분산 시스템에서 확장성 있는 순서화 방송통신 프로토콜

(Scalable Ordered Broadcast Protocol in the Distributed System)

한 인 † 홍 영 식 ††
(In Han) (Young Sik Hong)

요 약 본 논문은 대규모 분산시스템에서 전체 순서화를 유지하는 방송통신 프로토콜을 제안한다. 기존의 토큰기반 방송통신 프로토콜과는 달리 제안된 프로토콜에서 토큰은 방송도메인을 구성하는 링을 묵시적으로 회전하지는 않는다. 대신에 토큰 전송을 위한 세 개의 메시지를 사용한다. 토큰은 방송메시지를 위한 순서번호 정보를 갖고 있으며, 오직 토큰을 소유한 노드만이 방송메시지를 전송할 수 있다. 방송메시지를 전송하려 하는 노드는 토큰을 소유한 토큰노드로 토큰요청메시지를 보내고, 토큰노드는 요청한 노드에게 토큰을 전송한다. 토큰을 요청한 노드가 토큰을 소유하게 되면 토큰의 순서정보를 이용하여 방송메시지를 전송한다. 이 메시지를 수신한 모든 노드는 토큰노드가 변경되었음을 인식하게 되고, 새로운 토큰노드로 토큰을 요청하게 된다. 그러나, 새로운 토큰노드로부터 방송메시지를 수신하기 전에 과거의 토큰노드로 토큰요청메시지를 전송하는 노드가 존재할 수 있다. 이 문제를 해결하기 위해 제안된 프로토콜은 과거 토큰노드가 새로운 토큰노드에게 다음 토큰요청노드를 알리는 메시지를 사용한다. 시뮬레이션 결과를 통해 기존의 방송통신 프로토콜보다 제안된 프로토콜이 대규모 분산시스템에서 더욱 효율적이라는 것을 알 수 있다.

키워드 : 분산시스템, 방송통신, 프로토콜, 전체 순서화, 토큰

Abstract In this paper, we present an efficient broadcast protocol, called Btoken, that ensures total ordering of messages and atomicity of delivery in the large scaled distributed systems. Unlike the existing token-passing based protocol, Btoken does not circulate a token around the ring, instead, it uses three kinds of control messages for token transmission. The token has a sequence number field for broadcasting message, and the only site having the token can broadcast a message. When a site wishes to broadcast, it must send a message to the token site requesting the token. The token site sends a message with the token to the requester. When the requester receives the token, it becomes the current token site and broadcasts a new message after setting sequence number derived from a field of the token into its message. Upon reception of it, any operating member is informed the position of the token site and will send token requesting message to the new token site. However, the other site may request the token to the old token site prior to receiving the broadcast message from the new token site. To resolve this problem, Btoken uses a message which is sent to the current token site by the old token site notifying who is the next token requester. Results of our simulation of the protocol show that Btoken is more efficient in the large scaled distributed system compared to existing broadcast protocols.

Key words : Distributed system, Broadcast protocol, total ordering, token

1. 서 론

방송통신은 하나의 노드에서 방송시스템을 구성하는 모든 노드로 동시에 메시지를 전송하는 통신 수단으로 이더넷기반 LAN에서는 일반적으로 TCP/IP의 UDP를 사용한다. UDP를 이용한 메시지 전송은 TCP를 이용한

· 이 논문은 동국대학교 연구년(1999년-2000년) 지원에 의한 것임.
† 정 회 원 : (주)뱅크타운 기술연구소 시스템체계연구실장
ihan@banktown.com
†† 종신회원 : 동국대학교 컴퓨터. 멀티미디어공학과 교수
hongys@dongguk.edu
논문접수 : 2001년 4월 2일
심사완료 : 2002년 1월 15일

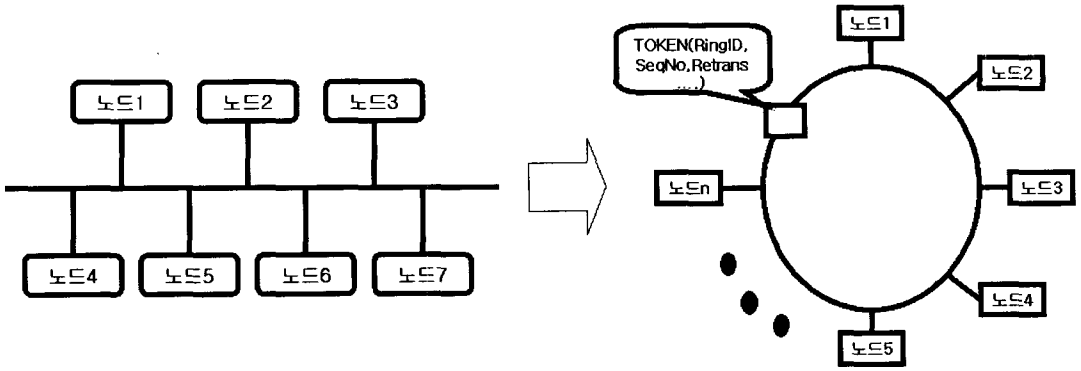

그림 1 Totem의 물리적 네트워크 구조와 논리적 링 구성도

메시지 전송과는 달리 메시지가 수신노드에서 올바르게 수신되었음을 확신할 수 없는 비신뢰 메시지 전송방법을 사용하고, 두 개 이상의 노드에서 동시에 서로 다른 메시지를 전송할 경우 모든 노드에서 같은 순서로 메시지를 수신한다는 것을 보장할 수 없다는 단점을 갖고 있다.

