Journal of Korea Multimedia Society Vol. 9, No. 3, March 2006(pp. 323-332)

다수 지연규격을 지원하는 시작시각 기반 공정패킷 스케줄러

김 태 준†

요 약

실시간 멀티미디어 응용의 서비스 품질을 보장하는 공정 패킷 스케줄링 알고리즘은 타임스탬프 계산시 사용되는 패킷의 기준시각 측면에서 종료시각 방식과 시작시각 방식으로 나뉜다. 전자는 트래픽 흐름의 속도에 반비례하는 레이턴시 특성으로 인해 트래픽 흐름의 예약속도 조정으로 다양한 지연규격을 지원할 수 있어 대부분의 스케줄러에 적용되고 있으나 과잉예약에 따른 대역폭 손실이 발생한다. 반면 후자는 과잉예약에 따른 대역폭 손실은 없으나 흐름의 수에 종속되는 레이턴시 특성으로 인해 다수의 지연규격을 수용하기 어려운 문제점이 있다. 본 논문에서는 다수 지연규격(multiple delay bounds)을 효과적으로 지원할 수 있는 시작시각 기반 공정 패킷 스케줄러를 제안하고, 제안된 스케줄러의 성능특성을 분석한다. 분석결과 종료시각 기반 스케줄러보다 최대 50% 높은 이용도 특성을 보였다.

A Start-Time Based Fair Packet Scheduler Supporting Multiple Delay Bounds

Tae-Joon Kim†

ABSTRACT

Fair packet scheduling algorithms supporting quality-of-services of real-time multimedia applications can be classified into the following two schemes in terms of the reference time used in calculating the timestamp of arriving packet; the Finish-Time (FT) and Start-Time (ST) schemes. The FT scheme, used in most schedulers, that has the property of an inversely rate-proportional latency is suitable to support various delay bounds because it can adjust the latency of a flow with raising the flow's reserved rate. However, the scheme may incur some bandwidth loss due to excess rate reservation. Meanwhile, although the ST scheme does not suffer from the bandwidth loss, it is hard to support multiple delay bounds because of its latency property relying on the number of flows. This paper is devoted to propose a ST scheme based scheduler to effectively support multiple delay bounds and analyze its performance comparing to the FT scheme based scheduler. The comparison results show that the proposed scheduler gives better utilization by up to 50%.

Key words: Packet Scheduler(패킷 스케줄러), Fair Queuing(공정큐잉), Fairness(공정성), Latency(레이턴시), Utilization(이용도)

1. 서 론

인터넷 전화, 인터넷 영상회의와 같은 고 수준의 서비스 품질을 요구하는 실시간 멀티미디어 통신 서비스가 널리 보급 확산되고 있다. 품질을 보장할 수 있는 자원예약 기반의 종합서비스(InterServ)와 확

※ 교신저자(Corresponding Author) : 김태준, 주소 : 충남 천안시 부대동 275번지(330-717), 전화 : 041)550-0209, FAX : 041)556-6447, E-mail : tjkim@kongju.ac.kr

접수일 : 2005년 8월 16일, 완료일 : 2005년 11월 8일
† 정회원, 공주대학교 정보통신공학부

장성이 우수한 차등서비스(DiffServ)의 두 가지 품질 지원 모델이 제시되었다[1]. 종합서비스 모델에서 품질 보장은 각 라우터에 탑재되는 공정 패킷 스케줄러에 의해 이행된다. 따라서 공정 패킷 스케줄러는 트래픽 흐름의 요구 속도를 보장하고 요구 지연규격을 위반하지 않아야 한다. 여기서 트래픽 흐름 (이하 흐름)이란 실시간 멀티미디어 통신 서비스에 수반되는 각각의 미디어 스트림(stream), 예로 음성, 영상 스트림 등을 의미하고, 지연규격은 라우터, 즉 스케줄링 서버에서 그 흐름에게 허용되는 최대 통과(transit) 지연시간을 의미한다. 참고로 공정 패킷 스케줄러가 탑재되는 라우터를 통상 QoS(Quality-of-Service) 라우터라 부른다.

이상적 패킷 스케줄링 알고리즘인 GPS(General Processor Sharing)[2]를 구현하는 두 가지 접근방식이 있는데, 하나는 WFQ(Weighted Fair Queuing)[3]에 적용된 타임스탬프 방식이고, 다른 하나는 WRR (Weighted Round Robin)[4]에 적용된 라운드로빈 (round-robin) 방식이다. 타임스탬프 방식은 구현이 복잡한 반면 라운드로빈 방식은 레이턴시(latency: 최대전달지연) 특성이 나쁜 단점이 있다. 타임스탬프 방식은 또한 도착하는 패킷의 타임스탬프 계산시 사용되는 패킷의 기준시각 관점에서 예상되는 전송 종료시각을 사용하는 종료시각 방식과 예상되는 전송 시작시각을 사용하는 시작시각 방식으로 나뉜다. 본 논문에서는 편의상 전자를 FT(Finish-Time)방식, 후자를 ST(Start-Time) 방식이라 부른다.

FT 방식은 WFQ에 처음 적용되었다[3]. WFQ는 우수한 레이턴시 특성을 제공하나 타임스탬프 계산이 너무 복잡하였다. SCFQ(Self-Clocked Fair Queuing)[5]에서 자가클럭(self-clock)을 도입하여 이 문제를 해결하였으나 레이턴시 특성이 저하되는 또 다른 문제점이 대두되었다. 두 가지 문제점을 동시에 해결할 수 있는 방안이 가상 서버시간 기반의 RPS(Rate Proportional Servers) 이론으로 정립되어[6] 다양한 패킷 스케줄러의 개발에 적용되었다[7-9]. 한편 ST 방식은 SFQ(Starting-time Fair Queuing) [10] 및 QLR(Queuing latency Rate)[11]에 적용되었는데, SFQ는 SCFQ와 마찬가지로 자가클럭을 사용하나 QLR은 자가클럭 대신 RPS 이론에서 제안된 서버 가상시간을 사용한다.

