대역 공평성 보장을 위한 Core-Stateless 기법 연구

정회원 김 화 숙*, 김 상 하**, 김 영 부*

A Study of Core-Stateless Mechanism for Fair Bandwidth Allocation

Hwa-Suk Kim*, Sang-Ha Kim**, Young-Bu Kim* *Regular Members*

요 약

라우터에서 공평 대역 할당은 네트워크 폭주 상황에 반응하여 흐름을 제어하는 트래픽을 그렇지 않은 트래픽으로부터 보호한다. 그러나 Weighted Fair Queueing, Flow Random Early Drop 등의 전통적인 공평 대역 할당 방법은 상태 관리나 버퍼 관리, 스케쥴링 등을 흐름 단위로 수행하므로 초고속 네트워크의 백본에서 사용되는 경우 구현의 복잡성 뿐 아니라 흐름의 증가에 따른 네트워크의 확장성 문제를 야기한다.

최근 구현의 복잡성 문제를 극복하고, 네트워크의 확장성과 견고성을 확보하기 위해 코어 라우터에서의 흐름 단위 관리를 배제하는 Core-Stateless 메커니즘들이 제안되었다. Core-Stateless Fair Queueing과 Rainbow Fair Queueing 등은 Core-Stateless 네트워크에서 근접한 공평 큐잉을 실현하는 대표적 메카니즘들이다.

본 논문에서는 기존의 Core-Stateless 공평 대역 할당 메커니즘들에 비해 단순해진 패킷 레이블 할당 방법과 패킷 폐기 방법을 사용하여 실제 구현 가능성을 높인 새로운 Core-Stateless 알고리즘인 Simple Layered Fair Queueing(SLFQ)를 제안하고 몇가지 형태의 시뮬레이션과 그 결과를 다른 메커니즘의 의 결과와 비교하면서 대역 공평성 보장 정도를 확인한다. 마지막으로 제안된 알고리즘의 향후 응용 가능성을 제시한다.

키워드: Core-stateless, 대역공평성, Fairness, CSFQ

ABSTRACT

Fair bandwidth allocations at routers protect adaptive flows from non-adaptive ones and may simplify end-to-en d congestion control. However, traditional fair bandwidth allocation mechanisms, like Weighted Fair Queueing and Flow Random Early Drop, maintain state, manage buffers and perform packet scheduling on a per-flow basis. Th ese mechanisms are more complex and less scalable than simple FIFO queueing when they are used in the interi or of a high-speed network.

Recently, to overcome the implementation complexity problem and address the scalability and robustness, sever al fair bandwidth allocation mechanisms without per-flow state in the interior routers are proposed. Core-Stateless Fair Queueing and Rainbow Fair Queueing are approximates fair queueing in the core-stateless networks.

In this paper, we proposed simple Layered Fair Queueing (SLFQ), another core-stateless mechanism to approxi mate fair bandwidth allocation without per-flow state. SLFQ use simple layered scheme for packet labeling and h as simpler packet dropping algorithm than other core-stateless fair bandwidth allocation mechanisms. We presented simulations and evaluated the performance of SLFQ in comparison to other schemes. We also discussed other are as to which SLFQ is applicable.

Ⅰ. 서 론

인터넷의 기본 개념은 네트워크의 단순화를 통한 확장성에 있다. 이러한 특성으로 인해 인터넷의 폭주 제어 및 재전송 등의 알고리즘은 모두 단말에서 처리되도록 설계 및 구현되어 있다. 그 대표적인 예

* 한국전자통신연구원 네트워크연구소 네트워크구조팀(hwskim@etri.re.kr, ybkim@etri.re.kr)
** 충남대학교 컴퓨터과학과 네트워크연구실(shkim@cclab.cnu.ac.kr)
논문번호 : 030063-0212, 접수일자 : 2003년 02월 13일

가 TCP와 같은 종단간 폭주 제어 기능이다.

그러나 이러한 종단간 폭주제어만으로는 네트워크의 폭주 발생을 완전히 막을 수 없고, 네트워크의 상태와 관계없이 무차별적으로 유입되는 트래픽 흐름에 의한 폭주로 인해 종단간 폭주제어를 수행하는 트래픽의 흐름이 더욱 더 제약을 받게 된다. 비디오, 오디오 등의 인터넷 응용들이 대중화됨에 따라 폭주 제어를 하지 않는 UDP 형태의 트래픽이 인터넷에 대량으로 유입되는 것이 보편화되었으므로 이러한 트래픽에 의한 대역 할당의 불공평은 더욱 심하게 나타난다. 따라서 네트워크에서 노드 자체 자원의 유용성을 제어할 필요성이 대두되었다. 라우터에서 공평 대역 할당은 네트워크 폭주 상황에 반응하여 흐름을 제어하는 트래픽을 그렇지 않은 트래픽으로부터 보호함과 동시에 종단간 폭주 제어 메커니즘을 단순화하는 데 이용될 수 있고, 또한 다양한 종류의 종단간 흐름 제어 메커니즘이 네트워크 상에 공존하도록 해 준다[1].

이전까지 연구된 공평 대역 할당은 Weighted Fair Queueing(WFQ)[2]과 같은 흐름 단위의 큐잉 메커니즘들이나 Flow Random Early Drop(FRED)[3]과 같은 흐름 단위의 폐기 메커니즘들로 구분할 수 있다. 이 메커니즘들은 현재 인터넷에서 주로 사용되고 있는 폭주시 큐의 후반부를 폐기(tail-drop)하는 전통적인 FIFO(First In First Out) 큐잉에 비해 구현이 대단히 복잡하다. 특히 이 공평 대역 할당 메커니즘들은 흐름 단위의 상태 관리와 흐름 단위의 동작들을 필요로 한다. 패킷이 수신되면 라우터는 이 패킷이 속한 흐름 단위로 구분하고 흐름의 상태 변수를 수정하고, 흐름의 상태에 따라 정해진 동작을 수행한다. 이 동작은 FRED와 같이 단순하게 패킷을 폐기할 것인지 큐로 입력할 것인지를 결정하는 동작이거나 WFQ와 같이 우선 순위를 가진 큐를 조작하는 복잡한 동작이 될 수도 있다. 이러한 동작들을 단순화하기 위한 연구들이 진행되어 왔으나 흐름 단위의 관리나 제어는 여전히 남아 있어, 초고속 네트워크의 백본으로 사용되는 경우 이러한 기능은 구현의 복잡성 뿐 아니라 흐름의 증가에 따른 네트워크의 확장성에도 문제를 야기한다.

이러한 단점을 극복하기 위해 최근에 대두된 것이 Core-Stateless 공평 대역 할당 메커니즘이다. Core-Stateless Fair Queuing(CSFQ)[4]을 대표로 하는 Core Stateless 공평 대역 할당 메커니즘은 필요한 흐름 단위의 관리 및 제어를 에지 라우터에서만 수행함으로써 코어 라우터의 부하를 줄여 주는 메커니즘이다. 코어 라우터에서 필요한 패킷 상태 정보는 에지 라우터로부터 패킷 헤더에 실어 전달된다. 따라서 흐름의 수가 증가하더라도 코어 라우터에서는 흐름의 분류 및 상태 관리 없이 동일하게 처리되어 네트워크의 확장성을 보장할 수 있다. 각 Core-Stateless 공평 대역 할당 메커니즘들은 패킷 상태 정보를 할당하는 방법과 코어 라우터에서 공평 대역 유지를 위해 폭주시에 패킷을 폐기하는 방법을 다르게 정의하여 최선형(Best Effort) 트래픽 흐름간의 대역 공평성을 보장한다.