이러한 문제를 해결하기 위해 많은 분산시스템에서는 방송되는 모든 메시지에 순서 정보를 추가하여 전송한다. 이 순서정보는 수신노드에서 메시지들의 순서를 재배열하고, 손실된 메시지를 검출하여 재전송을 요구하는 메카니즘에 사용된다. 순서정보를 이용하여 방송메시지를 전송하는 대표적인 알고리즘으로 중앙노드 기반 방송통신 프로토콜[1,2,3,4,5,6]과 논리적 링 기반 방송통신 프로토콜[7,8,9,10]이 있다.

중앙노드 기반 방송통신 프로토콜[3,4,5]은 방송도메인을 구성하는 모든 노드들 중에서 오직 하나의 노드를 중앙노드로 설정한다. 방송메시지를 전송하고자 하는 노드들은 방송요청메시지를 중앙노드로 전송하고, 중앙노드는 방송요청메시지에 고유의 순차번호를 부여하여 방송한다. 또한 중앙노드는 재전송 요청에 응답하고 주기적으로 동기화 단계를 수행하게 된다.

중앙노드 기반 방송통신 프로토콜의 작업량 집중화와 단일노드고장의 문제점을 보완하기 위해 제안된 프로토콜이 논리적 링 기반 방송통신 프로토콜[7,8]이다. 이 프로토콜들은 방송도메인을 구성하는 모든 노드들을 하나의 논리적 링으로 구성하고, 토큰이라고 하는 방송권한을 링을 따라 회전시킨다. 토큰을 소유한 노드만이 방송메시지를 전송할 수 있고, 토큰 속에 유지되는 순차번호를 이용하여 전체 순서화를 유지한다.

본 연구에서는 논리적 링 기반 방송통신 프로토콜의 두가지 문제점을 보완하는 새로운 방송통신 프로토콜을 제안하고, Btoken이라고 명명한다. 논리적 링 기반 방송통신 프로토콜의 첫 번째 문제점은 방송도메인 구성 노드 수 증가로 인한 방송메시지 전송지연시간의 증가하는 것과 두 번째는 노드 고장으로 인한 논리적 링을 재구성하는 오버헤드가 존재하는 것이다.

Btoken 방송통신 프로토콜은 논리적 링 기반 방송통신에서 사용하는 토큰을 사용하지만 순차적으로 링을 회전시키지 않기 때문에 노드 수 증가로 인한 논리적 링 기반 프로토콜의 문제점을 보완한다.

본 논문은 2장에서 논리적 링 기반 방송통신에 대해 기술하고, 3장에서 본 논문에서 제안하는 방송통신 프로토콜에 대해서 설명하며, 4장에서 논리적 링 기반 방송통신 프로토콜과 제안된 프로토콜의 성능을 비교 실험한 결과와 분석을 기술하고 5장에서 결론과 향후연구에 대하여 기술한다.

2. 논리적 링 기반 방송통신 프로토콜

논리적 링 기반 방송통신 프로토콜[7,8,9,10]의 대표적인 시스템은 Totem과 RMP가 있다. 이러한 프로토콜은 이더넷 구조의 네트워크에서도 방송도메인을 구성하는 모든 노드들을 하나의 논리적 링으로 구성하고, 토큰이라고 하는 방송권한을 지속적으로 회전시키고 있다. 토큰을 논리적 링을 따라 회전시키는 이유는 토큰을 소유한 노드만이 방송메시지를 전송할 수 있고, 모든 노드들에게 공평한 방송기회를 제공하기 위해서 이다. 다음 [그림 1]은 논리적 링 기반 방송통신 프로토콜의 시스템 구성도와 토큰의 모습을 나타낸다.

[그림 1]과 같이 토큰은 방송메시지의 유일한 순차번호를 부여할 수 있는 정보를 갖고 있으며, [그림 2]와 같

이 정상수행상태, 정보수집상태, 확인과정상태, 그리고 회복상태의 4가지 상태를 갖는다. 정상수행상태는 노드의 고장이 발생하지 않는 상태이고, 정보수집상태와 확인과정상태, 그리고, 회복상태는 노드의 고장이 검출되어 새로운 논리적 링을 구성하기 위해 거치는 단계이다.

