그룹 기반 클러스터 시스템에서 가상 채널 예약을 통한 실시간 결함허용 연결 모델

홍윤주°, 정지영, 김성수
아주대학교 정보통신전문대학원 정보통신공학과

Real-time Fault-tolerant Connection Model Using Virtual Channel Reservation in Grouped Cluster System

Yunju Hong, Ji Yung Chung, Sungsoo Kim
Professional Graduate School of Information and Communication Technology, Ajou University

요 약

프로세서와 네트워크의 발달로 클러스터 시스템은 웹 기반 응용 분야 등에서 수요가 증가하고 있다. 화상 회의, VOD, 온라인 게임, 상품 및 주식거래 등과 같은 실시간 서비스를 제공하기 위해 클러스터 시스템에서의 라우팅은 결함이 발생한 경우에도 마감시간 안에 실시간 연결이 이루어져야 한다. 본 논문에서는 여러 개의 노드로 구성되어 있으며, 동일한 작업을 수행하는 노드들이 그룹을 이룬 클러스터 시스템에서 백업 채널과 백업 노드를 예약하는 결함 허용 실시간 연결 방법을 제안하고 신뢰도를 분석한다. 또한 일반 네트워크 환경의 라우팅만을 사용한 시스템과 백업 채널을 사용하는 시스템의 QoS를 비교하기 위해 시뮬레이션을 수행한다.

1. 서론

고속 네트워크와 프로세서의 발달로 클러스터 시스템의 보급이 크게 증가하였다[1]. 실시간 클러스터 네트워크 시스템은 클러스터 시스템을 사용하여 실시간 서비스를 제공하기 위한 시스템으로 전화망(Telephone switching), 제조 시스템(Manufacturing system), 실시간 웹 기반 서비스 등에서 사용되고 있다.

클러스터는 일반 PC나 워크스테이션급 컴퓨터를 서로 연결하여 마치 하나의 컴퓨터처럼 사용하는 것을 의미한다. 클러스터에 있는 노드들은 서로 통신하여 업무를 분담하여 수행하며 장애가 발생한 경우 부하 분배와 복구[2]등의 기능을 제공하여 상호 조정되는 방식으로 작동한다. 클러스터는 이러한 기능을 지원하기 위해 모든 노드의 정보를 공유하는 부분이 제공되며, 클러스터의 상태를 지속적으로 모니터 하는 부분을 포함하고 있다. 클러스터에 임의의 한 노드가 실패할 경우 다른 노드들은 통보를 받고 실패한 노드에 할당된 작업이나 자원을 대신 처리한다. 만일 클러스터 내의 연결이 제대로 이루어지지 않는다면 실시간 연결을 필요로 하는 멀티미디어 서비스 등에서 데이터의 손실을 가져와 서비스의 질을 저하시키게 되며 전자 상거래 등의 응용 분야에서 경제적으로나 물질적으로 손실을 가져올 수 있다.

실시간 연결에서 가장 중요한 요소는 마감시간(Deadline)까지 메시지를 전달하는 것이다[3]. 대부분의 실시간 연결은 FDDI나 ATM과 같은 고속의 네트워크에서 QoS 단축형(QoS-Contracted), 연결 지향형(Connection Oriented), 예약 기반형(Reservation-Based)의 형태로 이루어지고 있다[4]. 네트워크는 자신의 정보를 이용하여 클라이언트가 요구하는 작업에 필요한 자원 등을 고려하여 경로(Path)를 선택한 후, 경로에 필요한 자원 등을 예약하며, 클라이언트가 요청한 서비스는 이 예약된 경로를 따라 이루어진다.

