킷들이 저장된 메모리 주소들을 독립적으로 관리하거나 연결 목록(linked list)의 형태로 공유 메모리에 각 패킷과 함께 저장해야 한다.

기존의 버퍼 관리 기법들은 각 가상 큐들의 크기를 제한함으로써 일정 크기의 버퍼 공간을 보장하도록 하는 것과 전체 버퍼 공간을 공유함으로써 버퍼의 이용률을 높이도록 하는 방식 사이에서 하나를 선택하면 다른 쪽은 포기해야 하는 근본적인 문제점을 가지고 있다. 따라서 공유 메모리 관리 기법 설계에 있어서 서로 상반되는 두 가지 목표들을 균형적으로 달성할 수 있도록 하는 것이 매우 중요하다.

현재까지 제안된 것으로 공유 메모리형 스위치를 위한 버퍼 관리 기법은 대체적으로 다섯 가지 방식을 들 수 있다.

첫째 방식은 완전 공유법 (Complete Sharing: CS)으로써 전체 메모리를 각각의 큐들이 제한없이 원하는 만큼 사용할 수 있으나 버퍼가 가득 찰 경우 새로 유입되는 패킷들은 버려지게 된다. 완전 공유법은 구현 및 관리가 간단하며 과부하 상태의 큐가 없고 포트간의 처리량이 유사할 경우 높은 공유 효과를 기할 수 있다는 장점이 있다.

두 번째 방식은 완전 분할법(Complete Partitioning: CP)으로써 대규모의 공유 메모리를 각 출력 포트별로 분할된 별도의 메모리처럼 관리하는 방법이다. 완전 분할법 또한 구현과 관리가 간단하며 악의적인 트래픽에 대해 정상적인 트래픽을 보호할 수 있는 장점이 있으나 메모리 공유 효과를 전혀 기대할 수 없다.

세 번째 방법은 정적 임계치(Static Threshold: ST)를 사용하는 것으로써 개별 큐의 크기를 고정된 임계치로서 제한한다[2]. ST는 각 출력포트에 대한 큐의 현재 길이가 주어진 임계치보다 작으면 도착 패킷의 저장을 허용한다. ST는 구현이 간단하다는 장점은 있으나 고정된 임계치를 사용하기 때문에 트래픽 상태 변화에 대한 적응력이 없다는 단점을 갖고 있다.

네 번째로는 도착 패킷들을 버퍼가 가득 찰 때까지 받아들인 뒤, 버퍼가 가득 찬 다음 들어오는 패킷들은 기존에 버퍼에 들어 있던 패킷들 중 하나를 선택하여 그 자리에다 덮어쓰는 방법이다[3]. 이 방법을 PO(Pushout: PO)이라 한다. PO는 항상 버퍼의 이용률을 최대화할 수 있을 뿐 아니라 패킷 유실률면에서도 다른 버퍼 관리 기법과 비교하여 우수한 성능을 가진다. 그러나 PO는 각각의 패킷마다 도착시간과 소속되어 있는 플로우를 포인팅하도록 해야 하는 등, 처리 과정이 복잡하여 고속 스위칭에 부적합할 뿐 아니라 패킷 우선 순위(priority) 제어의 구현이 어렵다. PO는 가장 우수한 성능을 보이나 고속 스위치를 위한 버퍼 관리기법으로서 실제적으로 구현이 어렵기 때문에 다른 버퍼관리 기법의 성능 평가를 위한 비교 기준으로서 자주 거론된다.

