서브넷에서 마스터 노드의 자율적인 선출 알고리즘

김기용 [1], 류연승 [1], 이동호 [2]
[1] 명지대학교 컴퓨터소프트웨어학과
seven@mju.ac.kr, ysryu@mju.ac.kr
[2] 국방과학연구소
happydongho@add.re.kr

An Autonomous Master Node Election Algorithm in a Subnet

KiYong Kim[1], YeonSeung Ryu[1], DongHo Lee[2]
[1]Dept. of Computer Software, Myongji University
[2]Agency for Defense Development

요 약

긴급 업무를 위한 네트워크 시스템에서는 네트워크 구성요소의 고장을 즉시 극복할 수 있는 메커니즘이 제공되어야 한다. 고장 극복을 위해서 네트워크 경로를 이중화하고 경로를 주기적으로 감시하여 고장 발생 시에 대체 경로를 사용하는 방법이 연구되어 왔다. 이를 위해서 서브넷 상의 경로를 감시하는 마스터 노드가 필요한데 마스터 노드의 고장 시 이를 극복할 방법이 필요하다. 본 논문에서는 마스터 노드가 있는 단일 서브넷에서 마스터의 고장 시에 모든 노드가 참여하여 자율적으로 새로운 마스터를 선출하는 알고리즘을 연구하였다.

1. 서론

전투체계 네트워크는 다수의 탐지 장치로부터 획득되는 정보를 바탕으로 전술편집, 위협평가, 무장할당 등의 작업들을 실시간으로 처리, 그 결과를 이용하여 실시간으로 무장체계를 통제하는 실시간 정보처리 시스템이다. 전투체계를 구성하는 장치(노드)들 사이에서 신속하고 정확하게 정보를 교환토록 하는 네트워크의 가용성은 전투체계의 핵심요소라 할 수 있다. 일반적으로 가용성을 높이기 위한 방법으로 이중화된 네트워크 시스템을 구축하고, 네트워크의 링크의 고장에 대비하여 하드웨어적 또는 소프트웨어적인 고장 극복 기법을 적용한다. 고장 극복 기법은 전투체계 네트워크 뿐만 아니라 산업용 실시간 프로세스 제어 어플리케이션 시스템과 같은 긴급 업무 네트워크 시스템에서 매우 중요한 시스템 성능으로 간주되고, 이를 위한 다양한 연구 개발 및 표준화 노력이 현재 진행 중이다.

대부분의 소프트웨어적 고장 극복 기법들은 이중화된 네트워크에서 노드와 노드 사이의 연결 상태를 하트비트 메시지를 통해 감시한다. 하트 비트 메시지는 각 노드들이 서로 교환하거나 마스터 노드가 통합적으로 취합하는 방법이 있다. 각 노드는 경로의 고장을 발견하면 대체 경로를 사용하여 통신하게 된다.

* 본 연구는 방위사업청과 국방과학연구소의 지원으로 수행되었습니다.(계약번호 UD070019AD)

본 연구진은 단일 서브넷 내에서 마스터 노드를 두고 노드 간의 연결 상태를 주기적으로 확인하기 위해 하트비트 메시지들을 취합하는 고장 극복 메커니즘을 연구하고 있다. 그러나, 마스터 노드가 고장이 나면 네트워크 전체에 큰 영향을 주게된다. 본 논문에서는 서브넷에서 마스터 노드가 고장났을 때 각 노드가 자율적으로 마스터 노드를 선출하기 위한 알고리즘에 대해 연구하였다.

본 논문의 2 장에서는 관련 연구로서 살펴본 분산 시스템에서의 조정자(coordinator) 선출 알고리즘들에 대해 기술하였다. 3 장에서는 제안한 자율적인 마스터 노드 선출 알고리즘에 대해 설명하고, 4 장에서는 마스터 노드의 선출에 소요되는 시간에 대한 성능분석을 기술하고, 5 장에서는 결론과 향후 과제를 설명한다.