FT와 ST 방식은 다음과 같이 레이턴시 측면에서 큰 차이점을 보인다; 전자는 흐름의 속도에 반비례하는 레이턴시 특성을 갖는 반면 후자는 흐름의 속도에 무관하나 스케줄러에 수용되는 흐름의 수에 비례하는 레이턴시 특성을 갖는다. FT 방식의 경우 각 흐름들의 레이턴시가 서로 독립적이기 때문에 다양한 지연규격을 수용하기 쉬운 반면 흐름의 속도에 의해 결정되는 레이턴시가 지연규격을 만족시키지 못할 경우 과잉예약을 해야 하므로 대역폭 손실이 발생한다. 한편 ST 방식의 경우 과잉예약의 문제는 발생하지 않으나 각 흐름의 레이턴시가 서로 종속되어 있어 다양한 지연규격을 수용하기가 매우 어렵다. 본 연구에서는 다수 지연규격을 효과적으로 지원할 수 있는 ST 방식 기반 스케줄러를 제안하고, 제안된 스케줄러의 성능특성을 분석 및 평가하고자 한다.

본 논문의 구성은 다음과 같다; 2장에서 관련연구의 소개와 더불어 ST방식의 문제점을 살펴보고, 3장에서 제안된 방식을 기술하며, 4장에서 제안된 방식의 레이턴시와 대역폭 이용도 특성을 분석한다. 그리고 5장에서 수치해석을 통해 성능특성을 평가하며 6장에서 결론을 맺는다.

2. 관련연구

공정 패킷 스케줄러의 성능특성은 일반적으로 레이턴시, 구현 복잡도 및 공정성의 세가지 지표에 의해 평가된다[2-11]. 이들 지표중 공정성은 짧은 시간 구간내 일어날 수 있는 불공평성을 평가하는 것으로 이는 서비스 이용자에게는 아무런 영향을 주지 않고 출력 트래픽의 간헐성(burtiness) 특성에만 영향을 미치므로 레이턴시와 구현 복잡도 지표에 비해 그 유용성이 매우 낮다. 따라서 본 연구에서 공정성 지표는 고려되지 않는다. 한편 구현 복잡도는 타임스탬프계산 복잡도과 패킷전송 복잡도에 의해 결정되는데, 전자는 스케줄러마다 다르나 후자는 최대 수용흐름의 수를 V 라 할때 V 개 흐름 큐의 선두 패킷중 타임스탬프가 가장 작은 패킷을 찾는 작업에 기인하므로 $\log V$ 번의 연산 동작을 의미하는 $O(\log V)$ 의 값을 갖는다[12].

먼저 FT 방식 관련 기존연구를 살펴보자. WFQ에서 임의 흐름 i 의 레이턴시 D_i 는 다음과 같이 계산된다[3];

$$D_i = \frac{L_i}{r_i} + \frac{L}{C} . \qquad (2.1)$$

여기서 r_i는 흐름 i의 요구속도, L_i는 흐름 i의 최대 패킷크기, L 는 모든 흐름의 최대 패킷크기, 그리고 C는 스케줄러 대역폭이다. 참고로 스케줄러 대역폭이란 출력링크의 전송속도를 의미하며, 본 연구에서 사용되는 두 가지 용어인 속도와 대역폭은 같은 의미를 갖는다. WFQ의 레이턴시는 대응 GPS 서버에서의 패킷 서비스 시간을 의미하는 L_i/r_i 항과 출력링크에서 최대 패킷 전송시간을 의미하는 L/C 항으로 구성되는 이상적인 값을 갖는다. 하지만 패킷이 도착할 때 마다 모든 흐름, 즉 최대 V 개의 흐름 상태를 파악해야 하므로 WFQ의 타임스탬프계산 복잡도는 $O(V)$가 된다. SCFQ방식에서 자가클럭 (전송중인 패킷의 타임스탬프)을 도입하여 타임스탬프 계산 복잡도를 $O(1)$로 대폭 줄였으나 $(L_i/r_i + VL/C)$로 계산되는[5] 레이턴시는 (2.1)에 비해 크게 저하된다.

RPS 이론[6]은 서버 가상시간을 도입하여 WFQ의 레이턴시 특성을 유지하면서 구현 복잡도를 줄이는 방법을 제시하였는데, 서버 가상시간이란 서버 활성구간 동안, 즉 서버가 지속적으로 패킷을 전송하는 구간의 시작시점부터 현재까지 전송된 전체 트래픽 량을 나타내는 시간의 함수로서 스케줄러의 기준시간으로 사용된다. 이 이론은 대부분의 FT 방식 기반 스케줄러 (이하 FT 스케줄러), 예로 SPFQ(Starting Potential Fair Queueing)[7], MSPFQ(Medium Starting Potential Fair Queueing)[8] 및 NSPFQ (New Starting Potential Fair Queuing)[9] 등에 적용되었는데, 이들은 모두 서버 가상시간의 정확성을 높이고 운영상의 복잡성을 줄이는데 주안점을 두고 개발되었다. 그 결과 이제 FT 스케줄러는 $O(\log V)$의 이상적 구현 복잡도[12]와 WFQ급 레이턴시 특성을 갖게 되었다.

한편 ST 방식을 사용하는 SPQ와 QLR에서 흐름의 레이턴시 D는 다음과 같이 계산된다[10,11];

$$D = \sum_{i=1}^{V} \frac{L_i}{C} . \qquad (2.2)$$

타임스탬프계산 복잡성은 자가클럭과 서버 가상시간과 같은 스케줄러 기준시간의 운영 복잡성에 기인하므로[12] 타임스탬프 계산시 사용되는 패킷 기준시각의 선택과 무관하다. 따라서 자가클럭을 사용하는 SPQ는 SCFQ와 동일한 구현 복잡도를 갖고, 서버 가상시간을 사용하는 QLR은 RPS 기반 스케줄러와 동일한 구현 복잡도를 갖는다.

스케줄러에서 지원되어야 할 지연규격의 수에 대해 살펴보자. 동일한 서비스라 하더라도 지연품질을 달리하여 서비스를 차별화 할 경우 종단간 지연규격이 서로 달라지게 되고, 동일한 지연품질의 서비스라 하더라도 트래픽 흐름이 경유하는 노드 수가 일정하지 않으므로 각 노드에서 요구되는 흐름의 지연규격은 흐름마다 다를 수 있다. 따라서 실제 환경에서 스케줄러는 다수의 지연규격을 지원해야 한다.