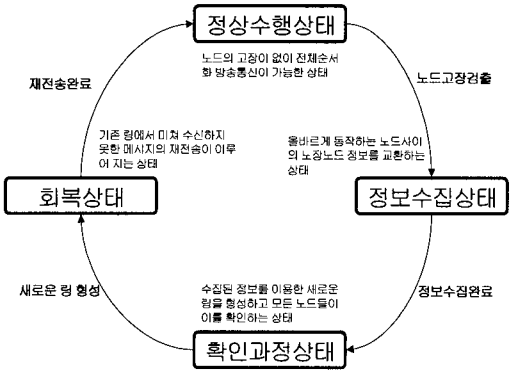

그림 2 Totem에서 노드들의 상태 전이도

정상수행상태에서 토큰을 수신한 노드는 자신이 수신하지 못한 메시지의 재전송 요구, 다른 노드에서 요구한 메시지의 재전송, 그리고 자신의 메시지를 방송하는 일을 할 수 있는 권한을 갖는다. 토큰을 수신한 노드는 자신이 유지하고 있는 메시지 순차번호와 토큰의 메시지 순차번호를 비교하여 토큰의 순차번호가 더 크면 메시지 자신이 수신하지 못한 메시지가 있음을 검출할 수 있고, 재전송 받기 위하여 토큰 속의 "재전송 요구 리스트" 정보에 손실된 메시지의 순차번호를 등록한다. 또한, 토큰의 재전송 요구 리스트 정보를 읽어 자신이 소유한 메시지를 다른 노드에서 재전송을 요구하였는지 조사하여 자신이 소유하고 있다면, 재전송 요구 리스트에서 해당 메시지 순차번호를 제거하고, 해당 메시지를 같은 순차번호로 방송메시지를 보냄으로써 재전송을 하게 된다. 방송결정이 일어난 노드에서 토큰이 수신되면, 수신된 토큰의 순차번호 정보를 증가시키고, 자신의 메시지에 토큰의 순차번호를 부여한다. 고유의 순차번호를 부여받은 메시지를 방송함으로써 전체 순서화를 유지하는 방송메시지를 전송하게 된다.

Totem과 같이 논리적 링 기반 방송통신 프로토콜은 논리적 링을 구성하는 노드의 고장이 발생할 경우 논리적 링이 끊어지게 되고, 링이 끊어짐으로써 지속적이고 순차적인 토큰의 회전이 불가능하게 된다. 이런 문제를 해결하기 위해 Totem에서는 시간초과 알고리즘을 이용하여 노드의 고장을 검출할 수 있고, 노드의 고장이 검출되면 새로운 링을 구성하는 알고리즘을 제안하고 있다.

정보수집상태는 고장난 노드를 제외한 노드들만으로 새로운 링을 형성하기 위해 노드들의 상태를 조사하는 단계로 각 노드에서 유지하는 고장노드 리스트와 정상노드 리스트를 방송통신을 이용하여 교환한다. 정보수집상태의 노드들은 자신의 정보와 다른 정보를 수신할 때마다 방송통신을 이용하여 이 사실을 다른 노드들에게 알린다. 따라서, 유한시간 내에 모든 노드들의 정보는 일치하게 되고, 모든 노드의 합의가 발생하면, 가장 작은 노드 식별자를 갖는 노드가 새로운 토큰을 생성하여 다음 노드로 토큰을 전송하면서 확인과정상태로 전이한다

확인과정상태에서 새로 생성된 토큰은 링을 두 번 회전하게 된다. 첫 번째 회전은 새로운 링을 생성하면서 과거 링에서의 정보를 수집하는 과정을 거치고, 두 번째 링을 회전하면서 새로운 링의 생성을 확인하고 과거 정보를 배포하는 역할을 한다. 두 번째로 링을 회전하는 토큰을 수신한 노드는 재전송 단계에서 과거 링에서 수신하지 못한 메시지의 재전송 처리가 발생한다.

기존의 링 기반 방송통신 프로토콜은 방송도메인을 구성하는 노드들을 하나의 논리적 링으로 구성하고 구성된 링을 따라 토큰이라고 하는 방송권한을 지속적이고, 순차적으로 회전시킨다. 따라서 방송메시지를 보내고자 하는 노드에서 방송결정을 한 후 토큰이 수신될 때까지 기다리는 지연시간이 발생하고, 이 지연시간은 링을 구성하는 노드의 수가 많아지면 비례적으로 증가하는 단점을 갖고 있다. 또한 링을 구성하는 임의의 노드에서 고장이 발생하면 노드 고장을 검출할 수 있으나 새로운 링을 구성하기 위해 너무 많은 메시지 교환과 시간이 소요된다는 단점을 갖고 있다.

3. BToken 방송통신 프로토콜

링 기반 방송통신 프로토콜은 중앙노드 기반 방송통신 프로토콜의 문제점을 보완하는 알고리즘이지만, 방송도메인을 구성하는 노드 수가 증가하면 방송결정 후 방송메시지를 전송할 때까지 소요되는 시간이 비례적으로 증가한다는 단점과 노드의 고장이 발생할 경우 새로운 링을 구성하기 위해 너무 많은 메시지가 교환된다는 단점을 갖고 있다. 이러한 단점을 보완하기 위해 본 연구에서는 방송권한인 토큰을 사용하면서도 방송도메인을 구성하는 노드들을 논리적 링으로 구성하지 않는 새로운 알고리즘, BToken을 제안한다. BToken은 방송도메인을 구성하는 노드들을 논리적 링으로 구성하지 않기

때문에 토큰의 지속적이고, 순차적인 회전이 발생되지 않는다. 따라서 방송결정을 한 노드는 토큰을 소유하기 위해 토큰을 소유한 노드에게 토큰 요청 메시지를 전송하고, 토큰을 소유한 노드는 요청노드에게 토큰을 전송해 줌으로써 방송결정을 한 노드에서 새로운 메시지를 방송할 수 있도록 한다. 다음 [그림 3]은 BToken 방송통신 알고리즘의 시스템 구성도이다.