마지막으로 구현의 간단함과 트래픽 환경에 대한 적응력을 가지고 있는 동적 임계치(Dynamic Threshold: DT)를 이용한 관리 기법을 들 수 있다[4]. 이 방법에서는 항상 출력포트에 대한 임계치를 현재 사용되지 않고 남은 버퍼 크기에 비례하여 설정한다. DT는 ST와 달리 트래픽 상황 변화에 대한 적응력이 있어 높은 메모리 공유 효과와 악의적 트래픽에 대한 순응 트래픽의 보호 기능을 동시에 기대할 수 있다. 또한 DT는 구현이 크게 어렵지 않다는 장점을 가지고 있다. 그러나 이 방법에서는 출력 포트간에 공유 메모리가 공정하게 할당될 수 있도록 하는 제어 기능만 있을 뿐 서비스 품질 (Quality of Service) 기능 지원을 위해 기본적으로 필요한 트래픽 플로우별 우선 순위에 기반한 자원 할당 기능이 갖춰져 있지 않다.

Ⅲ. 가중형 동적 임계치를 위한 버퍼관리기법

이 장에서는 참고문헌 [4]에서 제시한 동적 임계치를 이용한 버퍼 관리기법의 동작 메커니즘에 대해 비교적 구체적으로 설명하고 이를 확장하여 우선순위에 기반한 버퍼관리 방식에 대해 설명하기로 한다. 동적 임계치를 이용한 버퍼 관리기법에서 사용되는 제어 임계치의 크기는 현재 사용되지 않고 남은 전체 버퍼 공간에 비례하여 설정된다. 즉, 각 출력포트에 대한 큐는 사용되고 남은 버퍼를 임의의 함수 f에 대입하여 얻어진 값을 임계치로 설정하여 큐의 길이를 제한한다. 임의의 시간 t 에서, $T(t)$를 시간 t일 때의 임계치라 하고, $Q^i(t)$를 시간 t일 때의 출력포트 i에 대한 가상 큐의 길이라 하자. 또한 $Q(t)$를 모든 가상 큐들이 점유하고 있는 전체 버퍼공간, B를 전체 버퍼 공간이라 하면, 임계치 $T(t)$는 다음과 같이 나타낼 수 있다.

$$T(t) = f(B - Q(t)) = f(B - \sum_i Q^i(t)) \qquad (1)$$

동적 임계치를 이용한 버퍼관리기법에서 시간 t에 가상 큐 i에 도착한 패킷의 차단 여부는 다음과 같이 결정된다[4]. 가상 큐 i로 향하는 패킷들은 그 큐의 현재 길이가 임계치보다 작으면 유입이 허용되고 그렇지 않으면 차단된다. 즉, $Q^i(t) \geq T(t)$이면 가상 큐 i 에 도착하는 패킷은 차단된다. 하지만 이 방식은 차단형 버퍼관리 기법이므로 이미 저장되어 있는 패킷은 출력 시까지 삭제되지 않는다. 임계치를 구하는 가장 간단한 방법은 사용되고 남은 버퍼에 α배 곱한 값을 사용할 수 있으며 다음 식과 같이 주어진다.

$$T(t) = \alpha \cdot (B - Q) \tag{2}$$

본 논문에서는, 망 자원 중에서 공유 메모리형 출력 큐잉 교환기의 출력포트간 공정 분배를 위한 버퍼관리 기법 연구와 함께 차등화된 우선순위 지원을 위한 버퍼 관리 기법을 연구하였다. 특히 기존의 동적 임계법이 서로 다른 손실 요구치를 임계치에 반영하지 않았기 때문에 차등화된 우선순위 지원에 취약했던 점을 개선하였다. 기존 동적 임계법은 서로 같은 손실 우선순위를 갖는 트래픽이 여러 출력 포트에서 처리될 때 한정된 버퍼를 어떻게 효율적으로 사용하는가에 초점을 맞추고 있었다. 하지만 이는 어디까지나 동일한 손실 요구치 만을 위한 것이었다. 기존의 동적 임계법의 임계지 결정요소인 α값을 차등화하여 손실 요구치에 따라 α_p 을 따로 두게 되면 서로 다른 손실 요구치를 갖는 트래픽이 유입되어 우선순위를 달리 배정할 때 이에 따른 차등적인 관리를 할 수 있다. 여기에서 우선순위 p를 갖는 플로우에 대한 임계치 $T_p(t)$를 다음 식과 같이 설정한다.

