CORBA 시스템을 위한 적응적 부화균등화 기법

백승민[0] 김성천
서강대학교 컴퓨터학과
{jenesis[0], ksc}@arqlab1.sogang.ac.kr

Adaptive Load Balancing Method for CORBA System

Seung-min Baek[0] Sung-chun Kim
Dept. of Computer Science, Sogang University

요 약

기존의 정적 부하균등화 기법들은 동적 객체의 내재된 부하와 서버들의 다양한 상태 변화에 따른 능동적 부하분배가 불가능하다. 이를 위해 각 서버에서 프로세스가 자원을 점유하는 비율에 따라 해당 서버의 부하량을 판단하는 새로운 부하할당 기법을 제안하였다. 각 서버의 더미 프로세스는 자신이 프로세서를 점유하는 시점마다 해당 서버의 ID만을 전달함으로써 부하에 따른 차등하게 비동기된 서버보고를 수행한다. 이를 통해 각 서버는 자신의 부하량에 비례하여 부하균등화 참여율을 조정하여 성능저하를 최소화하고 부하량 판단 정보로 활용한다. 결국 제안기법은 웹 서비스의 부하의 성향에 무관하게 지능적인 부하할당을 가능케 한다.

1. 서 론

이질적인 분산 컴퓨팅 환경을 통합하고 산재한 공유 자원을 효율적으로 활용하여 멀티미디어 서비스를 제공할 수 있는 공통 환경을 제공하기 위한 많은 연구가 진행되고 있다. 이중 대표적인 것이 OMG(Object Management Group)가 개발하고 있는 CORBA(Common Object Request Broker Architecture)이다. CORBA는 분산 객체 컴퓨팅 환경의 표준으로서, 그 목적은 이질적 컴퓨팅 환경에 분산되어 있는 소프트웨어 컴포넌트들을 객체화하여 그 인터페이스를 ORB(Object Request Broker)에 등록하면, 클라이언트 프로그램이 ORB를 통해 원격 컴포넌트 객체를 마치 지역 객체처럼 접근하는데 있다.[1,2] CORBA에 대한 표준은 1991년 CORBA 1.1에 IDL(Interface Definition Language)과 ORB를 통한 클라이언트 서버간의 통신 방식이 정의되었고, 1994년 CORBA 2에서는 서로 다른 벤더간에 ORB가 연동할 수 있도록 IIOP(Internet Inter - ORB Protocol)가 정의되었다. 현재는 커포넌트 모델인 CORBA 3까지 정립된 상태이다. ORB는 CORBA 시스템의 핵심적 요소로, 이를 통해 서버와 클라이언트는 완전히 분리될 수 있으며 분산 시스템의 유연성과 확장성이 크게 향상될 수 있다. 하지만, 이러한 CORBA 시스템의 효용성을 위해서 분산 자원들을 효율적으로 관리할 수 있는 자원관리 전략이 필요한데, 그 핵심이 바로 각 자원의 상태를 실시간으로 모니터링하고 그에 따라 부하를 적절히 분배하는 부하균등화 기법이다.

현재 CORBA에서 predictability, security, transactions 그리고 fault tolerance 등을 위한 많은 솔루션들이 제안되고 있으나, 여전히 부하균등화를 비롯한 기본적인 분산 아키텍처 상세 표준이 제안되지 못하고 있는 실정이다. 본 논문에서는 CORBA 환경에서 부하균등화 기법에 대한 현재까지의 표준화 작업에 대하여 기술한 뒤, 그에 대한 확장으로서 구체적인 균등화 기법을 제안한다.

2. 연구배경

· 이 연구는 2001년도 서강대학교 교내 연구비 지원에 의하여 이루어졌음.

2.1 CORBA의 연동 원리

ORB의 구조를 이해하기 위해서 우선 CORBA 객체와 서번트(servant)라는 개념을 이해하여야 한다. CORBA 객체는 CORBA 시스템 안에 등록되어 객체 참조자를 통해 ORB에 의해 검색되고 클라이언트의 연산 요청이 전달되는 ORB 내의 가상의 존재라고 할 수 있다. 반대로 서번트는 CORBA 객체에 정의된 서비스를 제공할 수 있는 서버 내의 실제 프로세스를 말한다. 그림1은 CORBA의 작동원리 전체적으로 묘사한 것이다.

