DiffServ 망에서 QoS를 보장하기 위한 동적 가중치 할당 알고리즘

A dynamic WRR Algorithm for QoS Guarantee in DiffServ Networks

정동수, 김변곤, 조해성*, 정경택, 김남희, 이종인
군산대학교, 건양대학교*

Chung Dong-Su, Kim Byun-Gon, Cho Hae-Seong*,
Chung Kyung-Taek, Kim Nam-Hee*, Lee Jong-In
Kunsan National University, Konyang University*

요약

DiffServ망에서 많이 거론되고 있는 대표적인 스케줄러로 PQ(Priority Queue), WRR(Weighted Round Robin)등의 스케줄러가 연구되어져 있다. 그러나 이러한 스케쥴링 방식들은 약간의 단점을 가지고 있다. 본 논문에서는 PQ와 WRR의 단점을 보완하여 WRR 스케쥴러에 적용이 가능한 스케줄러 알고리즘을 제안한다. 본 논문에서 제안된 알고리즘은 DiffServ 망에서 각 클래스의 큐 상태를 체크하여 퍼지 이론을 적용한 제어 정책에 따라 WRR 스케쥴러의 가중치를 동적으로 할당하였다. 제안된 알고리즘의 성능평가를 위하여 네트워크 시뮬레이터(NS-2)를 이용하여 컴퓨터 시뮬레이션을 수행하였다. 시뮬레이션 결과 제안된 알고리즘은 EF 클래스의 패킷 손실률에서 WRR 스케쥴러 방식보다 향상되었으며, AF4 클래스에서는 PQ 방식보다 향상된 결과를 보였다.

Abstract

There are two traditional scheduling methods known as PQ and WRR in the DiffServ network, however, these two scheduling methods have some drawbacks. In this paper, we propose an algorithm that can be adopted in WRR scheduler with making up for weak points of PQ and WRR. The proposed algorithm produces the control discipline by the fuzzy theory to dynamically assigns the weight of WRR scheduler with checking the Queue status of each class. To evaluate the performance of the proposed algorithm, We accomplished a computer simulation using NS-2. In result, the proposed algorithm enhances the packet discard rate at the EF class than WRR scheduling method and the AF4 class than PQ scheduling method.

I. 서 론

QoS에 관련된 많은 연구 중에서 모든 패킷 흐름에 대해서 QoS를 제공해주기 위한 IntServ 모델은 현실적으로 구현이 복잡하고 오버헤드가 많기 때문에 확장성이 나쁜 단점이 있다[2]. 이러한 단점을 극복하기 위해 IETF(Internet Engineering Task Force)에서는 각기 다른 사용자 그룹에게 서로 다른 수준의 서비스를 제공할 수 있는 DiffServ 모델을 제안하게 되었다[1].

DiffServ는 복잡한 기능의 처리는 경계(edge) 라우터에서 수행하게 하고 코어(core) 라우터는 각 클래스의 PHPs(Per-Hop Behaviors)에 따라 패킷을 전달함으로써 확장성을 향상시켰다[2]. DiffServ에서 제공하는 서비스는 일반적으로 EF(Expedited Forwarding), AF(Assured Forwarding), DE(Default) 클래스로 나누어 진다. EF 클래스는 가장 우선순위가 높은 클래스로서 인터넷 전화, 화상회의 등에 알맞은 작은 패킷 지연시간과 지터, 작은 패킷 폐기율을 제공받는다. AF 클래스는 지연시간과 지터에 민감하지 않고 최선형 서비스보다는 좋은 서비스를 제공받으며, DE 클래스는 인터넷 최선형 서비스와 같은 수준의 서비스를 제공받을 수 있다.

PQ는 클래스의 우선순위에 따라서 서비스해주는 방식이므로 EF 클래스에 높은 수준의 QoS를 제공할 수 있으나 EF 클래스의 트래픽이 증가하면 우선순위가 낮은 클래스들의 QoS를 보장해 주지 못한다는 단점이 있다. 또한, WRR은 각 클래스의 가중치에 따를 서비스를 제공하기 때문에 각 클래스가 공평하게 서비스를 받을 수 있는 장점이 있지만, EF 클래스의 트래픽이 집중되는 경우에 EF 클래스의 QoS를 보장해 줄 수 없는 단점을 가지고 있다[3].

