클러스터 시스템에서 실행시간 예측을 통한 동적 부하 균등화

윤완오 정진하 최상방
인하대학교 전자공학과
wanoh38@hotmail.com, bully@hitel.net, sangbang@inha.ac.kr

Dynamic Load Balancing using Execution Time Prediction on Cluster Systems

Wan Oh Yoon Jia Ha Jung Sang Bang Choi
Dept. of Electronic Engineering, Inha University

요 약

네트워크 기술의 발전으로 저비용으로 고성능을 얻고자 하는 클러스터 시스템에 대한 연구가 많아지고 있다. SPMD(Single Program Multiple Data) 형태의 병렬 프로그램을 사용한 클러스터 시스템의 주된 성능 장애는 부하 불균등 현상이다. 본 논문에서는 이러한 문제를 해결하기 위해 마스터 노드가 정보를 모으는 횟수와 주기를 시뮬레이션을 통해 최적의 값으로 결정하고 그 주기 동안에 각 노드의 태스크 당 평균 수행시간을 계산한다. 통신비용의 오버헤드를 고려한 시스템의 실행시간을 평균 수행시간으로 예측하여 각 노드가 이동할 태스크의 수를 결정하는 동적 부하 균등 알고리즘을 제안한다. 제안한 알고리즘의 클러스터 시스템을 모델링하고 성능 분석을 위한 시뮬레이션을 한다.

1. 서론

네트워크 기술(ex. Gigabit Ethernet, ATM, Myrinet, SCI)이 발전함에 따라 고가의 고성능 클러스터 시스템을 대체 할 수 있는 저가의 고성능 리눅스 클러스터 시스템의 연구가 활발히 이루어지고 있다[4]. 이런 시스템은 각 노드에서 서로 다른 데이터로 동일한 연산을 하는 SPMD 프로그래밍을 처리할 때 주로 이용된다. SPMD 프로그래밍은 대표적으로 MPI, PVM과 같은 메시지패싱 방식 프로그램밍 환경이 지원하고 있다[3]. MPI는 최근 멀티프로세서 시스템이나 리눅스 클러스터에서 사용할 수 있는 표준화된 메시지 전달 함수의 라이브러리로 동기 및 비동기의 점대점 통신함수와 다양한 집단화 함수를 제공한다.

MPI를 이용한 이기종 클러스터 시스템에서 병렬 프로그램을 사용할 경우 동등한 부하 분산은 각 노드의 시스템 성능과 태스크의 수행시간이 달라 부하 불균등현상이 발생하므로 최적의 시스템 성능을 얻는 것은 불가능하다. 시스템의 성능을 최적화 시키기 위해서는 이런 부하 불균등현상이 일어나지 않도록 각 노드의 시스템 성능에 적합한 태스크를 할당받아 모든 노드가 비슷한 시간에 태스크의 수행을 마쳐야 한다. 그러나 각 시스템의 성능은 태스크 실행 이전에 예측할 수 없어 초기의 모든 태스크를 부하 불균등 현상이 발생하지 않도록 적절히 분배하기가 어렵다. 그러므로 실행시간 중에 수시로 각 노드의 부하량을 측정해서 적절한 부하 재분배를 이루어야 하는데 이를 동적 부하 균등화라고 한다[2].

클러스터 시스템은 부정확하게 연결된(loosely-coupled) 된 분산시스템으로 볼 수 있는데 그 만큼 통신비용이 상대적으로 많이 들므로 통신비용을 고려한 적절한 동적 부하 균등화를 하여야 한다[3]. 스위치 구조의 네트워크로 이루어진 클러스터 시스템에서 동적 부하 균등화는 여러 가지 정보를 이용하여 최적의 부하 균등화를 이루고자 한다. 이를 위해서는 각 노드의 성능과 통신비용을 고려하여 클러스터 시스템의 전체 수행 속도를 가장 빠르게 할 수 있는 태스크의 개수를 선택하여 적절하게 재분배 하여야 한다. 기존의 논문에서는 통신비용을 고려하지 않고 임계치를 고려한 부하균등화 알고리즘을 사용하였다[6].

본 논문에서는 MPI를 사용한 SPMD 병렬 프로그램을 이기종의 클러스터 시스템에 적용하여 최적의 이득을 얻기 위해 정보를 얻는 주기를 선택하고 통신비용을 고려한 동적 부하 균등화 알고리즘을 제안하고 이를 시뮬레이션 하였다.