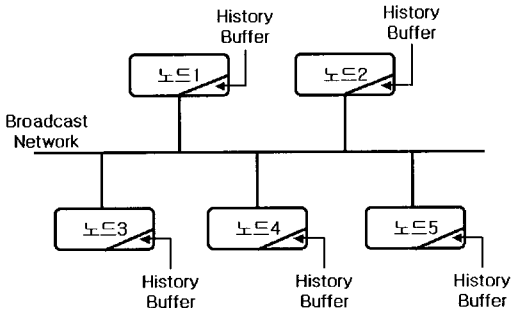

그림 3 Btoken의 시스템 구성도

[그림 3]에서와 같이 모든 노드는 "방송노드 정보 리스트(History Buffer)"를 갖고 있으며, 이 정보는 가장 최근에 방송한 노드를 가장 첫 번째 요소로 저장하는 스택과 유사한 구조로 되어 있다. 따라서, 각 노드에서 유지하는 방송노드 정보 리스트의 첫 번째 요소는 가장 최근에 방송한 노드 식별자가 저장되어 있고, 이를 통해 각 노드는 자신이 어느 노드에게 토큰 요청 메시지를 전송해야 하는지를 결정할 수 있다. 다음 [Algo.1]은 방송결정은 한 노드가 토큰을 소유하고 있지 않을 경우 토큰을 소요하고 있는 노드로 토큰 요청메시지를 전송하는 내용을 기술한 것이다.

알고리즘 1 Btoken의 방송결정에 대한 동작 과정

```
if ((WantedBcast() = TRUE ) AND (havingToken()
  = FALSE) then
   TokenSite ← getTopBuff()
   SendMsg( TokenRequestMsg, TokenSite )
end if
```

토큰을 소유한 노드는 토큰 요청 메시지가 수신되면 다음, [Algo.2]과 같은 과정으로 토큰을 요청한 노드에게 전송해 준다.

토큰 요청 메시지를 보낸 노드에서 토큰을 수신하게 되면 토큰의 순차번호 정보를 이용하여 자신의 메시지 순차번호를 결정하고 방송메시지를 전송한다. 새로운 방송메시지를 수신한 모든 노드는 송신자 식별자를 조사하여 자신의 토큰 소유노드 정보를 수정함으로써 방송결정을 할 경우에 최근에 방송메시지를 전송한 노드로 토큰 요청메시지를 전송하게 된다. 다음 [Algo.3]은 방송메시지를 수신한 노드에서 진행되는 과정을 기술한 것이다.

알고리즘 2 Btoken의 토큰소유노드가 토큰요청메시지 수신에 따른 동작

```
Recv(MSG)
if MSG.Type = "TokenRequestMsg" then
   if havingToken() = TRUE then
      while( NumOfBcast != 0 )
         MyMsg ← mkMyMsg()
         TOKEN.SeqNo ← TOKEN.SeqNo + 1
         MyMsg.SeqNo ← TOKEN.SeqNo
         Mroadcast( MyMsg )
         NumOfBcast ← NumOfBcast - 1
      end while
      SendMsg( TOKEN, MSG.Sender)
   end if
end if
```

알고리즘 3 Btoken의 방송메시지 수신 노드의 동작과정

```
RecvMsg( MSG )
if (MSG.type = "BroadcastMsg" ) then
   if (MSG.Sender != HisBuff[0]) then
      RemoveNode( MSG.Sender, HisBuff )
      PUSHisBuff( MSG.Sender )
   end if
end if
```

하지만, 모든 노드들은 각기 방송노드정보 리스트를 유지하기 때문에, 임의의 노드가 토큰을 요청하고 수신하여 방송메시지를 전송하기 전에 다른 노드가 과거의 방송노드에게 토큰을 요청하는 경우가 발생될 수 있다. 이 경우의 과정을 [그림 4]로 나타내었다.

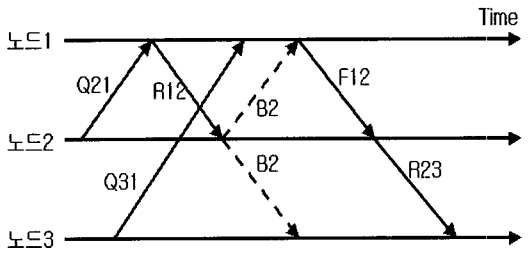

그림 4 Btoken의 메시지 송·수신 시나리오(노드 고장 없음)

[그림 4]에서 노드1이 토큰을 소유한 노드이고 노드2와 노드3의 방송노드 정보 리스트의 첫 번째 요소에 노드1의 정보가 저장되어 있다고 가정한다. 노드2에서 방송 결정이 발생하여 노드1에게 토큰 요청 메시지를 보내고, 노드1이 노드2에게 토큰을 전송한다. 토큰을 받은 노드2는 방송메시지를 보낼 수 있고, 이 방송메시지를 수신을 한 노드들만이 토큰을 소유한 노드가 노드1에서 노드2로 변경되었음을 알 수 있다. 그러나 노드3은 노드2의 방송메시지를 수신하기 전에 자신의 정보를 이용하여 노드1에게 토큰 요청메시지를 전송함을 알 수 있다. 하지만, 노드1은 이미 노드2에게 토큰을 전송한 상태이므로 토큰을 전송해 줄 수 없고, 노드2 역시 노드3이 토큰 요청메시지를 전송하였다는 것을 알 수 없다.

