차등화 서비스의 성능 개선을 위한 보상 기법

정종화[0] 하 란
홍익대학교 컴퓨터 공학과
{jhjung[0], rhanha}@cs.hongik.ac.kr

Development of Fair Scheduler for Quality Improvement in Wireless Differentiated Service

Jong-Hwa Jung ,Rhan Ha
Dept. of Computer Engineering, Hong Ik University

요 약

지연 차등화 모델은 각 클래스의 지연 차별 변수에 따라 차등화된 서비스를 제공한다. 따라서 각 클래스의 패킷별 평균 대기시간이 지연 차별 변수에 비례한다. 유선 환경의 지연 차등화 모델인 WTP를 무선 환경에 적용시킨 WWTP는 큐의 처음 패킷의 블록으로 인한 큐의 전송 중단 현상을 개선하였다. 이로 인해 클래스별 서비스 불공평성이 줄어든다. 그러나 여전히 채널 오류기간에 비례하여 전송의 편중과 중단현상이 발생하는 단점이 있다. 본 논문은 WWTP에서 발생하는 서비스 편중과 중단 현상을 개선하기 위한 보상 정책을 제공하는 스케줄러를 제안한다. 제안된 모델은 에러 기간동안 에러 프리 서비스를 가상적으로 시뮬레이션하여 정상 서비스와의 차이를 알아내고 이를 바탕으로 보상 정책을 수행한다. 결과적으로 지연 차등화 모델인 WWTP에서 발생하는 서비스 중단을 개선하면서 정상적인 채널상태와 유사한 서비스를 제공한다.

1. 서 론

IETF는 멀티미디어 응용의 다양한 성능 요구를 만족시키기 위해 통합 서비스[1,2]와 차등화 서비스[3,4,5]를 발전시켰다. 통합 서비스 접근 방법에서 RSVP[2]의 흐름당 자원을 요청 및 예약하는 기능은 개별 흐름마다 서비스 품질을 보장할 수 있게 해준다. 그러나 인터넷의 백본 라우터에는 동시에 수십만의 근원지-목적지 트래픽 흐름이 있을 수 있기 때문에 확장 가능하고 융통성 있는 서비스 차등화가 요구된다. 차등화 서비스는 크게 절대 차등화 서비스와 상대 차등화 서비스로 구분할 수 있다. 절대 서비스 차등화는 허락된 사용자에게 일정하게 정해진 성능을 제공하는 반면 상대 서비스 차등화는 우선순위 인덱스가 높은 클래스에게 낮은 클래스에 비해 상대적으로 더 좋은 성능을 보장한다.

상대적 서비스 차등화 모델을 지원하기 위해 대기 시간 우선권(Waiting Time Priority, 이하 WTP) 스케줄러[3]는 HOL(Head of Line, 이하 HOL) 패킷의 대기 시간을 기준으로 각 클래스의 지연 차등화 인자(Delay Differentiate Parameter, 이하 DDP)에 따라 나누어 전송의 우선순위를 결정한다.[3,4,5] WTP는 큐의 첫 번째 패킷의 블록으로 인해 클래스의 전송이 중단될 수 있으므로 무선 환경에는 적합하지 못하다.

WTP를 무선 환경에 적용시킨 무선 대기 시간 우선(Wireless-WTP, 이하 WWTP) 스케줄러[3]는 HOL 블록 현상을 개선하여 에러 흐름에 의한 클래스의 전송 중단 현상을 개선하였다.

WTP를 개선한 AWTP(Advanced-WTP)[4]는 인터넷의 다양한 패킷 크기를 고려하여 개별 패킷의 전송 시간까지 참조하여 우선순위를 구한다. 한 클래스의 패킷이 서비스된 후의 다른 클래스의 패킷들의 증가된 대기시간을 가상적으로 추정하여 최대 비례[4]를 구한다. AWTP는 WTP와 비교하여 DDP에 더 근사하는 차등화를 보여준다.

