서버기반 컴퓨팅 환경에서 사용자 패턴을 이용한 효율적 로드 밸런싱 구현

김성미, 박명순
고려대학교 컴퓨터 과학 기술 대학원
e-mail : kimsungmee@hotmail.com
myongsp@korea.ac.kr

An efficient Load Balancing using user patterns in Server Based Computing

Sung-Mee Kim, Myong-Soon Park
Graduate School of Computer Science & Technology, Korea University

요 약

클라이언트/서버 기반 시스템은 응용 프로그램 사용에 있어, 각 클라이언트 별 셋업 및 유지 보수 등의 문제점이 있다. 이런 문제점의 대안으로 서버기반 컴퓨팅이 제시되었다. 이 환경에서 클라이언트는 터미널 서비스를 위해 서버에 접속하고, 로드 밸런서는 이에 대해 적절한 서버를 할당해 주며 그에 따라 터미널 세션이 설정된다. 하지만 이 구조에는 터미널 세션의 유지와 사용자의 컴퓨팅 패턴 등에 의해 서버 부하가 가중되는 문제점이 있다. 따라서 본 연구에서는 클라이언트 별 터미널 서비스 사용량을 추출하고 그에 따라 사용자 패턴에 구분하여 로드 밸런싱 시 이를 적용함으로써 서버 자원의 분배를 보다 효율적으로 할 수 있게 되었다.

1. 서론

오늘날 클라이언트/서버 기반으로 구성된 기업 또는 학교의 전산 환경은 클라이언트에서 실행되는 응용프로그램이 점차 대형화지고 복잡해 짐에 따라 응용 프로그램 실행을 위한 클라이언트의 잦은 업그레이드 및 신규 구입과 설치, 그에 따른 시스템 셋업과 유지비용이 증가하는 문제점을 가지게 되었다. 따라서 이와 같이 클라이언트/서버 구조와 같은 분산환경에서 나타나는 문제점을 해결하기 위한 새로운 대안이 제시되기 시작했으며 그 중 하나가 바로 서버기반 컴퓨팅 환경이다[1].

서버기반 컴퓨팅은 모든 응용프로그램과 데이터가 서버에 설치되고 저장되며, 각각의 클라이언트는 서버에 접속하여 서버에서 실행되는 응용프로그램을 단순히 화면 값을 통해 마치 자신의 컴퓨터에서 실행하는 것과 같이 사용하는 방식이며 이러한 서버기반 컴퓨팅 환경을 구성하기 위해서는 가상의 데스크톱을 보여줄 수 있는 디스플레이와 사용자 입력을 받기 위한 입력장치, 네트워크 연결을 위한 인터페이스가 필요하다. 또한 이러한 장치에 의해 서버에 설치된 응용프로그램을 사용하는 것을 터미널 서비스라고 한다.

따라서 터미널 서비스 역시 다수의 사용자가 원하는 정보 또는 서비스를 얻기 위해 해당 서버에 접속하여 작업을 한다는 것이 마치 웹 서비스와 비슷하다고 생각 할 수 있다. 하지만 웹 서비스인 경우 정해진 사용자가 아닌, 불특정 다수의 이용자가 접속을 하는 것에 비해 터미널 서비스는 기업이나 학교, 또는 정부 기관등에서 효율적인 전산 작업 및 인프라 구성을 위한 컴퓨팅 환경이므로 해당 단체에 소속된 사용자들만이 이용할 수 있는, 즉 이미 사용자가 정해진 서비스라고 볼 수 있다.

또한 터미널 서비스에서 각 클라이언트 연결에 따른 독립성 유지는 터미널 세션(Session)이 담당하며 서버 자원의 효율적인 분산을 위해 일반적으로 로드 밸런서에 의해 가장 적절한 서버가 할당 되고, 클라이언트가 할당된 서버에 접속함으로써 연결된다. 하지만 터미널 세션의 특징은 한 번 설정된 세션으로

여러 응용 프로그램 작업뿐만 아니라 사용자가 로그오프 하기 전까지는 계속 유지가 되어 재 연결 되기 때문에, 일반적인 로드 밸런스 기법에 의한 서버 분배는 문제점이 있다. 따라서 제한적인 서버 자원을 보다 효율적으로 분산하기 위해, 이미 정해져 있는 사용자들의 컴퓨팅 사용량을 추출하여 사용자 패턴을 분석한다. 또한 실제 로드 밸런싱 때 이를 적용함으로써 보다 효율적인 로드 밸런싱 기법에 대해 연구한다.