이 경우, BToken에서는 "전달(Forwarding)메시지"라는 새로운 일대일통신 메시지를 이용하여 해결한다. 전달 메시지는 토큰을 다른 노드에게 전송한 노드가 토큰을 전송한 시점부터 토큰을 수신한 노드로부터 방송메시지가 수신될 때까지 자신에게 토큰 요청 메시지를 전송한 노드들의 정보를 새로운 방송노드에게 알리는 메시지로 [그림 4]에서 F12 메시지를 말한다. 전달 메시지를 수신한 노드2는 노드3이 방송 결정을 한 상태라는 것을 알 수 있고, 토큰을 노드3으로 전송할 수 있다. 따라서, 노드3은 토큰을 수신하여 방송메시지를 전송할 수 있다.

BToken 방송통신 알고리즘에서 토큰을 수신하기 위한 오버헤드로 토큰 요청메시지와 토큰 전송메시지가 존재하며, 경우에 따라 전달메시지가 추가적인 오버헤드로 작용하게 된다. 그러나, 방송도메인을 구성하는 노드들의 수가 증가하더라도 순차적인 토큰의 회전이 발생하지 않으므로, 노드 수가 증가하더라도 토큰을 수신하기 위한 지연시간이 증가하지 않는다.

토큰을 소유한 노드의 고장은 순서화 방송통신 불능이라는 매우 치명적인 문제를 야기한다. BToken 방송통신 알고리즘에서는 토큰을 소유한 노드가 고장이 발생하면, 시간초과 알고리즘으로 검출하고 새로운 토큰을 생성하는 알고리즘을 제안한다.

고장을 검출한 노드는 자신이 유지하고 있는 방송노드 정보 리스트의 두 번째 노드에게 토큰 요청 메시지를 전송한다. 일반적으로 토큰을 소유하지 않은 노드가 토큰 요청 메시지를 수신하고, 자신이 방송노드 정보 리스트의 두 번째 노드이면, 첫 번째 노드인 방송노드 즉, 토큰을 소유하고 있는 노드의 고장이 검출되었음을 될 수 있다. 최근의 방송노드가 고장이 발생하였다고 판단하는 기준은 일정시간동안 방송메시지가 수신되었는가를 이용하여 판별한다. 현 방송노드의 고장으로 판별되면, 자신이 기존 토큰의 버전번호보다 1만큼 큰 버전 번호를 갖고 순차번호를 0으로 설정한 새로운 토큰을 생성하여 토큰 요청 노드로 전송한다.

새로운 버전 번호를 갖는 토큰을 수신한 노드는 자신의 메시지를 방송하고 새로운 방송노드가 된다. 또한, 새로운 버전 번호의 토큰을 이용하여 전송한 방송메시지를 수신하는 모든 노드들은 과거의 토큰노드에서 고장이 발생하였음을 인식하고, 방송결정이 발생하면 새로운 토큰을 갖는 노드에게로 요청메시지를 전송한다. 다음 [그림 5]는 토큰을 소유한 노드에서 고장이 발생하였을 경우에 동작 과정을 나타낸 것이다.

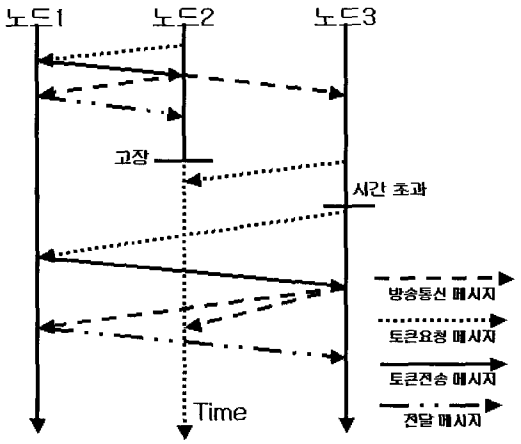

그림 5 Btoken의 노드 고장 검출시 메시지 송·수신 시나리오

[그림 5]에서 노드2는 노드1로부터 토큰을 수신하여 방송메시지를 전송한 후 전달 메시지까지 수신한 후 고장이 발생하였다. 노드3은 노드2의 방송메시지를 수신하였으므로 토큰을 소유한 노드가 노드2라는 것을 인식하고 방송결정이 발생하여 노드2로 토큰 요청 메시지를 전송한다. 하지만, 노드2는 고장이 발생하였기 때문에 노드3은 일정시간동안 응답메시지를 수신하지 못하게 되어 노드2의 고장을 검출하게 된다.

