실시간 네트워크 모니터링 기반 분산/병렬 컴퓨팅의 작업 할당 전략

정재홍° 김수자 박복자 송은하 정영식
원광대학교 컴퓨터 공학과
{jhjeong°, sujkim, ppojja, ehsong, ysjeong}@wonkwang.ac.kr

Task Allocation strategy for Distributed/Parallel Computing based on Realtime Network Monitoring

Jae-Hong Jeong° Su-Ja Kim Bock-Ja Park Eun-Ha Song Young-Sik Jeong
Dept. Computer Engineering, Wonkwang University

요 약

인터넷 가반 분산/병렬 처리 프레임 워크 PDP(Parallel/Distributed Processing Scheme on Web)는 네트워크 내 유휴 상태 호스트들을 활용하여 대용량 작업을 병렬로 처리한다. 본 논문에서는 이러한 서브 작업을 할당받는 자원이 동작하는 네트워크 환경을 모니터링 함으로써 수시로 변화하는 네트워크 환경에 대처하는 방안을 제시한다. 특히 네트워크 환경 모니터링 예측 결과를 PDP의 작업 할당 알고리즘에 적용하여 네트워크 과부하 및 결함 등으로 인해 발생되는 작업 지연 요소에 적응적 대처함으로써 전체 작업 수행 처리율 향상을 도모하는 방법을 제안한다.

1. 서 론

네트워크 기술의 발전으로 사용자들의 수가 기하급수적으로 증가하고 있다. 이에 따라 사용자들의 요구사항 또한 질적 양적으로 높아져 응용 분야가 넓어 졌으며, 요구되는 계산량도 늘어나게 되어 높은 컴퓨팅 파워가 필요하게 되었다. 그러나 고가의 슈퍼컴퓨터보다 저가의 다수의 컴퓨팅 자원을 사용하여 대용량 작업량을 지닌 어플리케이션을 분할하여 수행하는 병렬 처리로 인해 빠른 시간에 작업을 마칠 수 있게 되었으며, 전체 작업의 수행시간을 단축시킬 수 있게 되었다.

그러나 이러한 대용량 어플리케이션을 처리하는데 있어서 대량의 연산과 데이터 전송이 필요하다면 네트워크 대역 이용률에 따라 데이터 전송의 지연으로 인한 전체 작업의 수행 시간에 영향을 미친다. 본 논문에서는 이로 인해 발생할 수 있는 영향을 최소화하기 위해 어플리케이션의 분산/병렬 처리 연산을 수행하는 컴퓨팅 자원의 성능뿐만 아니라 자원들의 네트워크 과부하 및 결함에 대처하여 효과적으로 작업을 재할당하여 효율적인 작업 처리율을 향상시키기 위한 RNM에 대해 제안한다.

2. 분산/병렬 프레임워크

PDP(Parallel/Distributed Processing Scheme on Web)는 네트워크 환경에 있는 유휴 컴퓨팅 자원을 활용하여 많은 연산 수행을 필요로 하는 이미지 프로세싱을 수행하는 분산/병렬 프레임워크이다. PDP의 구성은 작업의 수행을 요청하는 요청자와 작업 수행에 참여하는 호스트(이하 작업에 참여하는 컴퓨팅 자원을 호스트라 칭함)와 요청된 작업의 분할과 분할된 작업을 호스트에 할당하고 호스트를 관리하는 관리자이다.

PDP는 작업 수행에 참여한 호스트에 성능 기반 작업 할당 알고리즘을 적용하고, 적응적 작업 재할당 기법을 제시하여 작업 부하를 줄인다. 또한, 연산 환경의 자율성에 따른 호스트들의 추가 및 이탈을 위한 동적 작업 프로세서 관리 기법을 제공하고 있으며, 그림 1은 PDP의 수행 환경 및 동작이다.