따라서 본 논문의 구성은 2 장에서 관련 연구로 터미널 서비스 및 세션의 개념 이해와 로드 밸런싱 기법 적용의 문제점을 파악하고 이를 해결하기 위해, 3 장에서는 사용자 패턴에 의한 효율적인 로드 밸런싱 기법에 대해 설명한다. 4 장에서는 제안한 기법의 실험 및 결과에 대해 설명하고 마지막으로 5 장에서는 결론 및 향후 연구 과제를 말한다.

2. 관련 연구

2.1 터미널 서비스

터미널 서비스란 모든 클라이언트의 응용프로그램 실행, 데이터 처리 및 데이터 저장을 서버에서 처리하고, 이를 정해진 프로토콜을 통해 사용자들에게 제공한다. 그림 1 은 프로토콜을 통해 터미널 서비스에서 사용자들에게 어플리케이션 실행 및 처리결과를 제공하는 방법을 나타내고 있다.

그림 1. 터미널 서비스 제공 방법

다음은 현재 대표적으로 사용되고 있는 터미널 서비스를 위한 프로토콜들이다[2].

- Microsoft Windows Terminal Services
  : RDP (Remote Desktop Protocol)
- Tilon TusKan Professional Server2004
  : ATC (Application Transmission Control)
- Citrix MetaFrame for Windows
  : ICA (Independent Computing Architecture)
- Tarantella Enterprise Express for Linux
  : AIP (Adaptive Internet Protocol)
- AT&T VNC for Linux
  : VNC (Virtual Network Computing)
- Sun Ray For Solaris
- Xfree86 on Linux

2.2 터미널 세션

터미널 세션이란 클라이언트가 서버 자원을 이용하기 위한 연결로써 서버자원의 한도 내에서 세션 연결이 이루어진다. 또한 사용 도중 사용자의 네트워크 이상으로 인해 연결이 종료가 되더라도 터미널 서버가 관리하는 메모리상에 여전히 기존 세션에 대한 것이 남아있기 때문에 다시 연결을 시도하게 되면 새로운 세션의 생성 없이 기존 세션 연결을 재 사용한다는 특징이 있다[3].

즉, 터미널 서비스에서 세션의 종료라 함은 메모리상에서도 그 연결이 소멸되는 시점인, 사용자가 로그오프 되어 작업 환경이 모두 프로필에 저장된 후라고 볼 수 있다.

2.3 터미널 서비스 로드 밸런싱

터미널 서비스 역시 클라이언트가 네트워크를 통해 서버에 접속함으로써 이루어지는 서비스이다. 따라서 클라이언트가 서버에 접속 시, 한정된 서버 자원을 효율적으로 분산 시키고 자원을 균등하게 분배하는 로드 밸런싱 기법이 무엇보다 필요하다.

일반적으로 사용자로부터 터미널 서비스를 위한 서버 접속 요청 시 부하분산은 로드 밸런서를 통해 가장 적절한 서버가 정해지고 이를 사용자에게 알려준다. 사용자는 로드밸런서로부터 받은 서버 주소를 가지고 바로 서버와 접속을 시도하며, 비로소 세션이 연결 된다. 이와 같이 처음으로 터미널 세션을 연결한다는 의미는 터미널 세션의 특성 상 어떻게 보면 그 사용자의 터미널 서비스를 이용하는 전 컴퓨팅 과정에 영향을 미친다고 볼 수 있다. 따라서 어떠한 로드 밸런싱 기법에 의해 가장 적절한 서버를 정하는가가 중요한 관건이다.

하지만 일반적으로 터미널 서비스에서 로드 밸런싱은 각 서버에 할당된 사용자가 서버의 프로그램을 사용하는데 있어 어떠한 사용량을 가지고 사용 하는지 전혀 고려되지 않은 채 접속 당시의 사용자가 서버에 미치는 리소스 부하량과 현재 클러스터 내에 속한 서버들의 리소스 부하량만(CPU 처리량, 메모리 또는 커널 메모리 사용량)을 고려한 단순한 로드 밸런싱 기법을 이용하고 있다.