노드2의 고장을 검출한 노드3은 자신의 방송노드 정보 리스트의 두 번째 노드인 노드 1에게 토큰 요청 메시지를 전송하고, 노드1은 노드3의 토큰요청메시지를 수신하여 노드2의 고장을 판별하게 된다. 노드1역시 일정시간동안 어떠한 방송메시지도 수신하지 못하였다면 노드2의 고장을 판별할 수 있다. 따라서 노드1은 새로운

토큰을 생성하여 노드3에게 전송하고 노드3은 자신의 메시지를 방송하게 된다

만약, 토큰을 갖고 있는 노드의 고장이 발생하지 않은 상태에서 네트워크의 부하나 시스템의 작업량 과다로 고장으로 검출되는 현상이 발생하더라도 방송메시지를 수신하는 노드들이 최고버전의 토큰을 이용한 방송메시지를 최종 방송노드로 결정하게 되므로 버전번호가 작은 토큰을 소유한 노드는 자신의 메시지를 방송한 후 다른 노드로부터 토큰 요청메시지가 수신되지 않고, 자신의 토큰보다 버전이 높은 토큰을 갖고 방송한 메시지가 수신되기 때문에 방송도메인에서 일시적으로 두 개 이상의 토큰이 존재할 수 있으나 오직 하나의 토큰만이 존재하게 되고, 방송도메인은 전체 순서화가 유지된다.

4. 실험 내용 및 결과 분석

기존의 논리적 링 기반 방송통신 프로토콜인 Totem과 본 논문에서 제안하는 BToken 방송통신의 성능을 비교하기 위하여 Linux 5.1 운영체제에서 ns-2.1.b4 시뮬레이션을 사용하여 시뮬레이션 하였다.

첫 번째 실험은 방송도메인을 구성하는 노드 수에 따른 성능 비교로 Totem의 경우 방송도메인을 구성하는 노드의 수가 증가하면, 토큰이 링을 1회전하기 위해 소요되는 시간이 증가하게 됨으로 방송결정을 한 노드가 토큰을 수신할 때까지 기다리는 지연시간이 증가하게 된다. 하지만, BToken의 경우는 방송도메인을 구성하는 노드 수와 무관하게 토큰 요청메시지와 토큰 전송메시지가 오버헤드로 작용함을 확인하는 실험이다.

Totem의 경우 M개의 순서화 방송메시지를 전송하기 위해 소요되는 통신비용은 전체 방송도메인을 구성하는 노드 수를 N이라 하고, 일대일 통신비용을 T_{Ptp}, 그리고 방송통신 비용을 T_{Bcast}라고 할 때, 다음 식(1-1)과 같이 계산할 수 있다.

$$T = M(\frac{N}{2} T_{Ptp} + T_{Bcast}) \qquad 식(1\text{-}1)$$

Totem에서는 방송결정을 한 노드가 토큰을 수신하기까지 평균 지연시간($\frac{N}{2} T_{Ptp}$)이 오버헤드로 작용하고, 이 값은 방송도메인을 구성하는 노드의 수에 비례적으로 증가함을 알 수 있다.

BToken에서의 순서화 방송메시지 전송비용은 다음 식(1-2)와 같이 계산할 수 있다.

$$T = 2(M - M_d) T_{Ptp} + M T_{Bcast} \qquad 식(1\text{-}2)$$

식(1-2)에서 M_d는 i번째 방송결정을 한 노드와 $(i+1)$번째 방송결정을 한 노드가 같은 경우의 수를 나타낸다.

방송결정이 연속적으로 발생할 경우 BToken에서는 토큰 요청메시지와 토큰 전달 메시지를 사용하지 않는다.

식(1-2)를 통해 BToken은 방송도메인을 구성하는 노드의 수와 무관한 통신비용이 소요됨을 알 수 있다. M_d 값은 방송도메인을 구성하는 노드의 수가 증가하면, 연속적으로 방송결정을 할 확률이 작아지기 때문에, 방송도메인을 구성하는 노드 수가 증가하면 아주 미세한 성능감소가 발생한다.

실험에 사용한 파라미터들은 다음 [표 1]과 같다.

표 1 노드 수 증가에 따른 성능변화을 위한 실험 파라메터

파라미터 내용	파라미터 값
네트워크 구조	10Mbps 이더넷
노드 수	4~20
방송결정 노드	임의의 노드
방송결정 시간 간격	평균0.5초의 지수분포의 임의 시간
순서화 방송메시지 수	1000 개
노드 고장과 재전송	없음
전송 메시지 크기	1 KB

다음 [그림 6]은 시뮬레이션 결과이다.

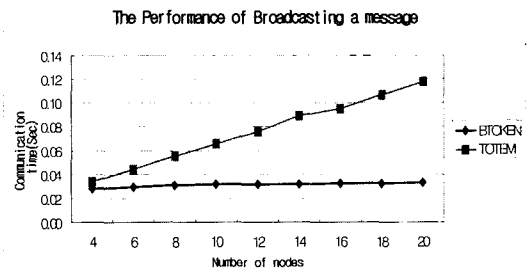

그림 6 노드 수 증가에 따른 성능변화 실험 결과 그래프

첫 번째 실험 결과인 [그림 6]을 통해 논리적 링 기반 방송통신 프로토콜인 Totem은 방송도메인을 구성하는 노드의 수가 증가할수록 방송통신 비용이 증가함을 알 수 있다. 반면에 BToken은 방송도메인의 노드 수가 증가하더라도 방송통신 비용이 거의 변화가 없음을 알 수 있다.