FT 방식의 경우 흐름 i의 지연규격을 Q_i라 할 때 (2.1)에 의해 결정되는 레이턴시 D_i가 Q_i를 위반, 즉 $D_i > Q_i$시 스케줄러가 흐름 i에 할당하는 대역폭을 증가시켜 $D_i \le Q_i$ 조건을 충족시킬 수 있으므로 각 흐름이 요구하는 다양한 지연규격을 쉽게 만족시킬 수 있다. 하지만 ST 방식의 경우 모든 흐름의 레이턴시가 (2.2)에 의해 동일한 값을 가지므로 각각의 흐름이 요구하는 서로 다른 지연규격을 별도로 지원할 수 없다. 그 대신 (2.2)에 의해 결정되는 레이턴시가 가장 엄격한 지연규격을 위반하지 않도록 흐름수 V를 줄여야 하는데, 이로 인해 과다한 대역폭 손실이 발생할 수 있다.

3. 제안방식

제안된 방식의 핵심은 각 지연등급 트래픽을 별도의 ST 방식 기반 스케줄러 (이하 ST 스케줄러)로 처리하는 것이다. 여기서 지연등급이란 하나의 지연규격 또는 여러 지연규격의 집합을 의미하는데, 여러 지연규격의 집합일 경우 가장 엄격한 지연규격이 그 등급의 지연규격이 된다. 제안된 방식을 MST (Multiple ST) 스케줄러라 부르자. 스케줄러에서 지원되는 지연등급이 M 개이고 각 지연등급의 지연규격을 $Q_1, Q_2 .., Q_M$ 초(second)라 하자, 여기서 $0 < Q_1 < Q_2 < .. < Q_M$ 이다.

3.1 MST 스케줄러의 구성

MST 스케줄러는 그림 1과 같이 등급판별기, 전단서버 및 후단서버의 세 부분으로 구성된다. 등급판별기는 도착하는 흐름의 요구지연규격으로 지연등급을 판별하여 그 흐름이 전단서버내 해당 부서버로 입력되도록 한다. 전단서버는 M 개의 부서버로 구성되며, 각 부서버는 독립된 ST 스케줄러로서 동작

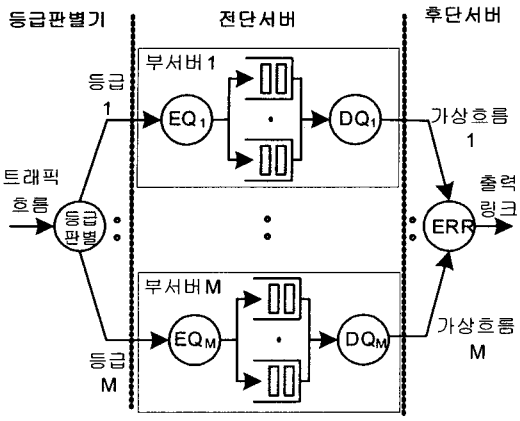


그림 1. MST 스케줄러의 구성도

하면서 해당 지연등급의 트래픽만 처리한다. 후단서버는 전단서버내 M 개 부서버가 출력링크를 공평하게 사용하도록 해주는 라운드-로빈 스케줄러로 성능특성이 우수한 ERR(Elastic Round Robin)[13]방식을 사용한다. 참고로 그림 1의 MST 스케줄러는 하나의 출력링크에 대한 것으로 실제 라우터에는 출력링크의 수만큼의 MST 스케줄러가 탑재된다.

임의 부서버 n은 도착하는 패킷에 대해 그의 타임스탬프를 계산하여 부착하고, 타임스탬프가 부착된 패킷을 해당 흐름의 큐에 입력하는 EQn(En- Queue 서버 n), 흐름별로 별도의 큐가 구비되는 큐집합 및 큐집합에 대기중인 패킷 중 가장 앞선 타임스탬프를 갖는 패킷을 찾아서 출력링크로 전송하는 DQn (De-Queue 서버 n)으로 구성된다. 각 부서버에서 출력되는 트래픽은 후단서버에서 하나의 가상 흐름으로 취급된다. 후단서버는 M 개의 가상흐름 큐를 구비하여 각 큐의 패킷을 ERR 방식으로 서비스 한다.

MST 스케줄러에서 취급되는 패킷은 트래픽 흐름에서 유입된 데이터 패킷과 흐름식별자, 요구속도, 요구지연규격 등의 관련정보를 모두 포함하는 내부 패킷이다. 실제 구현시 그 량이 많은 데이터 패킷의 취급부담을 줄이기 위해 데이터 패킷은 별도의 큐 관리자가 보관 및 관리하도록 하고, 관련정보만 갖는 작은 제어패킷으로 스케줄링 한 후 해당 데이터 패킷을 출력링크로 전송할 수도 있다.

3.2 MST 스케줄러의 동작

전단서버와 후단서버의 동작으로 나눠서 살펴보자. 참고로 등급판별기는 구현시 트래픽 흐름 처리 프로그램에 내장된다.

1) 전단서버의 동작

먼저 스케줄러의 기준시간으로 사용되는 서버 가상시간에 대해 살펴보자. 각 부서버는 독자적인 서버 가상시간을 운영한다. 각 부서버의 서버 가상시간은 그 부서버가 트래픽을 서비스하기 시작할때, 즉 서버 활성구간의 시작시점에 0의 값으로 초기화된다. 그리고 매 패킷의 전송 완료시 다음 전송할 패킷의 타임스탬프 값으로 갱신되고 각 패킷의 전송 중에는 실시간으로 증가된다. t 시점에 임의 부서버 n 의 서버 가상시간을 $P_n(t)$ 라 하고, $P_n(t)$ 의 갱신시점을 τ_j 라 하자. 여기서 j 는 전송 완료한 패킷 수를 의미하므로 0과 양의 정수 값을 갖는다.

