

최적 레이턴시 기반 공정 큐잉 알고리즘의 구현 복잡도 개선

종신회원 김 태 준*, 서 봉 수**

Improving the Implementation Complexity of the Latency-Optimized Fair Queuing Algorithm

Tae-joon Kim*, Bongsue Suh** *Lifelong Members*

요 약

서비스 품질을 보장하기 위해 널리 사용되는 WFQ(Weighted Fair Queuing)는 엄격한 지연한계를 요구하는 저속 트래픽 흐름의 경우 자원 이용도가 아주 나빠지는 구조적인 한계를 갖고 있다. 이러한 WFQ의 문제점은 최적화 되지 않은 트래픽 흐름의 레이턴시에 기인하는 것으로 밝혀졌고, 최적 레이턴시를 실현하여 자원 이용도를 높일 수 있는 LOFQ(Latency-Optimized Fair Queuing)가 도입되었다. 그러나 LOFQ는 새로운 흐름을 수락할 때마다 모든 흐름에 대해 최적의 레이턴시를 재 설정해야 하므로 흐름 수락시 $O(N^2)$ 의 복잡도를 가져 구현에 어려움이 있다. 본 논문은 이러한 복잡도를 $O(1)$ 수준으로 낮추는 방식을 제안한다. 제안된 방식은 가해진 트래픽 부하의 서비스 품질의 통계적인 특성을 이용하여 최적 레이턴시 지수 함수를 구한 후 이 함수를 이용하여 도착 흐름의 최적 레이턴시 지수를 간단하게 구하는 것이다.

Key Words : WFQ, fair queuing, packet scheduler, quality-of-service, resource reservation

ABSTRACT

WFQ(Weighted Fair Queuing) is the most popular fair queuing algorithm to guarantee the Quality-of-Service(QoS), but it has the inherent drawback of a poor resource utilization, particularly under the low rate traffic requiring a tight delay bound. It was recently identified that the poor utilization is mainly due to non-optimized latency of a traffic flow and then LOFQ(Latency-Optimized Fair Queuing) to overcome the drawback was introduced. The LOFQ algorithm, however, renews their optimal latencies for all flows whenever a new flow arrives, which results in the high implementation complexity of $O(N^2)$. This paper is to reduce the complexity to $O(1)$. The proposed method is first to derive the optimal latency index function from the statistical QoS characteristics of the offered load, and then to simply calculate the optimal latency index of the arriving flow using the function.

I. 서 론

서비스 품질(QoS: Quality of Service)의 보장을 요구하는 실시간 통신 서비스를 수용하기 위해 IETF(Internet Engineer Task Force)에서 자원예약

기반의 종합 서비스(IntServ) 모델을 제시하였고^[1], 자원예약을 위해 RSVP(Resource reSerVation Protocol)를 표준화 하였다^[2]. IntServ 모델에서 서비스 품질을 보장하기 위해서 RSVP-라우터가 사용 되는데, RSVP-라우터는 트래픽 흐름의 요구 속도를

* 공주대학교 정보통신공학부(bsuh@kongju.ac.kr) (° : 교신저자)

논문번호 : KICS2012-03-106, 접수일자 : 2012년 3월 1일, 최종논문접수일자 : 2012년 5월 31일

보장하고 허용되는 레이턴시(latency), 즉 지연(delay) 규격을 위반하지 않기 위해 공정 패킷 스케줄러를 탑재한다.

이상적 패킷 스케줄링 방식인 GPS(General Processor Sharing)^[3]는 WFQ (Weighted Fair Queuing)^[4]에 의해 구현되었다. WFQ는 흐름간의 상호 간섭을 차단하여 각 흐름의 품질 특성이 다른 흐름으로부터 영향을 받지 않도록 하는 흐름 분리와 차별화된 품질 보장을 지원하는 등 공정 큐잉의 요구사항을 충실히 따르기 때문에 IETF IntServ 모델의 RSVP-라우터에 적용되었다^[5]. 그러나 흐름의 스케줄링 속도를 높일 경우 스케줄링 속도와 요구 속도와의 차이만큼의 대역폭 자원이 손실되는 문제가 있다^[6]. QoS 보장을 요구하는 실시간 트래픽이 인터넷 전화와 같은 사람간 통신에서 이제 센서 네트워크와 같은 사물간 통신으로 발전하고 있다^[7,8]. 분산 제어나 감시, 상태 정보 등의 단순 데이터를 주고받는 사물간 통신은 영상 콘텐츠를 위주로 하는 사람간 통신보다 속도가 낮은 반면 중단간 지연 시간이 엄격해지는 특징을 갖는다. 따라서 사물간 통신 트래픽이 많은 부분을 차지할 미래 인터넷에서 품질보장 서비스를 제공하는데 있어 WFQ의 문제점은 더욱 심화될 것이다.

WFQ의 자원 이용도 저하 문제를 해결하기 위해 다양한 연구가 이루어졌다. 음성 트래픽의 지원을 위해 감시기를 갖는 WFQ 방식은 음성 흐름의 스케줄링 속도를 다소 과다하게 할당하는 대신 필요 시 감시기가 패킷을 드롭(drop)하도록 하였다^[9]. 가변 스케줄링 속도를 사용하는 최적 네트워크 서비스 곡선 방식^[10]과 크레딧(Credit) 기반 프로세서 공유방식^[11]은 일시적으로 예약된 속도보다 높은 스케줄링 속도를 요구하는 흐름들에 대해서 대역폭에 여유가 있는 다른 흐름의 대역폭을 빌려주는 기법을 사용하였다. 그러나 빌려줄 만큼 대역폭의 여유를 가진 흐름이 줄어들 경우 그 효과가 반감되는 문제가 있다. 한편 ^[12]에서 어떤 흐름의 여유 레이턴시를 다른 흐름에 할당되는 대역폭 자원을 줄이는데 재활용할 수 있는 LOFQ (Latency-Optimized Fair Queuing)가 제안되었다^[13].

LOFQ는 새로운 흐름 수락시 모든 흐름에 대하여 최적 레이턴시 값을 재 설정하느라 흐름 수락시 $O(N^2)$ 의 복잡도를 갖는 구현상의 복잡성 문제가 대두된다. 본 연구에서는 이러한 구현 복잡도를 $O(1)$ 수준으로 낮추는 방식을 제안한다. 제안된 방식은 가해진 트래픽 부하의 서비스 품질의 통계적인 특

성을 이용하여 최적 레이턴시 지수 함수를 구한 후 이 함수를 이용하여 도착 흐름의 최적 레이턴시 지수를 간단하게 구하는 것이다.