2. 부하균등모델 및 태스크 이동 비용

현존하는 구현된 클러스터 시스템은 대부분 스위치구조로 상호 연결되어 있으므로 본 논문에서 사용한 부하균등 모델도 10Mbps 이더넷을 사용한 스위치 구조의 마스터-슬레이브 모델을 사용하였다(그림1). 슬레이브 노드

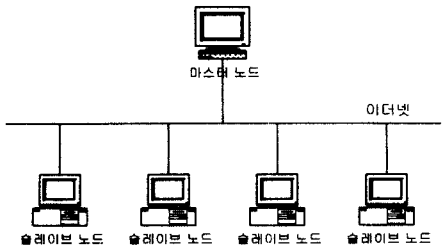

그림 1. 마스터-슬레이브 모델

사이의 태스크 이동에 따른 통신비용 C_t는 두 개의 구성

요소로 다음과 같이 나타낼 수 있다[6].

$$C_t = C_{tx} + C_{starttime}$$

C_t는 각 노드들의 태스크 이동에 따른 통신비용이며, 각 노드가 가지는 네트워크 드라이버의 성능과 10Mbps 이더넷 네트워크로 구성된 모델에서 TCP/IP 최대 전송률로 표현할 수 있다. $C_{starttime}$은 노드에서 태스크를 이동하기 위해 준비하는 지연 시간이다.

3. 주기정책 및 프로그래밍

마스터-슬레이브 모델을 적용한 본 논문에서 동적 부하균등화 알고리즘을 적용하기 위해서는 적절한 시기에 슬레이브 노드들의 정보를 모아 노드의 성능을 계산하고 수행시간을 예측하여 부하 균등화 정책을 결정해야 한다. 성능 예측을 이용한 알고리즘에서 정보 수집 횟수와 정보 수집 사이의 시간간격은 시스템 성능에 영향을 준다. 따라서 노드의 성능을 통한 실행시간의 효율적인 예측과 정보수집의 횟수를 효율적으로 정하기 위해 정보를 수집했을 때의 시간과 그 동안 슬레이브 노드가 처리한 태스크의 수를 계산하여 다음주기를 결정하는 비주기 정책을 사용한다. 병렬 프로그램을 위한 C 프로그래밍을 사용하여 시뮬레이션 하였으며 프로그래밍의 순서도는 다음과 같다.

(1) 초기에 마스터 노드는 동등한 태스크(N)를 슬레이브 노드에 분배하고 임의로 주어진 시간 T에 브로드캐스트를 하여 슬레이브 노드의 부하 정보 요청 메시지를 보낸다.

(2) 각각의 슬레이브 노드는 Global Gathering Method을 사용하여 각각의 노드 정보를 마스터 노드에게 보낸다.

(3) 마스터 노드는 받은 정보를 주어진 알고리즘에 적용하여 필요한 정책을 수행한다. 시스템이 처리해야할 전체 태스크 중에서 α개의 태스크를 처리했을 때마다 정보수집을 한다면 T동안에 처리한 태스크의 개수와 α의 비율로 다음 주기 T_{next}를 정한다.

(4) 계산된 정보를 슬레이브 노드에 브로드캐스트한다.

(5) 지시된 정보에 맞게 송신 노드는 부하 전달 가능 여부를 묻는 메시지를 보내고, 수신 노드의 응답에 따라 태스크를 이동한다.

4. 동적 부하균등화 알고리즘

먼저 통신비용을 고려하지 않은 동적 부하균등화를 제안한다. 동적 부하균등화를 하기 위해서 마스터 노드는 슬레이브 노드의 부하 정보를 이용하여 이주할 태스크를 계산해야 한다. 본 논문의 목적에 부합되는 클러스터 시스템에서 슬레이브 노드의 부하는 기존 분산 시스템에서 전송 정책의 임계값 기준으로 사용되는 CPU queue length, CPU utilization, I/O usage와 같은 절대적인 임계값을 사용할 수 없다. 그러므로 각 노드의 부하를 판단하는 기준으로 Run queue의 태스크 개수로 판단을 한다[3].

초기 P개의 노드에 동등하게 분배한 태스크 개수를 N이라 하고 비주기 T 마다 마스터 노드는 슬레이브 노드의 Run Queue에 있는 태스크 개수(n_i)의 정보를 모은다. T 주기에 모아진 정보를 통해 각 노드가 하나의 태스크를 수행하는데 걸리는 평균 시간을 계산한다.

$$Ttask_i = \frac{T}{N - n_i} \qquad i=1,\ldots,P \qquad (1)$$

슬레이브 노드들이 앞으로 수행해야 할 전체 태스크의 수 N_{total}를 계산하는데 이는 식 (2)와 같다.

$$N_{total} = \sum_{i=1}^{P} n_i \qquad (2)$$

클러스터 시스템이 최적의 성능을 얻기 위해선 모든 슬레이브 노드가 동시에 작업을 끝내야 한다. 모든 노드가 동시에 태스크의 수행을 끝내기 위해서는 식 (3)과 같이 노드의 성능에 비례한 태스크를 재 할 당 받아야 한다.