다음으로 임의 부서버 n 에서 패킷 도착시 수행되는 EQn의 동작을 살펴보면 다음과 같다:

i) 임의 흐름 i 의 k 번째 패킷 p_i^k 가 t 시점에 도착할 때 최근에 갱신된 서버 가상시간인 $P_n(\tau_{j-1})$ 와 갱신 이후 경과된 실시간을 이용하여 서버 가상시간을 계산한다;

$$P_n(t) = P_n(\tau_{j-1}) + (t - \tau_{j-1}), \tag{3.1}$$

여기서 $\tau_{j-1} \le t < \tau_j$ 및 $P_n(0) = 0$.

ii) 패킷 p_i^k 의 타임스탬프 TS_i^k, 즉 가상 시작시각 S_i^k 를 계산하여 부착하고, 타임스탬프가 부착된 패킷을 해당 큐에 수록한다;

$$TS_i^k = S_i^k = \max(F_i^{k-1}, P_n(t)). \tag{3.2}$$

여기서 $F_i^{k-1} = S_i^{k-1} + l_i^{k-1}/r_i$ 이고 l_i^{k-1} 는 패킷 p_i^{k-1} 의 크기이다.

마지막으로 매 패킷의 전송 완료시 수행되는 DQn의 동작을 살펴보면 다음과 같다:

i) 최근에 갱신된 서버 가상시간과 그때부터 경과된 실시간을 이용하여 서버 가상시간을 계산한다;

$$P_n(t) = P_n(\tau_{j-1}) + (t - \tau_{j-1}). \tag{3.3}$$

ii) 대기중인 흐름의 선두패킷들 중 가장 앞선 타임스탬프 $TS_{\min}$ 을 갖는 패킷을 찾아서 전송하고, 서버 가상시간 $P_n(\tau_j)$ 을 갱신한다;

$$P_n(\tau_j) = \max(P_n(t), TS_{\min}) . \quad (3.4)$$

2) 후단서버의 동작

ERR기반의 후단서버는 활성화된, 즉 큐에 대기중인 패킷이 있는 가상흐름들의 목록을 의미하는 활성목록을 사용하여 최대 M 개 가상흐름의 트래픽을 순차적으로 서비스 한다. 후단서버는 활성목록의 선두에 있는 가상흐름의 패킷을 서비스한 후, 계속 활성상태이면 그 가상흐름을 활성목록의 맨 끝으로 이동한다. 그리고 비활성 가상흐름에 패킷이 도착하면 활성목록의 맨끝에 그 가상흐름을 등록한다.

활성목록에 등록된 모든 가상흐름을 한번씩 순차적으로 서비스하는 것을 라운드라 하는데, 후단서버는 이러한 라운드를 반복하면서 서비스를 수행한다. 라운드 s 에서 가상흐름 n 의 전송 허용치를 $A_n(s)$, 라운드 s에서 현재까지 전송된 가상흐름 n 의 트래픽 량을 $Sent_n(s)$ 라 하자. 가상흐름 n 는 라운드 s 에서 $Sent_n(s) < A_n(s)$ 이면 다음 패킷의 전송을 시작할 수 있다. 따라서 $Sent_n(s)$ 가 $A_n(s)$ 보다 큰, 즉 허용치를 초과한 트래픽 전송이 일어날 수 있는데, 이를 신축적 허용치(elastic allowance)라 한다. 초과하여 전송된 트래픽 량, 즉 ($Sent_n(s) - A_n(s)$)를 초과카운터 $SC_n(s)$ 에 수록하여 다음 라운드의 전송 허용치 계산에 반영한다. 라운드 s 에서 서비스된 가상흐름의 초과카운터 중 가장 큰 값을 $MaxSC(s)$ 라 하는데, $A_n(s)$ 는 다음과 같이 이전 라운드의 초과카운트값에 의해 결정된다[13];

$$A_n(s) = w_n(1 + MaxSC(s-1)) - SC_n(s-1) , \quad (3.5)$$

여기서 $w_n \equiv C_n / C_{\min}$, C_n 는 가상흐름 n , 즉 부서버 n 에 할당된 대역폭, $C_{\min} \equiv \min_{i \in S(M)} C_i$, 그리고 $S(M)$ 은 M 개 부서버의 집합, 즉 1부터 M 까지 정수의 집합이다. 실제 서비스되는 최대 패킷 크기의 비트 수를 L 이라 할 때 $SC_n(s)$ 와 $MaxSC_n(s)$ 는 $(L-1)$ 이하의 값을 갖는다.

4. 성능특성 분석

M 개의 지연등급을 지원하는 MST 스케줄러에 대해 먼저 트래픽 흐름을 모형화하고 흐름의 레이턴시를 구한 후 대역폭 이용도를 분석하고, 구현 복잡도를 살펴본다. 임의 지연등급 n 에 할당되는 대역폭을 C_n 라 하고, 복잡성을 피하기 위하여 모든 흐름은 동일한 최대 패킷크기 L 을 사용하는 것으로 가정한다.

4.1 트래픽 흐름 모형화

서비스 품질 보장이란 약속된 속도(대역폭)와 종단간 전달지연규격을 만족시키는 것을 의미하므로 스케줄러에 도착하는 트래픽 흐름의 품질특성 역시 속도와 지연규격에 의해 모형화될 수 있다. 도착 흐름의 속도를 $[a,b)$ 내 분포하는 랜덤변수 R 로 정의하고, 지연규격을 $[Q_H, Q_L)$ 내 분포하는 랜덤변수 Q 로 정의하자. 여기서 Q_H 는 가장 엄격한 지연규격이고, Q_L 는 가장 느슨한 지연규격이다. 지연규격을 M 개의 지연등급으로 그룹화할 경우 지연규격 랜덤변수 Q 는 $S(M)$ 내 분포하는 지연등급 랜덤변수 N 으로 변환될 수 있다. 그러면 도착흐름은 R 과 Q, 또는 R 과 N 의 두 요소로 구성되는 2차원 랜덤벡터로 표현된다.

4.2 레이턴시 특성

MST 스케줄러에서 임의 지연등급 n 내 흐름 i의 레이턴시는 부서버 n 과 후단서버에서 그의 레이턴시들에 의해 결정된다. 먼저 이들 레이턴시 사이의 의존성을 살펴보자.

보조정리 1: MST 스케줄러에서 임의 지연등급 n 내 흐름 i 에 대해 부서버 n 에서 그의 레이턴시와 후단서버에서 그의 레이턴시는 서로 독립적이다.

