클러스터 DBMS를 위한 노드상태 모니터링 도구 및 부하 분산 관리기의 설계

남화진[1] 장재우
전 북 대 학 교 컴 퓨 터 공 학 과
{hjnam, jwchang}@dblab.chonbuk.ac.kr

Design of Node Monitoring Tool and Load Balancer for Cluster DBMS

Hwa-Jin Nam[1] Jae-Woo Chang
Dept. of Computer Engineering, Chonbuk National University

요 약

다수의 단일노드를 고속의 네트웍으로 연결하여 단일 시스템처럼 인식할 수 있는 클러스터 DBMS 를 효율적으로 관리하기 위해서는 클러스터 관리 도구가 필요하나 관련 연구는 미흡한 실정이다. 본 논문에서는 클러스터 DBMS 를 위한 관리도구로써 노드상태를 모니터링 할 수 있는 모니터링 도구를 설계한다. 아울러 이러한 모니터링 결과를 기반으로 각 노드의 부하를 측정하고 사용자의 요구를 적절히 분산시킴으로써 클러스터 DBMS 의 성능을 향상을 도모할 수 있는 부하 분산 관리기를 설계한다.

1. 서론

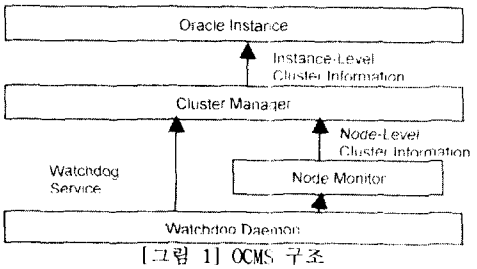
최근 인터넷 사용인구의 지속적인 증가로 단일 대용량 데이터베이스 서버로는 사용자의 접속 요구를 감당하기 어려워지고 있다. 이에 따라 여러 개의 단일 노드를 고속의 네트워크로 연결하는 클러스터 기반 DBMS 에 대한 연구가 활발히 진행중이다. 이러한 클러스터 기반 DBMS 는 대규모 데이터를 여러 노드에 분산 저장하고 일관성 있게 접근할 수 있는 메커니즘을 제공하며 또한 클러스터 시스템의 주요 특징인 고성능(High Performance), 고가용성(High Availability), 고확장성(High Scalability)을 가진다[1,2]. 이러한 클러스터 기반 DBMS 를 효율적으로 관리하기 위해서는 다음과 같은 기능을 가지는 관리도구가 필요하다. 첫째 다수의 노드로 구성된 클러스터 시스템을 단일 시스템처럼 인식할 수 있는 가상의 단일 시스템 환경(Single System Image)을 제공하여야 한다. 둘째, 전체 시스템 구성과 각 노드의 부하 및 자원의 활용상태의 파악이 용이하여야 한다. 셋째, 모든 노드의 성능을 최대한 발휘하기 위해 사용자의 요구를 적절히 분산시키는 스케줄링 방법이 필요하다.

이를 위해, 본 논문에서는 클러스터 기반 DBMS 를 위한 모니터링 도구와 부하분산 관리기를 설계한다. 이는 각 노드들에 대해서 IP 레벨의 단일 시스템 이미지를 제공함으로써 사용자의 트랜잭션이 어느 노드에서 수행 되는 가에 구애를 받지 않는 투명성(Transparency)을 제공한다. 또한 각 노드의 자원상태나 부하의 상태를 실시간으로 모니터링하는 모니터링 도구와, 이를 토대로 각 노드의 부하를 측정하고 사용자의 트랜잭션 요구를 효율적으로 분산하여 클러스터 DBMS 의 성능 향상을 도모하는 부하분산 관리기를 설계한다.