예를 들어 오피스 프로그램과 그래픽 프로그램 설치되어 있는 터미널 서버 3 대를 사용자 5 명이 사용한다고 했을 때, 사용자 5 명 중 User1, User2 는 담당하는 업무 특성상 두 프로그램을 업무 시간 내내 사용하는 heavy User 이다. 하지만 나머지 User3, User4, User5 는 단순히 오피스 프로그램만 사용하고 사용시간도 짧은 Light User 라고 가정했을 때, 만약 User1~User5 모두 처음에 단순히 워드를 사용하기 위해 서버에 접속하게 되면, 로드 밸런서는 모든 사용자를 서버 3 대에 순서적으로 할당할 것이다. 하지만 User1, User2 작업 특성 상 한번 연결된 세션을 이용해서 워드뿐만 아니라 다른 오피스 프로그램 및 그래픽 프로그램도 함께 사용할 수 있으며, 그렇게 되면, User1 과 User2 가 속한 서버는 Heavy User 가 속하지 않은 서버에 비해 부하량이 상대적으로 증가하는 결과를 가져온다.

따라서 본 논문에서는 터미널 서비스에서 보다 효

율적인 로드 밸런싱을 위해 사용자 별 서버 접속 시간 및 프로그램 별 사용시간에 따른 로그를 추출하여 사용자의 작업 패턴을 분석한다. 또한 이렇게 분석된 패턴을 기본으로 단순히 로드 밸런싱 시점 시 최소 리소스 부하량만을 고려한 부하 분산이 아닌, 사용자 패턴을 고려한 로드 밸런싱을 함으로써 서버 자원의 분배에 있어 성능 개선이 되었는지를 분석하고자 한다.

3. 사용자 패턴에 의한 로드 밸런싱 기법

3.1 사용자 패턴 분석을 위한 로그 추출

사용자 패턴 분석을 위해 필요한 사항은 어떤 사용자가 터미널 서버에 접속하여 어떤 프로그램을 얼마나 오래 사용함으로써 서버의 리소스를 점유 하였는지를 추출하는 것이다. 이는 터미널 서비스의 클라이언트 프로그램을 이용하면 해결할 수 있다.

우선 사용자는 인증을 위해 사용자 아이디로 로그인을 하고 작업 종료 후에는 로그 오프를 하는 것을 이용해, 서버 접속 시간을 구할 수 있다. 또한, 사용자에게 배포된 어플리케이션을 실행시키는 시간과 종료하는 시간을 이용해 프로그램 별 사용 시간을 추출할 수 있다.

이때, 시간에 따른 점유율을 생각해 볼 수 있다. 예를 들어 한 사용자의 일정기간 동안 기록된 총 서버 접속 시간이 50 시간인 반면, 사용 프로그램별 총 사용시간은 20 시간에 불과하였다면, 서버에 접속만 한 채, 실제적으로 서버의 리소스를 발생하는 컴퓨팅 작업은 많이 하지 않았다는 이야기이다. 따라서 사용시간별 가중치는 다음과 같은 방법에 의해 구할 수 있다. 먼저 모든 사용자의 프로그램 이용률을 구한 후 최대 이용률을 가진 사용자의 이용률을 1 이라 보고, 사용자 프로그램 이용률에 따른 상대적인 가중치를 구한다.

- UR(사용자프로그램이용률)=TPT(총프로그램사용시간) / TST(총서버접속시간)
- TW(사용 시간별 가중치)=UR/최대 UR

다음으로, 프로그램 별 가중치에 대해 생각해 볼 수 있다. 예를 들어 워드 프로그램과 그래픽 프로그램을 사용한다고 했을 때, 그래픽 프로그램이 더 많은 서버 리소스를 요구한다. 즉, 동일한 사용 시간이라 하더라도 프로그램별로 가중치를 달리해야 하며, 이는 프로그램 실행 시, 메모리 점유가 가장 큰 프로그램을 1 이라 보고, 메모리 점유율에 따른 상대적인 가중치 값을 구한다. 또한 가중치 값을 프로그램 별 사용시간에 적용함으로써 전체 프로그램 사용에 대한 리소스 점유 평균치를 구할 수 있다.

- PW(프로그램가중치)= PM(메모리 사용량)/최대 PM
- UW(사용자가중치)= 프로그램Σ(사용시간*PW)*TW
- 평균치=사용자Σ(UW)/사용자수

이렇게 해서 나온 결과를 기준으로 3 단계의 사용자 패턴, 즉 Heavy User, Medium User, Light User 로 분류한다. 여기서 Heavy User 란 평균치를 50 이라고 보았을 때, 80 이상인 사용자이며, Light 란 20 이하인 사용자로 정한다.