II. 기존 연구결과 분석 및 문제점 제시

WFQ 스케줄러에 도착하는 임의 흐름의 요구속도를 r , 스케줄러가 지켜야 할 흐름의 지연한계, 즉 지연규격을 b , 흐름의 최대 패킷 크기를 L 이라 하자. WFQ에서 임의 흐름의 고유 레이턴시 q^r 은 다음과 같이 주어진다^[4].

$$q^r = \frac{L}{r} + \frac{L^{\max}}{C^B}. \quad (2.1)$$

(2.1)에서 C^B 는 출력링크의 용량으로 스케줄러의 총 대역폭 자원 량을 의미하고, L^{\max} 는 스케줄러내 모든 흐름의 최대 패킷크기이다. WFQ는 요구속도와 지연규격을 동시에 만족시켜야 하므로 흐름에 할당되는 대역폭, 즉 흐름의 예약속도 s 는 (2.1)로부터 다음과 같이 계산된다. 참고로 스케줄러가 흐름을 예약속도로 스케줄링하므로 예약속도를 스케줄링 속도라고도 한다.

$$s = \max(r, r^C) \text{ 및 } r^C = \frac{LC^B}{bC^B - L^{\max}}. \quad (2.2)$$

(2.2)에서 r^C 는 $q=b$ 가 되는, 즉 흐름의 요구속도에 의해 결정되는 레이턴시가 바로 그 흐름이 요구하는 지연규격과 일치하는 r 값으로 임계 요구속도라 한다. 그러면 WFQ에서 임의 흐름의 고유 레이턴시 q^r 은 다음과 같이 그 흐름의 지연규격 b 를 위반하지 않는 값을 갖게 된다.

$$q^r = \frac{L}{r} + \frac{L^{\max}}{C^B} \leq b. \quad (2.3)$$

고유 레이턴시 q^r 과 지연규격 b 는 (2.2)와 (2.3)으로부터 $q^r=b$, $q^r>b$ 또는 $q^r<b$ 의 관계를 갖는다. [12]에서 WFQ의 문제점을 분석하기 위해 흐름을 그의 q^r 과 b 또는 r^C 와 r 사이의 관계 측면에서 다음과 같이 분류하였다: $q^r<b$ 또는 $r>r^C$ 인 속도제약(RR: Rate-Restricted) 흐름, $q^r<b$ 또는 $r<r^C$ 인 지연제약(DR: Delay-Restricted) 흐름 그리고 $q^r=b$ 또는 $r=r^C$ 인 최적(OP: Optimal) 흐름. DR 흐름의 경우 ($b-q^r$)만큼의 초과 레이턴시를 줄이기 위해 스케줄링 속

도를 높여야 하므로 그 만큼 대역폭이 낭비되는 반면 RR 흐름의 경우 $(b-q)$ 의 남은 레이턴시가 있지만 스케줄링 속도를 요구속도 이하로 낮출 수 없어 남은 레이턴시, 즉 잉여 지연 자원을 낭비하게 된다. 이러한 지연 자원의 낭비가 바로 WFQ의 자원 이용도를 저하시키는 주된 원인으로 분석하였다[12].

[12]에서 WFQ의 문제점을 해결하기 위해 LOFQ를 제안하였다. LOFQ는 레이턴시 지수 $\beta(\geq 0)$ 를 도입하여 임의 흐름 i 의 k 번째 패킷 P_i^k 의 타임스탬프 TS_i^k 를 다음과 같이 계산한다.

$$TS_i^k = \max[S_i^k, v(t)] + \frac{\min[1, \beta_i] L_i^k}{h_i}. \quad (2.4)$$

여기서 S_i^k 는 k 번째 패킷의 전송 시작시간, L_i^k 는 P_i^k 의 크기, h_i 는 흐름 i 의 스케줄링 속도, $v(t)$ 는 서버 가상시간이다.

흐름의 타임스탬프가 그의 레이턴시 지수 β 값에 따라 패킷의 전송 시작시간(start-time)부터 종료시간(finish-time)까지 임의의 값을 가질 수 있다. 참고로 $i \geq 1$ 일 경우 LOFQ는 바로 종료시간 기반 타임스탬프 계산방식을 적용하는 WFQ가 된다. $(0,1)$ 범위의 β 를 갖는 임의의 DR 흐름 i 는 WFQ 방식에 비해 $\Delta T = (1-\beta)L_i/h_i$ 만큼 타임스탬프 값을 줄여서, 여기서 L_i 는 흐름 i 의 최대 패킷크기, ΔT 만큼 패킷을 더 빨리 전송한다. 이 결과 흐름 i 는 ΔT 만큼 레이턴시를 줄일 수 있다. 하지만 다른 흐름에게 $\Delta D = (1-\beta)L_i/C^B$ 만큼의 추가지연, 즉 ΔD 의 레이턴시 바이어스를 초래한다. 그런데 RR 흐름의 경우 잉여 레이턴시로 추가지연을 상쇄할 수 있어 지연규격의 준수에 문제가 없다. 결과적으로 DR 흐름의 경우 스케줄링 속도를 높이지 않고도 자신의 레이턴시를 줄일 수 있어 대역폭 이용 효율을 개선할 수 있게 된다. 요약하면 WFQ에서 RR 흐름의 남은 지연시간은 쓸모가 없지만, LOFQ에서는 이를 지연 자원에 축적한 후 DR 흐름의 초과 레이턴시를 줄이는데 사용, 즉 잉여 지연자원의 재사용을 통해 성능개선을 달성하는 것이다.

흐름을 많이 수락할수록 흐름이 발생시키는 레이턴시 바이어스의 합 D 는 증가하지만 어떤 값 D^B 를 초과할 수 없다. 따라서 대역폭(BW: BandWidth)과 마찬가지로 레이턴시 바이어스를 일종의 자원으로 취급할 수 있다. 이를 지연 자원(DW: Delay Width)이라 하고, D^B 와 D 를 각각 스케줄러의 DW 용량과 스케줄러의 점유 DW라 한다. 참고로 D^B 는

스케줄러에서 가장 엄격한 지연규격의 값을 갖는다.