두 번째 실험은 각 프로토콜의 최적의 경우와 최악의 경우의 통신비용을 측정한 것이다. 이 실험을 통해 첫 번째 실험 그래프의 오차범위를 조사할 수 있으며, 각

시스템의 방송결정 시점이 통신비용에 미치는 영향을 조사할 수 있다.

Totem에서 최적의 경우는 토큰을 소유한 노드들이 방송결정을 하는 경우이고, 최악의 경우는 토큰을 이웃 노드로 전송한 시점에 방송결정을 하는 경우이다. BToken에서 최적의 조건은 매번 같은 노드에서 방송결정을 하는 경우이고, 최악의 경우는 매번 다른 노드에서 방송결정을 하는 경우이다.

Totem에서 최적의 경우는 모든 노드들이 토큰을 소유하고, 이웃노드로 전송하기 전에 방송결정을 하는 경우이다. 이 경우의 전송비용은 토큰을 수신하기 위한 전송비용이 0이므로 식(2-1)과 같은 성능을 보이고, 최악의 경우는 토큰을 이웃노드로 전송한 직후에 방송결정을 하는 경우로 식(2-2)와 같은 성능을 보인다.

$$T_{Best} = M\ T_{Bcast} \qquad \text{식(2-1)}$$

$$M(N-1)\ T_{Ptp} + M\ T_{Bcast} <= T_{Worst} < M\ N\ T_{Ptp} + M\ T_{Bcast} \qquad \text{식(2-2)}$$

BToken의 최적의 경우는 매번 하나의 노드에서만 방송결정이 발생하는 경우이다. 즉, $M_d = M$인 경우이고 최악의 경우는 매번 다른 노드에서 방송결정이 발생하는 경우로 $M_d = 0$인 경우이다. 각각의 통신비용을 계산하면 다음 식(2-3)과 식(2-4)와 같다.

$$T_{Best} = M\ T_{Bcast} \qquad \text{식(2-3)}$$

$$T_{Worst} = 2\,M\,T_{Ptp} + M\ T_{Bcast} \qquad \text{식(2-4)}$$

실험에서 사용한 파라미터는 다음 [표 2]의 값을 제외한 모든 내용은 [표 1]과 동일하다.

표 2 최적·최악의 경우의 성능변화 실험을 위한 파라메터

파라미터 내용	파라미터 값
방송도메인 구성 노드 수	4
방송결정 노드	상황에 맞게 조정

[표 2]의 실험 파라메터를 Totem의 성능에 관한 식(2-1)과 식(2-2)에 적용하면, 최적의 경우의 성능은 1000 T_{Bcast}으로 예상할 수 있고, 최악의 경우의 성능은 3000 T_{Ptp} + 1000 T_{Bcast} <= T_{Worst} < 4000 T_{Ptp} + 1000 T_{Bcast}으로 예상할 수 있다. 또한, 식(2-3)과 식(2-4)에 적용하여, Btoken의 최적의 경우에 대한 성능이 1000 T_{Bcast}임을 알 수 있고, 최악의 성능이 2000 T_{Ptp} + 1000 T_{Bcast}임을 알 수 있다.

다음 [그림 7]은 두 시스템의 최적의 경우와 최악의 경우의 순서화 방송메시지의 전송비용을 측정한 시뮬레이션 결과 그래프이다.

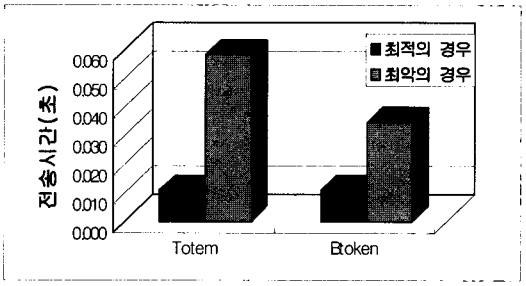

그림 7 최적·최악의 경우에 대한 성능변화 실험 결과 그래프

위 [그림 7]을 통해 Totem의 최적의 경우와 최악의 경우의 통신 비용차이가 BToken에 비해 상당히 크게 발생함을 알 수 있다. 이는 방송결정 시점과 방송결정 노드에 따라 통신비용의 차이가 크게 발생함을 의미한다. 따라서, Totem은 Btoken과 비교하여 사용자들에게 일정한 응답시간을 제공하지 못한다는 단점을 갖고 있음을 알 수 있다.

세 번째 실험은 노드의 고장이 순서화 방송통신에 미치는 영향을 조사하는 것이다. 노드의 고장이 발생하면 Totem은 정상 노드들이 서로 정보를 교환하여 새로운 논리적 링을 구성하고, BToken은 고장을 검출한 노드가 자신의 방송노드정보를 이용하여 고장난 노드를 제외한 가장 최근에 방송한 노드에게 토큰을 요청하여 새로운 토큰을 생성하는 알고리즘을 수행한다.