따라서, 본 논문에서는 PQ와 WRR의 단점을 보완할 수 있는 스케줄링 기법을 제안한다. 본 논문에서 제안된 알고리즘은 유동적으로 변화하는 각 클래스의 큐 상태를 체크하여 큐의 상태에 따라 각 클래스 큐의 가중치를 동적으로 할당함으로써 보다 효율적인 서비스가 가능하다. 유동적으로 각 클래스의 큐 상태를 체크하기위하여 퍼지이론을 적용하였으며, 퍼지이론을 통하여 퍼지 제어 규칙을 생성하여 클래스가 가지고 있는 큐의 가중치를 효율적으로 할당하도록 하였다.

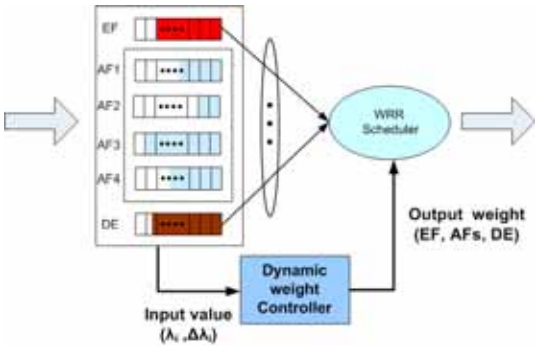
본 논문의 구성은 제 2장에서 제안된 스케줄링 알고리즘을 설명한다. 제 3장에서는 ns-2 시뮬레이터를 이용하여 기존 연구 및 제안된 알고리즘의 성능을 비교 분석한다. 마지막으로

제 4장에서 결론을 맺는다.

II. 제안된 알고리즘

그림 2.1은 제안된 스케줄러의 구조를 나타내고 있다.

동적 가중치 제어기는 각 클래스 큐의 셀 수(λ)와 변화량(Δλ)이 입력되면 퍼지 이론을 이용하여 큐의 상태정보 값을 계산하고 클래스의 우선순위와 상태정보에 따라서 각 클래스 큐의 가중치를 제어한다.



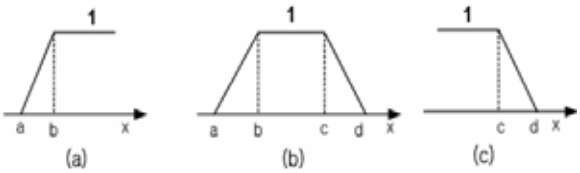
▶▶ 그림 2.1 제안된 스케줄러 구조
Fig. 2.1 Structure of proposed scheduler.

1. 퍼지 제어 규칙

각 클래스 큐의 가중치를 동적으로 할당하기 위하여 각 클래스 큐의 상태에 따라 두 개의 입력과 한 개의 출력을 갖는 퍼지 제어 규칙을 생성한다. 퍼지 제어 규칙 생성을 위한 두 개의 입력은 각 클래스 큐의 현재 셀 수(qLen)와 셀 수의 변화량(dq)이다. 출력 값은 각 클래스 큐의 상태에 따라 -1.0~1.0 사이의 큐 상태(QSt) 값으로써, 이러한 큐 상태 값을 임계치(threshold)와 비교하여 각 클래스 큐의 상태에 따른 제어를 수행하게 된다.

제안된 퍼지 제어 규칙을 생성하기 위하여 그림 2.2와 같은 사다리꼴 맴버쉽 함수를 수식 1과 같이 정의한다.

$$f(x;a,b,c,d)=\begin{cases}0 & \text{for } x \le a,\\ \dfrac{x-a}{b-a} & \text{for } a \le x \le b,\\ 1 & \text{for } b \le x \le c,\\ \dfrac{d-x}{d-c} & \text{for } c \le x \le d,\\ 0 & \text{for } x \ge d.\end{cases} \quad (1)$$



▶▶ 그림 2.2 함수 f 정의 (a) $f(x;a,b,+\infty,+\infty)$, (b) $f(x;a,b,c,d)$, (c) $f(x;-\infty,-\infty,c,d)$
Fig. 2.2 Definition of function f : (a) $f(x;a,b,+\infty,+\infty)$, (b) $f(x;a,b,c,d)$, (c) $f(x;-\infty,-\infty,c,d)$

퍼지 입력 변수로서 각 클래스 큐의 현재 셀 수와 변화량을 가지고 $|T(\text{qLen})|\times|T(\text{dq})|$와 같은 2차원의 배열이 형성된다. $|T(X)|$는 $T(X)$의 언어변수 항들의 수이다. 이와 같이 두 가지 변화량(qLen, dq)을 가지고 퍼지 입력 용어 집합을 다음과 같이 정의 하고,