본 논문의 구성은 다음과 같다. 제 2 장에서는 관련연구로서 OCMS 에 대해 소개한다. 제 3 장에서는 본 논문에서 설계하는 시스템의 전반적인 구조와 노드 상태 모니터링 도구의 설계에 대해 설명하고, 제 4 장에서는 부하 분산 관리기의 설계에 대해 설명한다. 마지막으로 제 5 장에서는 결론 및 향후연구를 제시한다.

2. 관련연구

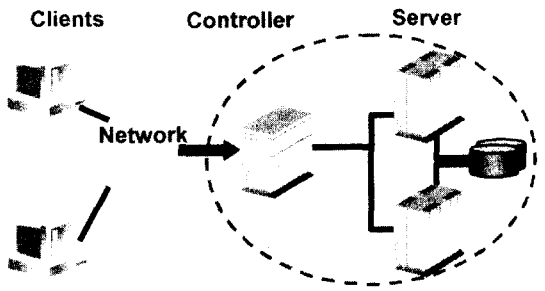
대표적으로 알려진 클러스터 DBMS 용 관리도구인 OCMS(Oracle Cluster Management Software) 는 오라클에서 리눅스용 오라클 8i Enterprise Edition 에 번들로 제공하는 클러스터 DBMS 용 관리도구이다[3]. 이는 Watchdog Daemon, Node Monitor, Cluster Manager 의 3 가지 컴포넌트로 이루어져 있다. Watchdog Daemon 은 각 노드의 하드웨어 리소스들을 모니터링하고, 이러한 정보를 상위 컴포넌트(Node Monitor, Cluster Manager)에 제공하며, 필요시 해당 노드를 reset 하거나 reboot 하는 기능을 제공한다. Node Monitor 는 Ping 메시지를 통해 다른 노드들의 상태정보 데이터베이스를 유지하며, 이를 통해 어떤 노드가 active 한지를 알 수 있다. 이 두 컴포넌트를 기반으로 Cluster Manager 는 클러스터 자원들을 모니터링해 오라클 인스턴스에게 클러스터의 일관된 시각을 제공하며, 인스턴스 간의 통신을 가능케 한다. 그러나, OCMS 는 오라클 8i 라는 특정제품을 위한 관리도구로서 범용적인 클러스터 DBMS 관리도구로 사용할 수 없다는 단점이 있다. [그림 1]은 OCMS 구조를 나타낸다.



[그림 1] OCMS 구조

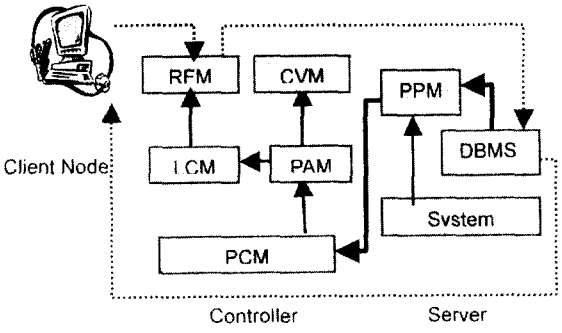
3. 노드 상태 모니터링 도구의 설계

클러스터 DBMS 는 하나의 컨트롤러 노드와 디스크 공유방식으로 여러 개의 서버노드가 클러스터형으로 구성되며 클라이언트는 컨트롤러 노드를 통해 클러스터 DBMS 에 접근하고 컨트롤러 노드는 적절한 스케쥴링 방법에 따라 사용자의 요구를 실제 서버 노드에 전달한다. 본 논문에서 사용하는 클러스터 기반 DBMS 의 구성도는 [그림 2]와 같다.



[그림 2] 클러스터 기반 DBMS 의 구성도

클러스터 DBMS 의 모니터링은 시스템측면 모니터링과 DBMS 측면 모니터링으로 나눌 수 있는데, 시스템 측면에서는 주요 자원의 사용상태를 모니터링 한다. 이러한 정보는 OS 에서 제공하는 /proc 파일 시스템에서 얻게 되며 CPU 활동상태, Memory 사용상태, Disk 의 마운트 상태, 특정한 프로세서의 수행 상태등을 모니터링 한다[5]. 또한 DBMS 측면에서는 현재 수행중인 트랜잭션의 개수, 접속한 사용자의 수, DBMS 가 수행하는 I/O 횟수, CPU 점유율, 평균 버퍼 캐쉬 적중률 등을 모니터링 하게 된다[4]. 이러한 모니터링 도구은 4 가지의 주요한 컴포넌트로 분류할 수 있는데 그 구성도는 [그림 3]과 같다.