LOFQ에서 임의 흐름의 레이턴시 q 는 (2.5)와 같이 레이턴시 지수 β 와 스케줄링 속도 $h(\geq r)$ 의 함수로 표현한다. 여기서 h 는 바로 그 흐름이 점유하는 대역폭 자원이 된다. (2.3)과 같이 $q \leq b$ 로 주어져 최적화 되지 않는 WFQ의 레이턴시 q 와 달리 LOFQ에서는 q 를 도입하여 $q=b$, 즉 요구 지연한계 b 와 일치되는 최적 레이턴시 q 를 실현하고자 하는 것이다.

$$q = D^B + \beta \frac{L}{h} = b. \quad (2.5)$$

β 의 레이턴시 지수를 갖는 흐름에 의해 증가되는 레이턴시 바이어스 값, 즉 그 흐름의 점유 DW가 되는 d 는 다음과 같이 주어진다.

$$d = \frac{\max(0, 1 - \beta)L}{C^B}. \quad (2.6)$$

(2.5)와 (2.6)으로부터 흐름의 점유 BW h 와 점유 DW d 는 서로 반비례하므로, 점유 BW를 늘리는 대신 점유 DW를 줄일 수 있고, 반대로 점유 DW를 늘리는 대신 점유 BW를 줄일 수 있는, 즉 DW와 BW가 상호 변환될 수 있음을 알 수 있다. 여기서 전자를 B2D 변환, 후자를 D2B 변환이라 한다. 그리고 늘어나는 BW 대비 줄어드는 DW의 비율을 B2D 변환효율 E^{B2D} , 늘어나는 DW 대비 줄어드는 BW의 비율을 D2B 변환효율 E^{D2B} 라 한다. 참고로 D2B 변환가능 흐름은 $\beta \in (0,1]$ 및 $h > r$ 인 h 를 가지며, B2D 변환가능 흐름은 $\beta \in (0,1)$ 인 값을 갖는다. E^{B2D} 와 E^{D2B} 는 다음과 같이 주어진다[12].

$$E^{B2D} = \frac{b - D^B}{C^B} = \frac{\beta L}{h C^B} = \frac{1}{E^{D2B}}. \quad (2.7)$$

LOFQ 스케줄러는 대역폭 자원만 관리하는 WFQ 스케줄러와 달리 지연 자원과 대역폭 자원의 두 가지 자원을 취급하며, B2B 변환을 통해 흐름 별로 보다 효율이 좋은 유형의 자원을 사용하도록 한다[13]. B2B 변환이란 임의의 두 흐름 i 와 j 에 대해 흐름 i 의 B2D 변환으로 가용 DW를 획득한 후 이를 흐름 j 의 D2B 변환에 투입하는 일련의 과정을 말한다. 흐름 i 의 점유 BW 증가량을 Δh_i , 흐름 j 의 점유 BW 감소량을 Δh_j 라 할 때 $E_{ij}^{B2B} = \Delta h_j / \Delta h_i = E_i^{B2D} * E_j^{D2B}$ 로 정의되는 E_{ij}^{B2B} 를 B2B 변환효율

이라 한다.

LOFQ 알고리즘을 살펴보면 새로운 흐름을 수락하기 전에 $E_{ij}^{B2B} > 1$ 인 임의의 두 흐름 i 와 j 가 있는 한 B2B 변환을 반복 수행하는 점유자원 최적화 과정을 거친 후 흐름 수락을 시도한다. 이러한 점유자원 최적화 과정은 바로 모든 흐름에 최적의 레이턴시 지수 값을 설정하여 이들 흐름이 최적의 레이턴시를 갖도록 하는 과정이다. [13]에 따르면 LOFQ 스케줄러는 WFQ 스케줄러 대비 35% 정도의 높은 대역폭 이용 효율을 제공할 수 있다.

LOFQ 알고리즘의 구현 복잡도는 패킷 전송시 복잡도와 흐름 수락시 복잡도로 나눌 수 있다. 전자 는 다음 전송할 패킷을 찾는 복잡도로 이는 흐름 별로 구성된 대기큐 열에서 타임 스탬프 값이 가장 앞선 패킷을 찾는 데 있어 요구되는 복잡도로서 WFQ의 그것과 동일하다. WFQ의 패킷 전송시 복잡도는 당초 $O(N^2)$ 에서 최근에 $O(1)$ 수준으로 개선 되었다[14]. $O(1)$ 수준인 WFQ의 흐름 수락시 복잡도와 달리 LOFQ의 경우 새로운 흐름 수락시 스케줄러에 기 수락된 모든 흐름에 대한 점유자원 최소화 동작으로 인해 N^2 번, 여기서 N 은 수락된 흐름 수, 의 B2B 변환이 일어날 수 있는, 즉 $O(N^2)$ 의 구현 복잡도를 갖는다.

III. 제안방식

LOFQ 스케줄러에 도착하는 임의의 흐름에 대해 스케줄러 자원 이용 효율을 극대화하는 레이턴시 지수 값은 그 흐름뿐만 아니라 스케줄러에 수용되어 있는 다른 모든 흐름들의 QoS 특성에 종속되기 때문에 구하기가 매우 어렵다. 따라서 본 연구에서는 스케줄러에 가해진 트래픽 부하의 QoS 데이터의 수집 및 통계적 분석을 통해 트래픽 흐름의 QoS 확률함수를 추정하고, 추정된 확률함수를 이용하여 스케줄러에 도착하는 흐름의 최적 레이턴시 지수 값을 $O(1)$ 의 구현 복잡도로 계산할 수 있는 최적 레이턴시 지수(OLI: Optimal Latency Index) 함수를 도입한다.

3.1. 흐름의 QoS 확률 함수

라우터에 탑재된 스케줄러 입장에서 볼 때 트래픽 흐름의 QoS는 요구속도 r 과 지연규격 b 로 구체화 된다. 여기서 $r \in [r^{\min}, r^{\max}]$ 및 $b \in [b^{\min}, b^{\max}]$ 이다. 그러면 스케줄러에 가해지는 트래픽 흐름은 요구속도 R 과 지연규격 B 의 두 랜덤변수를 갖는 랜

덤벡터로 모형화 할 수 있다. 이러한 랜덤벡터의 확률함수를 $f_{r,b}(r,b)$ 로 표현한다.