$T(\text{qLen}) = \{\text{GL, L, MD, H, GH}\}$

$T(\text{dq}) = \{\text{Decrease, Maintain, Increase}\}$

퍼지 출력 용어 집합으로 큐 상태(QSt)값의 용어 집합을 다음과 같이 정의할 수 있다.

$$T(\text{QSt}) = \{\text{NB, NM, ZO, PM, PB}\}$$

퍼지 입출력 맴법쉽 함수를 다음과 같이 정의한다.

$$\mu_{\text{GL}}(\text{qLen}) = f(\text{qLen};\ 0,\ 0,\ 0,\ 0.2\text{B})$$
$$\mu_{\text{L}}(\text{qLen}) = f(\text{qLen};\ 0,\ 0.2\text{B},\ 0.2\text{B},\ 0.4\text{B})$$
$$\mu_{\text{MD}}(\text{qLen}) = f(\text{qLen};\ 0.2\text{B},\ 0.4\text{B},\ 0.4\text{B},\ 0.6\text{B})$$
$$\mu_{\text{H}}(\text{qLen}) = f(\text{qLen};\ 0.4\text{B},\ 0.6\text{B},\ 0.6\text{B},\ 0.8\text{B})$$
$$\mu_{\text{GH}}(\text{qLen}) = f(\text{qLen};\ 0.6\text{B},\ 0.8\text{B},\ 1\text{B},\ 1\text{B})$$
$$\mu_{\text{D}}(\text{dq}) = f(\text{dq};\ -\infty,\ -\infty,\ -5,\ 0)$$
$$\mu_{\text{M}}(\text{dq}) = f(\text{dq};\ -5,\ 0,\ 0,\ 5)$$
$$\mu_{\text{I}}(\text{dq}) = f(\text{dq};\ 0,\ 5,\ +\infty,\ +\infty)$$
$$\mu_{\text{NB}}(\text{QSt}) = f(\text{QSt};\ -1,\ -1,\ -1,\ -0.5)$$
$$\mu_{\text{NM}}(\text{QSt}) = f(\text{QSt};\ -1,\ -0.5,\ -0.5,\ 0)$$
$$\mu_{\text{ZO}}(\text{QSt}) = f(\text{QSt};\ -0.5,\ 0,\ 0,\ 0.5)$$
$$\mu_{\text{PM}}(\text{QSt}) = f(\text{QSt};\ 0,\ 0.5,\ 0.5,\ 1.0)$$
$$\mu_{\text{PB}}(\text{QSt}) = f(\text{QSt};\ 0.5,\ 1.0,\ 1.0,\ 1.0)$$

위에서 규정한 용어 집합을 기반으로 표 2.1과 같은 퍼지 제어규칙을 만들고, 비퍼지화 단계를 거치면 각 클래스 큐의 상태정보를 알 수 있다.

[표 2.1] 퍼지 제어 규칙
Table 2.1 Fuzzy control rule.

	셀 수의 변화량(dq)	현재 셀 수(qLen)	요구 가중치(QSt)
1	D	GL	NB
2	D	L	NM
3	D	MD	Z0
4	D	H	Z0
5	D	GH	PM
6	M	GL	NB
7	M	L	NM
8	M	MD	Z0
9	M	H	PM
10	M	GH	PB
11	I	GL	NM
12	I	L	Z0
13	I	MD	PM
14	I	H	PB
15	I	GH	PB

2. 동적 가중치 할당 알고리즘

제안된 스케줄러 알고리즘은 우선순위가 가장 높은 EF 클래스를 기준으로 한다. EF 클래스의 큐 상태 정보 값이 임계치(thHigh) 보다 크다면 폭주가 발생하여 패킷 손실이 발생할 수 있으므로 AF 클래스 내에서 남는 가중치가 있는지 찾는다. 만약에 AF 클래스 큐의 상태 정보 값이 임계치(thLow) 보다 작다면 AF 클래스의 가중치를 감소시키고, EF 클래스의 가중치를 증가 시켜준다. 만약에 AF 클래스에 남는 자원이 없다면 DE 클래스의 가중치를 감소시켜 EF 클래스의 가중치를 증가시켜 준다. 이러한 방식으로 각 클래스 큐의 상태정보에 따라서 대역에 여유가 있는 AF 클래스나 우선순위가 낮은 DE 클래스의 가중치를 빌려옴으로써 EF 클래스의 폭주를 완화시킬 수 있고, 폭주가 완화되면 가중치를 반환하게 된다. EF 클래스 가중치의 반환과정은 AF 클래스에서 빌려온 가중치가 있다면 AF 클래스에 반환하고 AF 클래스에서 빌려온 가중치가 없다면 DE 클래스에 반환하게 된다. 그림 2.4는 제안된 알고리즘의 흐름도 이다.

