이동 에이전트 기반의 노드의 부하를 고려한 효율적인 스케줄링 방법

김용호° 김영균 오길호
금오공과대학교 컴퓨터공학부
{kimyh°, ygkim, gilho}@cespc1.kumoh.ac.kr

An Efficient Scheduling Method based on Mobile Agent in consideration of Node's Load

Yong-Ho Kim° Young-Gyun Kim Gil-Ho Oh
School of Computer Engineering, Kumoh National Institute of Technology

요 약

이동 에이전트는 네트워크 내에서 스스로 이동하면서 사용자 또는 다른 개체 대신 행동할 수 있는 컴퓨터 프로그램을 말하며 이동 에이전트가 적용될 수 있는 분야로는 전자 상거래, 정보 검색, 병렬/분산 처리, 네트워크 관리 등이 있다. 본 논문에서는 유휴 컴퓨팅 자원 뿐만 아니라 개별 사용자에 의해 사용중인 컴퓨팅 자원을 포함하는 분산/병렬 처리 환경에서의 스케줄링 방법에 대해 연구하였다. 컴퓨팅 자원이 사용 중일지라도 부하가 적을 경우 적절한 방법으로 분산처리 환경에 포함시킨다면 전체 컴퓨팅 자원들의 사용율은 증대할 것이다. 본 논문에서는 세 가지 스케줄링 방법을 적용하여 작업 노드들을 그룹으로 구성하고, 이동 에이전트를 통해 작업 그룹에 속한 노드들에게 분해할 수 없는 다수의 작업을 할당하여 처리토록 한다.

1.서론

이동 에이전트를 이용한 분산처리 환경에서 유휴 컴퓨팅 자원을 활용하는 연구들이 선행되어 왔다. 이는 사용 중이지 않은 컴퓨팅 자원을 분산처리 환경에 포함시킴으로써 컴퓨팅 자원의 사용율을 높이기 위함이나 사용 중인 컴퓨팅 자원에 대한 고려는 부족하다.

본 논문에서는 유휴 컴퓨팅 자원 뿐만 아니라 사용 중인 컴퓨팅 자원 중 낮은 부하를 나타내는 것을 분산처리 환경에 포함시켜 컴퓨팅 자원의 사용율을 높일 수 있는 효율적인 스케줄링 방법에 대해 연구하였다. 절대적인 성능과 부하를 고려한 성능에 따라 서로 다른 스케줄링 방법을 적용하여 실험하였으며, 분산처리 환경에 참여할 수 있는 컴퓨팅 자원들의 부하를 이동 에이전트를 이용하여 검사한 후 작업 처리에 적합할 경우 작업 그룹에 참여 시키고 작업을 할당한다. 분산 처리 되는 작업은 더 이상 분해할 수 없는 작업들로 하나의 작업은 하나의 노드에 할당되어 처리되며 작업을 처리 중인 노드는 작업을 추가로 할당 받지 않는다.

2.관련연구

2.1 이동 에이전트(Mobile Agent)

이동 에이전트는 네트워크 내에서 스스로 이동하면서 사용자 또는 다른 개체 대신 행동할 수 있는 컴퓨터 프로그램을 말한다. 이동 에이전트는 분산 컴퓨팅 환경에 동적으로 이식이 가능하며(Portability), 사용자가 요구하는 작업을 특정한 호스트로 이동하여 수행할 수 있으며(Code mobility), 다른 에이전트의 직접적인 지시나 간섭 없이도 스스로 판단하여 행동하는 자율성(Autonomy)을 가진다[1].

2.2 설계 시스템의 특성

본 시스템은 개별 사용자에 의해 사용 중인 지역 단위의 네트워크 환경에서 분산처리 환경을 구성하는 것이다. 개별 사용자에 의해 사용중인 컴퓨팅 자원을 사용하는 것이므로 보다 효율적인 작업 처리를 위해 작업 노드들의 부하를 측정하고, 이에 따라 뛰어난 성능을 보유한 노드들로 작업 그룹을 구성하여 주어진 작업을 분산 처리한다. 개별 사용자에 의한 사용 유무와 작업 노드의 부하 고려에 따라 세 가지의 그룹 구성법을 적용하였다.

3.시스템의 설계 및 구현

마스터 노드는 이동 에이전트를 이용하여 작업 노드들을 검사하고 이에 따라 최적의 작업 그룹을 구성하여 작업을 할당하게 된다.