[그림 3] 모니터링툴과 부하분산관리기의 주요 컴포넌트

PPM(Performance Probing Module)은 서버노드에서 시스템과 클러스터 DBMS 의 상태정보를 수집하고 Socket 방식으로 컨트롤러 노드로 전달한다. 컨트롤러 노드에서는 PCM(Performance Collection Module)를 통해 각 노드의 상태정보를 통합, 관리하고, PAM(Performance API Library Module)을 이용하여 각 노드의 상태정보를 Application 레벨에서 접근할 수 있도록 API 를 제공한다. CVM(Control and Visualization Module)은 상태정보를 보여주는 GUI 모듈로써 사용자는 이를 통해 클러스터 노드의 상태를 확인할 수 있다.

4. 부하 분산 관리기의 설계

부하 분산 관리기는 수집된 각 노드의 상태 정보를 기반으로 부하를 계산하고 실행 서버를 선택하는 LCM(Load Computation Module)과 Kernel Level 에서 Direct Routing 방식을 이용해 사용자의 요구를 실행 서버에 전달하는 RFM(Request Forwarding Module)으로 구성된다. Direct Routing 방식은 컨트롤러 노드가 클러스터 DBMS 의 게이트웨이 역할을 담당하고 컨트롤러 노드로 접근하는 사용자 요구 패킷을 실제 서버노드의 MAC 주소를 참조함으로써 전달하는 방식이다. [그림 3]은 부하 분산관리기를 구성하는 컴포넌트와 사용자의 요구 패킷의 경로를 보여준다.

부하 분산 관리기에서는 서버 노드들중 부하가 최소인 노드를 선택하기 위해 부하량과 관련이 있는 상태정보를 4 가지의 부하결정요소로 구분하고 이를 정규화함으로써 부하량을 상호 비교한다. [표 1]은 분류한 부하결정요소를 나타낸다

[표 1] 부하 결정 요소의 분류

부하결정요소		설 명
User Connection & Transaction	# of User Connection	현재 DBMS 에 접속하고 있는 사용자의 수
	# of Transaction	현재 DBMS 가 수행하고 있는 트랜잭션의 수
Disk Activity	Disk time	DBMS 가 일정시간동안 Disk 에 접근하는 횟수
Processor Utilization	CPU usage	DBMS 가 사용하는 CPU 의 점유율
Memory Usage	Available Byte	DBMS 가 사용가능한 메모리의 용량
	Buffer Cache Hit Ratio	평균 Buffer Cache 적중 비율

4.1 부하결정요소의 우선순위 부여과 정규화

본 논문에서는 분류한 부하결정 요소중 트랜잭션 측면 부하를 $Load_{Tran}$, 디스크 측면에서의 부하를 $Load_{Disk}$, 프로세서 측면의 부하를 $Load_{CPU}$, 메모리 측면의 부하는 $Load_{Mem}$ 으로 표현하고 각 요소들에 대해 우선순위를 부여한다. 즉, 클러스터 DBMS 의 부하량에 많은 영향을 미치는 요소에 높은 우선순위를 부여하는 것이다. DBMS 관점에서 볼 때 수행하는 트랜잭션의 수 및 사용자의 수는 DBMS 의 부하량을 전체적으로 평가할 수 있는 가장 중요한 부하결정 요소이다. 또한, DBMS 에서는 I/O 에 관련된 Job 의 수행이 빈번하므로 전체 트랜잭션 수행 비용의 많은 부분을 차지하는 디스크 측면의 부하도 주요 부하결정 요소이다. 본 논문에서는 위의 사항을 고려하여 다음과 같이 우선순위를 부여한다. 그러나 이와 같은 우선순위 부여는 트랜잭션의 수행 형태 및 수행 환경에 따라 유연적으로 변화될 수 있을 것이다.

