그리드 환경에서 고장 감내를 위한 효율적인 오버레이 네트워크 복구 기법

구현우○ 윤석호+ 홍영식
동국대학교 컴퓨터공학과,
일본 사이월드 (주)+
{ hwgoo○, frostysh, hongys}@dongguk.edu

Effective overlay network reconstruction approach for fault tolerance in Grid environment

Hyun-Woo Koo, Seok-Ho Yun+, Young-Sik Hong
Department of Computer Engineering, Dongguk University
Cyworld Japan Co.,Ltd. +

요 약

기존의 시스템 개발 환경은 동질성, 신뢰성 그리고 보안성 등을 중요시하는 중앙 집중 형태로 운영되어 왔다. 그러나 최근의 컴퓨팅 환경은 분산된 자원들 사이에서의 협업이나 자원 고유를 위한 상호 운영 방향으로 변모되어가고 있다. 이러한 상호 연결 시스템으로는 그리드 컴퓨팅이 활발하게 연구가 진행되고 있다. 그리드 환경에서 고려해야 할 사항은 필요 자원의 사용 대기시간을 줄이는 작업 분배 알고리즘과 고장 감내이며 이들을 중요한 연구 대상으로 하고 있다. 특히, 한정된 지역 정보만을 사용하는 n-Cycle 오버레이 네트워크는 효율적이고 고른 작업 분배 알고리즘을 제공하지만 고장 감내에 대한 대처를 하지 못하는 단점을 지니고 있다. 본 논문에서 부분 복구 기법을 제안함으로써 고장 노드에 의해 발생되는 작업 메시지의 누락율을 줄이고 전체 네트워크 토폴로지의 성능을 향상 시킨다. 또한 고장 노드가 발생하면 전체의 오버레이 네트워크를 재구성해야 하는 문제점을 해결한다. 실험을 통해 부분 복구 기법으로 노드의 고장에 따른 성능 저하율이 현저히 낮아짐을 보인다.

1. 서 론

기존의 시스템 개발 환경은 동질성, 신뢰성 그리고 보안성 등을 중요시하는 중앙 집중 형태로 운영되어 왔다. 그러나 최근의 컴퓨팅 환경은 분산된 자원들 사이에서의 협업이나 자원 고유를 위한 상호 운영 방향으로 변모되어가고 있다. 또한, 현재의 저장 시스템 환경은 지능망, 스위치 장치, 캐쉬 서비스 그리고 응용 서버를 이용한 상호 연결 방식으로 관심이 증대되고 있다. [1] 이러한 상호 연결 시스템으로는 그리드 컴퓨팅이 활발하게 연구가 진행되고 있다. 그리드 컴퓨팅의 연구는 보안성과 자원 발견 및 접근 그리고 자원 관리 측면에서의 성능을 고려하는 연구들이 진행되고 있다. [2]

그리드 환경에서 자원을 필요로 하는 작업들은 필요 자원의 대기시간을 줄일 수 있도록 하는 작업 분배 알고리즘에 의해 자원들을 할당 받고 실행된다. 필요 자원을 빠른 시간 내에 발견하고 접근하는 기법으로 오버레이 네트워크를 이용할 수 있다. 오버레이 네트워크는 인터넷과 같은 강한 연결 형태의 기존 네트워크에 그래프를 이용하여 가상의 망을 형성하는 구조이며 효율적인 자원 분배를 통해 작업의 대기 시간을 줄일 수 있다. 특히, 한정된 지역 정보만을 사용하는 n-Cycle 오버레이 네트워크는 효율적이고 고른 작업 분배 알고리즘을 제공한다.