3.1 작업 노드의 부하 측정

개별 사용자에 의해 사용 중인 노드를 분산처리 환경에 포함시키기 위해서는 적은 부하를 나타내는 노드만을 작업에 참여시켜야 할 것이다. 효율적인 작업 그룹 구성을 위해 작업 노드에 대한 부하 측정이 필요하다.

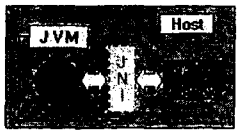


그림1. JNI를 이용한 작업 노드 부하 측정

3.1.1 하드웨어 자원 측정 방법

순수 자바 기능만으로는 작업 노드의 하드웨어 자원에 대한 정보를 확인할 수 없다[2]. 따라서 그림1과 같이 자바의 JNI를 통해 하드웨어 자원 검사 기능을 가진 C 라이브러리를 사용하여 CPU와 메모리에 대한 정보를 검사한다.

3.1.2 하드웨어 자원 측정 항목

작업 할당 및 작업 그룹 구성 시 고려되는 항목은 CPU의 평균 부하량과 메모리의 현재 가용 크기로 한정하였다.

3.1.3 성능 지수 산출법

작업 노드의 성능 지수는 식1과 같이 계산된다[3]. 식1에서 C_i는 i번째 노드의 CPU 처리 성능을 의미하며 M_i는 i번째 노드의 가용 메모리 크기를 의미한다. α, β는 가중치로서 노드의 전체 처리 능력에 대한 중요도에 따라 각 항목에 곱해지는 상수 값으로 그 합은 1이다.

3.2 작업 노드 선택 방법

개별 사용자에 의한 사용 유무, 노드의 절대적인 성능, 작업 노드의 부하 고려 여부에 따라 작업을 할당하기 위한 작업 그룹 구성 방법을 다음과 같이 세 가지로 구분하여 적용하였다.

3.2.1 부하 무시 절대적 성능 우선법

절대적 성능이라 함은 부하가 없을 때의 작업 노드의 성능을 의미한다. 개별 사용자의 작업 노드 사용에 의한 부하를 고려하지 않고 단순히 절대적인 성능만을 고려하여 작업을 할당한다. 모든 노드가 작업을 할당 받을 수 있는 대상이 되므로 별도의 작업 그룹을 구성하지 않는다. 표1에서 채운 막대 그래프는 작업 노드의 절대적인 성능 지수이고 비운 막대 그래프는 부하를 고려한 성능 지수이다. 이때 작업에 참여할 수 있는 노드는 1,2,3,4,5,6번 노드가 된다.

3.2.2 부하 고려 성능 우선법

개별 사용자의 작업 노드 사용에 의한 부하를 고려한다. CPU의 평균 부하와 메모리 가용 크기를 측정한 후 작업 노드의 절대적인 성능을 고려한 부하 제한 값을 차별 적용하여, 특정치 이상의 성능 지수를 나타내는 노드들로 작업 그룹을 구성한다. 표1에서 검은색 세로 선은 작업 그룹 참여가 가능한 최저 성능 지수를 의미하며 그 값이 50일 때 작업에 참여할 수 있는 노드는 1,2,4,6번 노드가 된다.

3.2.3 미사용 노드 우선법

작업 노드의 절대적인 성능과 부하를 고려하지 않고 단순히 개별 사용자에 의한 사용 유무에 따라 작업 그룹을 구성한다. 표1에서 작업에 참여할 수 있는 노드는 2,4,6번 노드가 된다.

표1. 작업 노드의 예

1	사용	100/75
2	미사용	95/95
3	사용	80/55
4	미사용	80/80
5	사용	70/50
6	미사용	70/70

3.3 작업 할당 방법

절대적 성능 우선법의 경우 작업 그룹을 구성하지 않으므로 모든 작업 노드들로 작업을 할당한다. 절대적인 성능이 가장 우수한 순서대로 작업을 할당한다.

부하 고려 성능 우선법의 경우 작업 그룹에 속한 노드만이 작업을 할당 받을 수 있다. 부하를 고려한 성능이 가장 우수한 순서대로 작업을 할당한다.

미사용 노드 우선법의 경우 작업 그룹에 속한 노드의 절대적 성능 순위에 따라 가장 우수한 순서대로 작업을 할당한다.

위의 세 가지 방법 모두 이미 작업을 처리 중인 노드에는 다른 작업을 추가로 할당하지 않는다.

표1의 경우에서 작업을 할당 받는 순서는 표2와 같다.