$$Load_{Tran} > Load_{Disk} > Load_{CPU} > Load_{Mem}$$

또한 각 요소의 값을 0.00 에서 1.00 사이의 값으로 정규화함으로써 상호의 비교가 가능하게 한다. $Load_{Tran}$ 은 DBMS 가 현재 처리하는 트랜잭션 개수와 접속한 사용자의 수를 고려한 부하량이며 정규화 과정은 다음과 같다.

$$Load_{Tran} = (\#\ of\ Transaction\ /\ MAX\ Transactions) * w_1 + (\#\ of\ User\ Connection\ /\ MAX\ User\ Connection) * w_2$$

위의 식에서 MAX Transactions 은 DBMS 가 처리할 수 있는 최

대 트랜잭션의 개수, MAX User Connection 은 DBMS 가 허용하는 최대 사용자 접속수, w_t 는 트랜잭션에 대한 가중치, w_c 는 사용자 접속수에 대한 가중치이고 $0 \le w_t, w_c < 1$ (단, $w_t + w_c = 1$)을 만족한다. 즉, $Load_{Tran}$ 의 값이 클수록 트랜잭션 개수와 사용자 수 측면에서 부하가 큼을 의미한다.
$Load_{Disk}$ 은 Disk I/O 횟수를 고려한 부하량이며 다음과 같이 정규화 한다.

$Load_{Disk}$ = (# of Disk Access time / User Define MAX Disk Access Time)

위 정규화 과정에서 User Define Max Disk Access Time 은 사용자가 정의한 최대 디스크 I/O 횟수이다. $Load_{CPU}$ 는 CPU 측면에서의 부하량을 의미하고 다음과 같은 정규화 과정을 가진다.

$Load_{CPU}$ = DBMS CPU Usage / Total CPU Usage)

$Load_{Mem}$ 는 메모리 측면에서의 부하량을 의미하고, 정규화 과정은 다음과 같다

$Load_{Mem}$ = (1 − Available Byte/MAX Available Byte)* w_b + (1 − Cache Hit Ratio) * w_h

위 식에서 MAX Available Byte 는 실행서버 중에서 사용 가능한 메모리가 최대인 것의 크기, w_b는 사용 가능한 메모리용량에 대한 가중치, w_h 는 Cache 적중률에 대한 가중치이고 $0 \le w_b, w_h \le 1$ (단, $w_b + w_h = 1$) 을 만족한다

4.2 실행서버 선택 알고리즘

부하 분산 관리기에서 서버 노드를 선택하는 방법은 우선순위에 기반한 방식과 가중치에 기반한 방식을 고려 할 수 있다. 우선순위에 기반한 방식은 부하결정요소에 우선순위를 두고 우선순위가 높은 요소부터 부하를 차례로 비교하여 부하가 적은 서버노드를 선택하는 방식이고 가중치에 기반한 방식은 각 부하결정요소에 가중치를 두고 각각의 부하결정요소에 가중치를 곱한 값을 모두 더함으로써 해당 서버노드의 총 부하량을 구하고, 이중 총 부하량이 가장 적은 서버노드를 선택하는 것이다.

본 논문에서는 우선순위 방식과 가중치 방식을 결합한 알고리즘을 사용한다. 즉, 부하결정요소에 우선순위와 가중치를 모두 부여하고, 우선순위가 높은 부하결정요소부터 비교하면서 최소값과의 차이가 임계값 이하인 서버노드만 후보 노드가 되어 다음 우선순위 부하결정요소를 비교한다. 만약 비교하는 모든 서버노드에 대해 최소값과의 차이가 임계값 이상이면 현재 후보노드에 대하여 가중치 방식을 적용하여 총 부하량을 계산하고 이중 가장 적은 부하량을 가지는 서버노드를 선택하는 방식이다. 다음은 제안하는 서버노드 선택 알고리즘이다.