증 명: 부서버 n 는 C_n 의 대역폭을 갖는 독립된 ST 스케줄러로 동작하므로 부서버 n 에서 흐름 i 의 레이턴시는 타 부서버와 후단서버의 영향을 받지 않는다. 후단서버 역시 C 의 대역폭을 갖는 독립된 ERR 스케줄러로 동작하므로 후단서버에서 임의 가상흐름의 레이턴시는 전단서버의 영향을 받지 않는다. 따라서 부서버 n 에서의 레이턴시와 후단서버에서의 레이턴시는 서로 독립적이다.

보조정리 1로부터 임의 지연등급내 흐름의 레이턴시는 다음과 같이 구해진다.

정리 1: MST 스케줄러에서 임의 지연등급 n 내 흐름의 레이턴시 D^n 는 다음과 같다;

$$D^n = \frac{V_n^{ST} L}{C_n} + \frac{(W - w_n)L + (L-1)(M-1)}{C} , \quad (4.1)$$

여기서 $W \equiv \sum_{n=1}^{M} w_n$ 이고 V_n^{ST} 는 부서버 n 에 수용되

는 최대 흐름 수이다.

증 명: 후단서버에서 지연등급 n 트래픽, 즉 가상흐름 n의 레이턴시를 구하면 ERR 방식을 적용하므로 다음과 같이 계산된다[13];

$$\frac{(W-w_n)L+(L-1)(M-1)}{C}. \tag{4.2}$$

대역폭이 C_n인 부서버 n에서 임의 흐름의 레이턴시는 (2.2)로부터 다음과 같이 주어진다.

$$\frac{V_n^{ST}L}{C_n}. \tag{4.3}$$

따라서 보조정리 1로부터 MST 스케줄러에서 지연등급 n내 흐름의 레이턴시는 부서버 n에서 그 흐름의 레이턴시와 후단서버에서 가상흐름 n의 레이턴시의 합이 되므로 (4.2)과 (4.3)에 의해 (4.1)이 증명된다.

4.2 대역폭 이용도

후단서버는 M개 부서버로부터 입력되는 트래픽을 순차적으로 서비스하므로 작업보존형 (work-conserving) 특성을 가져 대역폭 손실이 없다. 따라서 MST 스케줄러의 대역폭 이용도는 M개 부서버의 대역폭 이용도를 그들에게 할당한 스케줄러 대역폭 측면에서 가중 평균하여 구할 수 있다. M개의 지연등급을 지원하는 FT 스케줄러의 대역폭 이용도 역시 M개 지연등급 트래픽의 대역폭 이용도를 그들에게 할당한 대역폭 측면에서 가중 평균하여 구할 수 있다. MST 스케줄러의 성능 비교 모델인 FT 스케줄러의 대역폭 이용도를 먼저 구한 후 MST 스케줄러의 대역폭 이용도를 구한다. 한편 본 연구에서 사용되는 대역폭 이용도는 할당된 대역폭 중 트래픽이 요구하는 속도뿐만 아니라 요구 지연규격을 모두 만족시키면서 실제 트래픽 전송에 사용된 대역폭의 비율을 의미한다.

1) FT 스케줄러

요구속도가 r_i인 임의 흐름 i에 대해 그의 요구지연규격을 만족시키기 위해 필요한 최저속도를 흐름 i의 임계속도 r_i^{crt}, 흐름 i를 위해 할당하는 대역폭을 예약속도 r_i^{res}라 하자. 흐름의 요구속도가 임계속도보다 빠르면 요구속도로 예약하고, 느리면 임계속도로 예약해야 하므로 $r_i^{res}=\max(r_i,r_i^{crt})$가 된다. 참고로 $r_i<r_i^{crt}$인 경우 $(r_i^{crt}-r_i)$에 해당하는 대역폭만큼 더 많은 서버 대역폭을 예약해야 하는 과잉예약 현상이 발생하며 과잉예약분 만큼의 대역폭이 손실된다.

흐름 i의 지연규격을 Q_i라 할 때 (2.1)로부터 $r_i^{crt}=CL/(CQ_i-L)$가 되므로 각 지연규격은 자신의 고유한 임계속도를 갖는다. 따라서 $[Q_H,Q_L)$, 여기서 $Q_H<Q_L$, 내 분포하는 지연규격 랜덤변수 Q로부터 $[c,d)$, 여기서 $c\equiv CL/(CQ_L-L)$ 및 $d\equiv CL/(CQ_H-L)$, 내 분포하는 임계속도 랜덤변수 R^c를 얻을 수 있다. 편의상 지연규격 대신 그에 대응되는 임계속도를 사용하여 분석을 진행한다. 임의 지연등급 n의 임계속도 범위를 $[c_n,d_n)$라 하자.

지연등급 n 트래픽 흐름에 대한 총 예약속도가 대역폭 C_n을 초과할 수 없으므로 지연등급 n내 최대 수용 흐름 수 V_n^{FT}는 다음과 같이 계산된다;

$$\begin{aligned}V_n^{FT}&=\arg_k\{k\mid\sum_{i=1}^{k}r_i^{res}\le C_n \ \ and \ \ \sum_{i=1}^{k+1}r_i^{res}>C_n\}\\&=\left\lfloor\frac{C_n}{r_n^{res}}\right\rfloor,\end{aligned} \tag{4.4}$$

여기서 $r_n^{res}\equiv E[\max(R,R^c)\mid R^c\in[c_n,d_n)]$. 따라서 FT 스케줄러의 대역폭 이용도 ρ_{BW_FT}는 다음과 같이 구해진다;

$$\rho_{BW\_FT}=\frac{1}{C}\sum_{n=1}^{M}C_n\rho_{BW\_FT}^n, \tag{4.5}$$

여기서 $\rho_{BW\_FT}^n\equiv\frac{1}{C_n}\sum_{i=1}^{V_n^{FT}}r_i=\frac{r_n^{req}}{C_n}\left\lfloor\frac{C_n}{r_n^{res}}\right\rfloor$ 및 $r_n^{req}\equiv E[R\mid R^c\in[c_n,d_n)]$.

