CORBA 로드밸런싱 서비스를 사용한 범용 클러스터링 시스템

차원팔 최경호 김정선
한양대학교 컴퓨터공학과
{hccha8, choilh, jskim}@cse.hanyang.ac.kr

General-purpose Clustering System Using Standard CORBA Load Balancing Service

Hyunchel Cha Jungho Choi Jungsun Kim
Dept. of Computer Science and Engineering, Hanyang University

요 악

인터넷과 개인용 컴퓨터가 대중화되어 사용자들이 늘어나고 있으며, 컴퓨터 기술이 더 이상에 달려오며 대량의 정보를 안되며 전문가의 지식을 빠르게 얻게 되어 큰 효과적이라는 것이 인기를 끌고 있다. 하지만 기존의 클러스터링 시스템의 사용이 높은 성능을 바탕으로 특징 플랫폼의 중앙처리기가 아닌 특정 플랫폼에서 데이터 전송으로 사용하기도 전부지했다. 특히 점보 및 프로토콜 별도의 플랫폼에 의존적인 시스템은 사용이 복잡하고 비속성의 제약은 매우 대단하다. 이에 따라 규격화를 위해 고려된 규격과 교육을 알기 쉽게 유지할 수 있는 방법이 요구되며 이러한 시스템을 효과적으로 구현하기 위해서는 표준 기반의 COTS (commercial-off-the-shelf) 미들웨어의 적용이 반드시 필요하다.

본 논문에서는 이에 따른 문제를 본선체계에서의 프론트엔드의 구성 요소로 CORBA 로드밸런싱 서비스를 이용하여 유지보수 및 확장성이 용이하고 빠른 성능과 동시에 서비스가 가능하게 하여 서버 종류의 여덟개기서의 수행을 위한 클러스터 노드의 동적 추가/삭제가 가능한 범용 클러스터링 시스템을 설계하고 구현한다.

1. 서론

지난 수년간 컴퓨터의 보급이 급속도로 이루어지고 있으며, 학교나 기관 단체의 경우, 수천에 수백 대의 컴퓨터들이 사용되고 있다. 이와 더불어 네트워크의 발달로 인해 속도도 빨라지며 비용을 점차 감소하고 있다. 이런 네트워크 기술의 성장으로 인해 그 활용분야가 넓어지고 사용자가 많아지며 따라 네트워크 트래픽이 100% 성장을 기록하고 있다. 이에 따라 서비스의 처리능력도 개선되어 가고 있지만, 실제적으로 처리능력의 개선이 힘들다고 한다.

처리능력을 개선할 수는 여러 방법이 있는데, 본선에 가깝게agy하게 서버들을 계속 교체하는 것은 비용이 낮아지려면 단점이 있다. 다른 방법은 기존에 사용하던 수의 서버를 이용하여 클러스터로 묶어 많은 작업을 분산해서 처리하는 것이다. 이 방법은 많은 비용으로 높은 성능을 얻을 수 있으므로 현실성 뿐만이 없지 않을 수 있다.

하지만, 다양한 서버와 다양할 수 있는 플랫폼이 존재하고, 어떤 서버가 나아지게 되면 배치에서 특정 플랫폼 혹은 특정 어플리케이션을 중요한 클러스터링 시스템의 비속성에 더 없다. 그러므로 본 논문에서는 CORBA를 이용하여 유지보수 및 확장성이 용이하고 다양한 플랫폼에서 사용이 가능하며 여러 종류의 어플리케이션의 수행을 위한 클러스터 노드의 동적 추가/삭제가 가능한 범용 클러스터링 시스템을 설계하고 구현한다. 2장에서는 클러스터링 시스템 개선에 있는 CORBA에 대해서 서술하고, 3장에서는 CORBA 기반의 범용 클러스터링 시스템 설계 및 구현 방법, 4장에서는 실험 결과를 바탕으로 5장에서 제안한 결과를 정리하여 결론을 끝낸 것으로, 병렬처리 또는 배치, 배분, 고정 등의 서버의 제어적으로 다시하기 위한 방법이다. 일반적으로 클러스터링(Clustering)의 개념이 서버체계에서 인증받는 경향이 있으나, 클러스터링(Clustering)은 PCA와 클러스터링 및 요인 기술의 인증라인을 사용할 수 있다. 클러스터링(Clustering)의 이상으로는 단일의 PCA를 통해 처리하고, 그 결과 저장처리 등을 수행하고, 주로 오류신호 기반의 운영체제와 소프트웨어를 사용하여, 각각 대 성능에서 기존의 대량 컴퓨터를 능가하고, 압고 데이터 확장성이 우수하다는 것이다. 특성을 살펴보면