[ALGORITHM : Server Node Selection]

```
ServerNode_Selection()
INPUT : 노드의 부하량
OUTPUT : 선택된 서버 노드의 ID
VARIABLES :
i = 부하결정요소 번호                k = 서버노드의 ID
N = 노드의 개수                      ε = 임계값
W_i = Load[ ] 의 가중치
Load[list[k]]_i = 리스트 k 번째 노드의 요소우선순위가 i 인 부하량
Load[list[k]]_total = 리스트 k 번째 노드의 총 부하량
Prev_len = 이전 우선순위방식에서의 리스트에 등록된 노드의 개수
PROCEDURE :
/****각 노드로 구성된 리스트를 생성한다****/
For all k (0≤ k < N) {
  Make_list(k);
  Prev_len = Len(list);
}
/****모든 부하결정요소에 관하여****/
For all i (0≤ i < 4) {
  /****모든 노드에 대해 최소부하 값을 구한다. ****/
  For all k (0≤ k < Len(list))
    Min_Load_i = MIN(Load[list[k]]_i );
  /****모든 노드값과 최소부하값을 비교하여 임계값***/
  /****이상이면 후보노드 리스트에서 삭제한다****/
  For all k (0≤ k < Len(list)) {
      If((Load[list[k]]_i - Min_Load_i) > ε ) {
          Delete_List(k);
      }
  }
      /**** 모든 노드가 임계값 이상이면 우선순위****/
      /**** 방식을 종료하고 가중치방식을 적용한다**/
  If (Len(list) == 1) {
      For all k( 0≤ k < Prev_len)
          For all j ( 0 ≤ j ≤ i)
              Load[list[k]]_Total += Load[list[k]]_j * W_j;
        /**** 총 부하량이 최소인 노드를 선택한다****/
      For all k( 0≤ k < Prev_len)
            Select k such as Load[list[k]]_Total is Minimum
      Break;
  }
  else
      Prev_len = Len(list);
}
```

이를 통해 클러스터 DBMS 의 각 서버노드는 균등한 부하상태를 유지함으로써 클러스터 DBMS 의 전반적인 성능향상을 가져올 수 있다.

5. 결론

본 논문에서는 클러스터 기반 DBMS 의 효율적인 관리를 위한 모니터링 도구와 부하 분산 관리기를 설계하였다. 이는 IP 레벨에서 클러스터 DBMS 를 가상의 단일 시스템으로 구성하는 방법과 각 서버노드의 상태를 시스템측면과 DBMS 측면으로 나누어 모니터링을 수행하는 모니터링 도구를 설계하였다. 또한 이러한 서버노드의 상태 정보를 4 가지의 부하 결정요소로 분류하고, 정규화 과정을 통하여 부하량을 결정하여 우선순위 방법과 가중치 방법을 결합한 방식을 통해 최적의 실행 서버를 선택하는 부하분산 관리기를 설계하였다.

향후 연구로는 본 논문에서 설계한 모니터링 도구와 부하분산 관리기를 구현하고 다양한 실험을 통해 각 부하결정 요소별로 적절한 가중치를 결정하고 이를 바탕으로 성능 평가를 수행하는 것이다.

6. 참고 문헌

[1] 김진미, 온기원, 김학영, 지동해 , " 클러스터링 컴퓨팅 기술" , 1999

[2] Rajkumar Buyya , High Performance Cluster Computing Vol 1,2

[3] "Oracle 8i Administrator's Reference Release3(8.1.7) for Linux Intel", chapter 7 Oracle Cluster Management Software

[4]" Windows NT Performance Monitor" , Microsoft Corporation

[5] Cluster Computing White Paper , http://www.ieeetfcc.org