본 논문에서 다루고자 하는 공평 대역 할당 연구도 이 Core-Stateless 공평 대역 할당 메커니즘에 근거하고 있다. 인터넷이 백본 네트워크로 대두되고, 고속화되는 상황에서 네트워크의 확장성과 견고성(robustness)은 더욱 중요한 특성이 된다. 특히 현재 연구되고 있는 Multi Protocol Label Switching (MPLS)나 Differentiated Services(Diff-Serv)의 경우도 이러한 백본 네트워크의 고속화, 광역화를 위해 에지 노드에서 흐름 단위의 기능을 수행하고 코어 노드의 기능을 단순화하려는 메커니즘이라 볼 수 있다. MPLS에서는 트래픽 엔지니어링은 통해 트래픽 특성을 구분하고 경로 및 대역을 할당하여 Quality of Service(QoS)를 보장하고 Diff-Serv의 경우에도 트래픽의 특성을 몇 단계의 클래스로 나누고 코어 라우터에서 그 클래스에 따른 처리를 수행함으로써 QoS를 보장한다. 그러나 이러한 네트워크들에서도 최선형 트래픽에 대한 처리는 단순히 일정 대역이 할당되고 고전적인 FIFO에 의해 처리되므로 기존의 인터넷에서 발생하는 흐름 간의 불공평 및 폭주로 인한 성능 저하는 동일하게 발생한다. 따라서 최선형 트래픽에 대한 공평 대역 할당 메커니즘은 이러한 QoS보장을 전제로 하는 백본 네크워크의 성능 향상을 위해서도 여전히 필요하고 특히 네트워크의 확장성과 견고성을 위해 코어 라우터에서의 흐름 단위 관리를 배제하는 Core-Stateless 메커니즘이 요구된다.

이를 위해 Ⅱ장에서는 Core-Stateless 네트워크의 구조와 이 네트워크를 기본으로 기존에 연구된 공평 대역 할당 알고리즘들의 장단점을 살펴보고, Ⅲ장에서는 본 논문에서 제안하고자 하는 새로운 공평 대역 할당 메카니즘을 상세히 설명한다. Ⅳ장에서는 몇가지 형태의 시뮬레이션과 그 결과를 통해

본 논문에서 제안된 메커니즘의 대역 공평성 보장 정도를 확인하고, V장에서는 이 제안 알고리즘의 향후 응용 가능성 면을 살펴보고, 마지막으로 결론을 맺는다.

Ⅱ. Core-Stateless 네트워크와 공평성 보정 알고리즘

1. 네트워크 구조

Core-Stateless 네트워크 구조는 그림 1과 같이 나타낼 수 있으며, 흐름 단위의 상태 관리를 수행하는 에지(Edge) 라우터와 흐름 단위의 상태 관리를 수행하지 않는 코어(Core) 라우터로 구성된다.

Core-Stateless 네트워크는 하나의 도메인 형태를 이루며 에지 라우터는 가입자에 직접 연결되거나 도메인과 도메인 간의 경계면에 연결되는 라우터가 되고, 도메인 내부의 다른 라우터는 코어 라우터가 된다.

에지 라우터는 패킷을 흐름 단위로 분류하고, 흐름 단위로 속도를 예측하여 계산하며, 속도 또는 속도에 의해 계산된 흐름 정보를 패킷 헤더(packet header)에 실어 코어 라우터로 전달한다. 코어 라우터는 흐름 단위의 상태 관리없이 이 레이블 정보를 사용하고 필요시 라우터 내의 통합(aggregate) 상태 정보를 기준으로 이 레이블을 수정하여 다음 라우터로 전달한다. 레이블에 실어 전송되는 패킷 정보는 공평 대역 할당을 위한 정보가 되므로 패킷 흐름 속도를 기준으로 구해지나 실제 데이터는 알고리즘마다 다르게 표현된다.

또한 Core-Stateless 네트워크에서는 공평성 보장을 위해 필요한 흐름 단위의 버퍼링이나 스케줄링을 하지 않고 각 라우터는 확률적인 폐기 알고리즘을 사용하는 FIFO 큐잉을 사용한다 패킷을 폐기하는 확률은 레이블에 실려온 정보와 라우터에서 통합 트래픽의 측정에 의한 값 즉 라우터의 상태에 의해 결정된다. 패킷 폐기 알고리즘도 각 공평 대역 할당 메커니즘의 레이블 정보와 통합 데이터에 따라 다양하게 나타난다.

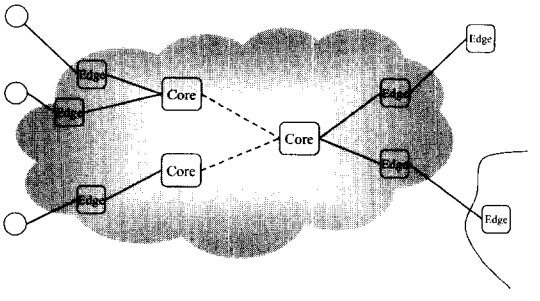


그림 1. Core-Stateless 네트워크의 구조
Figure 1. Architecture of Core-Stateless Network

2. 기존 Core-Stateless 공평 대역 할당 알고리즘

2.1 Core-Stateless Fair Queueing

Core-Stateless Fair Queueing (CSFQ)[4]은 Core-Stateless 네트워크를 기반으로 대역 공평성을 지원하는 대표적인 메카니즘이다. CSFQ의 코어 라우터는 출력 포트 단위로 구해진 공평 분할 속도를 기준으로 각 흐름의 대역 공평 할당을 보장한다. CSFQ를 지원하는 에지 라우터에서 코어 라우터로 전달되는 패킷의 헤더에 실리는 레이블은 패킷이 속한 흐름의 속도이다. 에지 라우터는 수신된 각 패킷에 대해 해당 흐름의 속도를 계산하고 이를 레이블 형태로 패킷 헤더에 실어 전송한다.

t_i^k와 l_i^k를 흐름 i의 k번째 패킷의 도착 시간 및 길이라 할 때 해당 흐름의 예측되는 도착 속도는 식 (1)과 같다

$$r_i^{new} = (1 - e^{-T_i^k / K})(l_i^k / T_i^k) + e^{-T_i^k / K} r_i^{old} \quad (1)$$

여기서 $T_i^k = t_i^k - t_i^{k-1}$이고 K는 상수이다. 지수 가중치 $e^{-T_i^k / K}$를 사용하여 지수 평균을 구함에 의해 예측된 속도가 실제 속도에 점근적으로 수렴하게 되고 오차의 한계를 정할 수 있게 해 준다. 이 지수 가중치를 사용하는 도착 속도 예측 방법은 이후의 모든 Core Stateless 공평 대역 할당 메커니즘에서 그대로 사용된다.

CSFQ의 코어 기능에서 폭주시에 패킷을 폐기하는 확률은 패킷의 레이블에 실려온 속도와 라우터에서 출력 포트의 공평 분할 속도(Fare share rate) 를 기준으로 구해진다. 패킷에 실려온 레이블이 공평 분할 속도보다 클 경우 ($\frac{r(t)-\alpha(t)}{r(t)}$) 비율로 폐기된다.

폭주 상태는 입력되는 모든 패킷에 대해 지수 평균을 사용하여 계산한 통합 도착 속도 $\hat{A}$를 정의하고 가 링크 속도보다 크면 폭주 상태로 판단한다.

공평 분할 속도는 출력 포트가 폭주 상태가 아닐 경우 가장 큰 흐름속도가 되고 폭주 상태일 경우는 폐기되지 않고 큐에 입력된 트래픽 량 $\hat{F}$(지수 평균을 사용하여 계산)와 링크 속도에 대한 비율에 의해 증가되거나 감소된다.

$$\hat{\alpha}_{new} = \begin{cases} \max_{1 \le i \le n} r_i & \hat{A} < C \\ \hat{\alpha}_{old} \dfrac{C}{\hat{F}} & \hat{A} \ge C \end{cases} \quad (2)$$

출력되는 패킷의 레이블은 폭주를 경험한 패킷 흐름의 속도를 반영하기 위해 패킷에 실려온 레이블 값과 공평 분할 속도 값 중 작은 값으로 수정된다.

CSFQ는 이후에 나타나는 Core-Stateless 네크워크에서의 공평 대역 할당 알고리즘들의 근간이 되는 알고리즘으로서 Core-Stateless 네트워크에서의 라우터의 기능과 성능 목표치의 기준이 된다. 다만 CSFQ의 코어 노드에서 공평 분할 속도를 계산하기 위해서 패킷 단위로 지수 평균을 계산해야 하는 부분은 개선의 여지가 있다고 생각된다.

2.2 Core-Stateless Random Early Detection

Core-Stateless Random Early Detection (CS-RED)[7]은 RED의 폭주 통지 기법과 CSFQ의 분산 기법을 결합한 방법이다. TCP와 폭주 제어를 하지 않는 흐름을 서로 나누어 기능을 수행한다.