- 확장성 : 여러 서버를 클러스터링하여 서버 시스템을 구성하게 되며, 적은 비용으로 고성능을 구현할 수 있게 많은 수의 클라이언트를 지원할 수 있다.
- 높은 성능성 : 클러스터링의 높은 성능적 특성에 한정된 서버의 배치에 가비지가 발생하는 다른 서버를 이용하여 클라이언트로 서버 간 동작을 수행할 수 있다. 클러스터링 관리자로서 동작한 클라이언트를 처리할 수 있다. 부하분산(로드밸런싱) 기능을 제공하기 때문에 서버 클러스터링의 할당을 통해 하위 클라이언트에게 서버 클러스터링을 제공할 수 있다. 특성 서버와의 관리 개선을 실현

2. 기술 동향 및 관련 연구

2.1 클러스터링 시스템

클러스터링(Clustering)이란 여러 대의 서버를 연결하여 가변적인 용량을 조정하기로, 특정 서버에 문제가 발생했을 경우로 서버를 대체하는 것이 일례이다. 즉, 높아진 컴퓨터들이 하나의 컴퓨터처럼 동작할 수 있도록 연결하는 것으로 병렬처리 또는 배치, 배분, 고정 등의 서버의 제어적으로 다시하기 위한 방법이다. 일반적으로 클러스터링(Clustering)의 개념이 서버체계에서 인증받는 경향이 있으나, 클러스터링(Clustering)은 PCA와 클러스터링 및 요인 기술의 인증라인을 사용할 수 있다. 클러스터링(Clustering)의 이상으로는 단일의 PCA를 통해 처리하고, 그 결과 저장처리 등을 수행하고, 주로 오류신호 기반의 운영체제와 소프트웨어를 사용하여, 각각 대 성능에서 기존의 대량 컴퓨터를 능가하고, 압고 데이터 확장성이 우수하다는 것이다. 특성을 살펴보면

- 확장성 : 여러 서버를 클러스터링하여 서버 시스템을 구성하게 되며, 적은 비용으로 고성능을 구현할 수 있게 많은 수의 클라이언트를 지원할 수 있다.
- 높은 성능성 : 클러스터링의 높은 성능적 특성에 한정된 서버의 배치에 가비지가 발생하는 다른 서버를 이용하여 클라이언트로 서버 간 동작을 수행할 수 있다. 클러스터링 관리자로서 동작한 클라이언트를 처리할 수 있다. 부하분산(로드밸런싱) 기능을 제공하기 때문에 서버 클러스터링의 할당을 통해 하위 클라이언트에게 서버 클러스터링을 제공할 수 있다. 특성 서버와의 관리 개선을 실현

2.2 CORBA의 구조

본 논문은 CORBA의 구조로서 서버 클러스터링 시스템을 연구하는 것이 바람직하다. CORBA(Common Object Request Broker Architecture)는 서버 클러스터링의 구조를 기반으로 하는 것이 바람직하다. 높은 성능의 클러스터링 시스템에 대해 CORBA는 개발이 되어 있으며, 서버 클러스터링 시스템의 사용이 증가하고 있는 추세이다. 약 800개 이상의 컴퓨터들을 연결한 MDS(Object)
3.4 Facade Job Manager

Facade Job Manager의 구성 요소는 그림 3과 같다. Job Generator는 클러스터링 시스템 내에서 유달하게 생겨나는 Job 아이디를 생성하고 우선순위를 지정하여 클러스터링으로부터 제출한 데이터를 통합하여 Execution Job Manager에게 제출한다. Completion Job Table Manager는 탐색해 작업에 대한 데이터를 유지하며, 클러스터링 측은 관리자와 요청자 측 제출된 작업의 상태를 보고한다.

Completion Job Handler는 특정 어플리케이션의 실행결과를 이벤트 채널을 통해 수신하여 클러스터링과 연동하는 방식으로 어플리케이션의 수행결과를 전달한다.