본 논문에서 다루는 그룹 기반 클러스터 연결 시스템에서의 라우팅은 클러스터 내의 노드들이 그룹 형태로 연결이 이루어져 있으며 클러스터 그룹 사이에서 메시지를 전달하는 경우 반드시 중심 클러스터 그룹을 통과하여야 한다. 이와 같은 그룹 기반 클러스터 시스템은 분산 환경에서 협동적인 작업을 수행하거나 웹 기반 에플리케이션을 제공하는 시스템에서 역할이 증가되고 있다. 그룹 기반 클러스터 시스템에서 노드들 사이의 결함 허용 실시간 통신을 제공하기 위해서는 결함이 발생한 노드를 고립시키고, 시스템을 재정의 한 후 네트워크의 나머지 부분만을 사용하여 라우팅 하여야 한다. 우리는 이를 실시간으로 제공하기 위하여 두 개의 가상 채널(Virtual channel)을 결함이 발생하기 전에 정의하여 사용한다.

본 논문의 2장에서는 기존의 실시간 통신 방법에 관하여 알아보며, 3장에서는 본 논문에서 다룰 시스템의 구성에 관한 모델과 결함 발생 모델에 관하여 알아보며 4장에서는 이에 대한 분석을 한다. 5장에서는 시뮬레이션 결과에 관하여 살펴보고 6장에서는 제시한 모델의 결과에 관하여 결론을 맺는다.

2. 관련 연구

여러 개의 노드를 가지는 다중홉(Multihop) 네트워크에서 실시간 연결을 위해 메시지를 복사하여 동시에 여러 개의 채널로 전송하는 방법이 있다[5]. 이 방법은 서비스의 끊임없는 결함을 허용하고 복구하기 위해 좋은 성능을 발휘하지만, 멀티미디어 서비스와 같은 대용량의 분야에서는 비용이 많이 든다. 이 방법의 단점을 해결하기 위해 중복 채널을 사용하여 모든 패킷을 중복하지 않고 몇 개의 패킷만을 중복하여 결함을 방지하는 방법이 있다[6,7]. 이러한 방법들은 영구적인 결함을 탐지하고 결함을 복구하는데 효과적이며, 비용을 절감할 수 있다.

실시간 연결을 위해 소스 노드가 모든 네트워크에 브로드캐스트(Broadcast)하여 새로운 채널을 예약하는 방법이 있다[6]. 소스에서 메시지를 보낸 채널의 결함을 탐지하면, 새로운 채널을 찾기 위해 전체 네트워크에 브로드캐스트한다. 이 방법은 새로운 채널이 예약되어 있지 않기 때문에 네트워크가 큰 오버헤드를 갖는다.

[5]를 응용한 방법으로 소스 노드에서 백업 채널을 예약하지만 각각의 노드에서 결함 발생 시 발생하지 않은 다른 경로를 찾기 위해 주위의 경로를 탐색하는 방법이 있다[7]. 이는 백업 채널을 예약해 놓기 때문에 자원의 활용도를 높일 수 있을 뿐만 아니라 각각의 노드들이 다른 채널을 찾기 때문에 효율적인 실시간 통신이 가능하다.

본 논문에서는 이전의 결함허용 실시간 연결에 관한 연구를 바탕으로 여러 개의 노드로 구성된 클러스터 시스템에서 결함 허용 실시간 연결 제공 방법을 제안한다. 클러스터의 노드들은 애플리케이션 작업의 특성에 따라 여러 개의 그룹으로 나누어지며 각각의 노드들은 다른 노

This work is supported in part by the Ministry of Information & Communication of Korea.("Support Project of University Foundation Research<2000>" supervised by IITA)

노드들과의 연결을 필요로 한다. 우리는 이와 같은 시스템에서 실시간 서비스를 제공하기 위해 소스 노드(Source node)와 목적 노드(Destination node) 사이에 두 개의 가상 채널을 예약한다. 가상 채널은 노드들 사이의 링크(Link)의 부하를 줄이기 위해 노드의 결함이 탐지된 후 백업 채널을 Cold-standby로 사용하며, 링크의 대역폭(Bandwidth)을 고려하여 특정 노드의 결함에 의해 백업 채널로 전환시 결함이 발생한 노드에 관련된 채널뿐만 아니라 전체 시스템의 채널을 전환한다. 일반적인 네트워크에서는 목적 노드에 결함이 발생한 경우 가상 채널은 결함을 허용하지 못한다. 그러나, 본 논문에서 제안하는 클러스터 시스템은 목적 노드에 결함이 발생한 경우 같은 그룹의 다른 노드로 작업의 전이가 가능하기 때문에 추가 비용없이 실시간 서비스를 제공할 수 있다.