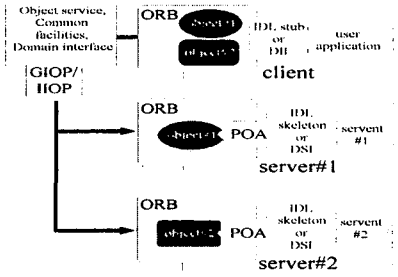

그림 1 CORBA의 연동 원리

2.2 Load Balancing in TAO CORBA

현재 가장 높은 성능과 널리 사용되는 CORBA 시스템으로는 TAO이다. TAO[The ACE(Adaptive Comm. Environment) ORB]는 CORBA 2.6 표준을 따르는 Washinton 대학에서 개발된 open-source의 시스템이다. 그림2는 TAO 시스템에서의 부하균등화 기법을 개괄적으로 나타낸 것이다. TAO 시스템에서의 부하균등화 기법은 adaptive on-demand 방식의 기법이다. 클라이언트들이 우선 부하균등화기에 요청을 보내면, 부하균등화기는 서버들의 상태를 모니터링하고 있다가 최상의 서버를 선택하여 요청을 릴레이하는 동시에 클라이언트에게 해당 서버의 위치를 알려주어 이후 직접적으로 해당 서버에 요청을 직접

적으로 할 수 있도록 한다. 부하균등화기는 지속적으로 서버들을 모니터링하면서 만약 조건이 변화할 경우 클라이언트가 다른 서버에게 요청을 할 수 있도록 관리한다.[3,4]

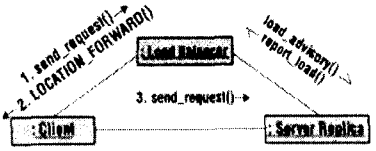

그림 2 Load Balancing in TAO

부하균등화기가 최적 서버를 선택하는 방법에는 여러 가지가 있으나 크게 비적응적 방법과 적응적 방법으로 나눌수 있으며, 전자로는 라운드로빈이나 랜덤선택 방법들이 있으며 후자로는 CPU 사용량과 같은 runtime 정보들을 사용하는 방법들이 있다. 또한 적응적 방법은 정보수집의 방법, 즉 서버 모니터링 방법에 따라 다음과 같이 정리할 수 있다.

동기적 정보수집 기법
(Synchronous Information-Collecting scheme)
-모든 서버는 부하분배기의 정보수집을 위한 동기화에 참여해야 한다
1-1. Demand based
-동기적 정보수집 비용 과다
-모든 요청이 같은 정보수집 비용 감수
-최소 큐 길이의 서버로의 할당 보장
1-2. Periodic-Informing based
-정기적인 동기적 정보수집 비용
-요청 작업 계수보다 정보수집 간격이 작으면 무의미
-정보수집 간격이 길수록, 평균 요청부하의 크기가 작아 서버의 큐 길이의 변화가 클수록 태스크 할당의 정확도가 떨어짐

현재 TAO를 위한 부하균등화 기법에서 각 서버들의 상태 정보수집 기법은 명확하게 제시되지 않았을 뿐 아니라, 그 성능 또한 구체적으로 증명되지 않은 상태이다. 또한 위와 같이 전통적인 동기적 정보 수집 방법은 그 효용성이 극히 제한적이라는 것을 알 수 있다. 본 논문에서는 기존의 불명확한 부하균등화 기법을 대체할 새로운 방식의 균등화 기법을 제한한다. 제한 기법은 각 서버의 상태에 따라 차등하게 비동기된 정보 수집을 바탕으로 부하균등화를 수행한다.

3. 비동기적 차등 정보수집 기법(AGIC)

제안 기법은 기존의 동기적 정보 수집의 단점들을 수정하면서도 보다 정확한 부하균등화를 이루기 위한 비동기적 정보수집 기반 기법이다. 기존 동기적 기법들의 가장 큰 문제는 모든 서버가 정보수집을 위해 조건에 따라 서비스를 중지해야 한다는 것과 수집한 정보 자체또한 그 정확성 및 이용 방법이 모호하다는 점이다. 결국 기존 동기적 기법들은 그 비용에 비해 이득이 제한적이라는 문제가 있다.

