제33회 한국정보처리학회 춘계학술발표대회 논문집 제17권 제1호 (2010. 4)

분산 VOD 서버 환경에서 히스토리 기반의 동적 부하분산 스케줄러

문종배
한국과학기술정보연구원
e-mail: jbmoon@kisti.re.kr

A History-based Scheduler for Dynamic Load Balancing on Distributed VOD Server Environments

Jongbae Moon
Korea Institute of Science and Technology Information

요 약

최근 사용자의 멀티미디어에 대한 요구의 증가가 VOD (Video-on-Demand) 서비스를 발전시키게 되었다. VOD는 엔터테인먼트나 원격 교육, 광고 및 정보 등 많은 분야에서 사용되고 있다. 이러한 VOD 서비스는 많은 디스크 I/O와 네트워크 I/O를 요구하며 기존 웹 서버 시스템과 비교했을 때 오랜 시간 동안 서비스를 해야 하는 특징을 가지고 있다. 또한 VOD 서비스는 많은 네트워크와 디스크의 대역폭을 요구하며, 서비스의 QoS에 민감해서 사용자 응답시간이 길어지면 사용자 요청의 취소율이 높아지게 된다. 따라서 불만족스러운 서비스의 증가로 네트워크 부하만 증가하게 된다. 이러한 기존 웹 서버 환경과는 다른 부하의 패턴이 있는 VOD 서비스 환경에서는 부하를 균형적으로 분배하여 서비스의 QoS를 높이는 것이 매우 중요하다. 본 논문에서는 분산 VOD 시스템 환경에서 부하를 효율적으로 분산하기 위해 계층형 분산 VOD 시스템 모델과 사용자 요청 패턴의 히스토리와 유전 알고리즘을 기반으로 한 스케줄러를 제안한다. 본 논문에서 제안한 계층형 분산 VOD 시스템 모델은 서버들을 지역적으로 분산하고 제어 서버를 지역마다 설치하여 지역에 있는 VOD 서버들을 관리하도록 구성한다. 사용자 요청을 지역 서버군 내에서 분산시키기 위해서 히스토리를 기반으로 한 유전 알고리즘을 사용한다. 이러한 히스토리 정보를 기반으로 유전 알고리즘의 적합도 함수에 적용하여 VOD 시스템을 위한 유전 알고리즘과 유전 연산을 구현한다. 본 논문에서 제안한 부하 분산 알고리즘은 VOD 서비스 환경에서 사용자 요구에 대한 부하를 보다 정확하게 예측하여 부하를 분산할 수 있다. 본 논문에서 제안한 계층형 분산 VOD 시스템의 부하 분산 알고리즘의 성능을 테스트하기 위해 OPNET 기반 시뮬레이터를 구현한다. 라운드로빈(round-robin) 방식과 랜덤(random) 방식과의 비교 실험을 통해 본 논문에서 제안한 부하 분산 알고리즘의 성능을 평가한다. 비교 실험을 통해 본 논문에서 제안한 알고리즘이 보다 안정적인 QoS를 제공하는 것을 보여준다.

1. 서론

최근 몇 년 동안 컴퓨터와 네트워크 장비의 발달로 인터넷이 급속도로 발전하고 있다. 또한, 최근에는 오디오나 비디오를 인터넷을 통해 전송하려는 노력이 시도가 있었고 지금은 웹과 초고속 통신망을 통해 동영상이 실시간으로 전송되고 있다. 최근에는 이러한 노력과 기술로 VOD (Video-on-Demand) 서비스가 활성화 되고 있다[1]. VOD는 원격지에 있는 사용자의 요구에 따라 영화나 뉴스 등의 영상 및 음성 기반의 서비스를 네트워크를 통해 제공하는 영상 서비스를 말하며, 주문형 비디오라고도 불린다.