3.2 사용자 패턴을 적용한 제안 알고리즘

터미널 서비스 환경에서 효율적인 부하 분산을 위한 제안 알고리즘은 다음과 같다. 우선 서버 자체에 가중치를 부여하여 Medium User 와 Heavy, Light User 를 위한 서버로 분류한다. 이렇게 두 가지 분류로 나눈 이유는 대체적으로 Heavy User 인 경우 세션 연결 시간뿐만 아니라 프로그램 사용 리소스도 많이 차지하는데 비해 Light 사용자는 상대적으로 사용시간 및 리소스 사용률이 적어서 전체적으로 그 두 사용자들과 Medium 사용자들과 대등한 관계를 보이기 때문이다. 따라서 로드밸런서는 미리 분석된 사용자 패턴 DB 와 접속을 위해 로드밸런서로 들어온 사용자를 매치해 각각 해당하는 서버에 할당한다. 이때, 만약 나뉜 서버가 두개 이상의 복수 개 일 경우, 서버는 그 당시 최소 리소스(CPU, 메모리, 커널 메모리 등)에 우선적으로 할당한다. 그림 2 는 메모리 사용량을 기준으로 한 제안 로드 밸런싱 알고리즘을 나타낸 것이다.

```
//서버정의
Server_id[1]=Heavy_Light server;
Server_id[2]=Heavy_Light server;
Server_id[3]=Medium server
Server_id[4]=Medium server

If (request_connection(user_type)==”Medium”){
   index=3;
   } // 사용자 패턴이 Medium 이면 3,4 번 서버에 할당
   else{
   index=1; // 사용자 패턴이 Heavy 또는 Light 인 경우
   } //1,2 번 서버에 할당
minMemoryServer(index);최소메모리 사용 서버 구하기

minMemoryServer(num)
{
   if(memory_usage[num]<memory_user[num+1]){
        select_lb_server(memory_usage[num]);}
        else { select_lb_server(memory_usage[num+1]);}
}// 메모리 비교 및 로드 밸런서 서버 선택
```

그림 2. 제안 로드 밸런싱 알고리즘

4. 실험 및 결과

4.1 실험 환경

1) H/W
   - 터미널 서버: 4 대(Xeon 2.4 X 2,1G Memory)
   - 로드 밸런서: 1 대
   - 스토리지, Directory & DB 서버: 각 1 대
2) S/W

- Windows 2003 Server Standard
- SQL Server 2000
- 프로그램: 한글 2005, MS 오피스, 포토샵 7.0

3) 사용자 수: 20 명

터미널 서버 및 스토리지, Directory& DB 서버, 그리고 로드 밸런서는 하나의 Active Directory 로 설정했으며, S1, S2 는 Heavy User 와 Light User 를 위한 서버, S3, S4 는 Medium User 를 위한 서버이다. 그림 3 은 실험을 위해 구성한 시스템 환경을 나타낸 것이다.

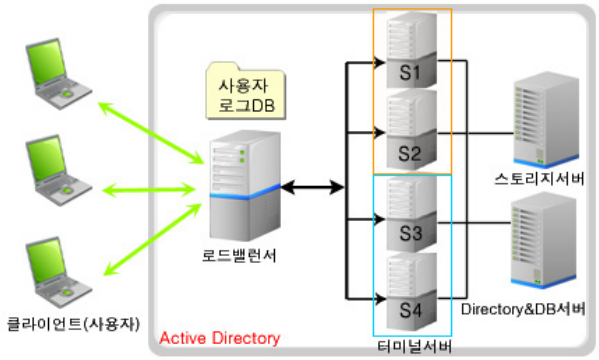

그림 3. 터미널 서비스 시스템 환경

사용자 패턴 분석을 위해 로드 밸런서 서버에 사용자 로그 DB 를 두었으며 사내 직원 20 명을 대상으로 일주일간의 사용량에 대한 로그 데이터를 이용하여 사용자 패턴을 분석하였다. 그림 4 는 추출된 로그 데이터를, 표 1 은 서버에 실험에 이용된 프로그램의 메모리 점유량을 나타낸 것이며, 표 2 는 사용자별 패턴 결과를 나타낸 것이다.