3. 클러스터 네트워크 모델

3.1 그룹 기반 클러스터 시스템

본 논문에서 제안하는 시스템의 모델은 중심 네트워크 그룹 N과 클러스터 그룹 C_i의 집합으로 이루어져 있다. 그림 1은 i={1, 2, 3, 4}인 그룹 기반 클러스터 시스템을 표현한 것이다. 중심 네트워크 그룹과 클러스터 그룹은 특정한 임을 동시에 수행하거나 협동작업을 통하여 하나의 애플리케이션을 수행한다고 가정한다. 또한, 클러스터 그룹 C_i 사이에는 직접적인 연결이 존재하지 않기 때문에 클러스터 그룹간에 연산이 이루어지기 위해서는 중심 네트워크 그룹 N을 통과하여야 한다.

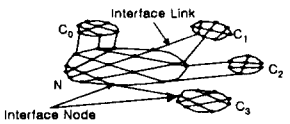


그림 1 i={1, 2, 3, 4}인 그룹 기반 클러스터 시스템

그림 1에서 N의 end 노드와 C_i의 노드를 연결한 에지(edge)를 인터페이스 링크(Interface Link)라 하고 그 노드들을 인터페이스 노드(Interface Node)라 한다. 이 모델에서는 인터페이스 링크와 그룹의 인터페이스 노드가 잘 정의되어 있다고 가정하여, 각각의 그룹의 경계가 분명하다.

3.2 연결 유형(Connection Type)

그림 1의 그룹 클러스터 시스템에서 연결은 모두 5가지 타입으로 요약되어진다. 그림 2에는 메시지를 소스 노드와 목적 노드로 구별하여, 소스 노드에서 목적 노드로 연결이 요청되는 경우를 표현하였다.

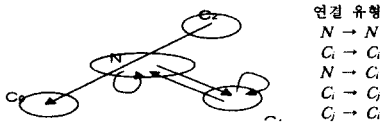


그림 2 연결 유형

중심 네트워크 그룹에서의 라우팅을 R_N, 클러스터 그룹 안에서의 라우팅을 R_G라 하면 C_i에서 C_j까지의 연결은 $R_G \rightarrow R_N \rightarrow R_G$로 표현된다. 이는 앞에서 언급한 바와 같이 모든 연결은 N을 통하여 이루어지기 때문이다. 그러나 각각의 그룹 안에서의 통신은 N에 영향을 받지 않는다. $N \rightarrow N$은 정상적인 중심 클러스터 그룹 내의 노드 연결과 관리, 백업 및 오프라인 관리를 위하여 하나의 노드를 임의로 제거할 경우와 결함에 의해 중심 클러스터 그룹의 노드가 끊어지는 경우에 발생한다. $C_i \rightarrow C_j$, $C_j \rightarrow C_i$의 연결은 클러스터 그룹간에 작업을 요청할 경우에 나타나며, 애플리케이션의 보안, 관리 등을 위하여 반드시 중심 클러스터 그룹 N을 경유하여야 한다. $N \rightarrow C_i$는 초기의 연결 및 대부분의 응용 프로그램을 수행하기 위해 일반적으로 나타난다. $C_i \rightarrow N$은 애플리케이션에서 중심 클러스터 그룹에 특별한 작업을 요청하였을 때, 다른 클러스터 그룹으로 이동 시, 중간 경유 과정으로 나타난다.

그림 2의 연결 유형에서 라우팅은 R_N, R_{Ci}, $R_{Ci} \rightarrow R_N$, $R_N \rightarrow R_{Ci}$의 4가지로 나타낼 수 있다. 네트워크에서 4가지 유형의 라우팅은 모두 같은 방법으로 이루어지기 때문에 5종류의 연결 유형은 한 노드를 기준으로 들어오는 메시지와 나가는 메시지의 두 가지 유형으로 나타난다.