3.4 작업의 종류

시스템에서 처리할 작업은 더 이상 분해할 수 없다. 하나의 작업은 하나의 노드에 할당되어 처리되며 동일 작업이 다수의 노드로 할당되지 않는다.

표2. 작업 그룹 구성의 예

	1,2,3,4,5,6	1,2,3,4,5,6	
	1,2,4,6	2,4,1,6	부하제한값 50
	2,4,6	2,4,6	

3.5 구현한 시스템의 구조

썬 마이크로시스템즈의 JSDK(Java Standard Development Kit) 1.3.1과 IBM의 ASDK(Aglets Software Development Kit) 2.0.2를 이용하여 구현하였다[5,6]. 시스템은 관리 에이전트, 부하 측정 에이전트, 작업 에이전트로 구성되며 각각의 역할은 다음과 같다.

•관리 에이전트 (Manager Agent)

관리 에이전트는 그림2처럼 마스터 노드에 정적으로 존재하면서 작업 노드들에 대한 IP 주소 관리, 작업 그룹 구성 및 유지를 수행하며 처리 대기 중인 다수의 작업을 위한 풀을 유지한다. 작업 할당을 위해 작업 에이전트를 생성하여 파견하고 작업 수행 결과를 전달 받는다.

•부하측정 에이전트 (Load Measurement Agent)

부하 측정 에이전트는 그림2처럼 작업 노드마다 하나씩 정적으로 존재하면서 노드의 부하 정보를 주기적으로 측정한다. CPU의 평균 부하량과 메모리의 가용 크기에 대한 정보를 JNI를 통해 C 라이브러리를 사용하여 측정한 후 그 값을 마스터 노드에 존재하는 관리 에이전트에게 메시지로 전달한다.

•작업 에이전트 (Worker Agent)

마스터 에이전트에 의해 생성되며, 관리 에이전트가 지정한 작업 노드로 이동하여 작업을 수행하고 그 결과를 마스터 노드로 귀환 후 관리 에이전트에게 전달한다.

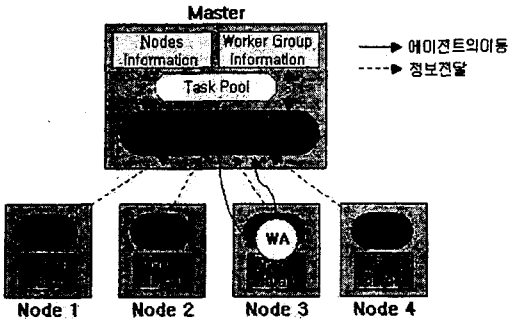


그림2. 시스템의 구조

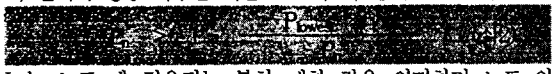
4.성능 평가

성능 평가를 위해 사용된 환경은 표3과 같다. 작업에 참여하지 않는 하나의 관리 노드와 서로 다른 성능을 가지는 8대의 작업 노드로 구성하였다. CPU의 성능 정보는 벤치마크 프로그램인 Wintune을 이용하여 측정하였다.

표3. 실험 환경

	펜티엄3-733 (2156)	256MB	1대
	펜티엄3-733 (2156)	128MB	2대
	펜티엄3-733 (2156)	64MB	1대
	펜티엄3-700 (2064)	128MB	1대
	펜티엄3-450 (1352)	64MB	3대
	펜티엄2-350 (1048)	64MB	1대
	10MBps Ethernet		

부하 고려 성능 우선법을 적용할 경우에 작업 참여가 가능 여부를 결정하는 부하 제한 값은 최저 사양을 가지는 작업 노드의 절대적 성능 지수를 기준으로 식2와 같이 계산된다.



L_n는 노드n에 적용되는 부하 제한 값을 의미하며 노드n의 부하 고려 성능이 L_n보다 클 경우 작업에 참여하게 된다. P_{lowest}는 최저 사양의 작업 노드가 나타내는 절대적 성능이고 P_n은 노드n의 절대적 성능이다. γ는 0과 1사이의 값을 가지는 상수($0<\gamma\leq 1$)로 부하 제한 정도를 조절한다. 본 논문에서는 α, β, γ를 각각 0.9, 0.1, 1.0으로 하였다.