2. 관련연구

분산 시스템에서 독립적으로 운영이 되는 노드들 간의 상태 정보나 데이터 정보를 교환하기 위한 통신 방법과 어떻게 각 노드들을 동기화하고 상호 작용할 것인가 하는 것은 분산 시스템의 성능을 결정하는 중요한 이슈들이다. 이런 경우 정상적인 작동을 위해 분산 시스템에서의 같은 일을 하는 노드들의 그룹에서 하나의 유일하고 중앙집중적인 조정자라는 특별한 프로세스를 두어 하부 작업을 다른 노드에 할당하거나 다른 노드 들을 관리하는 등의 특별한 일을 담당

한다. 조정자 프로세스를 리더라고 부르며 리더 선출 알고리즘은 고장 극복 분산 시스템 구축을 위해 매우 중요한 문제이다. 리더의 기능을 단지 하나의 프로세스에만 구현할 경우 이 프로세스의 장애가 발생하면 시스템 자체가 구동을 할 수 없는데, 시스템의 신뢰성을 향상시키기 위해 리더의 장애 발생시에도 분산 시스템에 속하는 모든 분산 프로세스들이 스스로 새로운 리더를 선출하는 기능을 갖추어야 한다.

리더 선출의 기본 아이디어는 정상적으로 작동하는 프로세스들 중에서 가장 우선순위가 높은 프로세스를 리더로 선출하고 나머지 프로세스들은 모두 선출된 리더에 동의하게 하는 것이다. 프로세스와 프로세스 사이의 메시지 전달 속도가 정확히 예측 되는 동기적 시스템에서 리더 선출과 같은 동의의 문제는 완전하게 해결 될 수 있으나, 메시지 전달 속도가 정확히 예측되지 못하는 비동기적 분산 시스템에서는 리더 선출과 같은 동의 문제는 해결될 수 없는 것으로 증명되었다.

리더를 선출하는데 있어, 네트워크 토폴로지 방식에 따라 다양한 방법의 선출 알고리즘이 제시되었다. 완전 접속망(fully-connected network) 알고리즘, 링 네트워크(ring network) 알고리즘, 스패닝 트리(spanning tree)에 기초한 알고리즘 등이 있다. 이 중에서 완전 접속망에 기초한 Garcia-Molina 의 불리 알고리즘(Bully Algorithm)이 대표적이다.

불리 알고리즘은 리더가 죽었을 경우, 살아있는 모든 프로세스가 리더 선출 과정을 시작하도록 하는 방식이다. 불리 알고리즘은 고장 복구 형으로 불리 알고리즘을 수행하는 프로세스가 도중에 죽는다고 하여도 이에 대처 할 수 있다. 죽었던 프로세스가 다시 살아날 경우 방금 살아난 프로세스도 선거에 참여하는 새로운 리더 선출 과정이 다시 수행되어 새로운 리더를 선출한다. 불리 알고리즘을 설명하면 다음과 같다.

a. 어떤 프로세스가 리더로서 더 이상 기능을 하지 않는 다는 것을 모든 프로세스들에게 알리면 선출 알고리즘이 시작됨.
b. 메시지를 받은 각 프로세스 P 는 자신보다 우선 순위가 높은 모든 프로세스들에게 메시지를 보내 살아있는지를 확인.
c. 어떤 프로세스들도 응답이 없다면 P 가 선출되며 리더가 됨.
d. 만약에 자신 보다 우선 순위가 높은 프로세스가 살아나 응답을 하면 자신보다 높은 어떤 프로세스가 리더로 선출되기를 기다림.

불리 알고리즘은 적은 수의 프로세스가 그룹으로 묶여 있고, 각 프로세스의 고장 잘못의 발생 빈도가 적은 시스템에서는 리더 선출의 기능을 정확하게 수행 할 것이지만, 그룹에 속하는 프로세스의 수가 많고, 각 프로세스의 고장 발생 빈도가 많으며 타임 아웃 시간 간격이 큰 시스템에선 처리속도는 크게 느려지게 된다.

따라서 불리 알고리즘을 개선한 알고리즘에 대한 여러 연구들이 있다. 타임아웃방식보다는 결점 탐지기를 이용한 수정된 불리_FD 알고리즘이 있다. FD 모듈을 이용하여 프로세스로부터의 직접적인 응답을 기다리지 않고 그 프로세스의 고장여부를 다운 시그널이나 업 시그널로 받는다. 따라서 기존 알고리즘 보다 빠르게 다른 노드들의 상태를 점검할 수 있다.

이 외에 다른 방식으로는 불리 알고리즘에서 조정자 프로세스가 고장 상태에서 다시 회복한 노드를 포함하는 과정에서 새로운 선출 과정을 거치지 않고 빠르게 새로운 조정자에게 복속시키는 알고리즘 등이 있다.

3. 제안하는 자율적 선출 알고리즘

본 연구진은 고장 극복을 위한 이중화 전투체계 네트워크에서 노드와 노드 사이의 연결 상태 감지를 통해 이중 경로에 대한 가용 경로를 제공하기 위한 마스터 노드 개념을 연구하고 있다. 이때, 마스터 노드의 고장에 대비하여 자율적인 마스터 노드 선출 알고리즘을 연구하였다.