트래픽 부하의 QoS 데이터를 수집하고, 수집한 데이터의 통계적 처리를 통해 흐름 QoS의 확률적 분포를 추정할 수 있다. 수집할 QoS 데이터는 스케줄러가 서비스를 완료한 트래픽에 대한 것, 즉 과거 데이터, 과 현재 서비스 중에 있는 트래픽, 즉 현재 데이터, 의 두 가지로 나눌 수 있다. 과거 데이터로부터 구한 확률함수를 $f_{r,b}^p(r,b)$, 현재 데이터로부터 구한 확률함수를 $f_{r,b}^c(r,b)$ 로 표시한다. 그러면 추정 확률함수 $f_{r,b}(r,b)$ 는 다음과 같이 구할 수 있다.

$$f_{r,b}(r,b) = \alpha f_{r,b}^p(r,b) + (1-\alpha) f_{r,b}^c(r,b). \quad (3.1)$$

여기서 α 는 과거 데이터의 반영 비율이다. 한 예로 α 값이 1이면 서비스를 완료한 부하의 QoS 특성만 반영한다. 최적 α 값을 구하는 문제는 라우터가 위치하는 사이트의 특성에 의존하여 본 연구의 범위를 벗어나므로 생략한다.

과거 데이터를 수집하는 방법을 살펴보자. 서비스가 종료되는 모든 트래픽 흐름에 대해 그의 r 과 b 에 대응하는 행렬 F^p 값에 그 흐름의 서비스 지속시간 T 의 값을 누적한다. 이렇게 하여 수집한 행렬 F^p 값을 F^p 의 전체 누적 값으로 정규화 하면 바로 $f_{r,b}^p(r,b)$ 가 얻어진다. 마찬가지로의 방법으로 행렬 F^c 를 사용하여 현재 서비스 중인 흐름에 대한 데이터를 수집하고 F^c 의 전체 누적 값으로 정규화 하여 $f_{r,b}^c(r,b)$ 를 얻을 수 있다. 참고로 모든 흐름에 대해 서비스 시작 시점의 시각이 그 흐름의 프로파일에 수록되므로 서비스 중인 흐름의 서비스 지속시간 값을 쉽게 구할 수 있다

3.2. 제안된 LOFQ 알고리즘

기존 LOFQ 스케줄러의 흐름 수락 알고리즘을 살펴보면 다음과 같다. 새로운 흐름이 도착할 때 먼저 스케줄러에 수용된 모든 흐름에 대해 점유자원 최적화 과정을 수행한다. 그리고 흐름의 점유 대역폭 자원을 최소화 하는 그러한 레이턴시 지수 값을 설정한 후 가용자원을 조사하여 흐름 수락을 시도한다.

본 연구에서 제안한 알고리즘은 그림 1과 같이 점유자원 최적화 과정 대신 (4.10)의 OLI 함수를 이용하여 최적 레이턴시 지수 값을 계산한 후 가용자원을 조사하여 흐름수락을 시도한다. 그리고 기존 알고리즘과 마찬가지로 대역폭 자원과 지연 자원

중 어느 한 자원은 충분한데 다른 자원이 부족하여 흐름 수락이 실패하는 예외적인 경우를 대비하기 위하여 필요시 자원변환을 수행한 후 흐름을 수락한다.

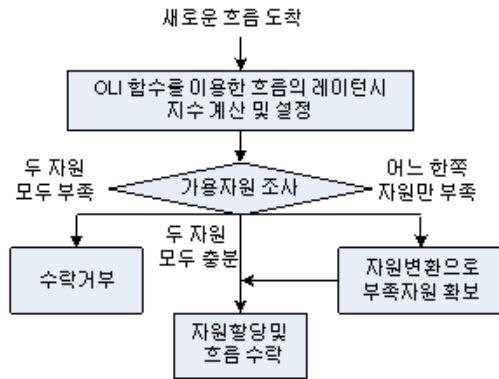


그림 1. 제안한 LOFQ 스케줄러 알고리즘
Fig. 1. Proposed LOFQ scheduling algorithm

OLI 함수는 QoS 확률함수 $f_{R,B}(r,b)$ 에 의존하므로 $f_{R,B}(r,b)$ 가 변할 때 마다 OLI 함수를 갱신해야 하고, $f_{R,B}(r,b)$ 는 가해진 부하의 QoS 특성에 의존하므로 부하의 QoS 특성이 변할 때 마다 $f_{R,B}(r,b)$ 를 갱신해야 한다. 따라서 OLI 함수의 갱신은 $f_{R,B}(r,b)$ 의 갱신과 갱신된 $f_{R,B}(r,b)$ 를 사용하여 OLI 함수를 다시 구하는 두 과정으로 나뉜다. OLI 함수의 갱신 시점에 대해 살펴본다. 부하의 QoS 특성은 시간에 따라 변할 수 있으므로 최근 갱신시점으로부터의 경과시간 T_{elapsd} 가 중요한 고려요소가 된다. 통계적으로 의미를 가지려면 충분한 데이터의 수집이 필요하므로 수집 데이터 량 Q_{data} 역시 중요한 고려요소가 된다. 그 외 사회적인 요인 등 다양한 고려요소가 있을 수 있다. 본 연구에서는 T_{elapsd} 와 Q_{data} 가 모두 어떤 임계 값 이상일 때 만 자동으로 수행되는 자동갱신과 오퍼레이터의 요구에 의해 수행되는 수동갱신의 두 가지 갱신 방법을 제시한다. 구체적인 임계값은 라우터 마다 다를 수 있으므로 본 연구에서 제외한다. 초기 OLI 함수는 미리 주어지는 QoS 확률함수의 기본값을 이용하여 구한다.