WWTP는 기존의 WFQ(Weighted Fair Queueing)[1,2]기반 스케줄러와 달리 지연되는 패킷은 대기 시간의 증가로 인해 우선순위가 증가한다. 따라서 채널 에러 상태로 인해 전송이 연기되는 패킷은 대기 시간이 증가하므로 채널이 정상 상태를 회복한 시점에 더 높은 전송의 우선순위를 가진다. 그러나 채널 에러 기간에 비례하여 에러 흐름의 보상 기간도 같이 증가하므로 우선순위가 낮은 클래스의 흐름은 보상 기간 동안 전송이 중단될 수 있는 문제점이 있다. 지연 차등화 스케줄러는 WFQ처럼 가중치와 가상 시간을 사용하지 못하므로 통합 스케줄러와 다른 방식의 보상정책이 필요하다.

본 논문에서는 WTP의 무선 환경 적용시 발생할 수 있는 서비스 중단 현상을 개선한다. 2장에서 제안된 스케줄러의 구조에 대해 설명한다. 3장에서는 보상 정책에 대해 설명한다. 4장에서는 실험결과를 통해 성능개선을 알아본다. 마지막으로 5장에서는 본 논문에 대한 결론을 맺는다.

2. 제안된 스케줄러의 구조

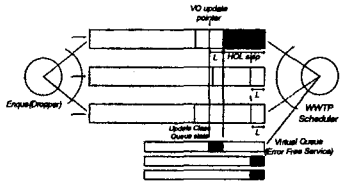

그림 1 제안된 스케줄러의 구조

그림 1은 제안된 스케줄러(WWTP+)의 구조를 보여준다. 각 클래스마다 하나의 가상 큐가 존재한다. 가상 큐는 실제 큐의 상태 정보만 저장한다. 스케줄러는 에러 기간동안 실제 큐의 상태를 가상 큐에 저장하고 에러 프리 서비스를 가상적으로 동작시킨다. 각 클래스 큐의 상태를 가상 큐에 저장하기 위해 가상 큐 갱신 포인터를 사용하며 포인터는 하나의 패킷 길이 L단위로 이동한다. 그러나 에러 흐름을 포함한 클래스에서 HOL skip[3]이 발생하면 실제 큐의 전송이 가상 큐보다 우선하는 경우가 발생하므로 실제 큐의 정확한 상태 정보 유지를 위해 다수의 패킷 정보가 한꺼번에 가상큐에 입력된다. 이와 함께 가상 큐 갱신 포인터가 지시하는 패킷의 가상 큐 입력 여부를 나타내는 변수를 함께 사용하여 큐의 상태를 저장한다. 스케줄러는 채널 상태 회복 시점에 가상 서비스 전송량과 실제 서비스의 전송량을 비교하여 각 클래스의 증가된 서비스 양을 계산한 다음 이를 바탕으로 보상을 수행한다.

3. 보상 정책

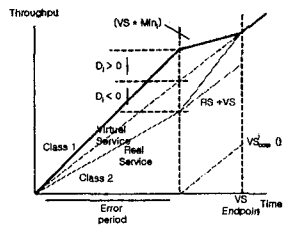

그림 2 보상 정책

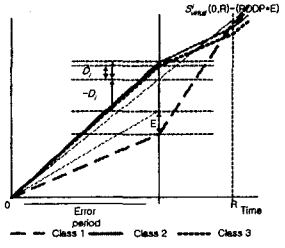

그림 3 보상량 조절

그림 2는 충분한 전송 대역폭과 가중치가 같은 두 개의 클래스를 가정한 상태에서 WWTP+의 보상 정책을 보여준다. 가상 서비스는 에러 발생 시점부터 서비스 증가분이 0이 되는 보상 종료 시점까지 계속 유지된다.