이러한 VOD 서비스는 기존 HTTP 서비스와는 다른 특징을 갖고 있으며, 부하 패턴 마찬가지로 다른 특징을 갖는다. VOD 서비스는 HTTP 서비스에 비해 시간이 매우 길고, 많은 네트워크와 디스크의 대역폭을 요구하는 특징이 있다. 또한 서비스의 QoS에 민감해서 사용자 응답시간이 길어지면 사용자 요청의 취소율이 높아지게 되고 네트워크 부하만 증가하게 된다. 따라서 기존에 많이 연구되었던 분산 환경에서의 부하 분산 알고리즘들[2][3][4]은 VOD 환경에 적용하기에는 적당하지 않다.

VOD 서비스에 맞는 부하 분산을 위해서는 동적으로 변하는 부하뿐만 아니라 사용자 요청 패턴을 분석하면 시스템의 부하를 예측할 수 있다. 예를 들어 VOD 파일의 하루 동안의 접속 패턴을 알면 하루 중 어떤 시간에 자원 소모가 많은 지 알 수 있다. 이러한 사용자 요청 패턴을 이용하여 CDN과 같은 서비스에서는 캐싱을 하는데 적용하고 있다. 따라서 사용자 요청을 히스토리로 저장하였다가 분석하여 사용자 요청 패턴을 생성한다면 부하를 분산하는 데 중요한 요소로 작용할 수 있다. 그러나 이러한 히스토리를 기반으로 한 VOD 부하 분산 방법은 개발되지 않았다. 따라서 본 논문에서는 VOD 환경을 위한 부하 분산 스케줄러를 제안한다. 본 논문에서 제안하는 스케줄링은 계층형 분산 VOD 환경에서 사용자 요청의 히스토리를 기반으로 하고, 유전 알고리즘을 사용하여 많은 사용자의 다양한 서비스 요청을 최적의 서버에 할당하도록 한다. 이러한 유전 알고리즘을 위해 히스토리를 기반으로 한 적합도 함수를 개발하여 적용한다.

2. VOD 시스템의 부하 분산 알고리즘

본 장에서는 계층형 분산 VOD 시스템의 전역 부하분산을 위하여 DNS를 이용하여 (그림 1)과 같은 최적의 서버 유지 알고리즘을 수행한다[5]. DNS에 등록된 제어서버는 n 대가 있다고 가정하며, 사용된 기호의 의미는 다음과 같다.

$T_{i,Load}$ = i번째 제어서버의 총부하량

α, β, γ : 가중치 $(\alpha+\beta+\gamma=100)$, TH: 임계값

S_i = i번째 제어서버

$L_{i,CPU}$: i번째 서버의 CPU부하량,
$L_{i,MEM}$: i번째 서버의 메모리부하량,

$L_{i,NT}$: i번째서버의 네트워크 부하량

```
while (1) {
  for(i = 1  to  n)
  {
    T_{i, Load} = α × L_{i, CPU} + β × L_{i, MEM} + γ × L_{i, NT}
    if ((S_i is in DNS List) and (T_{i, Load} > TH))
      then remove S_i from DNS list
    else if ((S_i is not in DNS list) and (T_{i, Load} < TH))
      then add S_i to DNS list
  }
}
```

(그림 1) 최적의 서버를 유지하기 위한 알고리즘

우선 i 번째 제어서버 S_i 의 총부하량을 구한다. 각 서버의 부하를 측정할 때에는 CPU와 메모리, 네트워크의 사용률을 이용하여 측정한다. 각 서버의 자원 사용량은 제어 서버에서 수행되는 모니터링 모듈이 수집한 정보를 사용한다. 분산 시스템이 제공하는 서비스의 내용에 따라 각 자원의 사용량에 다른 가중치를 두어 총부하량을 계산해야 한다. 온라인 방송이나 VOD 서비스와 같이 대용량의 파일 전송을 하는 서비스일 경우 CPU와 메모리 사용률 증가보다 네트워크 사용률의 증가폭이 크며, 온라인 게임과 같이 서버에서의 많은 처리를 요구하는 서비스일 경우는 네트워크 사용률의 증가보다 CPU 사용률의 증가폭이 크다. 따라서 총부하량을 계산할 때에는 서비스의 종류에 따라 사용률의 증가폭이 큰 자원에 가중치를 높여서 계산한다.