메시지 유형을 두 가지로 나눈 경우, 그룹 기반 클러스터 시스템은 소스 노드와 목적 노드가 존재하는 일반적인 네트워크와 라우팅 알고리즘이 크게 변하지 않는다. 결함이 발생한 경우, 일반 네트워크에서는 중간 노드만 결함을 허용하지만 그룹 기반 클러스터 시스템에서는 목적 노드까지 결함을 허용할 수 있다. 이와 같은 특성을 사용하여, 네트워크의 어느 노드에서 결함이 발생한 경우라 할지라도 사용자에게 실시간 서비스를 제공할 수 있다.

우리는 그룹 기반 클러스터 시스템을 소스 노드와 목적 노드로 이루어진 네트워크로 구성하기 위해 다음과 같은 가정을 한다.

(1) R_N, R_{Ci}, $R_N \rightarrow R_{Ci}$, $R_{Ci} \rightarrow R_N$은 같은 라우팅 알고리즘을 사용하며, 이들 노드의 부하는 연결에 영향을 주지 않는다.

(2) 임의의 노드 사이에서의 연결에 데드락(Deadlock)은 발생하지 않는다.

(3) 임의의 한 노드에 결함이 발생한 경우에 전체 클러스터 시스템에 정지가 나타나지 않는다.

두 노드가 서로의 그룹에게 작업을 요청하는 경우 데드락이 발생할 여지를 가지고 있다. 그리고 이와 같은 경우가 전체 시스템에 걸쳐 나타나는 경우 한 노드의 결함은 네트워크 전체의 중단을 가져올 수 있다. 그러나, 우리는 시스템에 이런 작업의 요청이 존재하지 않는다고 가정한다.

3.3 백업 채널 예약

본 논문에서는 중간 노드에 결함이 발생한 경우 미리 예약된 채널로 연결을 확립할 뿐만 아니라, 목적 노드에 결함이 발생한 경우에도 목적 노드를 이동시켜 채널을 연결함으로써 결함을 허용할 수 있다. 사실, 그룹 클러스터 시스템에서 한 채널의 중간 노드는 다른 채널의 목적 노드일 수 있기 때문에 하나의 노드에 결함이 발생한 경우 전체 시스템의 채널에 영향을 준다.

그림 3은 세 개의 채널이 존재하는 네트워크를 표현한 것이다. 그림 3-b는 노드 N_2에 결함이 발생한 경우 백업 채널로의 전환을 나타낸 것이다. 이 그림에서 채널 1만 백업 채널로 대체하여도 연결에 영향을 미치지는 않지만, N_5와 N_6사이의 링크에 부하가 커지는 단점이 있다. 이를 해결하기 위해 시스템에서는 링크의 부하가 대역폭을 넘지 않도록 채널을 예약하여야 한다.

a. 결함 발생 전 b. 결함 발생 후 백업 채널로 전환

그림 3 주채널과 백업 채널

클러스터 시스템에서 목적 노드에 결함이 발생한 경우 작업 부하 과정을 통하여 새로운 노드를 할당받는다. 그러나 작업 부하 과정은 시간 자원의 낭비가 큰 작업이기 때문에 실시간 서비스를 제공하기에 적합하지 않다. 따라서 우리는 목적 노드에 결함이 발생한 경우 같은 그룹 내의 다른 노드로 직접 전환하기 위해 백업 노드를 예약하여 백업 채널을 미리 예약한다. 이를 위해 주 노드(Primary Node) 외에도 백업 노드(Backup Node)를 예약하여 주 노드에 결함이 발생하는 경우, 백업 노드로 작업이 전이된다.

백업 채널 예약 시 다음과 같은 과정으로 알고리즘을 사용한다.

Step 1: 각각의 링크의 대역폭을 고려하여 링크의 최대용량 C를 설정한다.

Step 2: 모든 소스 노드에서 목적 노드까지 C를 넘지 않도록 주 채널을 결정한다.

Step 3: 모든 Node의 개수가 k일 때, 전체 네트워크에서 N_i를$(i \leq k)$ 제거한다.