에러 기간을 τ라고 가정하면 스케줄러는 에러 발생 시점 t부터 채널 회복 시점 $t+\tau$까지의 실제 큐의 누적 전송량 $S^i_{actual}(t, t+\tau)$와 가상 큐의 누적 전송량 $S^i_{virtual}(t, t+\tau)$를 유지한다. 채널 상태 회복 시점에서 클래스의 실제 서비스 양에서 가상 서비스 양을 빼서 각 클래스의 서비스 증가분인 D_i를 계산한다. 스케줄러는 D_i값이 0보다 큰 클래스(그림 2의 클래스 1)는 서비스 증가 클래스로 0보다 작은 클래스(그림 2의 클래스 2)는 에러 흐름을 포함한 서비스 감소 클래스로 간주한다. D_i가 계산된 후 보상이 시작된다. 이 때 WWTP는 대기 시간이 증가한 에러 회복 큐의 전송에 의해 서비스 중단이 발생한다. 개선된 방안은 보상 기간동안 대기 시간이 증가된 클래스의 전송 편중을 막기 위해 실제 큐의 전송이 가상 스케줄러의 결정에 따르게 한다. 그림 2의 RS(실제 서비스)+VS(가상 서비스)는 가상 스케줄러의 결정을 따를 경우 서비스 증가량을 보여주고 있다. 이에 따라 에러 회복 큐의 연속 전송이 방지되는 반면 편중 방지와 보상을 위해 실제 우선순위값은 무시된다.

서비스 증가 클래스는 가상 서비스에 의한 에러 프리 상태의 전송을 따르면서 서비스 증가분을 최대 서비스 감소 클래스에게 양보한다. 클래스 i의 최소 보장율을 α_i라고 하면 보상 기간동안 서비스 증가 클래스 i는 α_i에 보상 기간동안의 가상 서비스의 누적 전송량인 $VS^i_{comp}(t)$를 곱한 양의 서비스를 보장받는다. $S^i_{comp}(t)$는 클래스 i가 보상 기간의 어느 시점 t에서 받은 서비스의 누적치를 나타낸다. 서비스 증가 클래스의 $S^i_{comp}(t)$가 클래스의 최소 서비스 보장량을 넘어서면 서비스 감소 클래스에게 타임 슬롯을 양보한다. 이 때 서비스 감소 클래스는 에러 회복 흐름이 클래스의 대역폭을 모두 점유하는 상황을 막기 위해 서비스 증가 클래스가 양보하는 대역폭만 에러 흐름에게 양보한다. 이를 위해 큐의 앞에서부터 누적된 에러 흐름과 그 이후부터 연속되는 정상 흐름을 구별한다. 타임 슬롯 양보시 서비스 증가 클래스는 D_i를 감소시키고 서비스 감소 클래스는 D_i를 증가시킨다. 전체 서비스 감소분이 0이 되는 시점(그림 2의 VS Endpoint)에 보상이 종료되면서 가상 큐의 모든 내용이 비워진다. 에러가 발생하지 않으면 제안된 알고리즘은 WWTP와 같이 동작한다.

각 클래스의 큐잉량의 부족으로 타임 슬롯의 낭비가 발생하면 전체 서비스 증가량 보다 전체 서비스 부족분이 많아진다. 그림 3은 이런 상황에서 각 클래스의 서비스 증가량을 조절하는 예를 보여준다. 그림 3의 낭비된 전송기회 E에 해당하는 시간 β는 각 클래스는 대기 시간에 추가된다. E는 서비스 증가분을 서비스 감소량에서 빼고 남는 서비스 부족분이며 아래와 같이 구해진다.

$$E = -\left(\sum_{D_i<0} D_i + \sum_{D_i>0} D_i\right)$$

전송 기회의 낭비로 추가된 시간 β에 의해 채널 상태 회복 시점에 각 클래스는 다음과 같은 우선순위값을 가진다.

$$priority_i = \frac{w(p_i^h) + \beta}{\delta_i} \left(WTP\ priority_i = \frac{w(p_i^h)}{\delta_i}\right)$$