비동기적 차등 정보수집 기법
(Asynchronous Graded Information-Collecting scheme)
-시스템 전체의 동기화에 따른 정보수집 비용이 없다
-과부하 서버와 저부하 서버가 서로 차등적으로 부하분배기의 정보수집에 참여
-최소 부하량의 서버로의 할당을 최대한 보장

제안기법의 주 아이디어는 각 서버가 그 부하에 따라 차등적으로 부하균등화에 참여한다는 것이다. 고부하 서버일수록 현재 요청 할당과는 무관한 반면, 저부하 서버들은 해당 요청을 서비스하기 위해 경쟁해야 한다. 이러한 의미로 제안기법을 '비동기적 차등 정보수집 기법'이라 할 수 있다. 제안 기법의 구현은 매우 간단히 다음과 같은 작업을 수행하는 프로세스를 각 서버가 수행하므로써 이루어질 수 있다.

서버보고 프로세스(Server State Reporting Process)
1. 자신의 프로세서 점유 간격을 계산하고, 사전에 정의된 비율에 따라 해당 서버의 id를 부하분배기에 보고한다.
2. 자신에게 할당된 타임슬라이스를 다음 프로세스에게 바로 양보한다.

부하분배기는 각 서버의 비동기적 차등 보고를 받아 분석하여 가장 잦은 서버보고를 수행하며 보고 빈도가 일정한 서버에게 요청을 할당하도록 한다.

4. 비동기적 차등 정보수집 기법 분석

부하분배 기법의 효율성은 정보수집 비용과 부하균등화의 정확도 사이에 최적의 합의점을 찾는데 있다. 그러므로 각 서비스 요청에 대하여 이러한 정보수집 비용과 각 서버 사이의 작업 큐길이의 분산(variance)을 최소화할 수 있어야 클라이언트에게 타당한 서비스 응답 시간을 보장해 줄 수 있다. 이런 관점에서 본 장에서는 각 기법의 성능분석을 각 서비스 요청에 대하여 요구되는 응답 반환시간(turnaround time)의 모델링을 통하여 수행하고자 한다. 각 서비스 요청의 응답 반환시간 tt는 다음과 같이 일반적으로 정의될 수 있다.

tt = 부하분배기의 부하할당 서버결정 및 요청전달 시간 + 요청 서비스 수행 시간

1.부하분배기의 부하할당 서버결정 및 요청전달 시간
= 작업 큐에서의 대기시간 + 현재 부하할당을 위한 정보수집 비용 + 서버결정 알고리즘 수행시간 + 요청패킷 전달 시간
2.요청 서비스 수행 시간
= 요청 서비스의 작업시간(부하량) + 작업 큐에서의 대기시간 + 해당 서버의 서버보고 비용

우선 동기적 정보수집 기법에는 부하분배기가 각 서버에 정보수집 요청을 하고, 다시 각 서버가 부하분배기에 응답해서 최종적으로 부하분배기가 전체 시스템 정보를 수집하는 비용이 요구된다. 이러한 비용을 SO(Sync. Overhead)라 하면 이값은 부하분배기와 각 서버간의 통신 충돌이 없다면, n(서버수)에 비례하여 예측 가능한 범위의 값으로 가정할 수 있다. Demand based 기법에서는 모든 요청에 대하여 정보수집 작업을 요하므로 평균 정보수집 비용은 SO가 된다. 반면, Periodic-Informing based 기법에선 요청이 부하분배기에 어느 시점에 도착했느냐에 따라 다른 비용이 요구된다. 만약 요청이 정기적 정보수집 작업 중에 도착했을 경우, 요청 부하의 서버 할당은 정보수집 작업 종료시까지 대기하게 된다. 정보수집 간격이 p라면 요청 할당 시점에 정보수집 작업이 필요한 확률은 SO/(p+SO)가 되며, 대기 시간의 기대치는 SO/(p+SO)×SO가 된다. 한편, 요청이 정보수집 작업 외의 시간에 도착할 경우에는 부하할당은 전에 수집된 정보를 바탕으로 이루어지므로 정보수집 비용은 0이다.