S_i 가 DNS 리스트에 존재하고 총부하량이 임계값보다 크면 S_i 를 리스트에서 삭제하고, S_i 가 DNS 리스트에 존재하지 않고 총부하량이 임계값보다 작으면 부하가 다시 줄어들었다고 판단하여 다시 리스트에 추가한다. 그 외의 경우에는 서버의 현재 상태로 계속 유지하도록 한다. 임계값은 학습에 의한 값으로 관리자가 직접 정할 수 있도록 한다. 제어서버의 모니터링 모듈에 의해 측정된 부하정보를 기반으로 다음과 같은 갱신 정책들을 동적으로 선택하고 변경할 수 있도록 하였다.

- CPU 기반 방식: $\alpha=100, \beta=0, \gamma=0$, CPU 사용량이 대부분인 서비스의 경우 사용
- 메모리 기반 방식: $\alpha=0, \beta=100, \gamma=0$, 메모리 사용량이 대부분인 서비스의 경우 사용
- 네트워크 기반 방식:$\alpha=0, \beta=0, \gamma=100$, 네트워크 사용량이 대부분인 서비스의 경우 사용
- 복합 방식: $\alpha+\beta+\gamma=100$, CPU와 메모리, 네트워크의 사용량이 모두 무시하지 못할 정도로 사용되는 서비스의 경우, 관리자에 의해 가중치가 설정

지역 부하분산은 제어서버에서 관리하는 VOD 서버들 사이의 부하를 분산하는 방법이다. 다양한 서비스 형태의 요청에 대해 최적의 서버들을 할당하는 일반적인 방법이 없기 때문에 본 논문에서는 지역 부하분산을 위해 유전 알고리즘을 적용한다.

지역 서버군에서 서버들을 서비스의 유형별로 구성하였다. 서비스의 종류가 섞여서 요청되기 때문에 같은 종류의 작업들끼리 정렬할 필요가 있다. 서비스별 정렬이 끝난 후에는 모의 해 집단을 구성한다. 모의 해 집단은 m 개의 사용자 요청에 대해 n 개의 서버들에 할당하는 경우 중 랜덤하게 10%의 모의 해 집단을 구성한다.

각 모의해의 질을 평가하기 위해 적합도 함수를 작성한다. 적합도 함수는 요청된 작업이 완료되기까지의 시간을 최소화 하도록 만들어졌다. 이 때 예상되는 부하량을 히스토리에서 참조를 하도록 한다. 적합도 함수가 0에 가까울수록 사용자의 대기시간이 줄어들고 1에 가까울수록 사용자 대기시간이 길어지므로 0에 가까운 값을 갖는 해집합이 선택될 확률이 높다.

$$F(x) = \sum_{i=1}^{m} T_i \qquad \text{(식 1)}$$

T_i = P_i에서의 현재 부하 × P_i에서 작업시간 × 작업의 부하 히스토리

(P_i : i번째 프로세스, T_i : 예상 작업 시간) (식 2)

P_i 에서의 현재 부하 = P_i의 CPU 활용률 × 메모리 활용률 × 네트워크 활용률 (식 3)

작업의 부하 히스토리 = 이전 수행했던 동일 작업들의 부하의 평균 (식 4)

적합도 값 = $\frac{1}{F(x)}$ (식 5)

재생산은 한 세대의 개체집단으로부터 다음 세대의 개체집단을 만들기 위해 각 염색체의 적합도에 따라 반복 재생시키는 방법으로 적합도가 높은 개체일수록 다음 세대에 더 많은 자손을 가질 확률이 높게 선택되며, 이것은 주어진 환경에 잘 적응하는 개체만이 생존하는 적자생존의 자연선택원리를 적용한다. 재생산의 방법은 여러 가지가 있으나 그 중 가장 대표적인 것이 확률바퀴 방법이다. 이 방법은 총 적합도에 대한 개별 염색체의 적합도가 높은 순으로 원반의 크기를 결정하여 다음 세대로 재생산되도록 하지만 랜덤 변수를 이용하여 재생산 과정을 수행하기 때문에 적합도가 높은 염색체가 선택되지 않아 세대가 진행함에 따라 적합도 값이 적어지는 단점을 가진다. 본 논문에서는 이러한 단점을 보완하기 위해 적합도 값이 높은 엘리트 염색체에 대하여는 다음 세대에 그대로 재생산하는 엘리티즘(elitism)을 적용하고, 그 나머지 염색체에 대해서는 확률바퀴 방법을 사용한다.