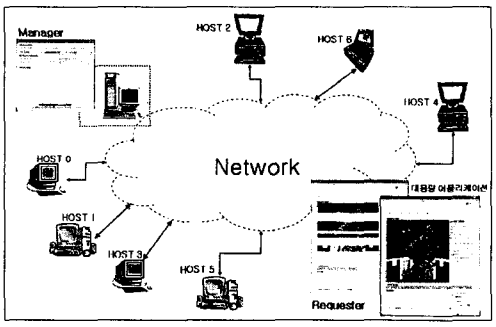

그림 1 PDP 수행 환경 및 동작

3. 네트워크 모니터 설계 및 구현

3.1 모니터링 자원 요소 결정 및 추출

본 논문에서 제시하고자 하는 모니터링을 위해서 호스트의 네트워크 대역폭(bandwidth)과 대역폭 이용률을 계

* 본 연구는 2003년 정보통신부에서 지원하는 기초기술연구지원사업(C1-2003-A1-2000-0013)으로 수행되었음.

산하여 호스트들의 네트워크 환경을 모니터링한다. 네트워크 대역폭 이용률을 계산하기 위한 네트워크 모니터로 RNM-PDP(RealTime Network Monitor for PDP)를 설계한다. RNM-PDP는 SNMP 폴링(poling)을 통해서 호스트 네트워크가 연결된 스위치(switch)의 MIB(Management Information Base)에서 네트워크 대역 이용률의 계산에 필요한 호스트의 네트워크 대역폭, 입·출력 데이터 크기, 그리고 대역 이용률 계산에 필요한 시간을 얻는다[1]. 표 1은 모니터링에 사용할 MIB-II 객체(object)[2]들이다. 이때 RNM-PDP의 SNMP 폴링에서 발생하는 네트워크 내 데이터의 양은 극히 적으므로 호스트들의 네트워크 트래픽에 영향을 주지 않는 것으로 간주한다. 얻어진 데이터는 누적된 데이터이므로 n번째 읽어 들인 데이터에서 n-1번째 읽어 들인 데이터와의 차를 구함으로써 단위 시간 동안 입·출력된 데이터의 양과 시간을 구한다.

표 1 네트워크 모니터링에 사용하는 MIB-II 객체

MIB-II 객체	설 명
ifSpeed	초당 비트 수로 측정되는 인터페이스의 현제 Bandwidth
sysUPTime	시스템의 네트워크 관리 영역이 마지막으로 재초기화된 이후의 경과 시간
ifInOctets	인터페이스에 도착한 전체 옥텟 수
ifOutOctets	인터페이스로 전송된 전체 옥텟 수
inOutDiscards	에러는 없지만 전송되지 못하고 버려지는 패킷 수
ipInDiscards	에러는 없지만 상위 계층에 전달되지 못하고 버려지는 패킷의 수

네트워크 이용률은 전이중방식 네트워크 이용률로 계산하며 계산식은 표 2와 같다.

표 2 대역폭 이용률 계산

FDU : 전이중방식 네트워크 이용률
B : 네트워크 Bandwidth
T : 시간
InData : T 시간 동안 입력 옥텟 수
OutData : T 시간 동안 출력 옥텟 수

$$FDU = \frac{MAX(InData, OutData) \times 8}{T \times 100 \times B} \ (\%)$$

3.2 RNM-PDP의 구조

RNM-PDP는 SNMP API를 사용하고 PDP와 상호 연동을 위해 Java로 구현하였으며, RNM-PDP의 클래스 구조는 그림 2와 같다.