$$\begin{aligned} T_p(t) &= \alpha_p \cdot (B - Q(t)) \\ &= \alpha_p (B - \sum_i Q^i(t)) \end{aligned} \tag{3}$$

따라서 동일한 전체 버퍼 점유상태에서도 플로우 별로 우선순위에 따라 임계치가 차등적으로 설정됨으로서 패킷 유실율면에서 차이가 발생하며 패킷 유실률 요구치에 맞춰 α_p 값을 적절히 설정하면 된다. 특히 α_p값이 2의 멱수 형태의 값을 갖도록 하면 시프트 레지스터를 이용하여 간단한 제어 메커니즘의 구현이 가능하다.

이는 두 가지 경우 이점을 갖는다. 첫째, 우선순위가 낮은 플로우에 과부하 트래픽이 유입될 경우 낮은 우선순위의 플로우가 메모리의 대다수를 점유하는 것을 제한함으로서 우선순위가 높은 트래픽에 대한 간섭을 최소화할 수 있다. 둘째, 우선순위가 높은 플로우에 과부하 트래픽이 유입될 경우 과부하 트래픽이라 할지라도 높은 우선순위의 트페픽에 대해서는 낮은 유실률이 유지될 수 있도록 할 수 있다.

Ⅳ. 시뮬레이션 결과 및 검토

이 장에서는 먼저 시뮬레이션 모델에 대해 기술하고 여러 가지 버퍼 관리 기법에 대한 성능비교 결과를 살펴보기로 한다.

4.1 시뮬레이션 모델

실험에 사용된 패킷 교환기는 2개의 입력포트와 2개의 출력 포트를 갖는다. 각 입력 포트별로 6개의 플로우가 인가되며 6개의 플로우는 세 종류의 우선순위 중에서 하나를 동일한 비율로 갖는다고 가정한다 (즉, $k=0,1,2$에 대해 우선순위 k인 플로우의 수는 2개가 된다). 이 중에서 우선순위별로 하나씩 모두 3개의 플로우가 출력 포트 1로 향하고 나머지 3개의 플로우는 출력포트 2로 향한다. 주어진 교환기의 각 출력포트는 플로우 개수만큼의 가상 큐를 가지고 있으며 (따라서 출력포트당 모두 6개의 가상큐가 존재) 이들 가상 큐들은 모두 하나의 메모리를 공유한다. 하나의 출력 포트 내에서는 FIFO 방식으로 패킷들이 처리된다. 그리고 모든 패킷들은 동일한 길이를 갖는 것으로 가정한다.

각 플로우는 하나의 on-off 소스로써 활성(active)과 비활성(idle) 기간으로 바뀌면서 활성기간 동안 패킷을 발생하게 된다. 활성기간과 비활성 기간은 각각 계수 α와 β의 지오메트릭(geometric) 분포를 갖는 랜덤 변수이다 (그림 3 참조). 반면 비활성 주기 기간 동안은 계수 β를 갖는 지오메트릭 분포가 된다. 여기서 각 활성기간 동안 적어도 하나의 패킷이 발생하고 비활성 기간에는 패킷이 발생하지 않고, 버스트 길이는 통계적으로 독립이라고 가정한다

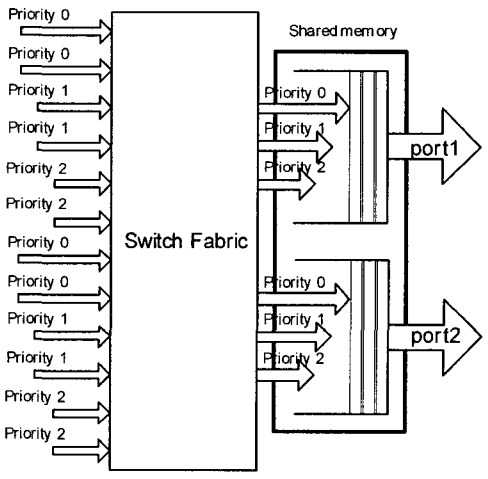

그림 2. 시뮬레이션 모델.
Fig. 2. The simulation model.