한 예로 R과 R^c가 모두 균등분포하고, 요구지연규격은 요구속도에 무관하며, $a\le c<d\le b$ 인 트래픽 부하 하에서 ρ_{BW_FT}를 구해보자. 먼저 r_n^{req}와 r_n^{res}를 계산하면 각각 다음과 같다;

$$\begin{aligned}r_n^{req}=E[R\mid R^c\in[c_n,d_n)]&=\frac{\int_{c_n}^{d_n}\int_{a}^{b}rf_R(r)drf_{R^c}(t)dt}{\int_{c_n}^{d_n}f_{R^c}(t)dt}\\&=\int_{a}^{b}rf_R(r)dr=\frac{a+b}{2}.\end{aligned} \tag{4.6}$$

$$
\begin{aligned}
r_n^{res} &= E[\max(R, R^c) \mid R^c \in [c_n, d_n)] \\
&= \frac{\int_{c_n}^{d_n}[t\int_a^t f_R(r)dr + \int_t^b r f_R(r)dr] f_{R^c}(t)dt}{\int_{c_n}^{d_n} f_{R^c}(t)dt} \\
&= \frac{d-c}{d_n - c_n}\int_{c_n}^{d_n}[t\int_a^t f_R(r)dr + \int_t^b r f_R(r)dr] f_{R^c}(t)dt \\
&= \frac{1}{d_n - c_n}\int_{c_n}^{d_n}(\frac{t^2 - 2at + b^2}{2(b-a)})dt \\
&= \frac{({c_n}^2 + c_n d_n + {d_n}^2) - 3a(c_n + d_n) + 3b^2}{6(b-a)}.
\end{aligned} \tag{4.7}
$$

따라서

$$
\begin{aligned}
\rho_{BW\_FT} &\cong \frac{1}{C}\sum_{n=1}^{M}\frac{C_n r_n^{req}}{r_n^{res}} \\
&= \frac{1}{C}\sum_{n=1}^{M}\frac{3C_n(a+b)(b-a)}{(c_n^2 + c_n d_n + d_n^2) - 3a(c_n + d_n) + 3b^2}.
\end{aligned} \tag{4.8}
$$

2) MST 스케줄러

먼저 임의 부서버 n가 수용할 수 있는 최대 흐름 수 V_n^{ST}를 구해보자. V_n^{ST}는 수용 흐름의 요구속도의 합이 C_n을 초과하지 않아야 하는 요구속도 수용조건과 (4.1)에 의해 결정되는 레이턴시가 지연규격을 초과하지 않아야 하는 지연규격 수용조건의 두 가지 조건을 모두 만족하는 최대 흐름 수가 된다. 요구속도 수용측면에서 흐름수의 상한치 $V_n^{ST_R}$는 다음과 같이 계산된다;

$$
V_n^{ST\_R} = \{k \mid \sum_{i=1}^{k} r_i \le C_n \ \ and \ \ \sum_{i=1}^{k+1} r_i > C_n\} = \left\lfloor \frac{C_n}{r_n^{req}} \right\rfloor. \tag{4.9}
$$

지연규격 수용측면에서 흐름 수의 상한치 $V_n^{ST_Q}$는 (4.1)로부터 다음과 같이 표현된다;

$$
\begin{aligned}
V_n^{ST\_Q} &= \max\arg_x\{\text{integer } x \mid \frac{xL}{C_n} \\
&\quad + \frac{(W - w_n)L + (L-1)(M-1)}{C} \le Q_n\}.
\end{aligned} \tag{4.10}
$$

이를 전개하면

$$
V_n^{ST\_Q} = \left\lfloor \frac{Q_n C_n}{L} - \frac{C_n(W - w_n)L + C_n(L-1)(M-1)}{LC} \right\rfloor. \tag{4.11}
$$

따라서 ρ_{BW_MST}는 다음과 같이 계산된다;

$$
\rho_{BW\_MST} = \frac{1}{C}\sum_{n=1}^{M} C_n \rho_{BW\_ST}^n, \tag{4.12}
$$

여기서 $\rho_{BW\_ST}^n \equiv \frac{1}{C_n}\sum_{i=1}^{V_n^{ST}} r_i = \frac{r_n^{req}}{C_n} V_n^{ST}$ 및 $V_n^{ST} \equiv \min(V_n^{ST\_R}, V_n^{ST\_Q})$.

4.3 구현 복잡도

스케줄러의 구현 복잡도는 스케줄링 알고리즘 수행상의 제약을 평가하기 위한 것으로 하나의 패킷을 처리하는데 요구되는 연산과정의 반복횟수로 표시되며, 가장 복잡한 내부장치의 구현복잡도에 의해 스케줄러의 전체의 구현 복잡도가 결정된다[12]. MST 스케줄러의 구현 복잡도를 살펴보자. 임의 부서버 n에서 구현복잡도는 EQn의 구현복잡도와 DQn의 구현복잡도 중 큰값이 된다. EQn은 수용되는 흐름의 수에 상관없이 도착하는 패킷마다 타임스탬프 계산과정((3.1)과 (3.2))을 한번씩 수행하므로 이의 구현복잡도는 $O(1)$이 된다. 반면 DQn은 매 패킷 전송에 앞서 수용된 흐름들의 선두 패킷 중 가장 앞선 타임스탬프를 갖는 패킷을 찾는데, 이때 수용 흐름의 수가 K라 할 때 $\log K$번의 분류(sorting)과정이 반복수행되므로 DQn의 구현 복잡도는 $O(\log K)$가 된다. 따라서 부서버 n의 구현복잡도는 $O(\log K)$가 된다. 한편 후단서버는 ERR기반 스케줄러이므로 $O(1)$의 구현복잡도를 갖는다[13]. 그런데 MST 스케줄러에서 M개의 부서버가 V개의 흐름을 수용하므로 모든 부서버가 동일한 흐름 수를 수용할 경우 $K = V/M$가 된다. 따라서 MST 스케줄러의 구현 복잡도는 가장 구현복잡도가 높은 DQn의 구현복잡도인 $O(\log K)$가 되고, $O(\log K) = O(\log V) - O(\log M)$ 이므로 $O(\log V)$의 이상적 구현복잡도를 갖는 FT 스케줄러의 그것에 비해 최대 $O(\log M)$만큼 복잡도를 낮출 수 있다.