3-1. 전체 구조

그림 1 은 이중화된 네트워크에서 마스터 노드를 추가한 구조이다.

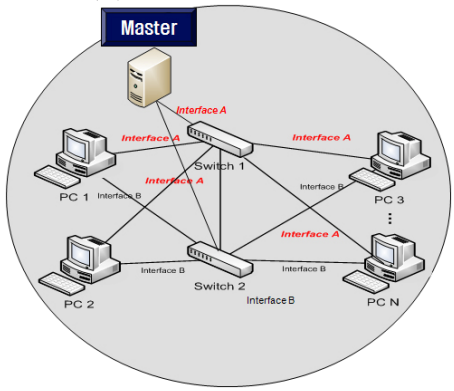

그림 1. 마스터 노드를 가진 한 서브넷에서의 전투체계 네트워크 구조

한 서브넷에는 두 개의 NIC 카드를 가진 n 개의 노드와 2 개의 스위치, 그리고 하나의 마스터 노드가 존재한다. 노드들과 마스터 노드는 이중화된 링크로 각각의 스위치에 연결되어 있다(인터페이스 A/인터페이스 B). 각 노드들은 주기적으로 하트비트 메시지를 한 서브넷 전체에 멀티캐스트로 보내게 된다.

마스터 노드는 서브넷 안의 모든 노드의 연결상태정보를 가지고 있으며, 가용 경로를 계산해 그 정보를 요청한 노드에게 알려준다. 자신의 서브넷 안에 있는 모든 노드들로부터 하트비트 메시지를 주기적으로 받게 된다. 이 하트비트 메시지를 이용하여 각각의 노드들에 대한 연결 상태 정보를 알 수 있다. 이를 수신한 마스터 노드는 자신이 만든 테이블에 각 노드들의 연결 상태와 노드-노드간의 가용한 전송경

로를 가지게 된다. 주기적으로 하트비트 메시지를 받으면서 자신의 테이블 내역을 설정한다.

만약, 서브넷에 마스터 노드가 새로 만들어져서 다른 노드들처럼 따로 존재를 하고 고정되어 있으면, 그 서브넷이 가지게 될 비용과 성능적인 측면에서 상당한 손해가 될 수도 있다. 기존 서브넷 네트워크에 마스터 노드를 추가해야 한다는 비용적 문제, 그리고 고정적으로 마스터 노드를 지정해둔다면, 고장 등으로 마스터 노드로서의 역할을 못하는 상태가 되면 네트워크의 불능 상태에 빠지게 될 것이다.

3-2. 자율적인 마스터선출 알고리즘

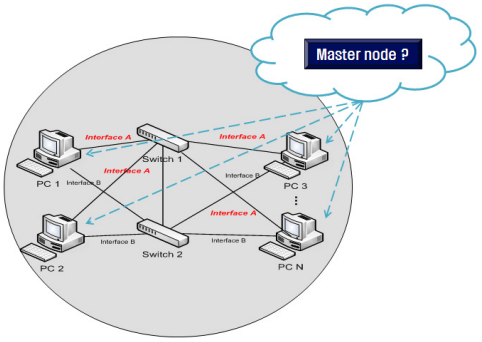

그림 2. 서브넷에서의 자율적인 마스터 노드

본 논문에서는 서브넷에서 별도의 마스터 노드를 따로 고정시키지 않고, 노드들이 자율적으로 마스터를 선출하여 이 문제를 해결하고자 한다. 마스터가 될 자격이 있는 후보 노드에 마스터의 기능을 추가, 동작시키고, 이 노드가 고장 등 이유로 마스터의 역할을 못하는 것을 탐지하면 서브넷 안의 다른 노드들이 새로운 마스터 차기 후보 노드를 선출하여 동적으로 마스터의 기능을 차기 후보 노드에게 이양시키는 것이다. 이렇게 하면 마스터 고장 시 새로운 마스터가 동작하게 되므로 고장 극복의 기능을 신뢰 할 수 있고, 새로이 마스터 노드를 만들지 않고 기존의 네트워크를 그대로 이용할 수 있어 성능과 비용에서 좋은 측면을 보일 것이다.

마스터를 위한 자율적 결정 프로토콜에서 각 노드들은 표 1 과 같은 상태를 각각 가지게 된다.