α와 β가 주어졌을 때, 평균 버스트 길이를 L_b, 평균

비활성 시간을 L_{idle}, 그리고 출력 링크에 정규화된 (normalized) 부하를 ρ라 할 때 각각의 값은 다음과 같이 주어진다.

$$L_b = \frac{1}{\alpha}$$
$$L_{idle} = \frac{1}{\beta} \quad (4)$$
$$\rho = \frac{L_b}{L_{idle} + L_b} = \frac{\beta}{\alpha + \beta}$$

그림 3에서 첫 on 구간의 길이는 5이며 (즉, 이 시간 구간에서 해당 트래픽 소스는 5개의 패킷을 연이어 발생) 1 슬롯 만큼의 휴지기간을 거쳐 다시 길이 2인 패킷 발생 구간을 나타내고 있다. 평균 버스트 길이라 함은 그림 3에서 보듯 랜덤하게 주어진,ㄴ 버스트 길이를 평균한 것이다.

본 논문에서는 교환기에서 통과된 패킷과 유실된 패킷의 비율을 나타내는 패킷 유실률(CLR)을 성능평가 지수로 정했으며 다음과 같이 주어진다.

$$CLR = \frac{\text{손실된 셀의 개수}}{\text{전송된 셀의 개수}}$$

실험에서 사용한 트래픽 조건은 다음과 같다. 전체 버퍼용량은 120 패킷, 평균 버스트 길이는 10 패킷으로 하였고 우선 순위 0의 플로우에 대한 트래픽 부하는 출력 대역폭의 0.3, 우선 순위 1의 부하는 출력 대역폭의 0.3, 우선 순위 2의 부하는 출력 대역폭을 기준으로 0.1에서 1.1까지 변하도록 하였다. 여기서 집중적으로 고려해야 할 사항은 낮은 트래픽 부하에서의 메모리 이용률과 낮은 우선 순위의 트래픽이 고부하 상태로 유입될 때 높은 우선 순위의 트래픽이 얼마만큼 영향을 적게 받고 보호를 받을 수 있는가 하는 점이다.

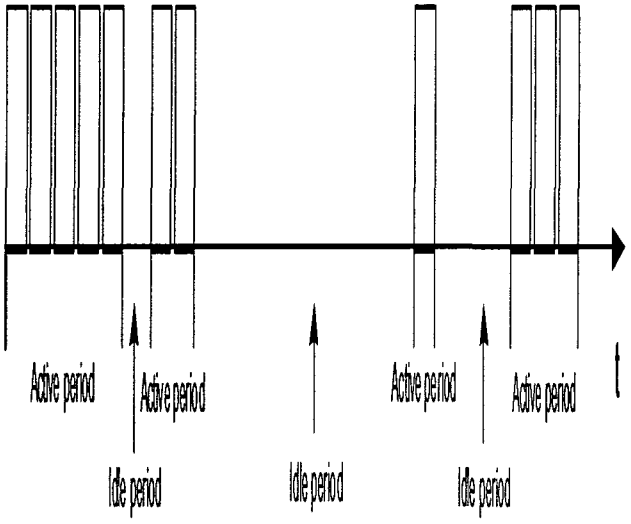

그림 148. On-off 소스 트래픽 발생 패턴.
Fig. 3. The traffic generation pattern for on-off sources.