β는 모든 클래스의 대기 시간에 동일한 값으로 추가되므로 증가된 대기 시간으로 인한 서비스 증가는 $1/\delta_i$에 비례한다. 반대로 서비스 감소분은 서비스 증가량에 역으로 비례한다. 따라서 클래스의 DDP(δ_i)값에 비례한 서비스 손실을 가진다. 정확한 보상을 위해 서비스 손실이 각 클래스의 DDP에 비례하도록 조정한다. 클래스 i의 상대적인 DDP 비율을 $RDDP_i$로 정의한다.

$$RDDP_i = \frac{DDP_i}{Total\ DDP}$$

새로 조정되는 서비스 증가분은 다음과 같다.

$$D_i = D_i + (E \times RDDP_i)$$

그림 3처럼 가상 스케줄링 종료 시점에 낭비된 전송기회 E는 각 클래스의 $RDDP_i$에 비례하도록 조정된다. D_i값의 조정으로 정확한 보상량을 계산함으로써 서비스 중단없는 보상이 가능하다. 표 1은 제안된 스케줄러의 알고리즘을 보여준다.

```
PROCEDURE wwtp_deque( )
  for(i = 0; i < MAX_CLASS; i++)  // 각 클래스의 우선순위 구함
    if(empty(i)) continue;
    pkt = Class i's Head_Of_Line packet;
    if(pkt can send)  // 전송 가능
      if((pkt's fid) ∉ A)
        A = A + {fid};  // A : Active flow set
        for(k=0; k < MAX_CLASS; k++)  find lagk;
        if( ( ∑ Di+ ∑ Di) < 0) adjust ∀Di;   //서비스 보상량 조절
      find priorityi;
      if(( | A |> 0) or (!empty(iviclass)))  // 에러개수가 0이상 또는 보상중
        Update_Virtual_Queue(i);  // 큐의 상태를 가상 큐에 저장
        find vpriorityi;
    else  // 전송 불가능
      find vpriorityi;
      if((pkt's fid) ∈ A)
        A \ (pkt's fid);          // 에러 흐름 개수 조정
      do  // 전송 가능한 다음 패킷을 검색
        current_pkt = pkt's next packet;
        if((current_pkt can send) and (current_pkt's fid ∈ A))
          find priorityik;  // 클래스 i의 k번째 패킷의 우선순위
          j=1;
        else if(current_pkt == TAIL) j=1;
        Update_Virtual_Queue(i);
      while(j=0);
  MAX_PRIORITY_QUEUE = MAX(k | priorityk > 0 and can send);
  MAX_VPRIORITY_QUEUE = MAX(k | vpriorityk > 0);
  if( | A | > 0 and lagi != 0.0)  // 서비스 보상
    MINi(t) = αi * VSCOMPi(t);  // 시간 t에서의 최소 보장 서비스
    if((SCOMPi(t) < MINi(t) and lagi < 0.0) or (lagi > 0.0))  // 정상 흐름
      MAX_PRIORITY_QUEUE = MAX_VPRIORITY_QUEUE;
    else  // 최대 서비스 감소 클래스에게 서비스 양보
      MAX_PRIORITY_QUEUE = MAXk∈A(k | lagk > 0 and can send);
  if(MAX_PRIORITY_QUEUE == -1)  pkt = NULL;  // 보낼수 없음
  else  // 보낼 수 있는 패킷이 존재
      pkt = classMAX-PRIORITY-QUEUE.deque(k);  // 처음 또는 k번째 패킷을 전송
      if(( ∑ Di == 0) and ( ∑ Di == 0)) clear ∀(virtual Queue);
      if(MAX_VPRIORITY_QUEUE != -1) vclassMAX-VPRIORITY-QUEUE.deque();
  return pkt;
END PROCEDURE
```

(표 1. WWTP+ 알고리즘)

4. 성능 분석

2Mb의 무선 대역폭에 6개의 무선 호스트를 가정하고 모든 흐름은 0.0205의 전송 지연과 512bytes의 패킷 크기, 10ms의 동일한 전송 지연을 가진다고 가정하였다. 비교대상은 제안된 스케줄러와 WWTP와 CIFQ[1]이며 각 스케줄러의 평균 전송 지연, 전송량, 흐름당 서비스 그래프를 비교한다. 실험에서 흐름 아이디 fid를 가진 흐름은 ((fid-1)%3)+1을 인덱스로 가지는 클래스에 속한다고 가정한다. 그림 4는 각 클래스의 대역폭을 달

리하여 전송량을 측정한 결과를 보여준다. backlog[1]가 증가할 수록 우선순위 인덱스가 높은 클래스로 전송이 편중되는 지연 차등화 서비스의 특징을 보여준다.