효율적인 작업 분배뿐만 아니라 그리드 환경에서 고려해야 할 문제는 고장 감내에 있다. n-Cycle이 비록 효율적이고 고른 작업 분배 알고리즘을 제공하지만 네트워크 구성 노드의 고장이 발생하는 경우 전체 네트워크를 재구성 해야 하는 문제를 가진다. 이에 본 논문에서는 n-Cycle를 이용한 자원 분배 알고리즘에서 노드 고장에 의해 발생되는 자원 낭비를 줄이고자 한다. 즉, 오버레이 네트워크를 구성하고 있는 노드에 대한 추가적인 정보를 이용하여 네트워크를 전체 재구성하는 것이 아니라 오버레이 네트워크를 구성하는 노드들 가운데 고장이 발생한 부분만을 복구 함으로써 전체 네트워크 재구성에 따른 시스템 자원의 낭비를 줄이고 고장 감내를 보장하는 기법을 제안한다.

본 논문의 구성은 다음과 같다. 다음 장 에서는 관련 연구를 살펴보고, n-Cycle 오버레이 네트워크에서의 노드 고장이 발생하는 경우의 문제점을 살펴본다. 3장에서는 시스템 자원의 낭비를 줄이고 고장 감내를 보장하는 부분 복구 기법을 제시한다. 4장은 실험을 통해 부분 복구를 적용한 오버레이 네트워크의 성능을 비교하고 마지막으로 결론과 향후 연구 과제를 언급한다.

본 연구는 정보통신부 대학 IT연구센터 육성 지원사업의 연구결과로 수행되었습니다.

2. 관련 연구

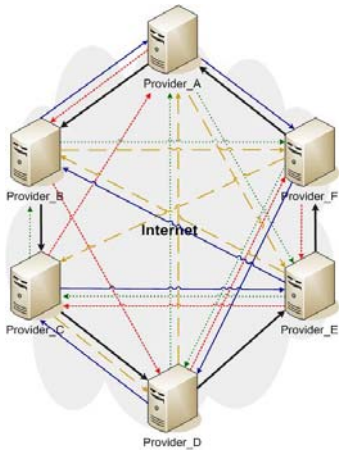
오버레이 네트워크는 효율적인 자원 분배와 같은 추가적인 네트워크 서비스를 수행할 목적으로 기존 네트워크의 노드들과 논리적 링크들로만 이루어진 가상 네트워크를 말한다. 이에 오버레이 네트워크에서의 이웃 노드들은 물리적인 이웃 노드가 아닌 논리적인 이웃 노드가 된다. 이러한 논리적인 이웃 노드들을 연결하고 구성하는 대표적인 기법으로는 분산 해시 테이블을 이용하는 것이다.

분산 해시 테이블의 각 노드는 논리적인 이웃 노드의 IP주소와 같은 다양한 정보들을 보유함으로 네트워크 토폴로지를 유지하게 된다. 분산 해시 테이블을 이용하는 오버레이 네트워크 기법으로 CAN[3], Chord[4], Pastry[5]과 Tapestry[6]가 있다. CAN(Content Addressable Network)는 인터넷 환경에서 해쉬 테이블을 이용한 분산 P2P 인프라이다. CAN은 확장성, 고장 감내, 및 자기 조직화를 위해 설계되었고 가상의 다차원 카티지안 좌표 공간을 이용한다. 좌표 공간은 존(zone)이라고 불리는 하이퍼 사각형으로 분할된다. CAN은 키로 사용되는 이웃 노드들에 관한 논리적인 다차원 좌표 공간과 IP 주소로 구성된 라우팅 테이블을 유지하고 관리한다. 각 노드는 이웃 노드의 좌표를 이용하여 목적지 좌표와 가장 가까운 이웃 노드를 선정하고 메시지를 전달한다. CAN은 시스템에 참여하는 노드의 위치가 균등하지 못하면 라우팅 비용의 편차가 매우 커질 수 있는 단점이 있다. Chord 시스템은 P2P 응용을 위한 분산처리 자원탐색 프로토콜로, 분산 탐색을 지원하며 consistent hashing을 이용하여 데이터의 삽입과 탐색을 수행한다. 핑거 테이블이라 불리는 라우팅 테이블은 소스 노드와 2의 지수승 거리에 있는 노드의 IP를 저장하여 관리한다. 개념적으로는 아주 완벽한 구성이지만 노드의 참여 및 탈퇴에 링크를 재 구성하는 일이 자주 발생하는 문제점이 있다. Pastry와 Tapestry는 앞서 소개한 구조화된 P2P 시스템인 CAN, Chord 와는 달리 프리픽스(Prefix)기반의 유사도를 바탕으로 목적지를 찾아가는 라우팅 방법을 사용한다. 각 노드는 자신과 연결성을 가지는 이웃 노드들의 아이디를 라우팅 테이블에 구성하고 목적지 주소와의 유사도를 높이는 방향으로 라우팅을 수행한다. 이 방식은 네트워크 구성에 따라 글로벌 라우팅에 실패할 가능성이 있다.