5. 성능 비교

성능비교에 앞서 공정 패킷 스케줄러가 지원해야 할 지연등급의 수와 지연규격에 대해 살펴보자. 대표적인 품질보장 서비스인 인터넷 전화의 경우 종단간 지연이 국내의 경우 150ms이하의 단일 품질등급만 정의하였으나 일본의 경우 A등급(100ms이하), B등급(150ms이하), 및 C등급(400ms이하)의 3 등급으로 정의되어 있다[14]. 앞으로 많은 사업자가 출현하여 인터넷 전화 서비스의 다양화, 고급화 등을 경쟁적으

로 추구할 경우 종단간 지연은 한층 더 차별화 될 것으로 예상되어 본 연구에서는 최대 30개의 지연등급을 가정한다. 종단간 지연은 각 전송선로에서의 전파지연과 각 노드에서의 전달지연의 합이 되는데, 본 연구에서는 종단간 전파지연이 60ms이고 홉 수가 30개(각 노드에서의 지연은 동일)이며, 종단간 지연시간이 [90ms, 150ms]내 임의 값으로 제한되는 국제 인터넷 전화 서비스를 고려한다.

이제 4장의 분석결과, 즉 (4.8)과 (4.12)를 사용하여 국제 인터넷 전화 서비스 환경하에서 기존 FT 스케줄러와 MST 스케줄러의 성능을 비교해보자. 구체적인 성능비교 환경은 다음과 같다; 스케줄러의 출력링크는 10Gbps의 속도를 갖고, 최대 30개의 지연등급을 지원하며, 각 등급의 지연규격은 1ms부터 3ms까지 0.1ms 간격으로 증가하는 값을 갖는다. 2Mbps급 영상, 200Kbps급 오디오 및 20Kbps급 음성의 세가지 흐름이 동일한 확률로 스케줄러에 도착하고, 도착 흐름의 지연등급은 그의 요구속도에 상관없이 지원되는 모든 지연등급에 균등하게 분포한다.

1000비트의 고정된 패킷크기 조건하에서 지연등급 수의 증가에 따른 대역폭 이용도 변화가 그림 2에 비교되어 있다. 이 그림에서 MST(M)은 M개의 지연등급을 모두 수용한 MST 스케줄러를 의미하고, MST(k)는 $k(<M)$개의 지연등급을 수용한 MST 스케줄러를 의미하는데, MST(k)의 경우 k번째 지연등급은 등급 k부터 M까지 $(M-k+1)$개의 등급 트래픽을 모두 수용하는 대신 이들 $(M-k+1)$개의 지연등급에 할당될 스케줄러 대역폭을 모두 할당받는다. 참고로 MST(1)은 바로 ST 스케줄러가 된다. 그림 2에서 수용 지연등급이 두 개 이상인 경우, 즉 MST($k(\geq 2)$)는 모두 FT 스케줄러보다 우수한 대역폭 이용도 특성을 갖고, 특히 MST(M)의 대역폭 이용도는 FT 스케줄러의 그것에 비해 최대 50% 이상 높다. 그리고 MST 스케줄러의 구현 복잡도는 $O(\log(V/30)) = O(\log V) - O(4.9)$ 이므로 FT 스케줄러의 구현 복잡도에 비해 $O(4.9)$만큼 낮아진다.

그림 3은 각 지연등급의 지연규격이 1ms부터 1.9ms까지 0.1ms 간격으로 증가하는 10개의 지연등급 하에서 패킷크기의 증가에 따른 대역폭 이용도를 비교한 것이다. 패킷크기가 증가할수록 대역폭 이용도가 저하되는데, 이는 패킷크기가 증가하면 동일한 지연규격을 얻기 위해 FT 스케줄러의 경우 각 흐름의 예약속도를 높여야 하고, MST 스케줄러의 경우 지연규격 수용측면에서 최대 수용 흐름 수를 더욱 줄여야 하기 때문으로 이해된다. 그림 3에서 MST($k(\geq 3)$)는 모두 FT 스케줄러보다 우수한 대역폭 이용도를 갖고, 특히 MST(10)는 FT 스케줄러에 비해 최대 50% 이상 높은 대역폭 이용도를 보인다.

그림 4는 모든 패킷을 RTP(Real Time Protocol)의 오버헤드인 432비트의 헤더와 유료부하를 실어나르는 바디로 구성할 경우 그림 3의 대역폭 이용도에 대한 유료부하 이용도를 도시한 것이다. 유료부하 이용도 ρ_{PL}는 다음과 같이 계산된다; $\rho_{PL} = \rho_{BW}(l-432)/l$, 여기서 ρ_{BW}는 대역폭 이용도이고 l은 패킷크기다.

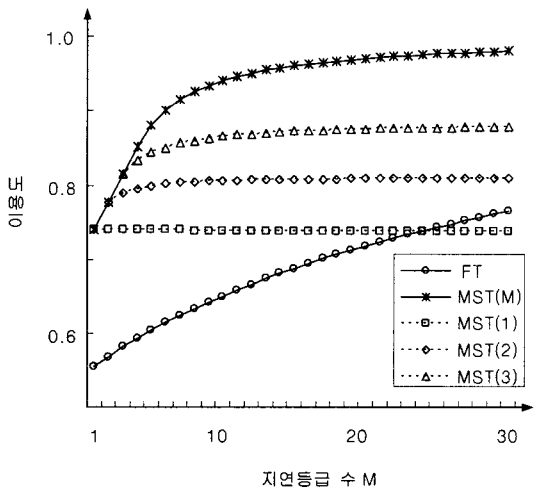


그림 2. 지연등급 수에 따른 대역폭 이용도 비교

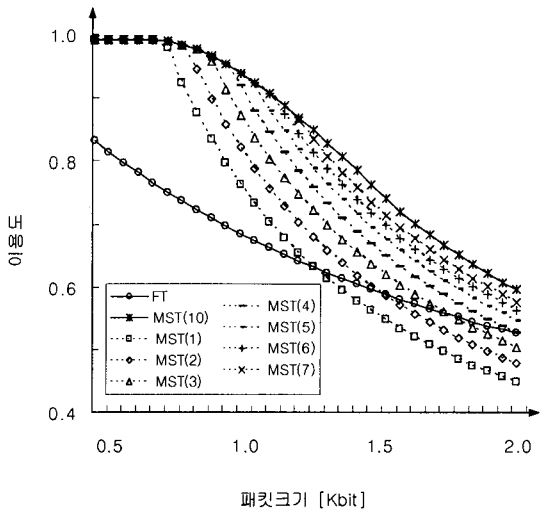


그림 3. 패킷크기에 따른 대역폭 이용도 비교($M=10$)