반면에, 비동기적 차등 정보수집 기법에서는 정보수집을 위한 전체 시스템의 동기 과정이 불필요하므로 SO와 같은 동기 비용은 존재하지 않는다. 하지만 현재 서버보고를 받는 중이라면 서버 할당은 대기해야 한다. 이 대기 시간은 약 ack/4(=ack/2×1/2)이 된다. 물론 그 가능성을 50%(1/2)로 가정하였지만, ack

가 실제로 동기 비용에 비해 매우 작은 값이므로 전체 대기시간에 큰 영향을 미치는 요소는 아니라고 할 수 있다. 이러한 정보수집을 위한 동기비용은 현재 요청에 대한 서비스 응답 시간에 무시할 수 없는 부담으로 존재한다. 특히 부하분배기 기반의 웹서버 시스템에서 가장 큰 문제인 부하분배기 병목현상을 가중시키는 요소이다. 그러므로 제안기법에서 이러한 동기비용 제거는 기존 기법에 비하여 큰 장점을 갖는다. 이것은 부하분배기의 작업큐의 평균 길이(bq)에 바로 반영되게 된다. bq의 값이 작을수록, 즉 부하분배기에서 요청이 대기하는 시간이 짧을수록 더욱 빠른 응답시간을 보장할 수 있다. 서버결정 알고리즘 수행시간은 세 기법 모두 서버수와 같은 정보 양을 유지하고 있고 최소 부하의 서버를 찾는 간단한 알고리즘이므로, 서버수를 n이라 할 때, 알고리즘의 시간복잡도는 O(n)으로 같다고 할 수 있다. 또한 요청 패킷의 서버 전달시간도 세 기법 모두 같은 네트워크 기반구조의 클러스터 시스템 상에서 구현되므로 같은 비용(Ptx)으로 가정할 수 있다.

제안 기법에서의 부하할당을 위한 서버결정 알고리즘은 다른 기법과는 약간 다르게 구현된다. 기존 기법에서 사용하는 정보는 일정 시점의 각 서버의 큐 길이인데 반해, 제안기법인 비동기적 차등 정보수집 기법에서 사용하는 정보는 현재부터 과거의 일정 시점까지의 차등적인 비율로 부하분배기에 보고해 오는 서버보고의 빈도이다. 이러한 정보는 한 시점의 큐길이가 아닌 그 변화도를 추정하는 것으로 측정방법은 간접적이지만 보다 간단하고 정확하게 해당 서버의 부하량의 추이를 추정할 수 있다. 한가지 주목할 것은 단순 빈도 비교만으로 각 서버의 부하량의 변화를 추정할 수 없다는 점이다. 만약 과거 일정 시점까지의 서버보고의 회수나 그 평균간격이 같다고 해서 부하량의 변화가 같다고 할 수 없기 때문이다. 최근에 변화가 많을수록 보다 부하량이 작을 가능성이 농후하기 때문이다. 그러므로 서버결정 알고리즘에서는 최근의 변화에 더욱 높은 가중치를 두어 각 서버의 부하빈도를 비교하게 된다.

하나의 요청은 서버에서 자신의 부하량 이외에 부가적인 오버헤드를 갖는데 그것은 작업 대기 큐에서의 대기시간과 시스템 인터럽트 시간이다. 이러한 대기 시간은 각 서버의 작업 스케쥴링 정책에 따라 다르지만, 본 논문에서는 서버의 작업 스케쥴링 정책을 RR기법으로 가정한다. 그리고 각 서버의 평균 큐 길이를 q, 각 요청의 평균 작업시간을 jt라고 하고, ts를 RR 스케쥴링의 타임슬라이스 크기라고 하면, 각 요청의 평균 큐 대기시간은 q×ts×⌈ jt/ts ⌉ 이다. 우리는 계산의 편의를 위해 서버 자원을 프로세서에 국한하며, 각 프로세스는 타임슬라이스를 모두 유효하게 소비한다고 가정한다. 시스템 인터럽트의 경우는 각 서버의 상태에 따라 다양하게 생성된다. 하지만 본 논문에서는 인터럽트 오버헤드를 부하분배기의 정보수집을 위한 각 서버의 서버보고에 관련된 시스템 인터럽트로 국한한다. 동기적 정보수집 기법의 경우 이러한 서버보고 인터럽트 오버헤드는 각 요청이 얼마나 작업 대기큐에 오랫동안 대기하는가에 비례하여 증가하게 된다. 그러므로 요청의 작업시간과 대기시간의 합을 동기적 정보수집 간격으로 나눈 값만큼 각 요청은 해당 서버의 서버보고로 인한 지연을 감수하게 된다. 이때 이러한 지연에 대하여서도 서버보고의 오버헤드가 증가할 수 있으나 편의를 위하여 무시한다.