분산 해시 테이블을 사용하지 않으면서 오버레이 네트워크를 구성하는 방법은 1차원 또는 계층적 임의 그래프를 이용하여 Flooding과 Random Walking 또는 확장된 링 검색 방법을 이용한다.

분산 해쉬 테이블을 이용한 오버레이 네트워크가 확장성에 우수함을 보이지만 테이블 정보를 유지하기 위한 오버헤드가 존재한다. 인터넷 환경에서는 분산 해쉬 테이블 방법을 이용하는 것 보다 랜덤 그래프를 이용한 방법이 더욱 널리 사용되고 있다. [7]

이러한 방법 중 하나인 n-Cycle 오버레이 네트워크는 분산 해시 테이블을 사용하는 오버레이 네트워크와는 달리 네트워크 토폴로지를 유지하는데 있어 단지 작업 메시지를 전달하기 위한 n개의 이웃 노드의 정보만을 이용한다. 여기서 이웃 노드 정보는 이웃 노드들의 IP 주소 정보 리스트가 된다. [8]



[그림 1] n-Cycle 오버레이 망 구조 (n=5, W=6)

[그림 1]은 노드의 수(W)가 6개인 경우에 5개의 사이클(n)을 구성하는 n-Cycle 오버레이 네트워크의 개략적인 구조를 나타낸다

각각의 노드는 네트워크 토폴로지를 유지하기 위하여 작업 메시지를 전달하는 업 스트림 노드와 노드의 상태 정보를 받는 다운 스트림 노드의 정보만을 유지한다. 만일 A→B의 링크가 있다면 작업이 A에서 처리되지 못하는 경우 B로 전달이 되고 작업을 전달 받은 B는 자신의 상태 정보를 A에게 전달한다. 이와 같은 구조로 n-Cycle 오버레이 네트워크는 n개의 서로 다른 해밀턴 사이클을 구성하며 각 사이클에서 노드의 순서는 무작위로 형성된다.

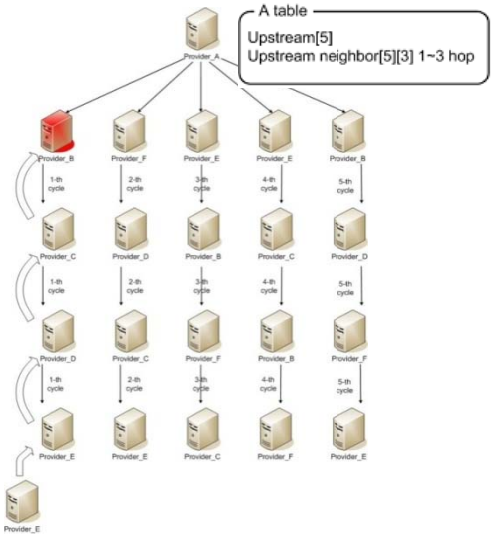
이러한 n-Cycle 오버레이 네트워크에서 중간의 노드가 고장이 발생하는 경우에는 오버레이 네트워크 토폴로지가 파괴되어 오버레이 네트워크를 재구성해야 하는 경우가 발생하는 점이 가장 큰 문제점이다. 이 경우 재구성에 따른 많은 시스템 자원의 낭비를 가져오고 재구성 시 전체 시스템의 성능 저하를 초래한다. 본 논문에서는 재구성이 필요한 경우에 고장이 발생한 노드가 포함되어 있는 사이클에 대해 부분복구를 통해 네트워크 토폴로지를 재구성치 않고 유지하는 방안을 제안한다.