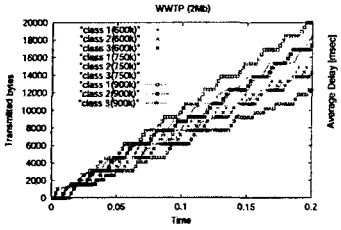

그림 4 대역 편중

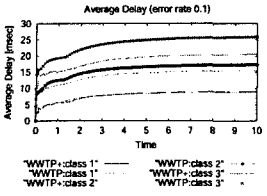

그림 5 평균 전송 지연

그림 5는 모든 흐름의 10%의 채널 에러를 가정한 경우에서 패킷당 평균 지연 시간의 비교 그래프이다. WWTP+는 채널 상태 회복후 보상 기간 동안 우선순위가 높은 클래스로의 전송 편중을 막으면서 보상하므로 우선순위가 낮은 클래스의 평균 지연이 WWTP에 비해 감소한다.

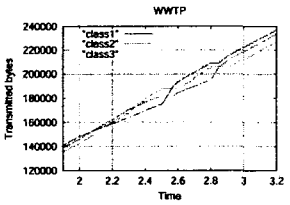

그림 6 WWTP

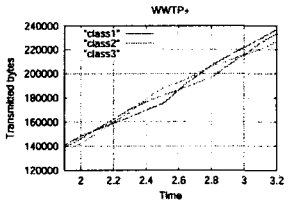

그림 7 WWTP+

그림 6,7은 동일 구간에서 클래스 1(시간 2~2.5)과 2(시간 2.3~2.8)의 한 흐름의 중복 에러를 가정한 경우의 WWTP와 WWTP+의 서비스 그래프이다. WWTP+는 WWTP(그림 6)의 시간 2.5와 2.8에서 나타나는 급격한 보상과 서비스 중단을 막으면서 보상을 제공하며 보상이 끝난 후 WWTP와 거의 동일한 전송량을 보인다.

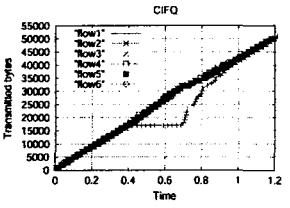

그림 8 CIFQ

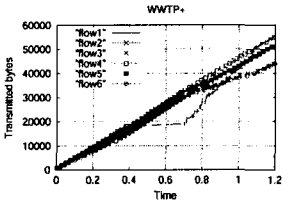

그림 9 WWTP+

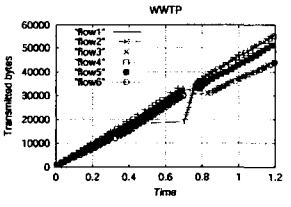

그림 10 WWTP

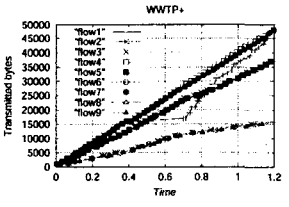

그림 11 WWTP+ (9 flows)

그림 8과 9는 통합 서비스(CIFQ)와의 비교 그래프이다. 패킷 크기는 1000bytes, 대역폭은 동일한 조건에서 시간 0.4부터 0.7까지 에러가 발생한 상황을 가정하였다. 흐름당 큐잉을 하는 통합 스케줄러와의 비교를 위해 흐름별로 서비스 그래프를 비교하였다. 흐름당 가중치에 의한 보상이 제공되는 CIFQ와 달리 WWTP+는 흐름이 속한 클래스의 DDP에 따른 차등화를 진행하면서 보상이 이루어진다. WWTP(그림 6,10)와 비교하여 WWTP+(그림 7,9)는 클래스와 흐름 단위의 서비스 중단을 모두 해결한다.