3.5 Execution Job Manager

Execution Job Manager의 구성요소는 그림 4와 같다. Job Acceptor는 Facade Job Manager로부터 Job을 접수하며, Job을 Job Table Manager에 저장한다. Job Table Manager는 Job Table을 관리하며, 각 Job의 저장 및 삭제 및 처리 상태를 관리한다.

Job Scheduler는 Job의 우선 순위(priority)에 따라 분배한 Job을 선택한다. Job Distributor는 Job Scheduler가 선택한 Job을 포드 발리 서를 통해 클러스터링 노드로 보내며, 전체적으로 Job을 집합되며 분배되는 순서는 그림 4와 같다.
3.6 Load Balancer


본 클러스터링 시스템에서 지원하는 로드 밸런싱 정책은 표 1과 같다.

![그림 5. Load Balancer와 Cluster Node]

<table>
<thead>
<tr>
<th>정책</th>
<th>내용</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round Robin</td>
<td>자바로 클러스터 노드에게 분배</td>
</tr>
<tr>
<td>Random</td>
<td>알리의 클러스터 노드에게 분배</td>
</tr>
<tr>
<td>Least Loaded</td>
<td>CPU 사용량이 가장 적은 클러스터 노드에게 분배</td>
</tr>
<tr>
<td>Memory</td>
<td>Disk 사용량이 가장 적은 클러스터 노드에게 분배</td>
</tr>
<tr>
<td>Network</td>
<td>Memory 사용량이 가장 적은 클러스터 노드에게 분배</td>
</tr>
<tr>
<td>Customized</td>
<td>요청 상태의 클러스터 테이블을 기반으로 Random 방식으로 분배하기로 한 클러스터 노드가 bus이 상태가 되면 분배를 다시</td>
</tr>
</tbody>
</table>

3.7 Cluster Node

클러스터링 시스템의 실행 중에 클러스터 노드가 등록되고 실행이 종료된 후에 삭제하도록 함으로써 동작에 추가적인 시간이 없다.

본 클러스터 노드에서는 특정 아름다운 셋을 실행할 수 있는 모듈을 만들고 재설날은 Job을 수행할 수 있도록 작성한다. Job이 완료되면 수행 결과는 Facade Job Manager의 Completion Job Handler에 전송한다.

4. 실험 및 결과 분석

4.1 실험환경

운영체계 : Windows2000, Linux, Solaris

어플리케이션 : 레이던스 엔진(Radiance Engine)

4.2 실험결과

현재 개발한 PC와 서버에서 많이 사용되는 Windows2000과 Linux, PC를 SunOS를 Solaris를 이용하여 본 시스템을 테스트해 보았다. 레이던스 엔진(Radiance Engine)은 미국 Lawrence Berkeley Laboratory(LBL)에서 개발한 프로그램으로 빌딩의 비밀리적으로 시뮬레이션 결과로부터 도출 및 추도 분석을 계산하고 가시화 하여주며 보행의 정량적, 정성적 평가를 하는 데 사용이 되어진다[9]. 본 실험에서는 Linux와 Solaris 운영체계에 PC와 서버에서 클러스터 노드들의 동작 아름다운 셋으로 레이던스 엔진을 사용하고 클러스터 노드의 동작 추가/삭제가 가능한 테스트로 향후 Windows 2000에서 Execution Job Manager를 실행한 것으로, 다양한 품질에서 다양한 정량적으로 로드 밸런싱이 가능하다고 테스트해 보고 가능성을 확인하였다.

5. 결론 및 향후 연구 과제

다양한 품질과 다양한 종류의 아름다운 셋이 수행될지 모르는 상황에서 특장 품질� 특정 특장 아름다운 셋에 종속된 클러스터링 시스템은 유용성을 확장성에 제약을 받게 된다. 본 논문에서는 품질과 특장 아름다운 셋에 영향을 받지 않고 다양한 품질로서 클러스터링 시스템을 손쉽게 사용할 수 있도록 복용 클러스터링 시스템을 설계하고 구현하였다. 향후계획로서 기존 시스템에 클러스터링 시스템의 상황을 효율적으로 관리하기 위한 모니터링 시스템과 실행상태가 지속적으로 클러스터링 시스템을 복구할 수 있는 기능을 추가한다면 더욱 효율적인 클러스터링 시스템을 이룰 수 있을 것으로 기대된다.

참고문헌


![표 1. 로드 밸런싱 정책]