3. 부분 복구 기법

관련 연구에서 살펴본 n-Cycle 오버레이 네트워크에서 노드의 고장이 발생한 경우의 해결 방안으로는 네트워크 전체를 재구성하는 방법과 고장이 발생한 부분에 관련된 정보를 유지하는 테이블을 갱신하는 방법이 있다. 첫 번째 방법인 전체 네트워크를 재구성하는 것은 많은 시스템 자원 낭비를 가져오고 재구성하는 동안 전체 시스템의 성능 저하를 초래한다. 또한, 두 번째 방법은 1개의 노드 고장을 처리하기 위해 유지해야 하는 테이블의 정보가 많아 여러 개의 고장이 발생한 경우 테이블 정보를 유지해야 하는 부하가 발생한다. 따라서 본 논문에서는 고장 감내를 보장하면서 시스템 자원

낭비를 줄이는 부분 복구 기법을 제안한다.

[그림 2]와 같이 부분 복구 기법은 고장 노드가 발생할 경우 작업을 전달하려고 하던 해당 사이클 정보만을 복구하여 테이블 정보를 갱신하고 해당 사이클로 다시 작업을 전달하게 된다. 이는 노드 고장에 따른 오류 노드를 포함하고 있는 테이블은 남아 있지만 각 노드에서 유지해야 하는 복구 테이블 정보를 줄이고 또한 복구를 위한 메시지 전달 횟수도 줄일 수 있게 된다.



[그림 2] 부분 복구 기법

n-Cycle 오버레이 네트워크를 구성하고 있는 자원 제공자(RP: Resource Provider)는 실제 작업 처리를 제공하는 역할을 하며 각 노드의 상태 정보를 유지 하면서 전달 받은 작업을 처리할 것인지 다른 자원 제공자에게 전달 할 것인지를 결정한다. 작업을 전달하기 위한 업 스트림 주소 리스트와 상태 정보를 전달하기 위한 다운 스트림 주소 리스트 및 사이클의 부분 복구를 위해 하위 이웃 노드 주소 리스트를 관리한다. 작업을 처리하는 경우, 처리하지 못하고 하위 노드로 전달하는 경우, 그리고 작업이 종료된 경우에 노드에 대한 작업 전달 가중치를 갱신하여 작업의 고른 분배를 유지한다. 작업 전달 가중치는 다운 스트림 주소 리스트내의 노드들에게 전달된다. 하위 이웃 노드 주소 리스트에는 연속으로 고장 처리 가능한 일정 홉만큼의 노드 정보를 유지한다. 즉, 하위 이웃 노드 리스트를 통해 노드의 고장이 발생했을 경우 부분 복구를 진행할 수 있다.

부분복구 알고리즘은 노드의 고장을 감지한 경우 전달받은 메시지의 작업 번호를 확인하여 연속된 고장의 개수를 카운트하고 이에 따라 작업 메시지를 해당 사이클의 테이블 정보를 이용하여 작업을 전달하고 테이블 정보를 갱신한다.

n-Cycle 오버레이 네트워크의 두드러진 장점 중 하나로 대부분의 노드들은 거의 $\lceil \log_n(|W|) \rceil$ 홉 안에서 서로 메시지를 주고받을 수 있다. 작업을 전달하는 방법으로는 무작위 탐색에 의한 작업 할당 방법과 가중치 확률 알고리즘을 사용한 방법이 있다.

먼저 무작위 탐색에 의한 방법은 만일 현재 작업 매시지를 받은 호스트가 작업을 처리할 수 있는 유휴 상태이면 요청을 받아들여 실행을 하고 그러지 않으면 업 링크 노드중 하나에 무작위 전달을 한다. 무작위 탐색 방법에서 홉 수는 네트워크의 평균 작업량 p 에 의존한다. 첫 번째 가정과 어떤 홉 수 h에 대하여 어떤 노드가 h 홉 이하의 전달 횟수를 가질 확률은 $(1-p)h$로 나타내어진다. 비록 이러한 방법이 어느 정도 만족할만한 평균 작업 처리량을 나타내지만 100%에 근접하지는 못한다. 하지만 알고리즘이 수행되는데 있어서 어떠한 네트워크 상태 정보도 필요로 하지 않는다는 것이다.