$$n'_1 \times Ttask_1 = n'_2 \times Ttask_2 = ... = n'_P \times Ttask_P \qquad (3)$$

각 노드들이 재분배 받아야 할 태스크의 수 n'_i 는 (2), (3) 두 공식을 이용하여 다음의 공식으로 유도 할 수 있다.

$$n'_i = \frac{1}{Ttask_i \sum_{j=1}^{P} \frac{1}{Ttask_j}} \times N_{total} \qquad i=1,\ldots,P \qquad (4)$$

이것은 통신비용을 고려하지 않았을 경우 부하균등화에 사용 가능하다. 이더넷을 이용한 클러스터 시스템에서 최적의 수행시간을 갖기 위해서는 각 노드에 재분배하는 태스크의 수의 통신비용과 노드의 성능을 고려하여 n'_i의 값을 구해야 한다. 본 논문에서는 통신비용을 고려한 각 노드의 시스템 수행 시간(E)을 예측하여 n'_i를 계산한다. 전체노드들의 평균 태스크 실행 시간 $T_{average}$를 $Ttask_i$ 와 노드의 개수 P를 이용하여 구한다. $T_{average}$보다 $Ttask_i$의 값이 큰 노드를 과부하 노드, 작은 값을 가지는 노드를 저부하 노드라고 정한다. 모든 슬레이브 노드가 식(4)에서 계산된 태스크 개수가 될 때까지 과부하 노드에서 저부하 노드로 태스크를 하나씩 증가(k)시키며 통신비용(C_t)을 고려한 E_i의 변화된 값을 계산한다. 클러스터 시스템의 전체 수행 속도는 각각 노드의 실행시간 E_i가 가장 큰 값과 일치함으로 노드의 선택은 E_i 값이 가장 큰 과부하 노드에서 E_i 값이 가장 작은 저부하 노드를 택하여 태스크를 이동한다(그림 2).

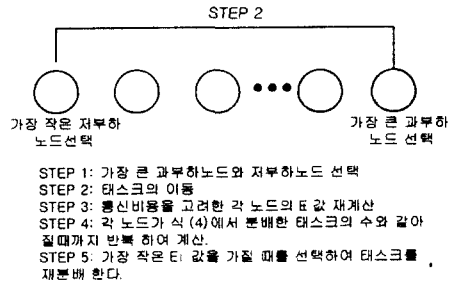

그림 2. 통신비용을 고려한 태스크 이동과정

과부하 노드 : $E_i = (n_i - k) \times T_{taski} + kC_t$

저부하 노드 : $E_i = (n_i + k) \times T_{taski} + kC_t$

하나의 태스크가 이주 할 때마다 전체 노드의 E_i값을 재 계산하여 노드를 선택하고 이주할 태스크를 증가하며 반복해서 계산한다. 모든 노드가 식(4)의 태스크 개수가 되었을 때 그때까지의 E_i값 중에서 가장 적은 경우를 선택하여 각각의 슬레이브 노드에 이동해야할 태스크 정보를 알려준다.

5. 시뮬레이션

본 논문에서는 실제 리눅스 클러스터 시스템의 구현에

앞서 제안한 알고리즘의 성능을 분석하기 위하여 C 프로그램을 사용하여 시뮬레이션을 하였다.

표 1 시뮬레이션 파라메터

Parameter	의 미	사용된 값
α	부하 교환 주기	200개의 태스크 수행
C_{msg}	부하 정보를 보내고 받는 비용.	각 노드의 태스크 실행 평균시간에 1~5%
C_{master}, $C_{receiver}$	Master Node와 Slave Node들간에 수행되는 통신 비용	$(N-1) \times C_{msg}$
C_t	Task 이주비용	단위시간 × task size
N	노드 개수	32
$N_{performance}$	각 노드의 성능	하나의 태스크를 수행하는 차이 0.1~0.45

Marginal Central Dynamic Load Balancing[4], Central Dynamic Load Balancing Strategy, Decentral Dynamic Load Balancing Strategy[5]들과 비교하기 위하여 부하 정보를 보내기 위한 Master-Slave 간의 통신비용(C_{msg})은 각 노드가 하나의 태스크를 수행하는 평균 시간의 1%에서 5%까지 변화시킨다.