먼저 TCP흐름에 대해서는 에지 기능은 CSFQ와 동일하게 예측된 속도를 패킷 레이블에 실어 전달하고, 코어 기능에서 패킷의 폐기는 표준 속도(V)와 폭주 정도($1/m$)에 의해 결정된다. 표준 속도는 큐에 입력된 패킷들 중에서 확률적으로 샘플링한 패킷들의 레이블의 중간값이다. 폭주 정도($1/m$)는 큐의 최대 크기(Q_{max})와 현재 큐의 길이($Q_{current}$)의 비율로 계산된다. 수신된 패킷 레이블(L)이 이 표준 속도보다 클 경우 패킷을 확률적으로 폐기한다. 패킷 폐기 확률은 패킷 레이블과 표준 속도의 비율(l)과 폭주 정도에 의해 계산된다.

폭주 제어를 하지 않는 흐름에 대해서는 차단 속도(mV) 이상의 레이블을 가진 패킷을 폐기한다.

이 메커니즘에서는 링크의 전파 지연 시간이나 입력 속도의 측정 간격 파라미터에 따른 CSFQ의 TCP 트래픽에 대한 공평성 성능 저하를 지적하고, 이를 향상시키기 위한 알고리즘과 표준 속도를 구하기 위한 파라미터 선택에 따른 공평성 정도를 보여 준다. 이 알고리즘은 TCP 흐름 간의 공평성 보장에 중점을 두어 공평 대역 이상으로 전송하는 CBR 흐름은 상대적으로 공평 대역 이하로 지나치게 저하된다.

2.3 Rainbow Fair Queueing

Rainbow Fair Queueing(RFQ)[5]은 색 표시 기법(color labeling scheme)과 버퍼 관리 메커니즘을 결합한 방법이다. 각 흐름을 전체적으로 연속되는 속도의 계층으로 나누고, 각 계층은 색으로 표현한다. 각 색 계층은 저속에서 더 세밀하게 나뉘어진다.

각 계층 i에 할당되는 대역 C_i는 식 (3)과 같이 계산된다.

$$C_i = \begin{cases} \dfrac{(1-a^{-1})^{N/b-INT(i/b)-1}}{ab} P \\ \dfrac{(1-a^{-1})^{N/b-1}}{b} P \end{cases} \quad (3)$$

여기서 N은 전체 계층의 수, a와 b는 계층 구조를 결정하는 상수로 N은 b의 배수가 되도록 결정되고, P는 네트워크에서의 최대 속도이다.

에지 라우터에서 레이블 할당을 위한 패킷 흐름의 속도는 CSFQ와 동일한 방법으로 예측되고, 이 도착 속도가 r일때 할당되는 레이블의 범위는 $0 \le i < j$로 j는 $\sum_{i=0}^{j} C_i \ge r$를 만족하는 최소의 값이다. 그리고 $C_i / \sum_{i=0}^{j} C_i$ 비율로 레이블 i값이 패킷에 할당된다.

코어 라우터는 색 한계값 C를 관리하고 패킷의 색 레이블이 C보다 크면 폐기한다. 폭주 상태가 되면 색 한계값은 감소되고 폭주가 해소되면 색 한계값은 증가된다. 속도를 기준으로 색 표시가 되므로 패킷의 폐기도 대체로 공평하게 이루어진다.

폭주는 큐의 한계값(threshold)과 현재 큐에 저장된 패킷량, 일정 시간동안 수신된 패킷량을 기준으로 계산된다.

RFQ의 레이블은 속도에 의해 계산되고, 또한 계층화되어 할당되므로 폭주시 상위 계층의 레이블을 가진 패킷의 폐기만으로 공평 대역 할당에 접근 가능하고, 코어 기능에서의 레이블 재할당이 필요 없는 장점이 있고, 작은 레이블 값이 폐기될 확률이 작으므로 패킷의 우선 순위가 필요한 응용에 사용

가능하다. 다만 레이블의 할당을 위한 계층의 용량 계산과 이를 위한 확률 계산이 복잡하고, 코어 라우터에서 수신되는 패킷의 수를 저장해야 하는 단점이 있다.

2.4 Queue Length based Fair Queueing

ueue Length based Fair Queueing(QLFQ)[6]도 RFQ와 같이 계층화 기법(layering scheme)과 버퍼 관리 메커니즘을 사용한다 따라서 QLFQ의 패킷에 실리는 레이블도 속도에 의해 계산된 계층값이다. QLFQ는 모든 계층의 용량을 동일하게 계산한다(linear layering). 따라서 하나의 계층 용량을 C라 할때 속도 r인 흐름에 할당되는 계층값들 즉 레이블 l의 범위는 $j*C>r$인 최소의 j에 대해 $0 \leq l \leq (j-1)$이고, 각 레이블 값 l은 $1/j$의 확률로 할당된다.

코어 라우터는 폐기 한계값 *Drop_thresh*를 관리하고 패킷의 레이블이 폐기 한계값보다 크면 폐기한다.

폐기 한계값은 현재 큐의 상태와 이전 큐의 상태, 이전 폐기 한계값의 함수로 계산된다. 현재 큐 상태는 폭주 정도를 반영한다는 원칙 아래 큐에 3개의 큐 한계값을 두어 폭주 상태를 4단계로 나누고 현재 큐 값의 위치에 따라 한계값의 증감량을 다르게 한다.

이 메커니즘은 계층의 용량을 동일하게 계산함에 의해 RFQ에 비하여 에지 라우터에서 레이블 할당시 계층 확률 계산 등을 단순화하였으나 코어 라우터에서의 한계값 계산 방법은 상대적으로 복잡해진다.

Ⅲ. 제안된 Core-Stateless 공평 대역 할당 기법

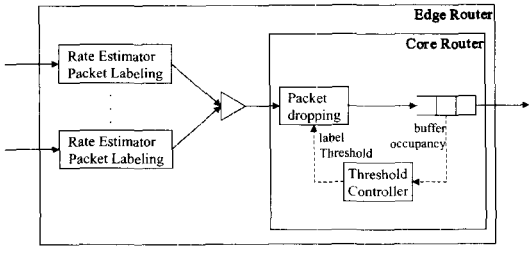


그림 2. SLFQ 에지 라우터 및 코어 라우터 기능 구조도
Figure 2. Functional Architecture of SLFQ

본 장에서는 기존 Core-Stateless 네트워크에서의 대역 공평성 보장 방법들의 장점들을 활용하고, 좀 더 구현하기 쉬우면서도 흐름 단위의 대역 공평성 보장에 준하는 메커니즘을 제안하고자 한다.

이 메커니즘은 코어 라우터의 흐름 제어 없이 흐름 속도의 계층화와 큐 관리를 통해, 흐름 단위의 관리 및 제어를 수행하는 FRED이상의 대역 공평성 보장을 도모하고, 더불어 CSFQ나 RFQ와 같은 다른 Core-Stateless 네트워크의 대역 공평성 보장 알고리즘을 단순화하기 위한 방법을 제공하는데 그 목적이 있다. 이후 이 알고리즘을 Simple Layered Fair Queueing(SLFQ)이라 한다.

SLFQ도 에지 라우터와 코어 라우터로 구성된 Core-Stateless 네트워크 구조를 그대로 사용한다. 이 구조 상에서 SLFQ를 지원하는 에지 라우터와 코어 라우터의 기능 구조를 나타낸 것이 그림 2이다.

SLFQ에서도 라우터의 기능은 크게 에지 라우터의 입력 포트에서 수행되는 에지 기능과 에지 라우터와 코어 라우터의 출력 포트에서 즉 모든 라우터에서 수행되는 코어 기능으로 나눌 수 있다.

에지 기능은 패킷을 흐름 단위로 구분하여 속도를 예측하는 기능과 그 속도를 기준으로 레이블 값을 계산하고 이를 패킷 헤더에 실어 코어 기능으로 전달하는 기능을 수행한다.

코어 기능은 FIFO 큐의 관리 기능과 함께 큐에 저장된 데이터 량을 바탕으로 레이블 한계값 제어를 수행하고, 계산된 레이블 한계값을 기준으로 패킷 폐기를 처리하는 기능을 수행한다.

1. 흐름 도착 속도 예측

기본적으로 대역 할당의 공평성 보장을 위해 필요한 흐름 단위 데이터는 각 흐름의 속도이다. 이 논문에서 다루고자 하는 트래픽의 대상도 별도의 자원 예약을 하지 않는 최선형 트래픽이기 때문에 각 흐름 속도의 예측은 Ⅱ장의 2.1절에서 설명한 CSFQ 메카니즘에서 사용하는 식 (1)의 지수 가중 평균을 그대로 사용하여 구한다.