4.2 성능 비교 및 검토

본 실험에서는 낮은 우선순위 트래픽의 부하가 변화할 때 여러 가지 버퍼 관리기법의 성능을 버퍼 이용률과 보호율면에서 비교해 보기로 한다. 버퍼 이용률은 전체적으로 낮은 부하 상태에서 패킷 유실률을 통해서, 그리고 보호율은 전체적으로 높은 부하 상태에서 우선순위 0 트래픽의 유실률을 통해 그 성능을 확인할 수 있다.

그림 4는 낮은 우선순위(즉, 우선순위=2) 트래픽의 부하가 0에서 1까지 변화할 때 개별 트래픽의 우선순위를 고려하지 않고 Pushout, CS, DT, ST, 그리고 CP 등의 버퍼관리기법들을 각각 적용시켰을 때 얻어지는 우선순위 0 트래픽에 대한 패킷 유실률을 도시하고 있다. CP의 경우 우선순위에 상관없이 각각 20 (단위는 패킷)의 버퍼 공간을 할당하였다. ST의 경우 임계치를 60으로 설정하였다. DT의 경우 α 값은 4.0으로 설정하였다. 그림에서 나타나는 바와 같이 낮은 트래픽 부하 상태에서는 Pushout과 CS은 가장 낮은 패킷 유실률을 보여주며 다음으로 DT, ST, CP의 순서로 패킷 유실률이 낮았다. 즉, 낮은 부하 상태에서는 완전 공유법과 Pushout이 높은 공유효과로 인하여 다른 관리기법에 비해 가장 양호한 성능을 가지게 된다. 반면에 높은 트래픽 부하 상태에서는 Pushout과 DT가 가장 낮은 패킷 유실률을 가지며 CP, ST, CS 등의 순으로 패킷 유실률이 높아짐을 알 수 있다. 공유 메모리를 위한 버퍼관리기법으로서 ST는 트래픽 부하가 낮은 경우 비교적 낮은 유실률을 보여주나 높은 트래픽 조건하에서는 Pushout이나 DT에 비해 상당히 높은 유실률을 보여주고 있다. CP의 경우 공유 효과가 없어서 낮은 트패픽 부하에서 놓은 유실률가지나 높은 부하 상태에서는 개별 가상 큐마다 버퍼공간이 독점적으로 할당되어 있으므로 패킷 유실률 면에서 비교적 양호한 성능을 보이게 된다. CS의 경우 낮은 트래픽 부하 상태에서 낮은 유실률을 보이지만 부하가 증가함에 따라 특정 큐가 버퍼공간을 급속히 점유함으로써 버퍼공간 사용에 따른 공정성이 매우 나빠진다. 그러나 DT의 경우 거의 모든 트래픽 부하 상태에서 Pushout에 가까운 우수한 성능을 보인다는 점을 알 수 있으나 낮은 우선 순위 (우선순위 2) 트래픽의 부하가 상승함에 따라 높은 우선순위 트래픽의 패킷 유실률이 비례하여 상승함으로써 우선순위에 기반한 패킷처리 기능이 전혀 지원되지 못하고 있음을 알 수 있다.

그림 5는 낮은 우선순위(즉, 우선순위=2) 트래픽의 부하가 0에서 1까지 변화할 때 Pushout, CS, DT, ST, 그리고 CP 등의 버퍼관리기법들을 개별 트래픽의 우선순위를 고려하여 적용시켰을 때 얻어지는 우선순위 0 트래픽에 대한 패킷 유실률을 도시하고 있다. CP의 경우 우선순위 기능을 추가하여 우선순위 0에서 우선순위 2까지의 플로우에 각각 30, 20, 10 (단위는 패킷)의 버퍼공간을 할당하였다. ST의 경우 임계치를 우선순위에 따라 각각 75, 60, 45으로 설정하였다. 가중형 DT (WDT)의 경우 우선순위