(i) N_i가 중간 노드일 때
- C를 넘지 않는 백업 채널을 찾는다.

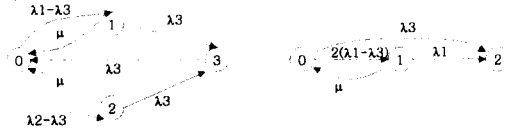
(ii) N_i가 목적 노드일 때
- 목적 노드를 백업 노드로 바꾼다(백업 노드는 같은 그룹의 다른 노드로 예약되어 있다.).
- C를 넘지 않는 백업 노드까지의 채널을 찾는다.

(각각의 채널은 C를 넘지 않는 최단 경로이다.)

4. 분석

4.1 신뢰도 분석

시스템의 신뢰도는 시간 0에서 t까지 서비스를 제공할 수 있는 확률이며 $R(t)$로 표현한다. 각각의 노드가 결함율이 λ인 포아송 프로세스(Possion Process)라면, 노드의 신뢰도 $R(t)$는 $e^{-\lambda t}$로 표현되고 채널의 경로가 n개의 노드로 이루어져 있다고 가정하면 채널의 신뢰도 $R(t)$는 $e^{-n\lambda t}$로 표현될 수 있다. 간단한 모델을 제시하기 위해 네트워크 구성 노드들의 결함율이 같고, 독립적이라 가정하면, 네트워크는 그림 4와 같다.



a. 모델 A　　b. 모델 B

그림 4 백업 채널을 가지는 네트워크의 마르코프 모델

그림 4-a에서 $\lambda 1$과 $\lambda 2$는 주 채널과 백업 채널의 결합율, $\lambda 3$은 두 채널의 공통인 부분의 결합율, μ는 채널의 복구율이다. 상태 0은 초기 상태를 표기한 부분이고, 상태 1은 주 채널에 결함이 발생한 경우, 상태 2는 백업 채널에 결함이 발생한 경우, 상태 3은 채널이 끊어진 경우이다. 그림 4-b는 주채널과 백업 채널이 같은 길이의 노드를 가지고 있다고 가정했을 경우의 모델이다. 한 채널의 신뢰도 $R(t)$는 $P(No\ Primary\ Fail)+P(Primary\ fail \cap No\ Backup\ fail)$이다. 공통인 부분의 노드를 고려하여 신뢰도를 평가하기 위해 함수 $c(a)$를 a채널의 노드의 개수, 주채널을 M, 백업 채널을 B, 공통인 부분을 S라 하고 채널의 신뢰도를 구하면

$$R(t)=1-P(failure\ in\ shared\ node)\cdot P(failure\ in\ the\ rest)$$
$$=1-\{(1-e^{-c(S)\lambda t})\cdot(1-e^{-c(M)\lambda t}-e^{-c(B)\lambda t}+e^{-c(M+B)\lambda t})\}$$

이다.

앞의 알고리즘에서 제거하는 노드의 개수를 증가시켜 백업 채널의 개수를 증가시킬 수 있다. 그림 5는 백업 채널의 개수에 따른 네트워크의 신뢰도이다. 그림 5에서 백업 채널의 개수가 증가할수록 신뢰도는 증가하지만 알고리즘을 실행시키는 데 걸리는 시간의 증가와 백업 채널 예약 공간의 제한으로 인하여 무한히 증가시킬 수는 없다.

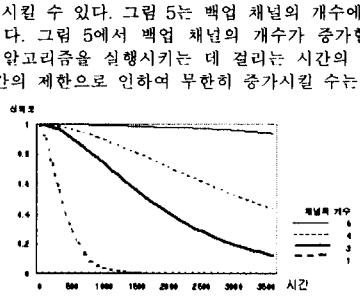


그림 5 백업 채널의 개수에 따른 신뢰도 분석

4.2 QoS 평가

실시간 통신에서의 QoS란 한 패킷의 신뢰도가 아니라, 마감시간 안에 전송되는 데이터의 양을 의미한다. 즉 노드에 결함이 발생한 경우에도 마감시간까지 패킷이 전송되어야 한다. QoS를 평가하기 위해 전송된 패킷이 마감시간까지 도착하여 들어오는 패킷의 비율을 전송률이라고 정의한다.