2. 레이블 할당

SLFQ에서는 RFQ와 같이 레이블 할당(Labeling)을 위해 속도의 계층화(layering) 기법을 사용한다. 속도를 계층으로 나누고 각 계층에 정수를 할당한다. 즉 l 계층의 용량은 Cl이고, 속도 r

인 흐름은 l_{max}개의 계층으로 나뉘어져 0~(l_{max}-1)의 계층 값을 가진다. 여기서 l_{max}는 $\sum_{l=0}^{l_{max}-1} C_l > r$인 최소의 l_{max}이다.

계층 분할 방법, 즉 C_l의 용량을 정하는 방법은 다양하게 나타날 수 있는데 가장 간단한 방법은 QLFQ와 같이 최대 전송 속도를 레이블의 수로 동일하게 분할하는 방법이다. 그러나 이 경우 저속의 흐름에 대해서 하나의 레이블에 의해 표현되는 계층의 용량이 속도에 비해 상대적으로 크다. 따라서 저속의 다수 흐름에 의한 폭주의 경우에 레이블의 감소는 필요 이상의 패킷 폐기를 초래하여 자원의 사용 면에서 비효율적이다. 따라서 레이블이 나타내는 속도의 용량이 저속은 작게, 고속에서는 크게 나타나야 한다. 이를 위한 방법으로 SLFQ에서는 로그 함수를 사용하여 속도 r인 흐름이 가질 수 있는 최대 레이블 값 l_{max}를 식 (4)와 같이 계산한다.

$$l_{\max} = \left\lfloor L_{\max} \frac{\log_2(r)}{\log_2(R_{\max})} \right\rfloor \quad (4)$$

여기서 L_{max}는 SLFQ에서 표현될 수 있는 레이블 범위 즉 계층의 수이고, R_{max}는 네트워크에서 나타날 수 있는 흐름 최대 속도이다. 이 식에서 실제 레이블 계산시 L_{max}와 $\log(R_{max})$는 네트워크에서 정의되는 상수 값이므로 $L_{max}/\log(R_{max})=K$로 두면 식 (4)는 식 (5)와 같이 정의할 수 있다.

$$l_{\max} = \lfloor K \log_2(r) \rfloor \quad (5)$$

SLFQ의 속도와 레이블의 관계도는 그림 3과 같이 나타낼 수 있다.

계층 l의 용량은 $C_l = 2^{\frac{l}{K}} - 2^{\frac{l-1}{K}}$이다.

수신된 하나의 패킷에 부여되는 레이블 값에 할당되는 레이블 값 l을 C_l/r확률로 임의(random)로 할당하기 위해 식 (6)과 같이 계산한다.

$$l = \lfloor K \log_2(pr) \rfloor \quad (6)$$

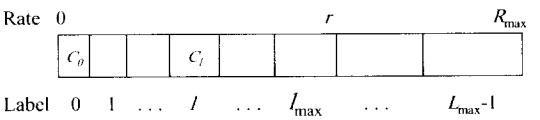


그림 3. 속도와 레이블 관계도
Figure 3. Relation between Rate and Label

여기서 p는 0과 1사이에서 균등하게 분포된 확률변수이다.

3. 패킷 폐기

SLFQ의 코어 기능에서는 레이블 한계값 이상의 레이블을 가진 패킷을 폐기한다. 즉 패킷에 실려온 레이블 l이 현재의 레이블 한계값 L_{th}보다 큰 경우 폐기한다. 이로써 각 흐름별로 동일하게 레이블 한계값 이하의 레이블 값을 가진 패킷만 입력되므로 각 흐름 간의 공평성 보장에 접근할 수 있다. 즉 SLFQ에서 레이블 한계값 L_{th}에 대해 각 흐름에 할당되는 공평 분할 속도는 $\sum_{l=0}^{L_{th}} C_l = 2^{\frac{L_{th}}{K}}$이다

4. 레이블 한계값 계산

SLFQ의 레이블 폐기 기준이 되는 레이블 한계값은 네트워크의 상태에 따라 판단된다. 네트워크가 폭주 상태로 판단되면 감소되고, 대역 자원을 충분히 활용하지 못한다고(under utilized) 판단되면 증가시킨다. 레이블 증가량과 감소량은 항상 1로 하여 단순화한다. 그러나 레이블과 속도의 관계에 따른 구조에 의해 저속과 고속에서 공평 속도의 증감량은 다르게 나타난다.

네트워크의 상태는 현재 큐의 상태로 판단한다. 그림 4과 같이 큐에 큐 한계값(Q_{th})을 기준으로 한계 영역(Q_{th} a~Q_{th}+a)을 두고 현재 상태값이 큐 한계 영역에 있을 때 자원이 이상적으로 사용되고 있다고 판단한다. 실제로 큐에 데이터가 남아있는 경우는 입력되는 패킷량이 출력되는 패킷량보다 많은 경우인 폭주를 의미하지만 순간적인 데이터 폭주로 인한 반응으로 다음의 데이터 고갈 상태를 일으키는 것을 방지하기 위해 큐의 현재 상태값이 큐 한계 영역 이하일때는 대역 자원이 충분히 활용되지 못하다고 판단한다. 그리고 큐의 현재 상태값이 큐 한계 영역 이상일 경우는 입력이 출력보다 지속적으로 많은 경우로 폭주 상태로 판단한다.

큐의 현재 상태를 큐 한계 영역으로 접근시키기 위

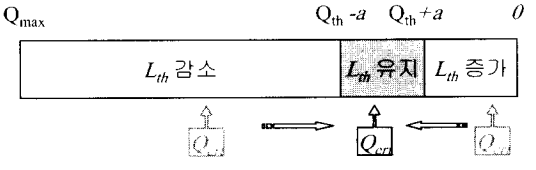


그림 4. 큐의 상태와 레이블 한계값
Figure 4. Queue State and Label Threshold

```
on receiving packet p :
  if (edge router)
    i =classify(p);
    r = estimate_rate(r, p);
    p.label = assign_label(r);
  if (p.label > Lth)
    drop(p);
  else if (Qcrt = Qmax)
    drop(p);
    Lth = Lth -1;
  else
    enqueue(p);
on departing packet p :
  if (Qcrt > Qth2) && (q_last >= Qcrt)
    Lth = Lth - 1;
  else if (Qcrt < Qth1) && (q_last <= Qcrt)
    Lth = Lth +1;
on empty queue :
  Lth = Lth +1;
```

그림 5. 코어 기능 알고리즘
Figure 5. Pseudo Code of Core Function

해 레이블 한계값을 조정한다. 큐의 현재 상태가 큐 한계 영역 이하이면 레이블 한계값을 증가하여 폐기되는 패킷량을 줄여 입력되는 패킷량을 증가시키고, 큐의 현재 상태가 큐 한계 영역 이상이면 레이블 한계값을 감소하여 폐기되는 패킷량을 늘여 입력되는 패킷량을 감소시킨다. 큐의 현재 상태가 큐 한계 영역 내에 있을 경우는 레이블 한계값을 유지한다.

그림 5는 코어 기능의 알고리즘을 나타내는 가상 코드이다. 레이블 한계값의 수정은 패킷이 출력되는 시간을 기준으로 적용함에 의해 별도의 타이머 사용을 줄여 구현의 단순화 효과를 가져온다.

레이블 한계값의 조정시에 현재 큐 상태값을 큐 한계 영역과 비교하는 것 이외에 이전 큐의 상태값을 비교한다. 이는 현재의 추세를 반영하여 과잉 반응을 막기 위한 것으로 큐 한계 영역 이상의 상태에서 레이블 한계값 감소시는 큐가 감소 추세인 경우를 배제하기 위함이고, 큐 한계 영역 이하의 상태에서 레이블 한계값 증가시는 큐가 증가 추세인 경우를 배제하기 위함이다. 또 극한 상황에서 빠르게 회복되도록 큐가 가득 차서 폐기가 일어날 경우 레이블 한계값을 감소하고, 출력할 패킷이 없을 경우 레이블 한계값을 증가한다.

Ⅳ. 시뮬레이션 및 결과 분석

이 장에서는 몇가지 네트워크 토폴로지에서의 시뮬레이션을 통해 나타나는 SLFQ의 결과를 분석한다. 시뮬레이션에서 SLFQ는 기존의 공평성 보장 알고리즘인 Deficit Round Robin(DRR), Random Early Drop(RED), Flow Random Early Drop(FRED), Core-Stateless Fair Queueing(CSFQ)의 결과와 비교한다.