III. 시뮬레이션 및 성능평가

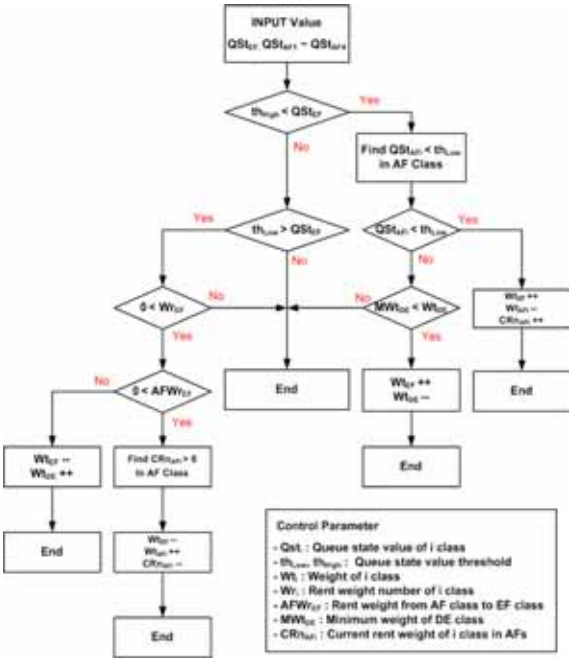


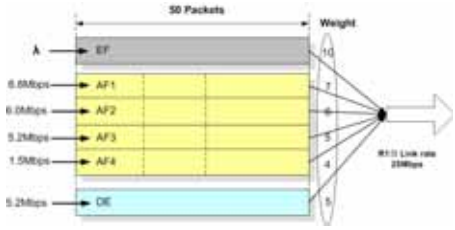
그림 2.4 제안된 알고리즘 gm름도
Fig. 2.4 Flowchart of proposed algorithm,

1. 시뮬레이션 환경

본 논문에서 제안된 알고리즘의 시뮬레이션은 NS-2의 DiffServ 노드를 기반으로 그림 3.1과 같은 망을 구성해서 시뮬레이션 하였다. 그림 3.1에서 Source 1~Source 6은 트래픽을 발생시키는 노드로써 트래픽은 포아송 과정으로 발생되며, 입력 트래픽 량은 그림 3.2와 같다.

R2~R3 사이의 링크에 bottleneck를 두고, 모의 실험을 하였으며 모든 라우터에서 큐를 관리하기 위한 방법으로는 EF 클래스에서는 Drop Tail, AF 클래스에서는 RED, DE 클래스에서는 Drop Tail 방법을 사용하였으며 시뮬레이션은 100초 동안 수행하였다.

▶▶ 그림 3.1 시뮬레이션 네트워크 모델
Fig. 3.1 Network simulation model.



▶▶ 그림 3.2 라우터 내부 큐 상태
Fig. 3.2 Inner queue state of router.

2. 시뮬레이션 결과 및 분석

시뮬레이션 결과는 DiffServ에서 많이 사용되어지고 있는 방식인 PQ 방식과 WRR 방식, 그리고 제안된 방식을 비교분석하여 나타낸다.

그림 3.3~3.6은 시뮬레이션 결과를 보이고 있다. 제안된 알고리즘은 WRR을 기반으로 가중치를 동적으로 제어하여 WRR의 단점인 EF 클래스의 손실률을 줄일 수 있고(그림3.3), PQ의 단점인 AF4 클래스의 손실률을 줄일 수 있었다(그림 3.5). DE 클래스는 최소 보장 서비스만을 하고 있기 때문에 효과가 조금은 감소하였다(그림3.6). 그 이유는 EF 클래스와 AF 클래스의 효과가 증가한 만큼 조금의 감소를 보이게 된 것이다. 하지만 제안된 알고리즘은 EF와 AF 클래스의 QoS를 보장해주면서도 DE 클래스도 PQ 방식과 비교하면 많은 성능 향상을 보이고 있다. 또한 AF1~AF3 클래스의 결과는 그림 3.4와 유사한 결과를 보이고 있는데, 이는 이들 클래스의 가중치는 영향을 받지 않기 때문이다.