선택연산은 교배에 쓰이는 두 개의 부모 해를 고르기 위한 연산자이다. 다양한 연산자들이 있으나 공통된 원칙은 우수한 해가 선택될 확률이 높아야 한다는 것이다. 우수한 해들과 열등한 해들 사이의 적합도 차이를 조절함으로써 선택 확률을 조절할 수 있는데, 이 차이의 정도를 선택압(selection pressure)이라 한다. 교배는 일점 교배가 많이 사용되지만 해집합이 그룹단위로 나누어져 있기 때문에 각 그룹마다 일점 교배를 하도록 한다. 변이는 해집합에서 특정 염색체들을 다른 값으로 바꾸는 작업이다. 후보 해집합이 모든 해집합을 다 표현할 수 없기 때문에 교배와 변이를 통해 다양한 해집합을 평가할 수 있도록 해준다.

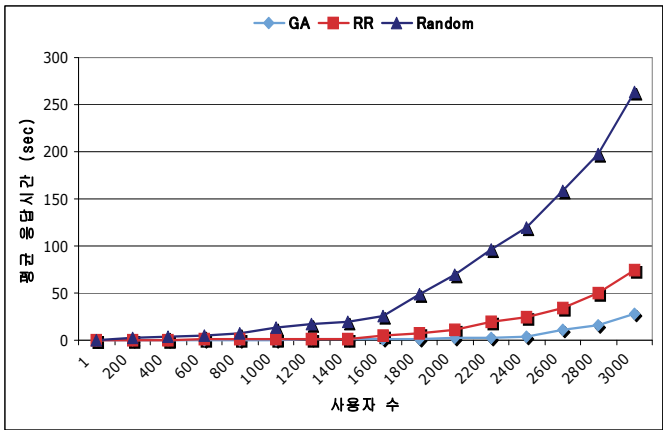
3. 실험 및 성능평가

본 논문에서 제안한 계층형 분산 VOD 환경에서 부하 분산을 위한 실험을 위한 시뮬레이터는 OPNET Technologies의 OPNET Modeler [6]와 C언어를 이용하여 구현하였다. 실험을 위해 사용한 환경은 다음과 같다. VOD 서비스는 영화와 뉴스, 온라인 강의의 3가지 서비스를 지원하도록 설정하였다. 서버 그룹은 15대의 서버들로 구성하였다. 클라이언트 노드는 1대에서 3000대를 생성하여 사용하였다. VOD 파일의 수는 100개를 사용하였다. 유전 알고리즘을 위한 요소들은 [표 1]과 같이 염색체의 길이와 교배 확률, 돌연변이 확률, 총 세대 수, 종료조건 등이 있다.

<표 1> 시뮬레이션을 위한 유전알고리즘의 요소

염색체 길이	20
교배 확률	0.9
돌연변이 확률	0.01
세대 수	500

첫 번째 실험은 사용자의 증가에 따른 응답시간의 변화를 살펴보는 실험을 수행했다. 사용자가 증가하면 VOD 요청이 증가하고 좀 더 다양한 유형의 사용자 요청이 들어온다.