DRR[13]은 WFQ의 한 형태로 흐름 단위의 큐에 저장된 데이터를 Round-Robin 방식으로 차례로 서비스하고 버퍼가 가득 찾을때 가장 긴 큐의 패킷을 폐기한다. 시뮬레이션에서 사용되는 알고리즘 중에 유일하게 흐름 단위의 큐잉 방식을 사용하는 알고리즘이다.

RED[14]는 FIFO 큐를 사용하면서 버퍼의 폭주가 일어나기 전에 패킷을 임의의 확률로 폐기하는 폭주 회피 알고리즘이다. RED는 두개의 버퍼 한계값을 관리한다. 지수 평균으로 계산된 버퍼 점유량이 첫번째 한계값 이하이면 패킷을 폐기하지 않고, 두번째 한계값 이상이면 모든 패킷을 폐기한다. 버퍼 점유량이 두 한계값 사이에 있을 경우 버퍼 점유량에 비례하여 폐기되는 확률을 늘인다.

FRED는 대역의 공평 할당을 위해 RED를 확장한 알고리즘이다[3]. 공평성을 보장하기 위하여 모든 흐름의 패킷이 버퍼에 적어도 하나 이상 남아있도록 상태를 유지한다. 폐기의 결정은 흐름 상태에 기반한다. 특히 이전에 가장 많이 폐기된 흐름의 패킷이나 평균 큐의 크기보다 긴 큐의 패킷을 폐기한다. FRED는 FRED1과 FRED2로 구현되는데 그 차이는 FRED2는 각 흐름에 최소의 버퍼수 할당을 보장한다.

모든 시뮬레이션은 시뮬레이터 ns-2[15]로 구현되었다. CSFQ와 FRED의 시뮬레이션 코드는 CSFQ 참고 사이트[16]로부터 가져온 것을 그대로 사용한다. DRR, RED 소스 코드와 TCP, UDP 트래픽 생성을 위해서는 ns-2 패키지를 그대로 이용한다.

시뮬레이션을 위한 파라미터로 각 출력 포트의 용량은 10Mbps, 링크의 전파 지연 시간은 1ms, 버퍼의 크기는 64Kbyte로 하고, FRED와 RED의 두 한계값은 각각 16Kbyte(25%)와 32Kbyte(50%)로 한다. 생성되는 TCP, UDP 패킷의 크기는 1000byte이다. SLFQ를 위한 파라미터로 최대 계

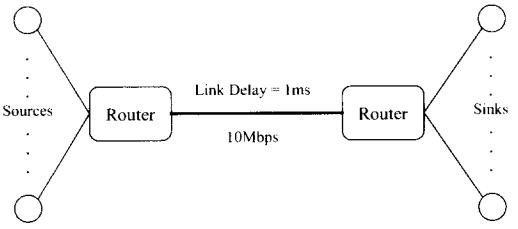


그림 6. 하나의 폭주 링크를 가진 네트워크 토폴로지 (토폴로지 1)
Figure 6. Network topology with one congested link (Topology 1)

층의 수 L_{max}는 256이고, 네트워크에서의 최대 흐름 속도 R_{max}는 10Mbps로 한다. 또 버퍼의 한계값은 버퍼 전체 크기의 25%로 한다.

흐름 단위의 관리를 통해 공평 대역 이상의 UDP 트래픽에 대해 제약을 가하여 TCP 흐름에 공평 대역을 보장할 경우 TCP의 goodput도 공평 대역으로 수렴함을 S.Floyd의 논문 "Promoting the Use of End-to-End Congestion in the Internet"[11]에서 시뮬레이션을 통해 보여주고 있다. 따라서 본 논문에서 시뮬레이션 결과의 흐름간 공평 대역 할당 정도는 UDP 또는 TCP등의 흐름 종류에 관계없이 각 흐름의 처리량(Throughput)으로 판단한다.

1. 단일 폭주 링크를 가진 네트워크

먼저 그림 6과 같은 형태의 단순한 네트워크 구성에서 시뮬레이션을 시작한다. 10Mbps 용량의 하나의 병목 링크로 N개의 흐름이 되는 토폴로지상에서 세가지 형태의 시뮬레이션을 시도한다.

첫번째 시뮬레이션은 32개의 서로 다른 속도의 UDP 흐름이 하나의 링크를 공유하는 경우의 대역 할당 상태를 확인한다. 각 흐름에 0~31까지의 번호를 할당하고 흐름 i의 속도는 이상적인 공평 분할 속도의 i+1배 즉, $(i+1)*10/32$Mbps로 전송한다. 예를 들어 흐름 0은 10/32Mbps, 흐름 1은 2*10/32Mbs, 흐름 31은 10Mbps가 된다.

그림 7는 10초 동안의 각 흐름별 평균 처리량, 즉 목적지에 도달한 트래픽 량을 보여준다. 그래프 상에서 DRR은 이상적인 대역 공평성 분할에 가장 가까운 형태를 나타낸다. RED와 FRED1은 흐름별 대역 공평성을 전혀 지원하지 못하고 높은 속도의 흐름이 높은 대역을 차지한다. CSFQ, FRED2, SLFQ는 각 흐름의 평균 할당 대역이 공평 분할 속도에 근접함을 보여준다.

단일 폭주 링크 토폴로지에서 두번째 시뮬레이션은 폭주 제어를 전혀 하지 않는 UDP흐름이 TCP 흐름들에 미치는 영향을 살펴보기 위한 실험이다. 0번 흐름은 UDP 소스로부터 링크 속도인 10Mbps로 전송하고, 나머지(1~31번) 흐름들은 TCP소스로부터 전송한다.

그림 8은 10초 동안의 시뮬레이션 결과를 보여준다. DRR 다음으로 CSFQ가 UDP 트래픽의 속도를 공평 분할 속도의 2배 이하로 제어하여 TCP 흐름이 공평 분할 속도에 근접하도록 제어하였고, 그 다음으로 FRED1과 SLFQ가 UDP 흐름의 속도를 각각 링크 속도의 12%와 17% 정도로 제어하고 TCP 흐름들을 나머지 링크 대역에서을 공유한다. FRED2은 UDP 흐름이 고평 분할 속도의 10배 정도가 되어 TCP 흐름들의 속도가 저하되나 TCP 흐름 간의 공평성은 유지된다. RED는 UDP가 링

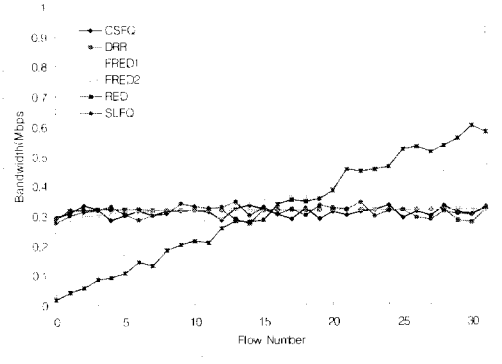


그림 7. 토폴로지 1에서 32 UDP 흐름의 평균 처리량
Figure 7. Average Throughput of 32 UDP flows in the Topology 1

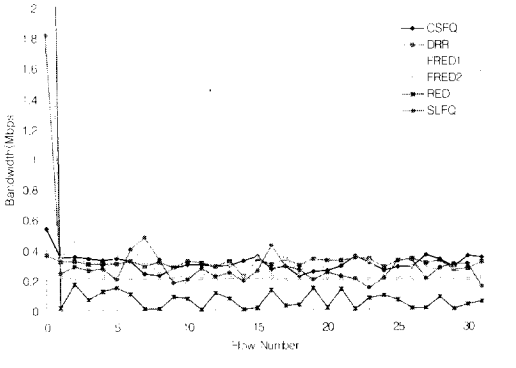


그림 8. 토폴로지 1의 1 UDP, 31 TCP 흐름의 평균 처리량
Figure 8. Average Throughput of 1 UDP and 31 TCP flows in the Topology 1

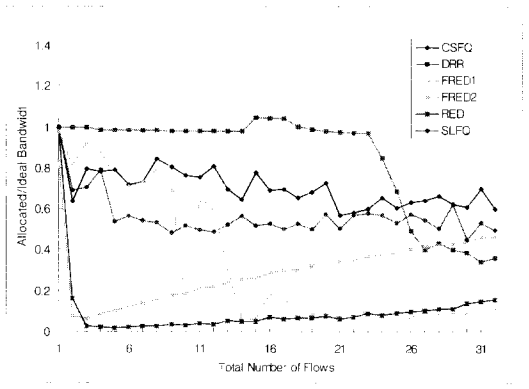


그림 9. 토폴로지 1의 1 TCP, (N-1)UDP에서 TCP 흐름의 평균 처리량
Figure 9. Average Throughput of TCP flow out of 1 UDP and 31 TCP flows in the Topology 1

