차등화 서비스를 위한 리눅스 커널 스케줄러 설계 및 구현

김다현O, 송진석, 이민석
한성대학교 컴퓨터공학과
coremakerO@gmail.com, kkopuk@gmail.com, minsuk@hansung.ac.kr

Design and Implementation of Linux Kernel Scheduler For Differentiated Services

Dahyun KimO, Jinsuck Song, Minsuk Lee
Department of Computer Engineering, Hansung University, South Korea

요 약

IT와 네트워크의 발전에 힘입어 많은 서비스를 인터넷을 통해서 제공하고 있다. 인터넷 사용자가 급증하면서 수많은 서비스 요청에 대한 효과적인 처리 방법에 많은 관심이 집중되고 있다. 본 논문에서는 라우터에서 패킷을 대상으로 사용된 Percentile 스케줄 기법을 리눅스 커널 스케줄러에 적용하여, 다양한 서비스 요청에 대해서 적절한 기준에 따라 다른 등급으로 분류하고 차등적인 서비스를 제공하고자 하는 연구를 진행하였다. 논문에서는 리눅스의 기본 스케줄러에 *nice()*시스템 콜을 이용하여 차등화 스케줄을 적용한 것과 Percentile 스케줄 기법을 리눅스 커널 스케줄러에 적용한 것을 비교 실험하였다. Percentile 스케줄 기법을 적용한 경우 기존의 리눅스 커널에 비해서 등급에 따른 응답 시간의 차이가 명확한 것을 관찰할 수 있었다.

1. 서론

서비스라는 분야는 시대가 흘러가면서, 더욱 다양하고 복잡한 형태로 발전하였다. 특히 IT와 네트워크의 발전은 수많은 서비스가 인터넷을 통해서 제공되도록 하였고, 이로 인해 인터넷을 이용하여 제공되는 서비스의 품질을 향상시키기 위한 노력들이 진행되고 있다.

1.1 연구 동기

사용자는 서비스 형태에 상관없이 타인에 비해서 더 나은 서비스를 받고자 하는 것이 공통적인 욕구이다. 이러한 욕구를 충족시켜주고 사용자의 수요 경쟁을 유발시킬 수 있다면 보다 많은 이윤을 창출할 수 있을 것이다. 기업에서에서도 웹 서비스에 이러한 방법을 적용하고자 하며, 기존의 정책이 아닌 새로운 정책이 적용될 때 보다 많은 이익이 발생할지에 대한 관심이 발생되고 있는 시점이다. 기존 웹 서비스에서 사용하는 운영체제 스케줄러의 정책은 '공평성'에 상당한 비중을 두고 있다[1]. 모든 태스크에게 공평한 실행 시간 할당량과 실행 순서를 부여한다. 하지만 본 연구에서는 고객에 따라 명백히 품질의 차이가 발생하는 서비스를 위해 의도적으로 태스크의 우선순위를 제어한다.

1.2 연구 목표

고객에 따라 등급을 부여하며, 부여된 등급으로 서비스의 품질에 차이가 나도록 한다. 여기서 이야기하는 서비스 품질의 대상으로 본 논문은 응답 시간을 선택하였으며, 등급의 차이는 곧 응답 시간의 차이라고 하겠다. 컴퓨터 외부의 요인들을 제외하고, 응답 시간에 영향을 주는 것으로는 네트워크 패킷 흐름, 응용 서버 내에서의 요청에 대한 내부 프로세스 스케줄, 그리고 마지막으로 운영체제의 태스크 스케줄러가 있으며, 본 논문에서는 운영체제내의 태스크 스케줄러를 변경하여, 서비스를 고객의 등급에 맞게 차등화 된 응답 시간을 제공할 수 있도록 설계/구현하기로 한다.

2. 관련 연구

차등화 스케줄 기법을 위해서는 운영체제 및 기법을 선택해야한다. 본 논문에서는 리눅스 환경 하에서 Percentile 기법을 이용한다. 우선 리눅스 내의 스케줄에 관련된 연구 내용과 Percentile 기법에 대해서 알아본다.

2.1 리눅스 스케줄러

본 연구에서 사용한 리눅스 커널은 버전이 2.6.16이며, 스케줄 알고리즘으로 O(1) 기법을 사용하였다. 리눅스의 기본 스케줄 정책은 시분할 기법을 토대로 한다. 시분할 기법은 CPU 시간을 슬라이스로 쪼개고, 실행 가능한 각 프로세스마다 슬라이스를 하나씩 할당하여 프로세스 여러 개를 시간 다중화 방식으

로 실행한다. 우선순위에 따라 순위를 매기는 작업 또한 중요한 스케줄 기법 중 하나이다. 태스크는 각 역할에 따라 크게 실시간 프로세스와 일반 프로세스로 나누어진다. 본 논문에서의 연구 대상은 일괄적으로 처리되어야 하는 일반 프로세스이며, 리눅스에서 일반 프로세스의 우선순위를 제어하는 시스템 콜은 *nice()*를 사용한다. *nice()* 시스템 콜이 사용하는 단위는 나이스 값이다. 다음 [표 1]은 나이스 값에 따라 태스크에 주어진 일련의 정책이다.