그림 11은 같은 조건에서 클래스당 3개의 흐름을 주고 클래스 내 흐름 편중을 비교하였다. 전체 대역폭에 비해 입력 패킷의 양이 증가하여 그림 9에 비해 우선순위 인덱스가 높은 큐로 전송이 더 편중되었다. 보상이 끝난 후 각 클래스의 서비스는 에러 프리 서비스에 근접한다. 같은 조건에서 CIFQ는 그림 8과 같은 그래프를 보여준다. 즉 WF^2Q[2]에 기반한 CIFQ가 가상 시간과 가중치에 기반한 보상을 제공한다면 WWTP+는 큐의 상태에 기반한 보상을 제공한다.

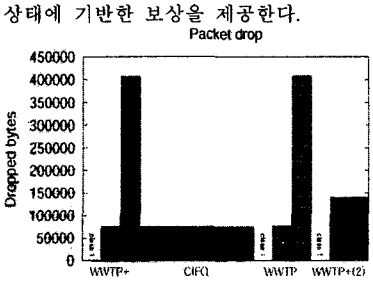

그림 12 Packet drop

그림 12는 각 알고리즘별 버려지는 패킷의 양을 보인다. 대역 1.75 Mb를 가정하였으며 나머지 조건은 앞과 동일하다. CIFQ는 가중치가 같으므로 모든 흐름이 같은 손실을 보이며 WWTP(+)는 우선순위 인덱스가 높은 큐의 손실이 적었다. WWTP+(2)는 WWTP+에 전체 큐 크기가 같다는 가정하에 각 클래스의 DDP에 비례한 큐 크기를 주었을 경우의 손실을 나타낸다. 이 때도 우선순위가 높은 클래스로 편중이 발생하지만 전체적으로 WWTP에 비해 패킷 드롭이 감소한다.

5 결론

본 논문에서는 무선 환경에서 상대적 지연 차등화를 제공하는 WWTP를 사용할 경우 발생할 수 있는 서비스 중단 현상을 개선하였다. 제안된 구현은 AWTP와 결합하여 다양한 패킷 크기를 가진 인터넷 환경에서 DDP에 더 정확하게 일치하도록 변경될 수 있다. 제안된 스케줄러는 짧은 오류와 긴 오류에 모두 작용하여 서비스 중단 현상을 개선한다. 그러나 무선 차등화 서비스의 대역 공평성 보장을 위해서는 유선 네트워크의 대역폭 중계기등과 관련된 연구가 요구된다.

참고문헌

[1] T. S. Eugene Ng, I. Stoica, and H. Zhang, "Packet fair queueing algorithms for wireless networks with location-dependent errors", Proceedings of IEEE INFOCOM'98, pp. 1103-1111, March 1998.

[2] J. C. R. Bennett and H. Zhang, "Worst-case fair weighted fair queueing", Proceedings of IEEE INFOCOM'96, pp. 120-128, March 1996.

[3] M. R. Jeong, K. Kakami, H. Morikawa, T. Aoyama, "Wireless Scheduler Providing Relative Delay Differentiation", Proc. The Third International Symposium on Wireless Personal Multimedia Communications (WPMC'00), Bangkok, Thailand, pp.1067-1072, Nov. 2000.

[4] Y.-C. Lai, W.-H. Li, and A. Chang, "A Novel Scheduler for the Proportional Delay Differentiation Model by Condiseri ng Packet Transmission Time", 2002.

[5] S.Blake, D.Black, M.Carlson, E.Davies, Z. Wang and W. Weiss,"An Architecture for Differentiated Services", RFC 2475, Dec. 1998.

[6] R. Braden, D. Clark, S. Shenker, "Integrated Service in the Internet Architecture: an Overview", IETF, RFC 1633, July 1994.