크 대역의 70%를 차지하여 31개의 TCP 흐름이 30%를 고유하며 공평성도 찾아볼 수 없다

동일 토폴로지에서 세번째 시뮬레이션은 하나의 TCP 흐름이 네트워크의 폭주에 반응하지 않는 다수의 UDP 트래픽에 의해 어떻게 보호되는지를 보기 위한 실험이다. 흐름의 수 N을 1~32까지 달리한 32번의 실험을 통해 결과를 수집한다. 이때 하나의 흐름은 TCP 소스로부터 생성하고 나머지 N-1(흐름 1~N-1)개의 흐름은 UDP 소스로부터 공평 분할속도(10/N Mbps)의 2배의 속도로 생성한다. 그림 9는 10초동안의 시뮬레이션 수행 결과로 각 N개의 흐름을 가진 32번의 시뮬레이션에서 TCP 흐름에 할당된 속도를 보여준다. 할당된 TCP의 속도는 공평 분할 속도에 대한 비율로 나타내었다. 여기서 특이한 점은 DRR은 22개의 흐름 이하의 흐름이 존재하는 경우 공평 분할 속도에 가깝게 TCP 흐름에 대역을 할당하나 그 이후 현저히 떨어진다. TCP는 흐름에 할당된 버퍼가 3개 패킷 이하일 때 그 성능이 현저히 떨어지기 때문이다. CSFQ와 SLFQ는 흐름 수가 많을 경우에도 공평 분할속도의 60%와 50% 이상의 대역으로 TCP 흐름을 유지시켜 준다. 즉 CSFQ와 SLFQ가 일정 수 이상의 많은 트래픽 흐름 상에서 DRR보다 더 나은 성능을 보여준다. 이는 짧은 시간에 입력된 소수의 TCP 패킷의 버퍼링을 허용함에 의해 TCP의 버스트성에 더 잘 반응한다고 볼 수 있다. 전체적으로도 SLFQ는 CSFQ보다는 낮지만 FRED1이나 FRED2보다는 훨씬 더 높은 성능을 갖는다.

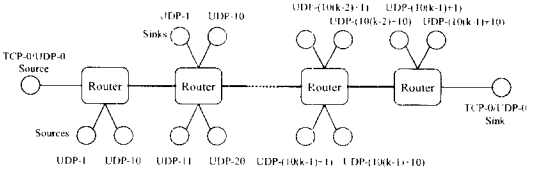


그림 10. 다중 폭주 링크를 가진 네트워크 토폴로지 (토폴로지 2)
Figure 10. Network topology with multiple congested link (Topology 2)

2. 다중 폭주 링크를 가진 네트워크

이 절에서는 그림 10와 같은 형태의 다중 폭주 링크를 가진 토폴로지에서의 시뮬레이션과 그 결과를 분석한다. 이 시뮬레이션을 통해 하나 이상의 폭주 링크를 경험한 트래픽 흐름에 대한 공평 대역 할당의 영향을 분석한다. 0번 흐름은 첫번째 노드에 연결된 소스에서 발생되어 차례대로 연결된 라우터를 거쳐 마지막 라우터에 연결된 착신지에 도착한다. 각 링크에는 이전 라우터와 연결된 10개의 UDP 소스들로부터 발생한 흐름들이 다음 라우터와 연결된 착신지들로 향하도록 하고 각 흐름의 속도를 2Mbps로 발생시킨다. 링크의 용량이 10Mbps이므로 모든 링크가 폭주 상태가 된다. 이 상황 하에서 0번 흐름의 특성에 따라 두가지 시뮬레이션을 수행한다.

첫번째 시뮬레이션은 첫 라우터로부터 마지막 라우터까지 다수개의 폭주 링크를 경험하는 0번째 흐름을 UDP 소스로부터 발생시키는 시뮬레이션이다. 이 UDP-0의 속도는 각 링크의 공평 분할 속도가 되는 10/11Mbps이다. 그림 11은 통과되는 폭주 링크 수에 따른 UDP-0의 처리량을 이상적으로 할당될 공평 대역의 비로 나타내었다. SLFQ는 DRR, CSFQ 다음으로 FRED2와 비슷하고 FRED1보다

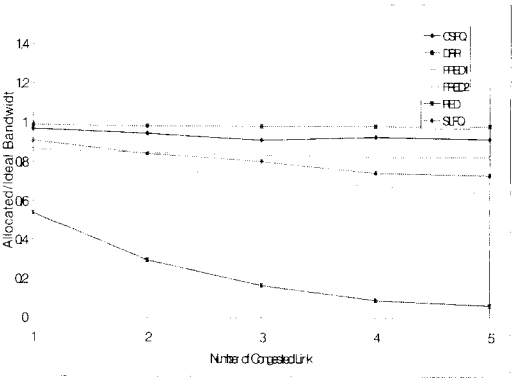


그림 11. 토폴로지 2의 UDP-0 처리량
Figure 11. Throughput of UDP-0 flow in the Topology 2

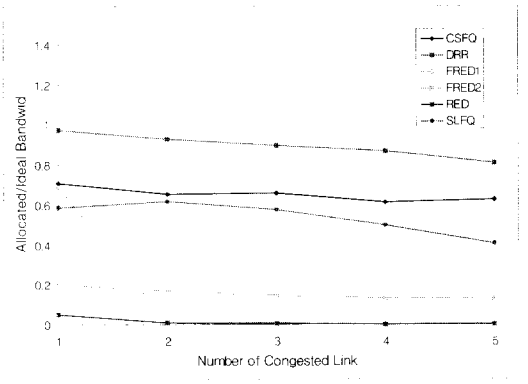


그림 12. 토폴로지 2의 TCP-0 처리량
Figure 12. Throughput of TCP-0 flow in the Topology 2

나은 공평성 보장의 효과를 보여준다. 공평 대역 할당을 제어하지 않는 RED는 아주 낮은 처리량을 나타낸다.

두번째 시뮬레이션은 첫번째 시뮬레이션의 트래픽 UDP-0를 TCP-0로 대체한 것이다. 그림 12은 폭주 링크 수에 따른 TCP 흐름의 처리량을 공평 대역으로 정규화하여 그래프로 나타내었다. 전체적으로 폭주를 많이 경험할수록 UDP에 비해 TCP의 처리량이 떨어진다. 이 그래프에서도 SLFQ는 DRR, CSFQ 다음으로 FRED1, FRED2보다 월등히 나은 성능을 나타낸다.

3. 링크 단위의 공평 대역 할당

이 절에서는 링크 단위의 공평성 할당을 보여주기 위해 그림 13과 같은 형태의 토폴로지를 구성하고 시뮬레이션을 시도하였다. 링크가 폭주 상태가 되어 패킷 손실이 발생하면 CSFQ에서는 흐름의 새로운 속도를 반영하기 위해 패킷의 레이블을 공평 분할 속도로 재할당하지만, SLFQ는 패킷의 레이블을 재할당할 필요가 없다. 레이블 한계값 이상의 레이블을 가진 패킷만 손실되고 자연히 그 이하의 레이블 값을 가진 패킷만 남은 상태가 되어 새

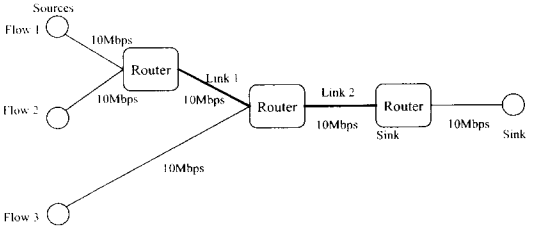


그림 13. 네트워크 토폴로지 3
Figure 13. Network topology 3

표 1. Link 2의 흐름별 처리량(Mbps)

		Flow 1	Flow 2	Flow 3
CSFQ	UDP	3.3304	3.3280	3.3280
	TCP	3.4376	3.2240	3.3240
SLFQ	UDP	3.2704	3.2976	3.4304
	TCP	3.4928	3.1960	3.2763

로운 속도가 이미 반영된 상태가 된다

각 링크는 10Mbps 용량을 가지며 전파 지연시간은 1ms이다. 3개의 흐름이 각 링크에서 공평 분할 속도 이상의 대역을 요구한다면 이상적인 경우 흐름 1, 2는 링크 1에서는 5Mbps, 링크 2에서는 3.33Mbps의 속도를 할당받을 수 있으므로 두 링크 모두에서 손실을 경험한다.