IV. 최적 레이턴시 지수 함수

4.1. 점유자원 최적화 과정 모델링

LOFQ 방식에서 도입한 흐름의 레이턴시 지수 β 는 자원 변환이 가능한 (0,1] 범위의 값과 자원변환이 불가능한 0 또는 1 보다 큰 값의 두 경우로 나

눌 수 있다. 자원변환이 가능한 흐름들에 대해 수행되는 점유자원 최적화 과정을 자세히 살펴본다. B2D 변환 효율이 서로 다른, 즉 $E_i^{B2D} > E_j^{B2D}$ 인 임의의 두 흐름 i와 j에 대해 $E_{ij}^{B2B} = E_i^{B2D} * E_j^{D2B} = E_i^{B2D} * (1/E_j^{B2D}) > 1$ 이 된다. B2B 변환효율이 1보다 크므로 흐름 i가 B2D 완전변환 되거나, 흐름 j가 D2B 완전변환 될 때까지 B2B 변환이 계속 수행된다. 여기서 B2D 완전변환이란 레이턴시 지수가 1이 될 때까지 B2D 변환이 일어나는 것을 말하고, D2B 완전변환이란 스케줄링 속도가 요구속도와 같을 때까지 D2B 변환이 일어나는 것을 의미한다.

흐름 i의 B2D 완전변환에 의해 얻어지는 지연 자원 량 ΔO_i , 흐름 j의 D2B 완전변환에 필요한 지연 자원 량 ΔI_j 로 표기한다. 그러면 흐름 i와 j에 대해 B2B 변환을 계속 수행하면 ΔO_i 와 ΔI_j 의 값에 따라 다음과 같은 3가지 결과가 나온다: i) $\Delta O_i = \Delta I_j$ 일 경우 B2D 변환으로 얻어진 지연 자원 량을 모두 D2B 변환에 투입하므로 흐름 i는 B2D 완전변환 되고 흐름 j는 D2B 완전변환 된다. ii) $\Delta O_i < \Delta I_j$ 일 경우 B2D 변환으로 얻어진 지연 자원 량을 모두 D2B 변환에 투입하여도 여전히 흐름 j는 D2B 변환의 여지가 있으므로 흐름 i만 B2D 완전변환 된다. iii) $\Delta O_i > \Delta I_j$ 일 경우 흐름 i의 일부 D2B 변환만으로도 흐름 j의 D2B 완전변환에 필요한 지연 자원 량을 충분히 얻을 수 있으므로 흐름 j만 D2B 완전변환 된다.

앞에서 ii)의 경우 $E_k^{B2D} > E_j^{B2D}$ 이고 흐름 k의 B2D 완전변환으로 얻어지는 지연 자원 량이 ($\Delta I_j - \Delta O_i$) 보다 큰 그러한 흐름 k를 생각하자. 그러면 흐름 k와 j 사이에 B2B 변환을 지속하면 흐름 k의 B2D 변환으로 얻어지는 지연 자원 량이 흐름 j가 D2B 완전변환에 필요한 지연 자원 량 ($\Delta I_j - \Delta O_i$) 보다 많기 때문에 흐름 j가 D2B 완전변환 된다. 앞에서 iii)의 경우 $E_i^{B2D} > E_l^{B2D}$ 이고 흐름 l의 D2B 완전변환에 필요한 지연 자원 량이 ($\Delta O_i - \Delta I_j$) 보다 큰 그러한 흐름 l을 생각하자. 그러면 흐름 i와 l 사이에 B2B 변환을 지속하면 흐름 l의 D2B 완전변환에 필요한 지연 자원 량이 흐름 i의 B2D 완전변환으로 얻어지는 지연 자원 량 ($\Delta O_i - \Delta I_j$) 보다 많기 때문에 흐름 i가 D2B 완전변환 된다. 이와 같이 다른 흐름들과의 B2B 변환을 통해 ΔO_i 와 ΔI_j 의 값에 상관없이 흐름 i는 B2D 완전변환 되고 흐름 j는 D2B 완전변환 된다.

위와 같은 최적화 과정을 모든 흐름에 대해 적용하면 최적의 E^{B2D-C} 값이 존재하여 이 값보다 큰

B2D 변환효율을 갖는 B2D 변환가능 흐름들은 모두 B2D 완전변환 되고, 이들을 제외한 D2B 변환가능 흐름들은 모두 D2B 완전변환 된다. 여기서 B2D 완전변환 되는 흐름을 A 그룹, D2B 완전변환 되는 흐름을 B 그룹이라 하고, 자원변환이 불가능한 흐름을 C 그룹이라 한다. 참고로 지연 자원을 생성하는 A 그룹 흐름은 바로 RR 흐름에 해당되고, 생성된 지연 자원을 투입하여 점유 대역폭 자원을 줄이는 B 그룹 흐름은 바로 DR 흐름에 해당한다. C 그룹을 더 세분화 하여 0의 레이턴시 지수값을 갖는 흐름을 C₁그룹, 1보다 큰 값을 갖는 흐름을 C₂ 흐름이라 한다. B2D 변환 효율이 E^{B2D-C}의 값을 갖게 하는 지연규격을 b^T로 표기하고 이를 임계 지연규격이라 부른다.

4.2. 최적 레이턴시 지수 함수 도출

먼저 각 그룹의 흐름이 점유하는 자원 량을 분석한다. A 그룹 흐름의 점유 BW h^A와 점유 DW d^A를 구해보자. B2D 완전변환에 의해 레이턴시 지수가 1이 되므로 (2.5)와 (2.6)으로부터 다음과 같이 계산된다.

$$h^A = \frac{L}{b-D^B} \text{ 및 } d^A = 0. \tag{4.1}$$

B 그룹 흐름은 D2B 완전변환에 의해 h=r 가 되고, (2.5)로부터 β=(b-D^B)r/L가 되므로 (2.6)에 의해 {L-(b-D^B)r}/C^B의 DW를 점유한다. 이로부터 B 그룹 흐름의 점유 BW h^B와 점유 DW d^B 계산식은 다음과 같다.

$$h^B = r \text{ 및 } d^B = \frac{L - (b - D^B)r}{C^B}. \tag{4.2}$$

C₁ 그룹 흐름은 0의 레이턴시 지수 값을 가지므로 (2.5)와 (2.6)으로부터 점유 BW h^{C1}와 점유 DW d^{C1}의 계산식은 다음과 같다.

$$h^{C1} = r \text{ 및 } d^{C1} = \frac{L}{C^B}. \tag{4.3}$$

C₂ 그룹 흐름은 1보다 큰 레이턴시 지수 값을 가지므로 (2.5)와 (2.6)으로부터 점유 BW h^{C2}와 점유 DW d^{C2} 계산식은 다음과 같다.