따라서, 우선순위가 높은 클래스의 패킷 폐기율이 낮으면서 우선순위가 낮은 클래스에서도 성능이 향상이 되었음을 알 수 있다. 따라서 남는 가중치를 효율적으로 분배하여 사용하고 있음을 알 수 있다.

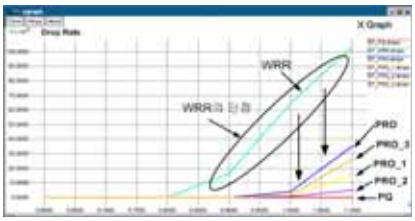
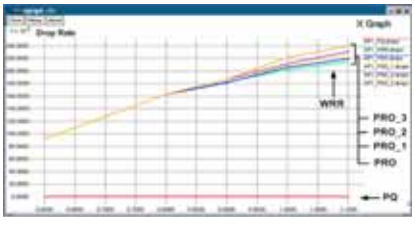


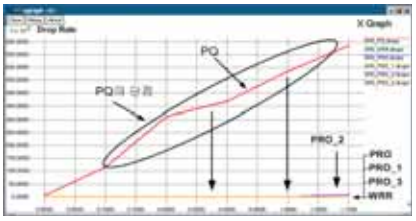
그림 3.3 EF 클래스의 패킷 폐기확률
Fig. 3.3 Packet drop rate of EF class.



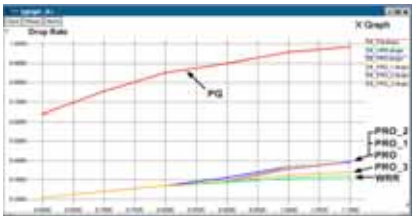
▶▶ 그림 3.4 AF1 클래스의 패킷 폐기확률
Fig. 3.4 Packet drop rate of AF1 class.

IV. 결 론

본 논문에서는 차별화된 서비스를 제공하기 위한 DiffServ의 스케줄러로서 다양한 스케줄러들이 있지만 그 중에서 PQ(Priority Queue)와 WRR(Weighted Round Robin)이 사용되어질 때의 단점들을 보완하고자 각 클래스에 입력되는 큐의 상태에 따라 각 클래스의 가중치를 공평하게 할당하는 방법을 제안하였다.



▶▶ 그림 3.5 AF4 클래스의 패킷 폐기확률
Fig. 3.5 Packet drop rate of AF4 class.



▶▶ 그림 3.6 DE 클래스의 패킷 폐기 확률
Fig. 3.6 Packet drop rate of DE class.

본 논문에서는 WRR기반의 스케줄러에 동적 가중치 할당 알고리즘을 적용시킴으로써 DiffServ 구조에서 가장 우선순위가 높은 EF 클래스의 QoS를 보장해 주기 위하여 AF 클래스 중에서 남는 영역의 가중치를 우선 할당하고 남는 영역이 없는 경우에 DE 클래스의 가중치를 할당하여 EF 클래스의 QoS를 제공하였다.

현재 사용이 되고 있는 스케줄러 방식인 PQ 보다는 DE 클래스에서 패킷 폐기율은 대략 55% 정도의 성능 향상을 보였다. 그리고 기존 WRR 방식 보다는 EF 클래스에서 패킷 폐기율이 평균 6.5% 정도의 성능 향상을 보였다

▌참 고 문 헌▐

[1] S. Blake, D. Black, M. Carlson, Wang and Weiss, "An Architecture for Differentiated Services," RFC 2475, Dec. 1998.
[2] K. Nichols, S. Blake, F. Baker, D. Black, "Definition of the Differentiated Services Field(DS Field) in the IPv4 and IPv6 Headers," RFC 2474, Dec. 1998.
[3] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, "Assured Forwarding PHB," RFC 2597, June 1999.
[4] V. Jacobson, K. Nichols, Cisco Systems, K. Poduri, Bay Networks, "An Expedited Forwarding PHB," RFC 2598, June 1999.
[5] G. Cheng, K. Ku, Y. Tian, and N. Ansari, "Core- Stateless Proportional Fair Queueing for AF Traffic," Proc. of the IEEE GLOBECOM, December 2004.