하나의 Task가 이웃 노드로 이동하는데 걸리는 시간(C_t)는 C_{msg}의 10배로 하였다. 각 노드의 성능은 구현하기 위한 이기종 클러스터 시스템과 부합하기 위해 하나

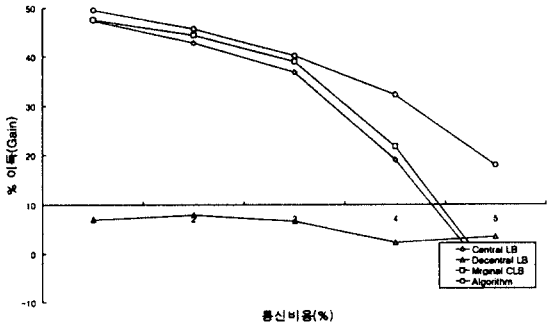

그림 3. 통신비용에 따른 비균등 그룹의 이득

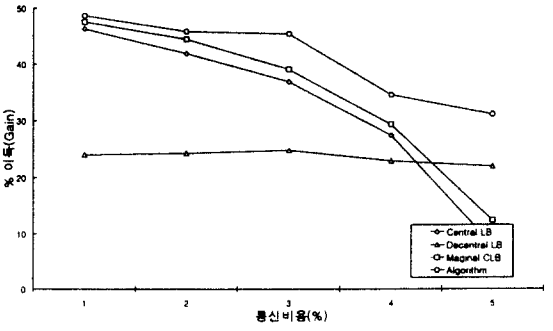

그림 4. 통신비용에 따른 균등 그룹의 이득

의 태스크를 수행하는 차이를 0.1에서 0.45 사이의 임의의 값으로 설정하였다. 정보 교환 주기 α는 시뮬레이션을 통해 약 100개의 태스크를 수행할 때마다 정보를 수집하도록 하였다.

네트워크 구성은 31개의 노드를 1, 2, 4, 8, 16 만큼 개수의 그룹으로 연결한 비균등 그룹과 균일한 개수로 연결한 균등 그룹으로 정하여 시뮬레이션 하였다.

그룹과 그룹사이의 태스크 이동은 스위치를 지나가므로 그룹내의 태스크 이동보다 통신비용을 40% 증가시켰다. 시뮬레이션에서 초래되는 모든 시간은 단위시간으로 가정한다. 그림 3과 4는 부하 균등화를 하지 않았을 경우에 각 노드의 task수는 100개이며 전체 수행시간은 21000에서 25000 단위 시간으로 비균등 그룹과 균등 그룹을 가정하였을 때 부하 균등화를 하지 않았을 경우에 대한 이득의 시뮬레이션 결과이다. Marginal Central Dynamic Load Balancing[4]의 α값은 가장 이득율이 높은 48%일 경우로 가정하였다.

6. 결론 및 향후계획

본 논문에서는 PC 리눅스 클러스터 시스템의 구현을 위해 각 시스템의 성능에 따른 실행시간 예측을 통한 동적 부하 균등화 기법과 정보를 모으는 주기를 프로그래머가 선택할 수 기법을 보였다. 태스크의 이동에 따른 통신비용은 실질적인 정확한 값을 얻어낼 수 없으므로 실행시간에 대한 상대적 통신비용을 이용했다. 전체적인 시스템 성능은 균등 그룹과 비균등 그룹 모두에서 비교대상보다 실행시간의 향상을 가져오는 것을 볼 수 있다. 균등그룹보다 비균등 그룹에서 성능이 떨어지는 것은 그룹과 그룹 사이의 태스크 이동이 비균등 그룹에서 더 많기 때문에 통신비용의 오버헤드가 추가되기 때문이다. 통신비용이 증가할 수록 태스크의 이동에 따른 비용 증가로 인하여 실행속도가 증가하는 것을 알 수 있다.

시뮬레이션을 통해 제안된 알고리즘의 성능을 시험해 보았으므로 향후엔 본 알고리즘을 적용한 PC 클러스터 시스템을 구현하여 성능 향상의 이득을 알아보아야 한다. 구현을 통하여 효율적인 주기의 선택과 태스크 이동에 따른 정확한 비용을 알 수 있다면 보다 나은 성능의 알고리즘을 구할 수 있을 것이다.

7. 참고 문헌

[1] Rajkumar Buyya, "High Performance Cluster Computing Volume 1 - Architectures and Systems"

[2] Marc H. Willebeek-Lemair, Anthony P. Reeves, "Starategies for Dynamic Load Balancing on Highly Parallel Computers", IEEE Transactions on Parallel and Distibuted Systems, Vol 4, No. 9, Setember 1993.

[3] Nenad Nedeljkovic, Michael J. Quinn "Data-Parallel Programming on a Network of Heterogeneous Workstations" 1992 IEEE, ISSN:0-8186-2970-3/92

[4] 정훈진, 최상방, "클러스터 시스템에서 SPMD를 위한 동적 여분 부하 균등화", 한국정보과학회 논문지 제26권 제4호

[5] 박경우, 김병기, "분산 시스템에서 동적인 혼합 부하 균형 알고리즘", 한국정보과학회 논문지 제21권 제4호

[6] M. Cermele, M. Colajanni, G. Necci, "Dynamic Load Balancing of Distributed SPMD Computations with Explicit Message-Passing", IEEE97, ISSN:0-8186-7879-8/97