$$h^{C2} = r \text{ 및 } d^{C2} = 0. \tag{4.4}$$

스케줄러에 도착하는 흐름은 QoS 특성에 의해 소속 그룹이 결정된다. 이러한 특성을 살펴보기 위해 흐름 그룹별로 b의 범위를 구해보자. 각 그룹의 레이턴시 지수, 요구속도 및 지연규격을 각각 β^X, r^X 및 b^X로 표기한다. 여기서 X는 A, B, C₁ 또는 C₂ 중 하나이다. 점유자원 최적화 결과 A 그룹 흐름은 B2D 완전변환 되므로 b^A>b^T. β^A∈(0,1)이고 h≥r이므로 (2.5)로부터 b^A<(D^B+L/r). 따라서 b^A∈[b^T,min{b^{max},(D^B+L/r)}]가 된다. 점유자원 최적화 결과 B 그룹 흐름은 D2B 완전변환 되므로 b^B≤b^T β^B∈(0,1] 이고 h≥r이므로 (2.5)로부터 b^B≤(D^B+L/r). 따라서 b^B∈[b^{min}, b^T]가 된다. C₁과 C₂ 그룹에 대해 살펴보자. β^{C1}=0이므로 (2.5)로부터 b^{C1}=D^B=b^{min}. β^{C2}>1이므로 (2.5)로부터 b^{C2}>(D^B+L/r). 따라서 b^{C2}∈[(D^B+L/r),b^{max}]. 이들 결과를 정리하면 다음과 같다.

$$b^A \in [b^T, \min\{b^{\max}, (D^B + L/r)\}] \text{ 및 } b^B \in (b^{\min}, b^T]. \tag{4.5}$$

$$b^{C1} = b^{\min} \text{ 및 } b^{C2} \in [\max\{b^{\max}, (D^B + L/r)\}, b^{\max}]. \tag{4.6}$$

임계 지연규격 b^T를 구하는 과정을 살펴본다. 점유자원 최적화란 바로 스케줄러의 흐름 수용능력이 최대가 되도록 각 흐름에 지연 자원과 대역폭 자원을 적절히 할당하는 것이므로 흐름 수용 능력이 최대가 되는 그러한 b^T 값을 구하면 된다. 스케줄러에 도착하는 임의의 흐름이 점유하는 대역폭 자원과 지연 자원 량은 랜덤변수가 되며, 이들을 각각 H와 D로 표현하면 이들의 기대값은 각각 E[H]와 E[D]가 된다. C^B의 BW량과 D^B의 DW량으로 정량화되는 스케줄러의 용량은 두 자원 중 어느 하나가 부족하면 다른 자원에 여유가 있어도 더 이상의 흐름을 수용할 수 없으므로 스케줄러의 흐름 수용 능력은 min{C^B/E[H], D^B/E[D]}로 계산된다.

다음으로 E[H]와 E[D] 값을 구한다. (4.1) ~ (4.4)로부터 그룹 A 흐름만 h^A의 BW를 점유하고 그 외 모든 그룹 흐름은 요구속도 r의 BW를 점유한다. (4.5)와 (4.6)에 정리된 각 그룹별 지연규격 b의 범위를 적용하면 E[H]는 다음과 같이 정리된다.

$$E[H] = \int_{r^{\min}}^{r^{\max}} \left[\int_{b^A} f_{R,B}(r,b)db + \int_{b^{\min}}^{b^T} f_{R,B}(r,b)db + \int_{b^{C2}} f_{R,B}(r,b)db \right] dr. \tag{4.7}$$

(4.1) ~ (4.4)로부터 그룹 B와 C₁ 흐름만 DW를

점유하며, (4.5)와 (4.6)에 정리된 그룹별 r 과 b 의 범위로부터 $E[D]$ 는 다음과 같이 구해진다.

$$E[D] = \int_{r^{\min}}^{r^{\max}} \left[\int_{b^B} d^B f_{R,B}(r,b)db + d^{C1} f_{R,B}(r,D^B) \right] dr. \quad (4.8)$$

b^T 값이 최적값 보다 작으면 E^{B2D-C} 값이 줄어들어 A 그룹 흐름의 비율이 낮아지는 반면 B 그룹 흐름의 비율은 높아진다. 그 결과 (4.8)로부터 $E[D]$ 값이 증가한다. 반대로 b^T 값이 최적값 보다 크면 E^{B2D-C} 값이 증가하여 A 그룹 흐름의 비율이 높아지는 반면 B 그룹 흐름의 비율이 낮아진다. 그 결과 (4.7)로부터 $E[H]$ 값이 증가한다. 따라서 최적 b^T 값은 바로 $C^B/E[H]$ 와 $D^B/E[D]$ 값을 서로 일치하게 하는 그러한 값이 된다. (4.7)과 (4.8)로부터 간단한 수치해석을 통해 최적 b^T 값을 구할 수 있다. 참고로 LOFQ 스케줄러의 흐름 수용 능력은 바로 이때의 $C^B/E[H]$ 또는 $D^B/E[D]$ 값이 된다.

마지막으로 최적 레이턴시 지수 함수, 즉 OLI 함수를 구한다. A 그룹 흐름은 레이턴시 지수가 1이 되고 C_1 그룹은 0이 된다. B 그룹과 C_1 그룹은 (2.5)의 식에 의해 레이턴시 지수가 구해지므로 스케줄러에 도착하는 임의의 흐름의 최적 레이턴시 지수를 구하는 OLI 함수는 (4.9)와 같이 표현된다.

$$\beta = \begin{cases} 1 & \text{for } b^A \ni b \\ \frac{r(b-D^B)}{L} & \text{for } b^A \text{ or } b^{C2} \ni b \\ 0 & \text{for } b^{C1} \ni b. \end{cases} \quad (4.9)$$

V. 성능 고찰

인터넷에서 유통되는 실시간 미디어는 수 Kbps 급의 저속 음성부터 수백 Kbps급의 영상까지 다양하므로 트래픽 흐름의 요구속도 r 은 [1,300]Kbps내에 균등하게 분포하는 것으로 가정한다. 종단간 지연이 150ms이내[15]인 VoIP의 경우 코덱 지연 등을 감안할 경우 인터넷 지연시간은 수십 ms 이내가 되어야 하고[16], 대부분의 경우 종단간 라우터의 수가 15개를 넘지 않을 것이므로 각 라우터에 할당되는 지연규격 b 는 직관적으로 [1,16]ms내에 균등 분포하는 것으로 가정한다. 그리고 모든 흐름의 패킷 크기는 모두 L byte, $C^B=10$ Gbps, $D^B=1$ ms로 가

정한다. 수치해석과 시뮬레이션 방법을 이용하여 성능을 평가하며, 시뮬레이션은 SMPL(Simulation Model Programming Language)[17]에 공평 패킷 스케줄러 루틴을 추가하여 수행한다.