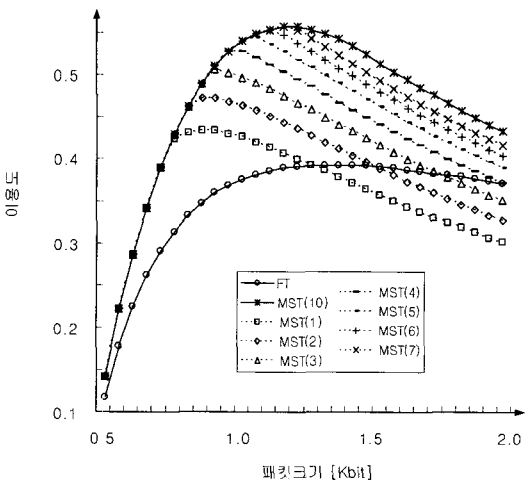


그림 4. 패킷크기에 따른 유료부하 이용도 비교 (M =10)

그림에서 유료부하 이용도는 특정 패킷크기에서 최대값을 갖는데, 이는 패킷크기가 작아지면 패킷내 유료부하의 비율이 낮아져 유료부하 이용도가 저하되고, 패킷크기가 증가하면 대역폭 이용도가 낮아져 유료부하 이용도가 역시 저하되기 때문이다. 그림 3의 비교결과와 마찬가지로 MST 스케줄러가 FT 스케줄러 보다 우수한 유료부하 이용도 특성을 보인다.

6. 결 론

타임스탬프기반 공정 패킷 스케줄러는 종료시각(FT)과 시작시각(ST) 방식으로 나뉘는데, ST 방식은 우수한 대역폭 이용도 특성을 가지나 다수 지연규격의 지원이 어려워 제대로 활용되지 못하고 있다. 이러한 문제를 해결할 수 있는 MST 스케줄러가 본 논문에서 제안되었다.

제안된 MST 스케줄러는 다수의 부서버를 갖는 전단서버와 ERR(Elastic Round Robin)방식으로 전단서버내 부서버들이 출력링크를 공평하게 사용하도록 해주는 후단서버로 구성되며, 각 부서버는 독립된 ST 방식 기반 스케줄러로 동작하므로 임의의 지연규격을 지원할 수 있다.

제안된 MST 스케줄러의 레이턴시 특성을 분석하고 이로부터 대역폭 이용도를 구하였다. 또한 구현 복잡도를 분석하였는데 기존 FT 방식 기반 스케줄러에 비해 최대 O(log(지연등급 수)) 만큼 구현 복잡도를 낮출 수 있었다. 실제적인 트래픽 환경에서 성능 특성을 비교한 결과 MST 스케줄러는 기존 FT 방식 기반 스케줄러 보다 최대 50% 이상 높은 대역폭 이용도 특성을 보였다.

참 고 문 헌

[1] X. Xiao and L. M. Ni, "Internet QoS: A Big Picture," *IEEE Network*, Vol. 13, No. 2, pp. 8-18, 1999.

[2] A. K. Parekh, *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*, PhD [2] A.K. Parekh, *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*, PhD dissertation, Massachusetts Institute of Technology, 1992.

[3] A. Demers, S. Keshav, and S. Shenker, "Design and analysis of a fair queuing algorithm," *Proc. ACM SIGCOMM*, pp. 1 - 12, 1989.

[4] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE JSAC*, Vol. 9, No. 8, pp. 1265-1279, 1991.

[5] S. J. Golestani, "A Self-Clocked Fair Queuing Scheme for Broadband Applications," *Proc. IEEE INFOCOM* '94, pp. 636-646, 1994.

[6] D. Stiliadis and A. Varma, "Rate Proportional Servers: A Design Methodology for Fair Queueing Algorithms," *IEEE/ACM Trans. Networking*, Vol. 6, No. 2, pp. 164-174, 1998.

[7] D. Stiliadis and A. Varma, "Efficient Fair Queuing Algorithms for Packet-Switched Networks," *IEEE/ACM Trans. Networking*, Vol. 6, No. 2, pp. 175-185, 1998.

[8] Dong-Yong KWAK, Nam-Seok KO, and Hong-Shik PARK, "Medium Starting Potential Fair Queueing for High-Speed Networks," *IEICE Trans. Communications*, Vol. E87-B, No. 1, pp. 188-198, 2004.

[9] Dong-Yong Kwak, Nam-Seok Ko, Bongtae Kim, and Hong-Shik Park,"A New Starting Potential Fair Queuing Algorithm with O(1)

Virtual Time Computation Complexity," *ETRI Journal*, Vol. 25, No. 6, pp. 475-488, 2003.

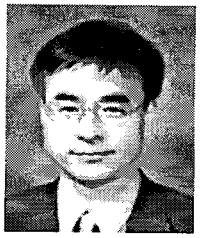
[10] P. Goyal, H. M. Vin, and H. Cheng, "Start-time fair queueing: A scheduling algorithm for integrated services packet switching networks," *IEEE/ACM Trans. Networking*, Vol. 5, No. 5, pp. 690-704, 1997.

[11] H. Zhou, *Real-Time Services over High Speed Network*, Ph.D Dissertation, Curtin University of Technology, Australia, 2002.

[12] J. Xu and R. J. Lipton, "On Fundamental Tradeoffs between Delay Bounds and Computational Complexity in Packet Scheduling Algorithms," *ACM SIGCOMM'2002*, pp. 279-292, 2002.

[13] S. S. Kanhere, H. Sethu, and A. B. Parekh, "Fair and efficient packet scheduling using elastic round robin," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 13, No. 3, pp. 324-326, 2002.

[14] 이은곤, "인터넷 전화 국내외 제도화 현황 및 정책적 시사점", 정보통신정책, 제16권, 제17호, pp. 1-17, 2004.



김 태 준

1980년 2월 경북대학교 전자공학과 졸업
1982년 2월 한국과학기술원 전자공학 석사
1999년 8월 한국과학기술원 전자공학 박사
1982년 3월 한국전자통신연구원
1996년 3월 천안공업대학 교수
2005년 3월~현재 공주대학교 정보통신공학부 부교수
관심분야 : 고속통신망, VoIP, 트래픽제어