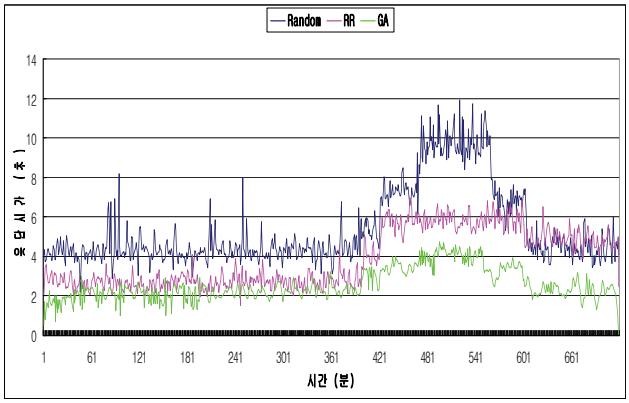
(그림 2) 사용자 증가에 따른 응답시간의 비교

(그림 2)는 사용자가 1에서 3000까지 증가하면서 평균 시작 응답시간의 변화를 측정한 그림이다. 세 가지 알고리즘이 사용자가 증가하면서 어느 정도까지는 모두 비슷한 성능을 보이다가 점점 성능이 차이나면서 증가하는 것을 볼 수 있다. 사용자가 어느 정도 증가할 때까지는 서버들의 성능으로 충분히 안정적인 QoS를 제공할 수 있지만 그 이상이 되면 제어 서버의 대기큐와 VOD 서버의 대기큐에서 대기하는 시간이 길어지기 때문에 사용자가 느끼는 평균 시작 응답시간은 길어진다. 랜덤(Random) 스케줄링이 가장 급격하게 응답시간이 증가하는 것을 볼 수 있다. 사용자 요청이 많아지면 한정된 서버에 부하가 많이 증가한 서버에 많은 사용자 요청이 할당되기 때문이다. 라운드로빈(RR) 스케줄링과 히스토리 기반 유전 알고리즘(GA)은 응답시간의 변화가 크지는 않지만 계속 증가하는 것을 볼 수 있다. 본 실험에서 히스토리 기반 유전 알고리즘은 평균 시작 응답시간이 40초 이하였다. 실제 VOD 서비스의 경우 40초 정도의 지연시간동안 케이블 방송에서처럼 광고를 내보낸다면 실제 환경에서도 충분히 기다릴 수 있는 시간이다. 사용자의 요구가 더 많아지면 VOD 서버의 수를 증가함으로써 해결할 수 있다.

히스토리 기반 VOD 부하 분산 알고리즘의 성능을 측정하기 위해 하루 동안의 사용자 패턴에 대한 히스토리를 기반으로 한 실험을 하였다. 사용자의 요구 패턴은 히스토

리를 참조하여 얻을 수 있다. 본 실험에서는 사용자 요청이 오후 6시정도부터 급격하게 증가하는 패턴을 이용하였고, 사용자를 최대 1000대까지 증가시키면서 실험하였다.

(그림 4)는 사용자 요청 패턴에 따른 응답시간의 변화를 측정한 실험 결과이다. 가로축은 시간을 나타내며 오전 10시부터 오후 22시까지의 시간을 의미한다. 사용자 요청이 많은 부분에서는 세 방법 모두 사용자 응답시간이 길어지고 있는 것을 볼 수 있다. 랜덤 방식과 라운드로빈 방식 부하 분산은 사용자가 사용자 요청이 6시부터 늘어나면서 응답시간이 급격히 증가하는 것을 볼 수 있다. 랜덤 방식은 사용자의 요청이 많지 않을 경우에도 응답시간의 변화가 많을 뿐만 아니라 사용자의 요청이 많아지면 이전 서비스에도 영향을 받기 때문에 응답시간은 더 증가된다. 히스토리를 이용한 유전 알고리즘은 사용자 요청이 많았을 때의 부하를 예측하여 부하를 분산하기 때문에 각 VOD 서버의 부하를 고르게 활용하면서 사용자 응답시간이 많이 증가하지 않는다. 첫 번째 실험에서처럼 사용자를 3000대까지 증가시킨다면 응답시간의 차이가 더 커질 것이다. 따라서 본 논문에서 제안한 알고리즘은 기존 부하 분산 알고리즘보다 안정적인 QoS를 제공할 수 있다.