부분 복구 기법에서 사용한 알고리즘은 가중치 확률 알고리즘으로 업 스트림 노드에서 수집된 가중치 정보를 바탕으로 적절한 노드에 작업을 전달하는 방법이다. 무작위 탐색 방법과 달리 현재 노드에서 작업을 처리할 수 없으면 각각의 업 스트림 노드의 가중치를 비교하여 유휴상태일 확률이 큰 노드에게 작업을 전달하게 되는 것이다.

부분 복구를 이용함으로써 전체 네트워크 재 구성에 따른 성능저하를 줄이고 고장에 따른 평균 작업 처리량을 증가 시킬 수 있다.

4. 실험 및 분석

부분 복구를 이용하여 n-Cycle 오버레이 네트워크를 유지하는 제안 기법을 실험을 통하여 분석한다. 작업은 일정한 양과 주기적인 작업 메시지를 전달하고, 노드의 고장은 10%정도로 일정 시간대에 발생시킨다. 아래의 [표 1]은 부분 복구 기법을 실험하기 위한 환경 변수 값이다.

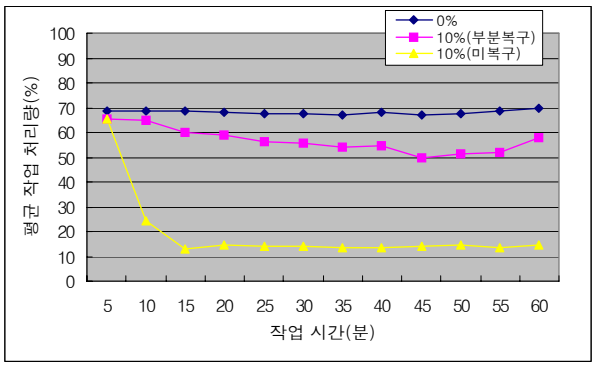
실험은 n-Cycle 오버레이 네트워크의 고장이 발생하지 않는 경우(0%)와 10%의 고장이 발생되었을 때 복구 과정이 실행되지 않는 경우 그리고 부분 복구 과정이 실행 되었을 경우를 비교 실험하였다.

표 1 Simulation Parameters

Input Parameters		
Network Topology	Number of nodes	100
	Overlay networks	5-Cycle
	Task Arrival	5 task/sec
	Task Servicing	Random , (10~20) sec/task
	Walking hop	Random walking
fault	Start Time	After 10 minute
	interval	20 sec/fault
	rate	10%
Output Parameters(Measurements)		
Avr. Throughput	노드당 평균 작업 처리량	
Drop Rate	작업 요청자의 작업 메시지 누락율	

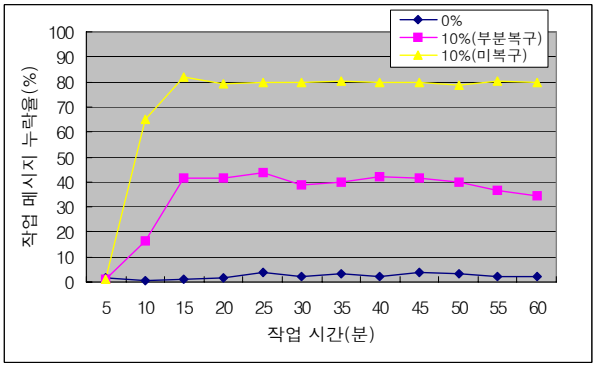
[그림 3]은 고장이 발생치 않은 경우와 10%고장이 발생한 경우 그리고 고장 발생을 부분 복구 한 경우의 노드당 평균 작업 처리량을 보여준다. 노드 고장이 발생하지 않은 경우는

평균적으로 약 68%의 작업 처리량을 나타낸다. 고장이 발생되고 복구 알고리즘을 적용하지 않은 경우에는 대략 평균 19%가량의 작업 처리량을 보인다. 반면에, 제안한 부분 복구 알고리즘을 적용한 경우 평균 약 56% 사이의 작업 처리량을 나타내어 복구 알고리즘이 적용되지 않은 경우와 비교하여 월등한 작업 처리량 증가를 볼 수가 있다.