Nice Value	Time Quantum
-20	800 ms
-10	600 ms
0	100 ms
+10	50 ms
+19	5 ms

[표 1] nice 값에 따른 시간 할당량

나이스 값은 최저 -20부터 최대 +19까지 총 40개로 구성되어 있으며, 각 값에는 시간 할당량이 결정되어 있다. 나이스 값이 낮을수록 해당 프로세스는 높은 우선순위로 실행된다.

리눅스 커널에서 컴파일 옵션으로 선점형 스케줄을 선택할 수 있다. 선점형 스케줄에서는 프로세스의 우선순위의 중요도에 따라서, 현재 작동하고 있는 태스크와 교환이 이루어질 수 있기 때문에 프로세스는 언제라도 자신이 가지고 있는 시간 할당량을 다 소비하지 못하더라도 대기 상태에 머물 수 있다. 본 논문에서는 Percentile 스케줄링 시 외부 프로세스 에 의해 간섭받는 것을 막기 위해서 선점을 할 수 없도록 하였다[2].

2.2 Percentile 스케줄 기법

Percentile 스케줄 기법은 패킷 별 우선순위를 기반으로 하여 네트워크 라우터에 사용된 스케줄링 기법이다. 이 기법은 라우터에 도착하는 패킷들을 서비스 품질에 따른 등급으로 구분하고, 각 등급에 최대 지연 시간 내에 전송되는 비율이 주어진 기준 값을 만족시키기 위해 사용하는 스케줄링 방법이다. 앞에서 말하고 있는 정해진 지연 시간을 바로 마감 시간이라 한다. 마감 시간은 각 패킷이 처리 완료 시간을 측정하여 평균값으로 정한다. Percentile 스케줄에서 등급 i에 속하는 패킷의 스케줄 우선순위는 다음과 같다.

$$F(i) = \frac{Si}{Ni \times P} \quad \cdots \text{[수식 1]}$$

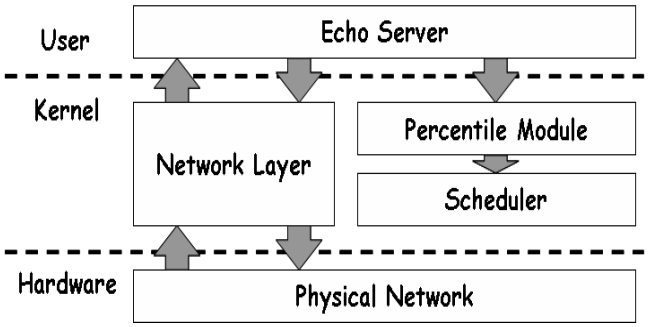
[수식 1] 공식에서 등급 i에 대해 Si는 마감 시간 안에 처리하여 성공한 패킷의 개수이며 Ni는 현재까지 누적된 등급 i로 분류된 패킷의 개수이다. P는 주어진 Percentile로서 비율과 비교될 기준 값을 뜻하며, 이는 마감 시간 안에 요구되는 처리 비율을 사용자가 임의로 정의한 것이다. *F(i)* 값이 낮을수록 높은 우선순위를 가지게 된다. 마감시간 안에 처리된 패킷의 수가 Percentile에 비하여 상대적으로 적은 등급의 패킷이 우선하여 처리된다[3].

3. 차등화 서비스를 위한 리눅스 스케줄링 연구

차등화 서비스 연구를 위해 Percentile 스케줄 기법을 실제 태스크 스케줄러에 적용한 형태와 방법, 실험 환경에 대해서 알아보고, 마지막으로 실험 결과에 대한 분석을 하도록 하겠다.