제안된 모델은 일반 네트워크 환경으로 구성되어 있기 때문에 결함 발생 시 소스 노드에서 백업 채널로 패킷을 재전송함과 동시에 각 노드에서는 일반적인 라우팅 알고리즘을 사용하여 새로운 경로를 찾아 패킷의 전송을 계속하고 목적 노드에서는 전송된 데이터 중 더 빨리 전송된 패킷을 선택할 수 있다.

5. 시뮬레이션

본 모델의 전송률을 평가하기 위해 각 링크를 통과하기 위해 필요한 시간을 단위시간 1로 하고, 마감시간을 임의로 선택하였다. 그림 6은 일반 라우팅 알고리즘을 사용하여 결함을 허용한 네트워크와 백업 채널을 예약하는 네트워크의 전송률을 비교한 그림이다.

각 노드의 결함율이 0.03일 때, 시뮬레이션 수행 결과, 백업 채널을 예약한 시스템에서는 노드의 개수가 증가하여도 전송률이 크게 저하되지 않는다.

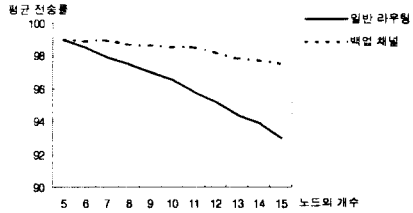


그림 6 전송률의 변화

6. 결론

클러스터 시스템에서 실시간 서비스의 제공은 프로세서의 성능뿐만 아니라, 시스템 내의 라우팅 방법에 따라 서비스의 QoS는 크게 영향을 받는다. 따라서 QoS의 요구를 만족시키기 위해 마감시간 안에 데이터를 전송할 수 있도록 연결이 이루어져야 한다.

본 연구에서는 그룹 기반의 클러스터 시스템의 형태를 단순화하여, 시스템 내의 어떤 노드에 결함이 발생하는 경우에도 채널의 확립에는 영향이 주지 않도록 백업 채널과 백업 노드를 예약하는 방법을 살펴보았다. 채널의 예약은 신뢰도를 증가시키고, 결함이 발생한 경우에도 마감시간 안에 들어오는 패킷의 전송률이 낮아지지 않는다.

우리는 향후 제한된 모델의 링크를 통과하기 위해 필요한 시간을 사실에 근거한 임의의 시간으로 대체하여 전송률을 살펴보고자 한다.

7. 참고문헌

[1] R. Buyya, "High Performance Cluster Computing: Architectures and Systems," Prentice-Hall, pp. 121, 1999.

[2] R. Friedman and D. Mosse, "Load Balancing Schemes for High-Throughput Distributed Fault-Tolerant Servers," Journal of Parallel and Distributed Computing, pp. 475-488, Dec. 1999.

[3] S. Han and K.G. Shin, "A Primary-Backup Channel Approach to Dependable Real-Time Communication in Multihop Networks," IEEE Transaction on Computer, pp. 46-61, Jan. 1998.

[4] B. Devalla, R. Bettati and W. Zhap, "Fault Tolerant Real-Time Connection Admission Control for Mission Critical Applications over ATM-based Networks," Proc. of the Sixth International Conference on Real-Time Computing Systems and Applications, 1998.

[5] P. Ramanathan and K.G. Shin, "Delivery of Time-Critical Messages Using a Multihop Copy Approach," ACM Trans. Computer System, vol. 10, pp. 144-166, May 1992.

[6] A. Benejea, C. Parris and D. Ferrari, "Recovering Guaranteed Performance Service connections from single and Multiple Faults," Technical Report TR-93-066, Univ. of California, Berkeley, 1993.

[7] J. Baker, "A Distributed Link Restoration Algorithm with Robust Preplanning," Proc. IEEE GLOBECOM, pp. 306-311, 1991.