처음에는 3개의 UDP 소스로부터 10Mbps속도로 데이터를 전송하고, 다음으로 3개의 TCP 소스로부터 데이터를 전송한다. 표 1은 링크2를 통해 10초 동안 평균적으로 수신된 UDP 처리량과 TCP 처리량을 흐름별로 보여준다. UDP 각 흐름의 속도는 CSFQ에 비해 성능은 다소 낮으나 공평 분할 속도 3.33Mbps에 근접한다. TCP 흐름은 UDP 흐름의 성능보다는 낮으나 버스트 특성에도 불구하고 대체로 공평 대역 할당에 근접함을 보여준다.

4. 시뮬레이션 결과 검토

이상의 절에서 몇가지 형태의 네트워크 환경 하에서 SLFQ의 공평 대역 할당에 대한 성능을 살펴보았다. DRR의 링크 단위의 큐잉을 통해 성능이 가장 우수하나 이는 구현의 복잡성으로 네트워크의 확장성을 지원하지 못하므로 단지 공평성 보장의 기준으로 시뮬레이션 되었다. CSFQ와 비교할때 CSFQ의 성능이 다소 우수하나 SLFQ는 CSFQ보다 코어 라우터의 기능을 한층 더 단순화하면서 공평성을 제공한다는 장점을 들 수 있다. SLFQ는 공평성을 지원하지 않는 RED와 비교할 경우 동일하게 FIFO 큐를 사용함에도 불구하고 대역 할당의 공평성을 보장하는 것이 확연히 드러나며 그 수준은 FRED와 비슷하거나 더 나은 성능을 나타낸다. 그러나 FRED는 코어 노드에서 패킷 흐름 단위의 관리가 필요하므로 네트워크 확장성의 저해 요소가 된다. 전체적으로 이 시험들의 결과 SLFQ를 통해 코어 노드에서의 상태 관리 없이, 또한 코어 네트워크의 단순화를 유지하면서 일정 수준의 대역 공평성을 보장할 수 있음을 보았다.

V. SLFQ의 응용

이전 장까지는 기본적인 SLFQ 메커니즘을 사용하여 TCP 또는 UDP 등의 최선형 트래픽에 대한 공평 대역 할당을 통해 네트워크 자원의 효율적인 사용과 함께 무차별적으로 입력되는 트래픽으로부터 네트워크의 폭주에 반응하는 트래픽 흐름을 보호하는 기능을 설명하였다.
이 장에서는 SLFQ가 인터넷의 확장성을 유지하면서 제공할 수 있는 다른 응용 면을 설명한다.

1. 가중치를 가진 흐름간 공평 대역 할당 보장

SLFQ는 가중치를 가진 트래픽에 대한 공평성 보장을 지원할 수 있다. 가중치 ω이고 속도 r인 흐름에 대해서는 기본 SLFQ의 메커니즘에서 속도 r/ω인 흐름과 동일하게 레이블을 할당한다. 따라서 이 흐름에 할당될 수 있는 레이블의 최대값은 식 (7)과 같다.

$$l_{max} = L_{max}\frac{\log_2(r/\omega)}{\log_2(R_{max})} \qquad (7)$$

이 경우 ω가 1보다 큰 경우는 ω가 1인 흐름보다 낮은 값의 레이블이 할당되어 그 만큼 폐기될 확률이 떨어지고, 또 반대로 ω가 1보다 작은 경우는 비례적으로 높은 레이블이 할당되어 그 만큼 폐기될 확률이 증가된다. 가중치는 에지 기능의 레이블 할당에만 영향을 미치게 되므로, 코어 기능은 기본 SLFQ 방식의 알고리즘의 수정 없이 가중치를 포함하는 흐름에 대해서도 공평성을 보장할 수 있다.

2. 네트워크 차원의 포괄적 대역 공평 할당 보장

SLFQ는 Core-Stateless 네트워크에서 링크 단위의 대역 할당의 공평성을 보장하기 위한 것이다. 즉 동일 링크 대역을 공유하는 각 흐름에 동일하게 링크 대역을 할당하고, 할당되지 않은 대역이 존재할 경우 더 많은 대역을 필요로 하는 흐름에 공평하게 할당한다. 이를 Max-Min 공평 분할이라고 한다. 예를 들어 하나의 10Mbps 링크를 각각 8Mbps, 6Mbps, 2Mbps 속도의 3개의 흐름이 공유할 때 식 (8)을 만족하는 최대의 공평 분할 속도 f 값을 구하는 것이다. .

$$\min(8,f)+\min(6,f)+\min(2,f)=10 \qquad (8)$$

이 예제에서 f 는 4가 되고 각 흐름은 4Mbps, 4Mbps, 2Mbps의 속도로 전송되는 것이 링크 단위의 Max-Min 대역 할당 방법이다.
이를 그림 14와 같은 네트워크를 통해 설명한다. 링크 용량을 1이라 하고 각 링크는 연결된 두 노드 번호로 지칭한다. 예를 들어 노드 1과 노드 2를 연결한 링크를 링크1-2로 표현한다. 각 흐름의 번호는 소스 및 착신지 번호로 표기한다. 즉 S1으로부터 D1으로 전송되는 흐름은 흐름 1이라 표현한다.
그림 14에 나타난 대역 할당 방법은 지금까지 논의된 링크 단위의 공평 대역 할당에 의한 것이다. 흐름 3은 링크 1-3에서는 흐름 1과의 경쟁에 의해 0.5로 할당되고, 링크 3-4에서는 경쟁에 의해 할당 대역이 0.5, 링크 4-6에서는 흐름 2와 흐름 4와의 경쟁에 의해 0.33이 할당되고 최종적으로 착신지에 0.33이 수신된다. 각 링크의 공평 분할 속도는 링크 단위로 독립적으로 할당된다.
그러나 네트워크 전체의 관점에서 볼 때 링크 단위의 공평 대역 할당이 항상 최적인 것은 아니다. 링크 1-3의 흐름 1에 할당된 대역 0.5는 전체 네트워크 관점에서는 최적의 할당이 되지 못한다. 링크 4-6의 폭주에 의해 흐름 3은 0.33의 대역이 할당되므로 이전 링크 1-3이나 링크 3-4에서의 0.5의 대역 할당은 의미가 없고, 따라서 흐름 3에 0.33의 대역이 할당 될 수 있다면 그림 15와 같이 흐름 1은 0.5가 아니라 0.67의 대역을 할당받을 수 있기 때문이다.
Network Border Patrol(NBP)[12] 메커니즘 등에서는 이러한 문제점을 지적하고 착신지까지 도달하지 못하고 중간에 손실되는 패킷에 의해 발생되는 네트워크의 폭주 및 불공평성을 해결하기 위해 네트워크에서 출력되는 속도 이상으로 입력되지 않도록 흐름 속도를 조절함에 의해 폭주 방지

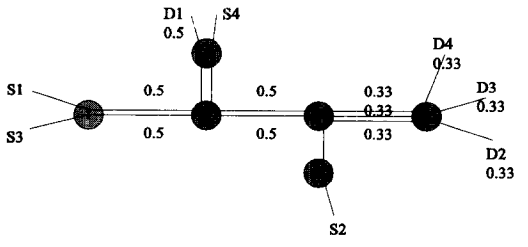


그림 14. 링크 단위의 공평 대역 할당
Figure 14. Allocation of Link Max-Min Fair-share

및 그에 따른 최소의 공평성은 보장한다. NBP 메커니즘에서는 순방향과 역방향 2개의 피드백 메시지와 입력 에지 라우터와 출력 에지 라우터의 기능을 정의한다. 출력 에지 라우터는 수신된 각 흐름의 속도를 측정하는 기능과 입력 에지 라우터로부터 순방향 피드백 메시지 수신시 각 흐름 속도를 실은 역방향 피드백 메시지를 생성하여 입력 에지 라우터로 전달하는 피드백 제어 기능을 수행한다. 입력 에지 라우터는 출력 에지 라우터로부터 수신된 역방향 피드백 메시지의 내용에 따라 각 흐름의 트래픽 속도를 조절하도록 제어하고, 주기적으로 순방향 피드백 메시지를 출력 에지 라우터로 전달하는 기능을 한다.