실험을 위해 수행 시간이 서로 다른 160개의 작업을 수행하였으며 작업 노드의 부하는 실험동안 일정하게 유지하였다. 총 수행시간은 표4와 같이 나타났으며 무부하 절대적 성능 우선법은 모든 작업 노드가 사용 중이지 않을 때 절대적인 성능을 우선하여 작업을 할당한 방법이다.

표4. 작업 수행 시간

[illegible]	169,512
[illegible]	239,054
[illegible]	301,466
[illegible]	381,680

각 노드의 절대적인 성능 및 부하를 고려한 성능과 각 방법에서 노드가 처리한 작업 분포는 그림3과 같다.

그림3. 노드별 성능 및 작업 처리 분포

작업 노드의 절대적인 성능 지수를 고려한 작업 노드 선택법의 경우에서, 작업 노드가 모두 사용 중이지 않을 때 작업이 할당되는 분포는 작업 노드의 절대적 성능지수 분포와 비슷하게 나타나므로 성능에 비례해서 작업을 할당 받는 것을 볼 수가 있다. 부하를 무시한 절대적 성능 우선법의 경우 절대적 성능 지수와 큰 차이를 나타내며 오히려 부하 고려 성능 지수와 유사한 분포를 나타낸다. 이는 큰 부하가 걸린 노드에서 작업 처리 속도가 느려지기 때문이며 바꿔 말하면 부하를 고려한 성능에 비례해서 작업을 할당 받는다는 것이다. 부하를 고려한 성능 우선법의 경우 2번 4번 8번을 제외한 모든 노드가 작업 그룹에 참여하였고 성능에 알맞게 할당 받은 것을 볼 수 있다. 미사용 노드 우선법의 경우 1번 5번 6번 7번 노드가 작업 그룹에 참여하였고 절대적 성능이 우수한 노드가 더 많은 작업을 할당 받은 것을 볼 수 있다. 실험 결과를 통해 부하가 있는 노드를 작업 처리에 포함할 경우 절대적인 성능이 아닌 부하가 고려된 성능을 기준으로, 보다 높은 성능을 나타내는 노드가 많은 작업을 처리하도록 해야 노드를 보다 효율적으로 사용할 수 있다는 것을 알 수가 있다.

5.결론

본 논문에서는 이질적인 성능을 가지는 작업 노드들로 구성된 지역 네트워크 환경에서 이동 에이전트를 이용한 분산처리 시스템을 구성하였고, 각 노드들의 효율적인 사용을 위하여 절대적인 성능과 부하를 고려한 성능에 따라 서로 다른 작업 스케줄링 방법을 적용하여 보았다. 실험 결과, 보다 우수한 성능을 나타내는 노드가 작업을 할당 받을 수 있는 기회를 더 가지도록 하는 것이 전체 컴퓨팅 자원을 효율적으로 사용하는데 도움이 된다는 것을 볼 수 있었다.

본 논문에서는 작업 노드의 부하를 고려하여 보다 효율적인 노드 사용을 위한 작업 스케줄링 방법에 연구하였다. 차후에는 노드간의 네트워크 속도와 작업 크기에 따른 전송 시간을 고려한 효율적인 작업 스케줄링 방법에 대해 연구할 계획이다.

참고문헌

[1]OMG, "Mobile Agent Facility Specification", http://www.omg.org/, pp12-13, 2000

[2]P. Bellavista, A. Corradi, "Java-based On-line Monitoring of Heterogeneous Resources and Systems", HP OpenView University Association 7th Plenary Workshop June 12-14, 2000.

[3] 김용호, 김영균, 오길호, "이동 에이전트 환경에서 최적화된 작업 할당 방법에 대한 연구", 한국정보과학회 가을 학술발표논문집, 28권, pp622-624, 10월, 2001년.

[4] 김용호, 김영균, 오길호, "이동 에이전트의 작업 시간을 고려한 작업 노드 선택법에 대한 연구", 한국정보과학회 봄 학술발표논문집, 29권, pp817-819, 4월, 2002년

[5] Aglet workbench, http://www.trl.ibm.com/aglets, IBM Japan, 2000.

[6] D. Lange, M. Oshima,"Programming and Deploying Java Mobile Agents with Aglets", Addison Wesley, 1998.

[7] R. Ghanea-Hercock, J. Collis, D. Ndumu, "Heterogeneous Mobile Agents for Distributed Processing", Workshop Agent based High Performance Computing, 1999

[8] C. Muthukrishnan, Suresh T.B, "A Multi-Agent Approach to Distributed Computing", Workshop Agent based High Performance Computing, 1999