- RNM-PDP_Controller : RNM-PDP의 전체 동작을 제어하는 역할과 PDP와의 인터페이스를 가지고 있어 PDP로부터 모니터링 할 호스트들의 IP 주소와 식별자를 부여받으며 모니터링 결과를 작업 할당에 반영할 수 있도록 PDP에 전달한다. PDP로부터 호스트 IP와 식별자를 전달받게 되면 모니터링 데이터를 얻는 Measure_Extractor 객체를 생성하고 데이터를 저장할 수 있도록 Data_Storage 객체를 전달한다.
- Measure_Extractor : RNM-PDP_Controller에 의해서 생성한다. 생성된 객체는 쓰레드를 통해서 사용자가 정한 시간 간격으로 네트워크 이용률 분석에 필요한 데이터를 SNMP API를 통해서 호스트가 속해 있는 네트워크 환경의 대역 이용률을 SNMP 데몬(demon)이 수행되고 있는 스위치로부터 가져와 Data_Storage에 저장한다. 데이터를 분석해서 네트워크 이용률을 계산하는 일도 담당한다.
- *Visualization* : 사용자 인터페이스 클래스이다. 사용자는 Host의 네트워크 모니터링 시간을 정의한다. 모니터링되고 있는 호스트의 네트워크 트래픽 데이터를 비교하며, 한 호스트 네트워크의 종합적인 모니터링 결과를 관측한다.
- TotalHostView : 호스트가 속해있는 네트워크 대역폭 이용률이나 전송되는 데이터양을 선택하여 비교할 수 있도록 사용자에게 그래프로 제공하는 클래스이다. 사용자 정의 시간 단위로 변화하는 데이터를 막대 그래프 형식으로 제공한다.
- HostView : 선택된 호스트의 정보를 그래프로 사용자에게 제공한다. 사용자가 선택한 호스트의 데이터를 *Visualization* 클래스로부터 전달 받아 호스트의 데이터의 시간 변동에 따른 변화를 그래프로 제공한다.

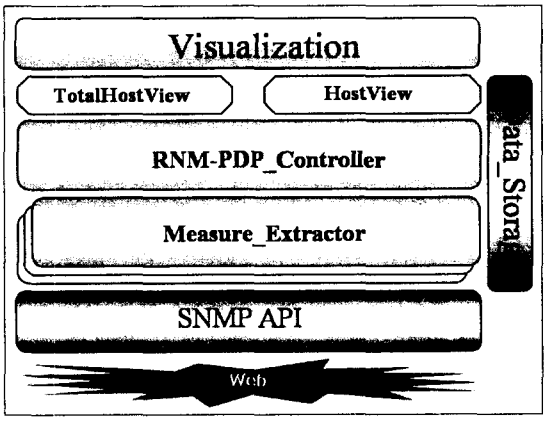

그림 2 RNM-PDP 클래스 구조

3.3 RNM-PDP 동작

RNM-PDP의 동작은 그림 3과 같다. PDP에 호스트가 작업 수행을 위해서 참여하면, PDP는 호스트의 식별자와 IP 주소를 인자로 RNM-PDP_Controller 클래스의 Add_Host 함수를 호출한다. Add_Host 함수는 Extractor_Manager 함수를 통해서 Measure_Extractor 클래스 객체를 생성한다. Measure_Extractor 클래스 객체는 SNMP API를 통해서 네트워크 모니터링 데이터를 Switch의 Host가 연결된 인터페이스에서 SNMP Agent를 통해서 MIB에서부터 가져와 데이터 저장소에 저장한다. 사용자 인터페이스인 *Visualization* 클래스는 데이터

가 데이터 저장소에 입력되기 시작함과 동시에 HostView 클래스와 TotalHostView 클래스를 통해서 시각화한다.

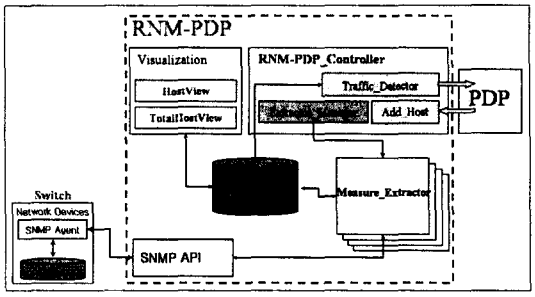

그림 3 RNM-PDP의 동작

Visualization 클래스에서 제공하는 두 가지 뷰(view)는 그림 4와 같다. 먼저, TotalHostView에서 제공하는 뷰로 사용자의 선택을 통해서 참여한 호스트들의 네트워크 환경의 대역 이용률과 데이터그램의 수를 비교한다. 두번째는 HostView에서 제공하는 뷰로써 참여 호스트 트리에서 호스트를 선택하여 선택한 호스트의 네트워크 모니터링 정보를 관측한다.