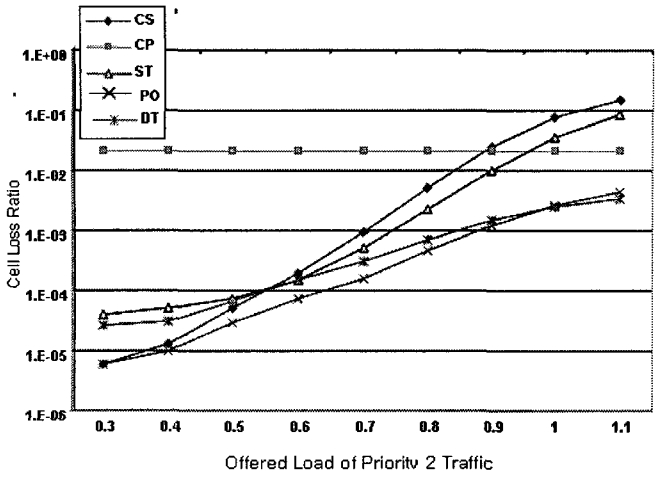

그림 149. 우선순위 기능을 적용되지 않은 경우 낮은 우선순위 트래픽 부하 변화에 따른 각 버퍼 관리기법 간의 패킷 유실률 비교.

Fig. 4. Packet loss ratios of several buffer management schemes without considering loss priorities when the low priority traffic load varies.

에 따른 α 값은 4.0을 기준으로 우선순위 0의 플로우는 8.0, 우선순위 1의 플로우는 4.0, 우선순위 2의 플로우는 2.0으로 설정하였다.

여기에서는 우선 순위에 기반한 패킷 처리 기능이 추가됨으로써 그림 4와는 확연히 다른 결과가 나타남을 알 수 있다. CP, CS의 경우 우선순위 기능이 적용되지 않으므로 전과 대등한 결과를 나타내고 있다. ST의 경우 낮은 우선순위 트래픽의 부하증가에 대해 약간의 유실률 향상을 보이나 여전히 낮은 우선순위 트래픽에 대한 높은 우선순위 트래픽의 보호 측면에서 차등화된 서비스

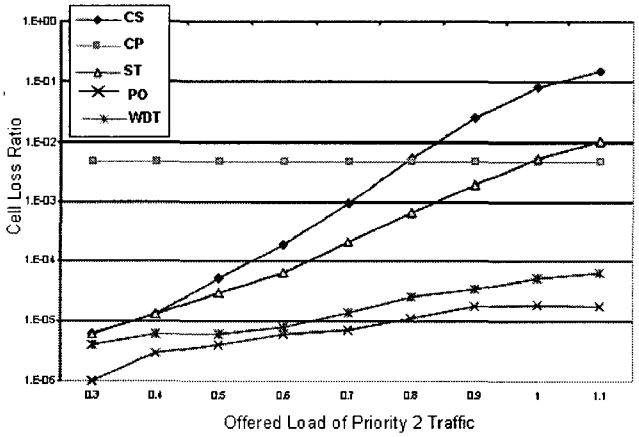

그림 150. 우선순위 기능을 적용한 경우 낮은 우선순위 트래픽의 부하 변화에 따른 각 버퍼 관리기법 간의 패킷 유실률 비교.

Fig. 5. Packet loss ratios of several buffer management schemes with considering loss priorities when the low priority traffic load varies.

품질 기능을 보여주지 못하고 있다. 한편 WDT의 경우 낮은 우선순위 트래픽의 부하변화에 대해 전체적으로 Pushout에 가까운 낮은 유실률 변화를 보여주고 있다. 따라서 우선순위에 따른 차등적인 버퍼 관리 기능이 제대로 수행됨으로써 우선순위 0 트래픽은 우선순위 2 트래픽에 의해 받는 영향이 현저히 감소하여 비우호적인 트래픽 환경에서도 정상적인 서비스가 가능함을 알 수 있다.