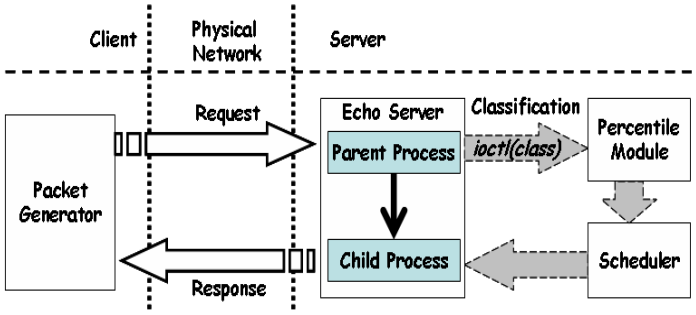
3.1 Percentile Scheduler의 적용

Percentile 스케줄 대상이 되는 클라이언트는 서비스의 종류에 따라 적절한 기준으로 등급화 되어있으며 해당 등급의 서비스를 요청한다. 스케줄러에서는 해당 등급에 따라 처리순서를 제어한다. 그 중 가장 핵심이 되는 프로세스는 바로 PM(Percentile Module)이다. PM은 Percentile의 대상 태스크들을 관리하는 모듈이다. [그림 1]은 PM을 포함한 Percentile 시스템에 대한 계층도이다[4].



[그림 1] Percentile System 계층도

Percentile 기법에 의해 부여된 우선순위는 기존의 리눅스 시스템에서 사용하는 우선순위와는 독립된 우선순위 기준을 사용한다. PM은 클라이언트의 요청을 처리하기위한 태스크의 등급 정보를 *ioctl()*을 통하여 서버로부터 전달받는다. 전달받는 정보는 해당 태스크의 등급과 태스크 디스크립터의 포인터 주소이다. PM은 *add_wait_queue_exclusive()* 함수를 이용하여 태스크를 해당 등급 큐에 삽입하고 대기 상태로 만들어준다. 수정된 커널 스케줄러가 태스크 요청 시 PM은 가장 우선순위가 높은 등급을 찾아내고 해당되는 등급에 있는 태스크 중에서 가장 먼저 들어온 태스크를 반환하게 된다. [그림 2]는 요청 패킷의 수신부터 응답 패킷의 전송까지의 흐름을 보여주고 있다.



[그림 2] Percentile System 흐름도

[그림 2]에서 물리적인 네트워크 부분에서는 패킷이 이동하고, 서버에서는 태스크 형태로 처리된다. 패킷 생성기가 있는데, 클라이언트에서 서비스를 요청하고, 그 응답을 받아 시간을 측정하기 위해 제작하였다.

본 논문에서는 리눅스 커널 스케줄러를 수정하여 차등화 서비스를 연구하였다. 앞으로 Percentile 스케줄러의 대상이 되는 프로세스를 Percentile 태스크라고 하겠다. Percentile 태스크는 Percentile 시스템 내부에서 PM이 관리하는 큐에 삽입되었다가 스케줄러 요청에 의해서 하나씩 실행된다. Percentile 태스크는 크게 PM에 삽입되는 과정과 실제로 스케줄러 요청에 의해서 실행되는 과정, 두 가지 과정으로 나누어진다. 따라서 리눅스 커널 스케줄러는 '시스템 시간'과 'Percentile 시간'으로 스케줄 과정을 나누고, 시스템 시간에는 Percentile 스케줄 대상 태스크를 네트워크 계층에서 PM으로 전달하는 과정이 이루어지고, Percentile 시간에는 Percentile 태스크가 스케줄링 되고 나서 실제로 실행되는 과정이 이루어진다. Percentile 스케줄러는 일반적으로 Percentile 시간에 새로운 Percentile 태스크를 PM으로부터 반환받아 실행한다. 하지만 Percentile 태스크가 soft_irq같은 실시간 태스크에 의해 선점당하여 완전히 처리되기 전에 실행 큐로 다시 삽입 된 경우, 다른 Percentile 태스크보다 먼저 처리하도록 하였다. 또한 Percentile 시간에도 실시간 태스크임을 나타내는 정적 우선순위가 100이하인 가진 태스크에 대해서도 즉시 처리하도록 하였다.

리눅스 커널에 Percentile 스케줄 기법을 적용하기 위해서는 2.3.1장에서 사용한 수식의 재정의가 필요하였다. 각 수식의 값을 변경하는 기준이 되었던 마감시간을 이용하지 않았다. 프로세스가 실행되어야하는 최대 지연시간 개념이 없이 임의로 프로세스의 실행시간이 완료되었을 때, 값들의 변경이 일어나도록 하였다. Ni의 경우, 전체 프로세스가 Echo Server를 통해 *fork()*하여 자식 태스크가 생성된 수만큼 증가하며, Si의 경우, 요청 태스크가 성공적으로 처리되고 스케줄러에 의해서 종료되는 순간 증가하게 된다. P는 [표 2]와 같이 사용자가 임의로 결정하였다.