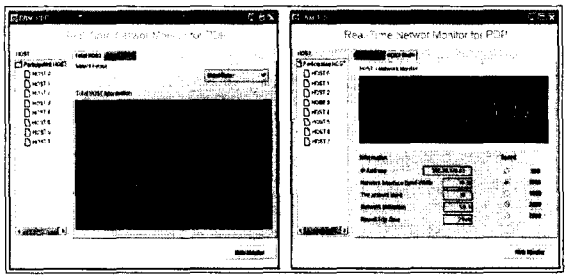

그림 4 RNM-PDP의 두 가지 뷰

3.4 실험 환경

호스트의 네트워크 환경을 모니터링하기 위한 테스트 베드를 그림 5와 같이 구성하였다.

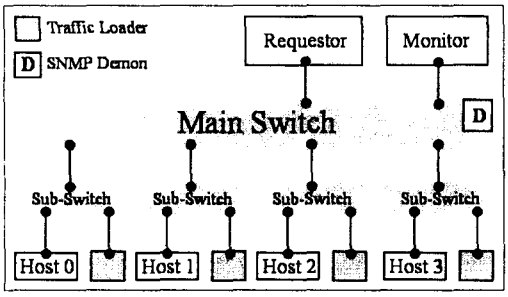

그림 5 테스트 베드

테스트 베드에서 호스트의 네트워크 대역폭 이용률을 측정한 부분은 Main Switch와 Main Switch에 연결되어 Sub-Network를 구성하고 있는 Sub-Switch 사이에 네트워크 대역폭을 모니터링한다. Main Switch는 SNMP Demon이 동작하고 있으므로 각 인터페이스에 입·출력되는 데이터 시간당 증가량으로 해당 인터페이스의 네트워크 대역 이용률을 구한다. 그림 5에서 Host는 PDP로부터 작업 요청을 받아 실제 작업을 수행하는 자원이고 동일한 Sub-Switch에 위치한 트래픽 로더는 실험을 위해서 네트워크 트래픽을 생성하는 자원이다. 각 Sub-Switch에 연결된 트래픽 로더는 사용자가 설정한 만큼의 트래픽을 생성하여 Main Switch를 통해서 네트워크로 전송하여 테스트에 필요한 네트워크 대역 이용률을 유지한다.

3.5 작업 할당

그림 6은 성능 기반 할당 알고리즘에 RNM-PDP에서 모니터링한 대역 이용률을 반영한 것이다. Host2는 Host0과 유사한 성능을 가지고 있지만 높은 대역 이용률로 인해 Host0 보다 적은 양의 작업을 할당받게 된다. 따라서 높은 대역 이용률에 의해 발생한 전체 작업 수행 시간의 증가를 낮은 대역 이용률을 가진 Host0의 추가 작업으로 영향력을 줄인다.

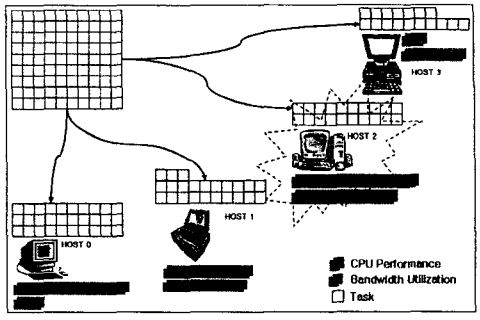

그림 6 대역폭 이용률을 반영한 작업 할당

4 결론

PDP에서 작업 할당 알고리즘은 호스트의 성능에 기반한다. 그러나 분산/병렬 처리가 네트워크 환경에서 수행되는 것을 고려할 때 호스트의 성능뿐만 아니라 호스트의 네트워크 대역폭 이용률을 반영하여 네트워크 환경의 높은 대역 이용률에 의한 처리 결과 전송의 지연으로 야기되는 전체 작업 처리율을 향상 시킬 수 있었다.

참 고 문 헌

[1] Hong Chen, Brett Tjaden, Lonnie Welch, Carl Bruggeman, Lu Tong, Barbara Pfarr, "Monitoring Network QoS in a Dynamic Real-Time System," IEEE 2002

[2] K. McCloghris, and M. Rose, "Management information base for network management of TCP/IP-based internets:MIB-II," RFC 1213, March 1991