패킷 크기 L 의 증가에 따른 임계 지연규격 b^T 값의 변화를 그림 2에 도시하였다. 패킷 크기가 작아질수록 b^T 값이 b^{\max} 값인 16ms로 증가하고, 반대로 패킷 크기가 커질수록 b^{\min} 값인 1ms에 수렴하는 모습을 보인다. 이러한 특성은 다음과 같이 분석할 수 있다. (2.5)로부터 패킷 크기가 작아질수록 흐름의 레이턴시가 감소하므로 지연제약 흐름, 즉 B 그룹 흐름의 비율이 줄어들고 반대로 A 그룹 흐름의 비율이 늘어난다. 그 결과 (4.7)과 (4.8)로부터 각각 $E[H]$ 값이 증가하고 $E[D]$ 값이 감소한다. 따라서 $E[H]$ 값을 줄이고 $E[D]$ 값을 늘려 흐름 수용 능력이 최대가 되도록 하기 위해 임계 지연규격 값이 b^{\max} 쪽으로 증가하게 된다. 반대로 패킷 크기가 커지면 (2.5)로부터 흐름의 레이턴시가 증가하므로 흐름 수용 능력이 최대가 되도록 하기 위해 임계 지연규격 값이 b^{\min} 쪽으로 수렴하게 된다.

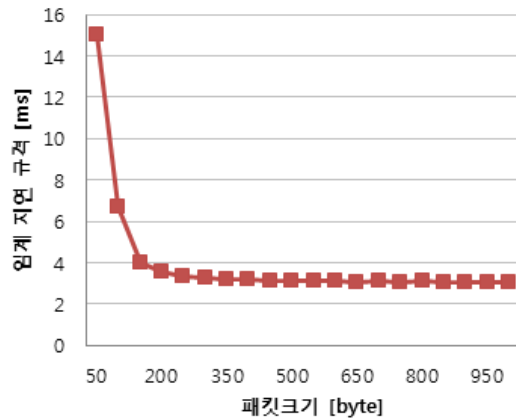


그림 2. 패킷 크기 변화에 따른 임계 지연규격
Fig. 2. Critical delay budget v.s. packet size

그림 3은 패킷 크기 증가에 따른 흐름 수용 능력을 WFQ의 그것과 비교하여 도시한 것이다. 패킷 크기가 증가할수록 수용능력이 떨어지는 모습을 보여준다. 이는 (2.5)와 (2.6)으로부터 흐름의 점유 BW와 DW가 모두 패킷 크기에 비례하여 증가하므로 그 만큼 수용할 수 있는 흐름의 수가 줄어들기 때문이다. 그림 3을 통해 해석적으로 구한 흐름 수용 능력이 시뮬레이션으로 구한 것과 일치함을 확인할 수 있다. 자원 이용 효율 측면에서 WFQ와 LOFQ를 상호 비교한 결과를 그림 4에 도시하였다. 곡선의 모습은 그림 3과 동일하다. 이는 흐름 수용

능력이 바로 스케줄러의 용량과 자원이용도의 곱으로 계산되기 때문이다. 그림 4에서 패킷 크기가 작아지면 WFQ로도 충분히 높은 자원 이용도를 얻을 수 있음을 관찰할 수 있다.

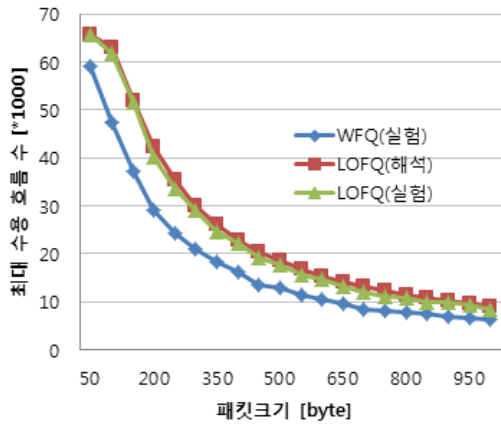


그림 3. 패킷 크기 변화에 따른 흐름 수용능력
Fig. 3. Capacity accommodating flows v.s. packet size

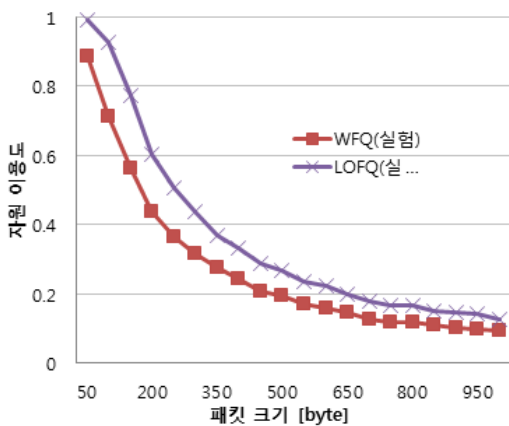


그림 4. 패킷 크기 변화에 따른 자원 이용도
Fig. 4. Resource utilization v.s. packet size

WFQ 대비 LOFQ의 우수성을 보다 쉽게 알아보기 위해 그림 5에 자원 이용도의 개선 정도를 도시하였다. 패킷 크기가 100바이트 이상이 되면 WFQ에 비해 30 ~ 40% 정도의 성능 개선의 효과를 얻을 수 있음을 관찰할 수 있다. 패킷 크기가 100바이트 미만인 경우 개선 정도가 낮은 것은 그림 4에서 살펴본 바와 같이 WFQ 만으로도 100%에 가까운 자원 이용도를 얻을 수 있어 그 만큼 성능 개선의 여지가 없기 때문이다. 참고로 실시간 스트림의 전송에 사용되는 기본 RTP 패킷의 경우 50 바이트의 오버헤드가 탑재된 전송효율을 고려하면 패킷 크기는 적어도 100바이트 이상은 되어야 하므로 LOFQ로 WFQ 대비 30% 이상의 자원 이용도 개

선이 기대된다. 이러한 결과는 참고문헌 [13]에 있는 기존 LOFQ 알고리즘의 성능 평가 결과와 일치한다.