등급	class1	class2	class3	class4	class5
P value	90	80	70	60	50

[표 2] 등급에 따른 P Value

P값의 의미는 전체 처리 시간 중 해당 등급의 태스크가 차지하고 있는 비중에 대한 기대치이다. 예를 들어 class1의 경우 전체 처리 시간에 대해서 해당 클래스의 실행 점유율이 90% 이상이 되어야 함을 의미한다.

3.2 실험 환경

실험은 커널 2.6.16 리눅스가 실행되는 [표 3]와 같은 사양을 가진 서버 PC 1대와 서비스 요청을 위한 클라이언트 PC 5대를 100Mbps 스위치 기반 이더넷으로 구성하였다. 한 가지 주목해야할 점은 현재 서버 PC의 성능이 클라이언트에 비해 상대적으로 낮다는 것이다. 이유는 연구 동기에서도 언급했듯이, '한정된' 자원 하에서의 차등화 효과에 대해서 알아보는 것이 중요한 목적 중 하나이기 때문에 서버의 자원을 최대한 적게 함으로써, 보다 명확한 실험결과를 얻고자 함이다.

분류	등급	CPU	Memory	OS
서버	-	700Mhz	256MB	Linux Kernel 2.6.16
Client1	Class1	2.8Ghz	1GB	
Client2	Class2	1.7Ghz	512MB	
Client3	Class3	1.5Ghz	512MB	
Client4	Class4	2.4Ghz	256MB	
Client5	Class6	2.8Ghz	512MB	

[표 3] 실험 PC 사양

3.3 실험 방법d

클라이언트로부터 패킷을 전송받아 진행하는 실험이므로, 간단하게 패킷을 생성하고 시간을 측정하는 패킷 생성기를 제작하였다. 매 실험에서 각 클라이언트는 패킷 생성기를 통해서 10개의 쓰레드를 이용하며 각 쓰레드는 1,000개의 패킷을 생성하여, 총 10,000개의 패킷을 전달하게 된다. 실험에서는 5개의 클라이언트가 사용되었기 때문에, 서버는 1회 실험에 총 50,000개의 패킷을 처리하는 것이 된다. 각 패킷에 대한 응답 시간을 각각의 클라이언트를 기준으로 측정하였다.

응답 시간은 각각의 클라이언트에서 패킷을 보낸 시점에서 보낸 패킷 도착시점까지를 펜티엄의 TSC(Time Stamp Counter)를 사용하여 측정하였다[5]. [그림 3]는 본 실험에 사용된 패킷 생성기에 대한 Pseudo Code 이다.

```
generated_packet()
{
        소켓 생성;
        서버 접속;

        while(정해진 패킷 수만큼 반복) {
                시작 시간 측정;

                요청 패킷 송신;
                응답 패킷 수신;

                종료 시간 측정;

                실제 응답 계산 및 저장;
        }
}
```

[그림 3] 패킷 생성기 Pseudo Code

실험의 비교대상으로, 기존의 리눅스 스케줄러에 등급별로 *nice()*를 이용하여 우선순위를 차등적으로 계산하고, 이를 적용하여 실험하였다.

3.4 실험 결과

*nice()*를 이용한 차등화 스케줄 방식과 Percentile 스케줄 방식에 대한 각각의 등급별 클라이언트 응답 시간을 비교/분석하였다.

3.4.1 *nice()*을 이용한 우선순위 조정

기존 리눅스 커널 스케줄러 대상으로 일반 태스크에 대한 나이스 값을 조절할 수 있는 *nice()* 시스템 콜을 이용하여 차등화 서비스를 실험하였다. 실험은 10회에 걸쳐 진행되었다. 리눅스 커널 스케줄러는 접속한 클라이언트에 등급을 부여한 후 *nice()* 시스템 콜을 사용하여 우선순위를 [표 4]와 같이 적용하였다.

등급	Class1	Class2	Class3	Class4	Class5
정적 우선순위	-20	-19	-18	-17	-16

[표 4] 등급별 정적 우선순위

평균 응답 시간을 측정한 결과 약간의 차등적인 응답 시간을 보여주고 있으나 만족할만한 결과를 보여주지는 못하고 있다. [표 5]은 각 클라이언트마다 측정된 평균 응답 시간이다.