그러므로 정기적 정보수집 기법에서는 각 요청이 각 서버에 머무는 시간에 비례하여, 정보수집 부담이 증가하게 된다. 이런 총 서비스 시간은 해당 서버의 평균 큐길이와 각 요청의 부하량에 비례하는데, 스케쥴링 기법의 효율성과 관계되는 것은 평균 큐길이이다. 결국 정기적 정보수집 기반 부하할당 기법에서는 해당 서버가 과부하일수록 해당 서버의 요청들은 더욱 가중한 정보수집 부담을 감수해야 한다. 하지만 비동기적 차등 정보수집 기법에서는 각 요청은 큐길이와는 무관한 정보수집 비용을 부담하게 된다. 제안기법에서 각 서버는 자신의 큐길이에 비례하여 정보수집 프로세스를 실행하기 때문이다. 다시말해 각 요청들은 자신이 필요로 하는 프로세서 타임슬라이스 수만큼 큐에서 순환하게 되는데, 각 순환에 한번씩만 해당 서버가 서버보고를 수행하기 때문이다. 지금까지 각 기법의 응답 반환 시간을 수학적으로 분석한 것을 다음과 같이 최종 정리할 수 있다.

1. Demand based (λ: 작업계수)

$$tt = (bq+1)\times(SO + O(n) + Ptx) + jt + q\times ts\times\lceil jt/ts \rceil + ((jt + q\times ts\times\lceil jt/ts \rceil)/\lambda)\times ack$$

2. Periodic-Informing based (k = p + SO)

$$tt = (bq+1)\times(SO/(p+SO)\times SO + O(n) + Ptx) + jt + q\times ts\times\lceil jt/ts \rceil + ((jt + q\times ts\times\lceil jt/ts \rceil)/k)\times ack$$

3. Asynchronous Graded Information-Collecting

$$tt = (bq+1)\times(ack/4+O(n) + Ptx) + jt + q\times ts\times\lceil jt/ts \rceil + \lceil jt/ts \rceil \times ack + ack/4$$

5. 결 론

위와 같은 분석을 통하여 우리는 제안기법인 비동기적 차등 정보수집 기법에서 각 요청에 대하여 정보수집 비용은 큐길이에 무관한 단지 ⌈ jt/ts ⌉ ×ack+ack/4 임을 알 수 있다. 이 비용은 모든 요청에 대하여 고정 값을 갖는 특징이 있다. 기존 기법에서는 각 서버의 큐길이가 길수록 각 요청의 서버에서의 대기 시간이 길어지게 되며 그에 비례하여 정보수집 부담이 증가하지만, 제안 기법에서는 일정한 부담만을 가진다. 이점은 각 서버 차원에서 보다 큰 의미가 있는데 그것은 과부하의 서버는 그만큼 정보 수집 비용에 참여하는 회수가 적어진다는 점이다. 그 이유는 큐길이가 길수록 서버보고의 주기가 길어지게 때문이다. 또한 사전 동기화로 인한 정보수집 비용에 비해 각 서버의 비동기적 서버보고의 비용은 매우 작다. 모든 것을 고려할 때 제안기법의 정보수집으로 인한 오버헤드는 고정적이지만 기존 기법에 비해 매우 작다는 것을 알 수 있다. 하지만 위와 같은 정보수집을 위한 오버헤드의 비교만으로 각 기법의 효율성을 비교하는 것은 문제가 있다. 세 기법 모두 부하분배기(bq)와 각 서버(q)에서의 평균 큐길이에 평균 응답반환 시간이 크게 좌우되기 때문이다. 그러나 일반적으로 부하분배기 기법에서 후자보다는 전자가 성능에 더 큰 영향을 미친다는 것과 제안기법의 스케쥴링 정확성을 타 기법에 비해 떨어지지 않는다고 가정할 때, 타 기법에 비해 정보수집 비용의 감소는 자명하다고 할 수 있다.

6. 참고문헌

[1]Steve Vinoski, "CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments", *IEEE Communications Magazine*, Feb. 1997.

[2]김석원, "CORBA 구조 및 제품현황", *정보과학회지*, 제17권 제7호, pp. 5-14, 7월 1999년.

[3]Ossama Othman, Carlos O'Ryan, and Douglas C. Schmidt ,"Strategies for CORBA Middleware-Based Load Balancing", *IEEE DS Online*, Vol2, Num.3, Mar. 2001

[4]Ossama Othman, Carlos O'Ryan, and Douglas C. Schmidt, "Designing an Adaptive CORBA Load Balancing Service Using TAO", *IEEE DS Online*, Vol2, Num.4