Ⅵ. 결 론

공유 메모리 관리를 위한 기존의 방법들은 가상 큐들의 최대 길이를 정적으로 제한함으로써 일정한 크기의 버퍼 할당을 보장하거나 전체 버퍼 공간을 공유하도록 함으로써 공유 버퍼의 이용률을 높이도록 하는 두 가지 부류로 나눌 수 있다. 완전공유 방식의 경우 낮은 트래픽 부하에서 높은 메모리 공유 효과를 보이나 트래픽 부하가 높아지면 특정 가상 큐가 공유 메모리를 과도하게 점유하는 것을 방지하는 의미의 보호 효과를 거의 기대하기 힘들다. 반대로 정적 임계치 방식의 경우 트래픽 조건 변화에 따른 적절한 임계치 설정이 불가능하다. 이에 반해 동적 임계치 방식은 거의 모든 트래픽 조건에 대해 적응성을 가짐을 알 수 있다.

본 논문에서는 공유 메모리의 공정 할당이라는 단순한 기능을 가지는 동적 임계치 방식을 확장하여 구현이 용이하고 서비스 품질기능 측면에서 우선 순위에 따른 차등적인 패킷처리 기능을 갖는 가중형 동적 임계치 방식을 제안하고 컴퓨터 시뮬레이션을 통하여 성능을 확인하였다. 끝으로 트래픽 플로우 별로 주어진 유실률 요구치를 만족시키는 인자값 α_p의 구체적인 설정 방안은 추가적인 연구가 필요한 부분이다.

참 고 문 헌

[1] F. A. Tobagi, "Fast Packet Switch Architectures for BroadBand Integrated Services Digital Networks", Proc. IEEE, vol. 78, no. 1, Jan. 1990, pp. 133-67.

[2] M. I. Irland, "Buffer management in a packet switch," *IEEE Trans. Commun.*, vol. COM-26, pp. 328-337, Mar. 1978.

[3] A. K. Thareja and A. K. Agarwala, "On the design of optimal policy for sharing finite buffers," *IEEE Trans. Commun.*, vol. COM-32, pp. 737-740, Jun. 1984.

[4] A. K. Choudhury and E. L. Hanhne, "Dynamic Queue Length Thresholds for Shared-Memory Packet Switches", *IEEE/ACM Trans. Commun.*, vol. 6, no. 2, Apr. 1998, pp.130-40.

[5] A. K. Choudhury and E. L. Hahne, "A Simulation

Study of Space Priorities in a Shared Memory ATM Switch", IEEE Journal High Speed Networks, vol. 3, pp. 491-512, No. 4, Nov. 1994.

[6] H. Y. Jung, B. C. Lee and K. C. Park, "Implementation of the Inter-Module Interface in an ATM Switching System", Proceedings of ITC-CSCC'97 Vol. 2, pp.79-82, Jul. 1997

[7] Joan Garcia-Haro and Andrezej Jajszczyk "ATM Shared-Memory Switching Architectures" IEEE Network, July/August 1994.

[6] K. Kumaran and D. Mitra, "Performance and Fluid Simulations of a Novel Shared Buffer Management Scheme", Proc. IEEE INFOCOM'98, Mar. 1988.

[7] A. Elwalid, D. Mitra and R. Wentworth, "A New Approach for Allocating Buffers and Bandwidth to Heterogeneous, Regulated 트래픽 in an ATM Node", IEEE JSAC, vol. 13, pp. 1115-1127, Aug. 1995.

김 창 원(Chang-Won Kim)
2000년 2월 건국대 전자공학과(공학사)
2003년 2월 건국대 전자공학과(공학석사)
관심분야: 정보통신망, 차세대 인터넷, MPLS망

김 영 범(Young Beom Kim)
1984년 2월 서울대 전자공학과(공학사)
1986년 2월 서울대 전자공학과(공학석사)
1996년 8월 미 메릴랜드주립대(공학박사)
1997년 9월 ~ 현재 건국대학교 전자공학부 부교수
관심분야: ATM, 통신망 트래픽 제어