그러나 FIFO 큐잉을 사용하는 NBP 메커니즘은 코어 기능에서 공평 대역 할당을 강제할 방법이 없으므로 Core-Stateless 폭주 회피 방법일 뿐 공평성 보장을 위한 방법은 아니다. 이러한 피드백을 이용한 Core-Stateless 폭주 회피 방법에 SLFQ의 공평성 보장 방법을 연동하여 포괄적인 Max-Min 공평 대역 할당이 가능한 Core-Stateless 공평 대역 할당 방법을 구현할 수 있다.

Ⅵ. 결 론

본 논문에서는 흐름 단위의 제어나 상태 관리 없이 대역 할당의 공평성 보장에 접근하기 위한 알고리즘으로 SLFQ를 제안하였다. SLFQ는 네트워크의 에지 라우터에서 속도에 의해 계산된 계층을 표현하는 레이블을 각 패킷에 실어 주고, 코어 라우터 기능에서는 레이블 한계값을 관리하고, 이 레이블 한계값보다 큰 레이블을 가진 패킷을 폐기한다. 속도에 의해 계산된 레이블의 계층 구조로 인해 패킷의 폐기는 대역 할당 공평성에 접근하게 된다.

또한, 본 논문에서는 SLFQ를 CSFQ, DRR, FRED, RED와 함께 몇 가지 형태의 토폴로지를 통해 시뮬레이션하고 성능을 비교하였다. SLFQ는 대체로 대역 공평성을 보장하는 형태를 보여주고 있고 성능면에서 DRR, CSFQ보다 다소 떨어지는 면은 있지만 흐름 단위의 제어를 하는 FRED와 유사하거나 더 나은 성능을 보여주고 있다. 그러나 RED와 비교하면 공평성 보장 특성이 확연히 드러난다.

SLFQ는 알고리즘의 복잡성 측면에서 에지 라우터에서는 흐름 단위의 관리를 하는 FRED 정도의 복잡성을 가지고, 코어 라우터에서는 이보다 단순한 확률적인 폐기 알고리즘을 수행하는 RED와 같은 복잡성을 가지면서도 FRED 이상의 공평 대역 할당을 보장한다. 또한 Core-Stateless 메커니즘과 비교시에도 에지 라우터에서의 레이블의 계층화를 통해 코어 라우터에서 모든 트래픽에 대한 지수 평균 속도를 구하는 계산과 레이블 재할당 기능을 수행하는 CSFQ보다 코어 기능을 한층 더 단순화하였고, 같은 계층화된 레이블을 사용하는 RFQ보다는 에지 라우터에서 속도로부터 레이블을 할당하는 과정을 단순화, 구체화하여 실제 구현 가능성을 높였다.

향후 SLFQ 응용 연구 분야로는 에지 라우터에서 레이블 할당시 패킷의 우선 순위가 필요한 응용의 goodput 향상을 위해 단순한 랜덤 분포에 의한 계층 할당이 아니라 버킷을 통해 패킷에 우선 순위를 할당할 수 있는 구체적 방법 연구와 간단한 피드백 알고리즘과 연동하여 네트워크 차원의 Max-Min 공평 대역 할당을 위한 알고리즘을 구체화하고 성능 분석을 하는 것이 필요하다. 또한 현재 방식의 흐름 단위로 구분하여 레이블을 할당하는 방식을 통합된(aggregated) 흐름 단위로 구분하여 레이블을 할당하도록 확장함에 의해 트래픽 경로별 공평 대역 할당을 가능하게 하고, Differ-Serv 나 MPLS 네트워크에서 최선형 트래픽에 할당된 대역에 적용, 또는 측정 기반의 호 수락 제어 알고리즘[9]에서 측정 데이터 흐름 간의 대역 공평 할당에 이용하여 네트워크의 확장성을 유지하면서도 네트워크의 성능 향상을 도모할 수 있다.

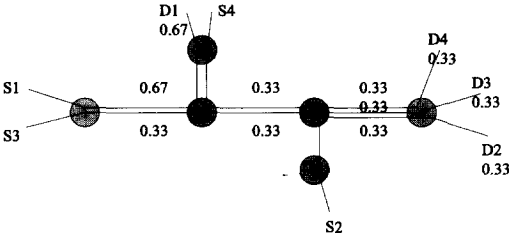


그림 15. 네트워크 단위의 공평 대역 할당
Figure 15. Allocation of globally Max-Min Fair-share

참 고 문 헌

[1] S.Floyd, "Promoting the Use of End-to-End Congestion in the Internet," *IEEE/ACM Tr. on Networking*, 7(4), pp.458-472, Aug. 1999.

[2] A.Demers, et al., "Analysis and Simulation of a Fair Queueing Algorithm," *Proceedings*

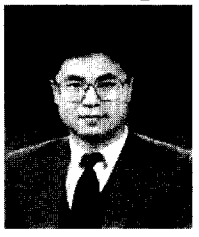
of ACM SIGCOMM'89, pp.3-12, Sep. 1989.
[3] Dong Lin, et al., "Dynamics of Random Early Detection," In Proceedings of ACM SIGCOMM'97, pp.127-137, Oct. 1997
[4] Ion Stoica, et al., " Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks," *Proceedings of ACM SIGCOMM'98*, pp.118-130, 1998.
[5] Ziruo Cao, et al., "Rainbow Fair Queueing: Fair Bandwidth Sharing without Per-Flow State," *Proceedings of IEEE INFOCOM 2000*, Vol. 2, pp. 922 -931, 2000.
[6] Zhai Mingyu, et al., " Fair bandwidth allocations through queue management in core-stateless networks," *Proceedings of IEEE GLOBECOM2001*, Vol. 4, pp.2248-2252, 2001.
[7] Takashi Kurimoto, et al., " Core-Stateless RED Algorithm for Improving Fairness in a Best-Effort Network," *IEICE Tr. Com*, E84-B(5), pp. 1413-1422, May 2001
[8] Ion Stoica, et al., "Providing Guaranteed Services Without Per Flow Management", *Proceedings of SIGCOMM'99*, pp.81-94, September 1999.
[9] V. Eleck, et al., "Admission Control Based on End-to-End Measurements," *Proceedings of IEEE INFOCOM2000*, pp.623-630, 2000.
[10] Woo-Seop Rhee, et al., "Two phase edge-to-edge distributed measurement based admission control mechanism in large IP networks," *GLOBECOM'01*, Vol.4, pp.2571-2575, 2001.
[11] Antoine Clerget, et al., "TUF: Tag-based Unified Fairness," *Proceedings of IEEE INFOCOM2001*, Vol.1, pp.498-507, 2001.
[12] Cellio Albuquerque, et al., "Network Border Patrol," *Proceedings of IEEE INFOCOM2000*, Vol.1, pp.322-331, 2000.
[13] M.Shreedhar, et al., "Efficient Fair Queueing using Deficit Round Robin," *Proceedings of SIGCOMM'95*, pp.231-243, Sep. 1995
[14] Sally Floyd, et al., "Random Early Detection for Congestion Avoidance," *IEEE/ACM Transaction on Networking*, 1(4), pp.397-413, July 1993
[15] UCB/LBNL/VINT, "Network Simulator - NS (Version 2)," http://www.isi.edu/nsnam/ns/
[16] Ion Stoica, "NS-2 Simulation for CSFQ," http://www.cs.cmu.edu/~istoica/csfq, 1998

김 화 숙(Hwa-Suk Kim) 정회원

1991년 2월 : 경북대학교 컴퓨터공학과 학사
2002년 8월 : 충남대학교 컴퓨터과학과 석사
1991년 1월~현재 한국전자통신연구원 선임연구원

<주관심분야> 유무선 통신, 인터넷, 광인터넷

김 상 하(Sang-Ha Kim) 정회원



1980년 : 서울대학교 화학과 학사
1984년 : U. of Houston 화학과 석사
1989년 : U. of Houston 전산학과 박사
1989년 : HNSX Supercomputers Inc. 자문위원
1990~1991 시스템공학 연구소 선임연구원
1992 ~ 현재 충남대학교 컴퓨터과학과 교수

<주관심분야> 컴퓨터 네트워크, 이동통신, 무선 인터넷

김 영 부(Young-Boo Kim) 정회원

1982년 : 한양대학교 전기공학과 학사
1984년 : 한양대학교 전기공학과 석사
1984년~현재 : 한국전자통신연구원 책임연구원, 광네트워크구조팀장

<관심분야> 광인터넷, 네트워크 구조 및 망 계획