등급	평균 응답 시간(sec)
Class1	0.003780
Class2	0.005004
Class3	0.005791
Class4	0.006061
Class5	0.007724
전체평균	0.005672

[표 5] *nice()* 이용 차등화 스케줄러 등급별 평균 응답 시간

3.4.2 Percentile 스케줄러

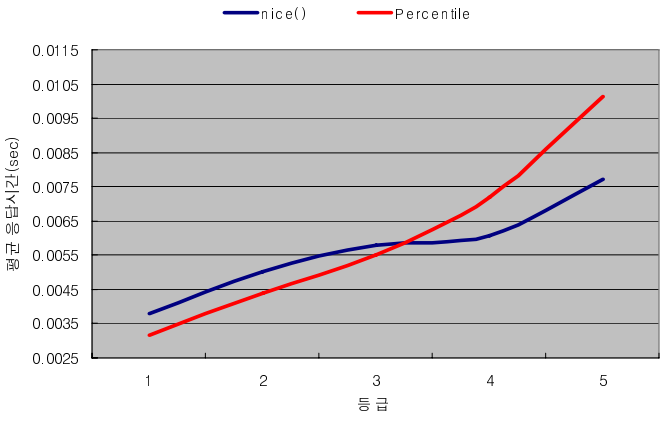
[표 6]은 PM과 수정한 커널 2.6.16 스케줄러를 이용한 시험에서의 평균 응답 시간을 측정한 결과이다.

등급	평균 응답 시간(sec)
Class1	0.003177
Class2	0.004401
Class3	0.005507
Class4	0.007197
Class5	0.010125
전체평균	0.006081

[표 6] Percentile 스케줄러 등급별 응답 시간

Percentile 시간의 스케줄에서는 각 클라이언트가 현재 요청된 서비스에 대해서 Percentile 기법으로 산출한 우선순위에 의해서 처리되었기 때문에 높은 등급이 항상 응답 시간이 빠른 것은 아니지만, 전체적인 응답 시간을 보면 상위등급의 클라이언트가 하위등급의 클라이언트에 비해서 평균적으로 응답 시간이 빠르게 측정되었다.

실제로 리눅스 시스템의 커널 스케줄러와 *nice()*을 이용하여 우선순위를 조정하였을 때와는 달리 각각 클라이언트의 응답 시간이 부여된 등급에 따라 확연하게 차등화 된 응답 시간을 보였다. [그림 4]은 두 가지 실험 결과를 비교한 그래프이다. x축은 등급을 나타내고, y축은 평균 응답 시간을 초 단위로 표현한 것이다.



[그림 4] 등급별 각 스케줄러 평균 응답 시간 비교 그래프

4. 결론

본 논문은 패킷 라우터에서 패킷을 대상으로 사용하던 Percentile 스케줄 기법을 리눅스 커널의 태스크 스케줄에 적용하고자 한 연구이다. 제안된 기법의 실제 구현과 성능을 평가하기 위해 실험을 수행하였으며, Percentile 스케줄 기법을 적용한 경우, *nice()*를 이용한 차등화 처리를 하는 스케줄러에 비해 등급에 따른 응답 시간의 차이가 명확했다. 이를 이용하여,

서론에서 언급했던 것과 마찬가지로 서비스를 고객에 따라 차등적으로 제공할 수 있게 되었다.

향후 연구과제로는 네트워크 계층과 디스크 입출력에 대한 Percentile 기법의 적용이 있다. 본 논문은 태스크 스케줄에 관련된 부분만을 대상으로 했기 때문에 이는 요청을 접수하고 응답을 만들어 내기 위한 서버내의 전 처리과정의 일부로써 서버 전체를 대상으로 볼 때 그 효과가 제한적이며, 더 나아가 요청 패킷의 수신부터 응답패킷의 송신까지 모든 곳에서 차등화 기법을 적용시키는 것이 궁극적인 목표이다. 그 중 서버 성능에 영향을 많이 끼치는 부분으로서, 네트워크 계층과 디스크 입출력 계층이 존재한다. 따라서 패킷을 직접 처리하는 네트워크 계층의 Percentile 스케줄 기법의 적용을 위한 연구가 필요하며, 응답 시간의 상당 부분을 차지하는 디스크 입출력 서비스 우선순위 제어를 통한 차등화 연구도 진행 되어야한다.

참 고 문 헌

[1] 김중규, “차등서비스 구조에서 동적 스케줄링을 이용한 공평성 제공, Journal of Computer & Communication”, Vol. 2, No. 1, 2003.

[2] Bovet Daniel Pierre, "Understanding the Linux Kernel, 3nd Edition", OReilly, 2006.

[3] 최동준, “리눅스에서의 Percentile 스케줄 기법 구현”, 한성대학교 대학원 석사 학위 논문, 2002.

[4] 유영창, “리눅스 디바이스 드라이버”, 한빛미디어, 2004.

[5] 윤찬희, 송진석, 김다현, 이민석, “시스템 시간 측정을 위한 TsMon 드라이버”, 한성대학교 공학연구 논문집 Vol. 4, No. 2 pp.59-66, August 2006.