(그림 4) 사용자 요청 패턴에 따른 응답시간의 변화

5. 결론

본 논문에서는 분산 VOD 시스템 환경에서 부하를 효율적으로 분산하기 위해 계층형 분산 VOD 시스템 모델과 사용자 요청 패턴의 히스토리와 유전 알고리즘을 기반으로 한 스케줄러를 제안하였다. 본 논문에서 제안한 계층형 분산 VOD 시스템 모델은 서버들을 지역적으로 분산하고 제어 서버를 지역마다 설치하여 지역에 있는 VOD 서버들을 관리하도록 구성하였다. 지역적으로 서버를 분산함으로써 여러 지역에 산재되어 있는 사용자들의 요청이 한 지역에 집중되는 것을 막을 수 있다. 지역 서버군에 있는 VOD 서버들은 제공되는 형태의 서비스 그룹으로 묶어서 서비스 할 수 있도록 구성하였다. 기본적으로 각 서비스에서 자주 요청되는 VOD들을 여유 서버를 두어 특정 서비스의 요청이 많은 시간에 활용할 수 있도록 한다. 이러한 여유 서버는 동적으로 부하가 많은 가상 서비스 그룹에 추가되었다가 삭제된다.

사용자 요청을 지역 서버군 내에서 분산시키기 위해서 사용자의 요청을 데이터베이스에 히스토리로 구성하여 저장하였다가 참조한다. VOD 서비스의 사용자 요청 패턴을 분석하면 시스템의 부하를 예측할 수 있기 때문이다. 부하 분산을 위해 일반 유전 알고리즘을 변형하여 본 논문에서 제안하는 형태의 시스템을 위한 유전 알고리즘과 유전 연산을 제안하고 구현하였다. 본 논문에서 제안한 계층형 분산 VOD 시스템의 부하 분산 알고리즘의 성능을 테스트하기 위해 OPNET을 이용하여 시뮬레이터를 구현하였다. 라운드로빈(round-robin) 방식과 랜덤(random) 방식과의 비교 실험을 통해 본 논문에서 제안한 부하 분산 알고리즘의 성능을 평가하였다. 실험은 사용자 응답시간과 자원 활용률, 패킷 손실률, 서비스 취소율에 대하여 비교 실험하였다. 비교 실험 결과 본 논문에서 제안한 알고리즘이 기존 알고리즘보다 안정적인 QoS를 제공하는 것을 보였다.

향후 연구 과제로는 다양한 평가 기준을 세우고 그에 따른 다양한 실험을 통하여 좀 더 정확한 성능평가가 필요하다. 또한 VOD 서비스는 서비스 중에 사용자의 요구에 따라 응답을 하는 상호관계가 있다. 이러한 사용자와의 상호관계를 부하를 측정하는데 적용하여 좀 더 정확한 부하를 예측할 수 있도록 하는 연구가 필요하다.

참고문헌

[1] Ma, K., Shin, K.G. “Multicast Video-On-Demand Services” ACM SIGCOMM Computer Communication Review 32(1), 31 - 43 (2002)

[2] Loukopoulos, T., Ahmad, I. “Static and Adaptive Distributed Data Replication Using Genetic Algorithms” J. of Parallel and Distributed Computing 64(11), 1270-1285 (2004)

[3] Rothlauf, F. “Representations for Genetic and Evolutionary Algorithms” In: Studies on Fuzziness and Soft Computing, vol. 104 (2002)

[4] Mundur, P., Simon, R., Sood, A.K. “End-to-End Analysis of Distributed Video-on-Demand Systems” IEEE Transactions on Multimedia 6(1), 129-141 (2004)

[5] Moon, J.-B., Kim, M.-H. “Dynamic Load Balancing Method Based on DNS for Distributed Web Systems” In: Bauknecht, K., Pröll, B., Werthner, H. (eds.) EC-Web 2005. LNCS, vol. 3590, pp. 238 - 247. Springer, Heidelberg (2005)

[6] OPNET Technologies Inc., http://www.opnet.com/