서버별 접속 시간

User01

시작 시간	종료 시간	사용시간 (분)	접속 IP
2/21/05 오후 9:13	2/21/05 오후 9:14	1	10.10.0.52
2/21/05 오후 9:16	2/21/05 오후 9:29	13	10.10.0.52

프로그램 사용량

한글 2005

사용자ID	프로그램 이름	접속 시간	종료 시간
User01	한글 2005	2/21/05 오후 9:13	2/21/05 오후 9:14
User01	한글 2005	2/21/05 오후11:17	2/21/05 오후11:17

그림 4. 추출 로그 데이터

표 1. 프로그램 별 메모리 사용량

프로그램명	사용량	프로그램명	사용량
한글 2005	12MB	MS 파워포인트	9MB
MS 워드	15MB	포토샵 7.0	24MB
MS 엑셀	9MB		

표 2. 사용자 별 패턴 결과

사용자	사용자	사용자	사용자	사용자
1: M	5: M	9: M	13: M	17: H
2: L	6: M	10: M	14: L	18: H
3: H	7: L	11: H	15: M	19: M
4: M	8: H	12: M	16: M	20: M

4.2 실험 결과

실험은 기존의 최소 리소스 부하방식과 제안방식을 비교하였다. 이때, 서버의 부하 측정은 메모리 사용량을 기준으로 하였으며, 사용자 20 명 모두가 서버에 접속한 후 1 시간 간격으로 서버의 메모리 사용량을 측정하였다. 표 3 은 사용자 패턴을 고려하지 않고 세션 연결 당시 서버의 최소 메모리 사용을 우선적으로 로드 밸런싱 했을 때 서버의 전체 메모리 중 사용량을 %로 나타낸 사용률이며 각 서버의 전체 평균값을 보았을 때, S4 가 다른 서버에 비해 상대적으로 많은 메모리 사용률을 보였으며 가장 적게 차지한 S2 와는 10%의 차이가 있음을 알 수 있다.

반면 표 4 는 사용자 패턴에 따라 Heavy 와 Light 사용자가 할당된 S1, S2 서버와 Medium 사용자가 할당된 S3, S4 서버의 메모리 사용률에 대한 변화를 나타낸 것이며 평균값을 기준으로 최소 메모리 사용률을 보인 S2 와 최대 메모리 사용률을 보인 S3 와의 격차가 3%로 줄어듦으로써 기존 방식에 비해 비교적 균일하게 부하 분산이 됨을 알 수 있다.

표 3. 최소 메모리 사용량 우선 로드 밸런싱

서버	0	1	2	3	4	5	6	7	8	평균
S1	46%	49%	53%	53%	51%	59%	60%	56%	56%	54%
S2	**40%**	**47%**	**52%**	**47%**	**47%**	**49%**	**54%**	**52%**	**50%**	**49%**
S3	45%	49%	58%	58%	55%	60%	60%	60%	58%	56%
S4	**47%**	**55%**	**62%**	**59%**	**58%**	**63%**	**63%**	**60%**	**61%**	**59%**

표 4. 사용자 패턴에 따른 로드 밸런싱

서버	0	1	2	3	4	5	6	7	8	평균
S1	43%	49%	55%	55%	53%	58%	58%	58%	56%	54%
S2	**39%**	**47%**	**54%**	**55%**	**50%**	**57%**	**58%**	**59%**	**56%**	**53%**
S3	**42%**	**50%**	**57%**	**58%**	**56%**	**58%**	**60%**	**60%**	**61%**	**56%**
S4	43%	51%	56%	56%	55%	57%	59%	58%	60%	55%

5. 결론 및 향후 과제

본 논문에서는 터미널 서비스 환경에서 클라이언트를 터미널 서버로 연결 시 사용자 패턴을 이용함으로써 터미널 세션의 특징으로 인하여 특정 서버에 가중될 수 있는 부하를 효율적으로 분배하여 기존 시스템의 성능 향상에 기여하였음을 검증하였다. 향후, 보다 정확한 사용자 패턴 분석을 위한 연구 및 터미널 서비스의 QoS 를 향상시키기 위한 연구가 필요하다.

참고문헌

[1] Microsoft, “Windows 2000 Server 구축 및 계획 가이드”, 2002.6.3

[2] 정윤재, 곽후근, 정규식. “씬 클라이언트 환경에서 터미널 서비스를 위한 적응적 서버 클러스터링”,정보과학회 논문지 1 - 정보통신, VOL.31 NO.06 pp.0582 ~ 0594, 2004.12

[3] Microsoft, “터미널 서버를 사용하는 세션 디렉터리와 로드 밸런싱”, 2002.8