[그림 3] 평균 작업 처리량 비교

[그림 4]는 각각의 경우의 작업 요청자의 작업 메시지 누락율 보여주고 고장이 발생되지 않은 경우 평균적으로 약 2.3% 정도의 메시지 누락율이 나타나고, 10%의 고장을 발생시킨 경우는 평균 약 72%를 보인다. 반면에, 부분 복구를 적용하였을 때는 평균 약 34%로 메시지 누락률이 발생하는 것을 확인할 수 있다.



[그림 4] 작업 메시지 누락율

두 개의 그래프에서 볼 수 있듯이 고장이 발생한 경우 부분 복구를 통해 고장이 발생하지 않는 경우에 근접하는 작업 처리량과 메시지 누락율을 관찰 할 수 있다.

5. 결론 및 향후 연구

본 논문에서는 그리드 환경에서 효율적인 작업 전달을 위해 사용되는 오버레이 네트워크의 부분 복구 기법을 제안하여 보았다. 전체적인 시스템 구조는 n-Cycle 오버레이 네트워크 시스템 구조에 따라 구현하였으며 추가적으로 부분 복구 알고리즘을 적용시켜 성능을 평가하였다. 제안한 알고리즘에 대한 처리로 시스템 처리량과 클라이언트의 작업 요청 메시지가 누락 되는 비율에서 개선된 성능을 얻을 수 있었다. 또한 메시지 누락률을 상당히 감소시켜 불필요한 작업 메시지 재전송을 줄일 수 있었다. 향후 연구로는 고장률에 따라 전체 네트워크를 재구성 하는 것과 부분 복구를 하는 방법을 비교하여 보고 어느 정도 수준에서 고장 감내가 가능한지에 대한 연구를 생각해 볼 수 있다. 또한 각 자원 제공자가 알고리즘에 따라 작업을 전달할 때 각 자원 제공 노드에서의 작업 스케줄을 통하여 단순히 작업을 전달하는 것 보다 어느 정도 대기 시간을 갖는 것이 더 좋은 성능을 보일 수 있기 때문에 작업 스케줄에 관한 연구도 추가적으로 보완 할 것이다.

참고 문헌

[1] I. Foster, C. Kesselman, J. M. Nick, S. Tuecke, “The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration”, JUNE 22, 2002

[2] I. Foster, C. Kesselman, and S. editors, “The anatomy of the grid: Enabling scalable virtual organizations”, International Journal of Supercomputer Applications, 2001.

[3] Sylvia Ratnasamy, Paul Franics, Mark Handley, Richard Karp, “A Scalable Content-Addressable Network”, Proceedings of ACM SIGCOMM, August 2001.

[4] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger,M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan, “Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications”, IEEE/ACM Transactions on Networking, Vol. 11, February 2003.

[5] Antony Rowstron, Peter Druschel, “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems”, Proceedings of the 18th IFIP/ACM International Conference on distributed Systems Platforms, November 2001.

[6] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, “Tapestry: A global-scale overlay for rapid service deployment”, IEEE Journal on Selected Areas in Communications, vol.22, no.1, pp.41-23, January 2004

[7] Eng Keong Lua, Crowcroft. J, Pias. M, Sharma. R, Lim. S, “A survey and comparison of peer-to-peer overlay network schemes”, Communications Surveys & Tutorials, IEEE, Vol.7, pp.72-93, 2005.

[8] Ladislau Boloni, Damla Turgut and Dan C. Marinescu, “n-Cycle: a set of algorithms for task distribution on a commodity grid”, IEEE International Symposium on Cluster Computing and the Grid, 2005