자율적인 마스터 노드 결정 방식은 그림 3 의 상태도에 의해 이루어진다. 각 노드 들은 맨 먼저 노드 초기화 INIT 상태를 지나 PREPARE 상태로 변경이 된다. 이때 후보 노드를 뽑는다. 후보 노드는 서브넷 내에서 제일 작은 IP 주소를 가진 노드인데, 만약에 이 후보 노드가 자격을 상실할 경우에 다음으로 낮은 IP 주소를 가진 노드가 후보 노드가 된다. 후보노드가 선정이되면 후보 노드는 WAIT 상태, 후보 노드가 아닌 노드들은 VOTE 상태로 변하게 된다. VOTE 상태가 되면 후보 노드에게 vote 메시지를 전송하게 된다. WAIT 상태의 후보 노드들은 타임아웃이 되기 전에 50%이상의 vote 메시지를 받으면, WIN 상태가 되어 마스터 노드로 선정된다. WIN 상태가 된 마스터 노드들은 자신의 서브넷의 모든 노드에게 win 메시지를 주기적으로 브로드캐스트를 한다. 만약에 win 메시지가 주기적으로 일반 노드에게 전달되지 못해서 타임아웃이 되면 다시 후보 노드를 찾아서 마스터 노드를 선정하게 된다.

표 1. 자율적인 마스터 선출에서의 노드의 상태

상태	설명	동작
PREPARE	후보 노드 선정중인 상태	후보 노드 선정
WAIT	후보 노드가 되었지만 50% 미만의 vote 메시지 수신 상태(Timeout 이전)	
VOTE	후보 노드가 아니 노드로 win 메시지를 수신하지 않은 상태(Timeout 이전)	후보 노드에게 vote 메시지 전송
WIN	후보 노드가 Master가 된 상태 (50%이상 vote 메시지 수신)	Win 메시지를 각 노드에게 송신
READY	후보 노드가 아닌 노드가 일반 노드로 확정된 상태	

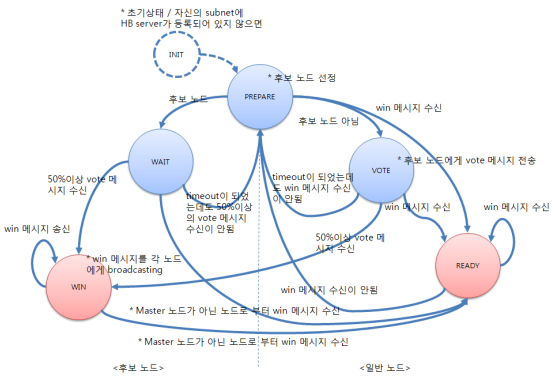

그림 3. 자율적인 마스터 노드 결정에서의 상태도

각 노드가 수행하는 마스터 노드 선출 알고리즘을 표 2 에 기술하였다.

4. 성능분석

본 절에서는 마스터 선출에 소요되는 시간을 분석한다. 첫 마스터의 선출이 성공했을 때에 소요되는 시간, 첫 마스터 선출이 실패할 때 두번째 시도에서 마스터가 선출에 소요되는 시간을 계산하고, N 번째까지의 마스터 선출에 소요되는 시간을 분석한다.

본 논문에서는 메시지 전달 속도가 정확히 예측된다는 것으로 가정한다. 이것은 정확히 예측되지 못하는 비동기적 시스템에서는 선출과 같은 문제는 해결되지 못한다는 것이 증명되어 있기 때문이다.

분석을 위해 사용하는 기호를 정의하면,

T_m : 노드간의 메시지 평균시간, 마스터가 win 메시지를 각 노드에 멀티캐스트 하는 시간 주기

TO_{win} : 마스터 노드로 부터 win 메시지가 오는 시간의 주기

TO_{master} : 마스터가 될 후보 노드가 50%의 vote 메시지를 기다리는 최대 시간

TO_{voter} : vote 메시지를 보낸 노드가 후보 노드의 win 메시지를 기다리는 최대 시간
($TO_{voter} > TO_{master} + T_m$)