VI. 결 론

LOFQ 알고리즘은 엄격한 지연규격을 요구하는 저속 흐름의 경우 자원 이용도가 아주 저하되는 WFQ 알고리즘의 구조적 한계를 개선하여 30% 이상의 자원 이용도 개선 효과를 얻을 수 있다. 하지만 흐름 수락시 점유자원 최적화 과정에 요구되는 $O(N^2)$ 의 높은 복잡도로 인해 구현에 어려움이 있음을 확인하였고 이러한 복잡도를 낮추는 하나의 방안을 제시하였다.

제시한 방식을 정리하면 다음과 같다. 트래픽 흐름의 서비스 품질을 요구속도와 지연규격의 랜덤 벡터로 모형화한 후, 가해진 부하의 서비스 품질 데이터를 수집하고, 수집한 데이터를 통계적으로 처리하여 랜덤 벡터의 확률함수를 추정할 수 있게 하였다. 추정된 확률함수를 이용하여 스케줄러에 도착하는 흐름에 최적의 레이턴시를 설정하게 해주는 레이턴시 지수값을 구할 수 있는 최적 레이턴시 지수함수를 도출하였다.

이러한 최적 레이턴시 지수 함수를 이용함으로써 기존 LOFQ 알고리즘에서 흐름 수락시 $O(N^2)$ 의 높은 구현 복잡도가 요구되는 점유자원 최적화 과정 없이 $O(1)$ 의 단순한 복잡도로 최적의 레이턴시 지수 값을 구할 수 있었다. 최적 레이턴시 지수 함수 기반의 LOFQ 알고리즘을 실험한 결과 기존의 LOFQ 알고리즘과 동일한 성능을 보였다.

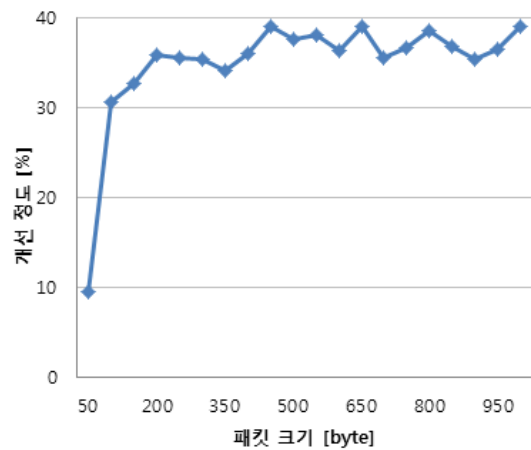


그림 5. 패킷 크기 변화에 따른 자원 이용도 개선 정도
Fig. 5. Improvement in utilization v.s. packet size

References

[1] X. Xiao and L. M. Ni, "Internet QoS: A Big Picture," *IEEE Network*, 13(2), pp.8-18, 1999.

[2] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, "Resource ReSerVation Protocol (RSVP) version 1 functional specification," *RFC 2205, IETF*, 1997.

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

[4] A. Demers, S. Keshav, and S. Shenker, "Design and analysis of a fair queuing algorithm," *Proc. ACM SIGCOMM*, 1989.

[5] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guaranteed Quality of Service," *RFC 2212, IETF*, 1997.

[6] M. Baldi and F. Risso, "Efficiency of Packet Voice with Deterministic Delay," *IEEE Comm. Mag.*, pp.170-177, 2000.

[7] A.Hangan, R.Marfiavici and G.Sebestyen, "Real-time Data Flow Scheduling for Distributed Control," *International Journal on Advances in Networks and Services*, 1(1), 2008.

[8] L. Tan and N. Wang, "Future internet: The internet of things," *Proc. ICACTE*, pp.376-380, 2010.

[9] A. Francini and F. M. Chiussi, "A Weighted Fair Queuing Scheduler with Decoupled Bandwidth and Delay Guarantees for the Support of Voice Traffic," *Proc. GLOBECOM*, 2001.

[10] Jens Schmitt. "Optimal Network Service Curves under Bandwidth-Delay Decoupling," *IEE Electronics Letters*, 38(6), pp.297-299, 2002.

[11] Do-Sung Jun, Jinwoo Choe and Alberto Leon-Garcia, "Credit-based Processor Sharing for Decoupled Delay and Bandwidth Allocation," *IEEE Comm. Letters*, 5(4), pp.178-180, 2001.

[12] T. Kim, "A Weighted Fair Queuing with Optimal Rate and Delay Allocation," *Proc. GLOBECOM*, 2006.

[13] T. Kim, "A study on improving latency-optimized fair queuing algorithm," *Journal of the Korea Multimedia Society*, 10(1),

pp.83-93, 2007

[14] S. Taniguchi, R. Kawate, K. Sato and T. Yokotani, "Architecture and Performance Evaluation of the Simplified WFQ," *Proc. Taiwan-Japan Joint Conference on Communications Technology*, 2007.

[15] *ITU-T Recommendation G.114: One-way transmission time*, 2003.

[16] S. Bae, "VoIP planning and evaluation through the E-model baed speech transmission quality analysis," Ph.D dissertation, *Kyonggi University*, 2004.

[17] M. H. MacDougall, "Simulating Computer Systems, Techniques and Tools," *MIT Press*, 1987.

김 태 준 (Tae-joon Kim)

중신회원



1980년 2월 경북대학교 졸업
 1982년 2월 KAIST 석사
 1999년 8월 KAIST 박사
 1982~1996년 ETRI 근무
 현재 공주대학교 근무
 <관심분야> 인터넷 엔지니어링

서 봉 수 (Bongsue Suh)

중신회원



1993년 2월 경북대학교 전자공학(공학사)
 1995년 2월 한국과학기술원 전기및전자공학(공학석사)
 1999년 8월 한국과학기술원 전기및전자공학(공학박사)
 1999년 7월~2004년 2월 한국 전자통신연구원 선임연구원
 2004년 3월~2005년 8월 안동대학교 정보통신공학부 조교수
 2005년 9월~현재 공주대학교 정보통신공학부 조교수, 부교수
 <관심분야> 컴퓨터 네트워크, 이동통신 시스템, 무선네트워크