표 2. 각 노드가 수행하는 마스터 선출 알고리즘

```
startup:
if state isn't INIT
then
        init ()
else
        state:=PREPARE
endif
while FOREVER
do
        switch (state)
                case PREPARE:
                        if received win msg is exist
                        then
                                state := READY
                                break
                        endif
                        if a current node is a candidate node
                        then
                                insertCandidateNode(currentNodeId)
                                candidateFlag := 1
                        else
                                insertGeneralNode(currentNodeId)
                                candidateFlag := 0
                        endif
                        if candidateFlag is 1
                        then
                                state := WAIT
                        else
                                state := VOTE
                        endif
                        break

                case WAIT:
                        timerWaitEvent()
                        for a timerWaitEvent is running
                        do
                                if received win msg from other node is exist
                                then
                                        state := READY
                                        timeOut()
                                        break
                                endif
                                if recvVoteMsg >= 0.5*NumOfAllNodes
                                then
                                        winFlag := 1
                                endif
                        done
                        if winFlag == 1
                        then
                                state := WIN
                        else
                                state := PREPARE
                        endif
                        break

                case VOTE:
                        timerVoteEvent()
                        sendVoteMsg()
                        for a timerVoteEvent is running
                        do
                                if recvOtherNodesVoteMsg >= 0.5*NumOfAllNodes
                                then
                                        state := WIN
                                endif
                                if received win msg is exist
                                then
                                        state := READY
                                        break
                                else
                                        state := PREPARE
                                endif
                        done
                        break
                case WIN:
                        timerWinEvent()
                        for a timerWinEvent is running
                        do
                                sendWinMsg()
                                if received win msg from other node is exist
                                then
                                        state := READY
                                        break
                                endif
                        done
                        break
                case READY:
                        if received win msg is exist
                        then
                                state := READY
                        else
                                state := PREPARE
                        endif
                        break
        endswitch
done
```

처음 마스터 선출 시도에서 마스터가 정상적으로 선출된다면 이때까지 소요되는 시간은 다음과 같다.

$2T_m + TO_{win} + TO_{master}$

첫번째 시도가 실패하면, 두번째 마스터 선출을 시도하게 되며 이때 마스터 선출까지의 소요 시간은,

$3T_m + TO_{win} + TO_{master} + TO_{voter}$

세번째 시도에 마스터가 선출된다면 소요되는 시간은,

$4T_m + TO_{win} + TO_{master} + 2TO_{voter}$

따라서, N 번째 시도에서 마스터가 선출된다면 이때까지의 소요 시간은,

$(N+1)T_m + TO_{win} + TO_{master} + (N-1)TO_{voter}$

와 같다.

5. 결론

본 논문에서는 마스터 노드가 있는 단일 서브넷에서 마스터 고장 극복을 위해 모든 노드가 참여하여 자율적으로 마스터를 선출하는 알고리즘을 제안하였다. 제안한 알고리즘은 노드들이 후보 노드에게 투표를 하고, 과반수 득표를 한 후보 노드가 마스터 노드로 선출된다. 제안한 방법은 마스터 선출에 소요되는 시간이 노드 수에 비례해 증가할 수 있다. 따라서, 예측 가능한 마스터 선출 소요시간을 제공하기 위해 각 노드가 항상 다른 노드의 상태 정보를 관리하게 하고 마스터 선출 시에 정상 상태의 후보 노드에게만 투표를 할 수 있는 방법을 연구 중이다. 향후 과제로서 제안하는 알고리즘을 시뮬레이션을 통해서 검증하고자 한다.

참고문헌

[1] "전투체계 네트워크 고장극복프로토콜 설계 기술 분석" 보고서, 국방과학연구소, 2007.5

[2] 송대연, 윤종호, 정한균, 김승환, "항공 데이터버스용 링크 우회방식을 가진 고장감내 이더넷 시스템의 성능 분석", 전자공학회, 2007

[2] 박성훈, "동기적 분산 시스템에서 효율적인 조정자 선출 알고리즘", 정보과학회 , 2004

[3] 김기, 이동훈, 최은미, "분산 시스템에서의 효과적인 코디네이터 선출 알고리즘", 정보과학회, 2003

[4] 신해준, 장재준, 김영탁, "MLPS 통신망에서의 신속한 장애복구를 위한 서브네트워크 기반의 세그먼트 단위 자동복구 기법", 한국통신학회, 2002

[5] J.Huang, S.Song, L.Li, P.Kappler,R.Freimark, J.Gustin, T. Kozlik, "An Open Solution to Fault-Tolerant Ethernet: Design, Prototyping, and Evaluation",IPCCC'99 ,IEEE, 1999

[6] Michal J. Fisher, Nancy A. Lynch, and Michael S. Paterson, "Impossibly of distributed consensus with one fault process", Journal of the ACM, 1985

[7] H. Garcia-Molina, "Elections in a distributed computer system", IEEE trans. on computer, 1982

[8] Chang, E.G. and Roberts, R. "An improved algorithm for decentralized extrema-finding in circular configurations of processors." Comms. ACM, 1979