다음 [표 3]은 실험에 사용된 파라미터를 정리한 것으로 다른 사항들을 [표 1]의 내용과 동일하다.

표 5 노드 고장으로 인한 성능변화 실험을 위한 파라메터

파라미터 내용	파라미터 값
방송도메인 구성 노드 수	노드 9,노드 10 (고장 무), 노드 10 (고장 유)
고장 종류	붕괴고장
고장 발생시간	임의의 시간
고장 노드	Totem : 임의의 노드 BToken : 방송 노드
방송결정 시간	방송메시지 수신 즉시 결정
방송 노드 결정	임의의 노드

실험은 노드 수 9개와 10개에서 고장이 발생하지 않는 경우에 1000개의 방송메시지를 전송하는 비용과 노드 수 10개에서 하나의 노드가 고장이 발생할 경우의 통신비용을 측정하였다.

다음 [그림 7]은 실험 결과 그래프이다.

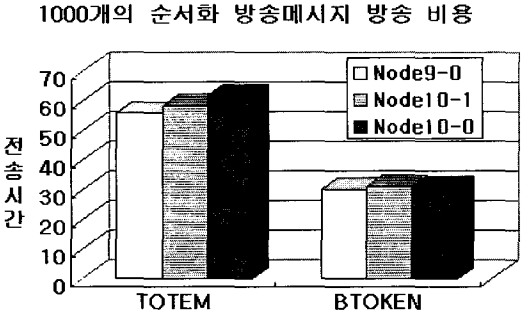

그림 8 노드고장으로 인한 성능변화 실험 결과그래프

위 [그림 8]에서 Node9-0과 Node10-0은 각각 노드 수 9개와 노드 수 10개에서 노드의 고장이 없는 경우를 나타내고, Node10-1은 노드 수 10개에서 임의의 시점에 1개의 노드가 고장이 발생하는 경우를 나타낸다. 위 결과 그래프를 통해 Totem은 노드의 고장이 없는 경우보다 노드의 고장이 발생하는 경우의 통신비용이 더 적게 소요되는 것으로 조사되었다. 이는 노드 고장으로 인해 오버헤드가 적은 것이 아니라 노드 고장으로 인해 방송도메인을 구성하는 노드의 수가 줄어들기 때문에 통신비용이 적게 소요되는 것이다. 즉, 노드의 고장이 발생하는 시점에 따라 통신비용 그래프의 모습이 다르게 나타날 수 있다. 하지만 BToken은 노드 수와 무관한 통신 성능을 나타내기 때문에 노드의 고장이 발생하면 새로운 토큰을 생성하기 위한 오버헤드로 인해 통신비용이 증가한다.

실험 조건을 토대로 Totem의 전송비용을 계산하면 다음과 같다. 단 노드의 고장은 500개의 메시지를 전송하고 발생한다고 가정한다.

$$T_{Node9-0} = 1000(4.5 \cdot T_{Ptp} + T_{Broad}) = 4500T_{Ptp} - 1000T_{Broad}$$

$$T_{Node10-0} = 1000(5 \cdot T_{Ptp} + T_{Broad}) = 5000T_{Ptp} + 1000T_{Broad}$$

$$T_{Node10-1} = 500(5 \cdot T_{Ptp} + 2T_{Broad}) + 9 \cdot T_{Broad} + 18 \cdot T_{Ptp}$$
$$= 4750T_{Ptp} + 1000T_{Broad}) + 9 \cdot T_{Broad} + 18 \cdot T_{Ptp}$$

일반적으로 $T_{Broad} < T_{Ptp}$ 임으로 다음과 같은 식이 성립된다.

$$T_{Node9-0} < T_{Node10-1} < T_{Node10-0}$$

따라서, 실험 결과인 [그림 8]의 Totem 결과 그래프에서 고장 감내 알고리즘의 오버헤드가 노드 수 감소로 인한 토큰 회전 시간 감소로 상쇄됨을 알 수 있다. 또한 노드의 고장이 초반에 발생할수록 노드 수 9의 결과그래프와 유사한 성능을 나타나고, 후반에 발생할수록 노드 수 10의 결과 그래프와 비슷한 성능이 나타난다.

5. 결론 및 향후 연구과제

방송통신 프로토콜을 이용한 분산응용 프로그램 작성은 매우 다양한 분야에서 이루어지고 있다. 이러한 방송통신을 통한 메시지 전송은 모든 수신 노드들에서 모두 같은 순서로 수신함을 보장할 수 있어야 하고, 노드 고장에 대비를 하여야 한다.

이런 문제를 해결하기 위해 널리 사용되는 방법이 순차번호를 이용한 순서화 방송통신 프로토콜이고, 가장 최근에 연구되어 사용되는 시스템이 Totem이라 할 수 있다. Totem은 방송도메인을 구성하는 노드들을 논리적 링으로 구성하고, 이 링을 따라 지속적이면서 순차적인 토큰을 회전시킴으로써 임의의 노드가 방송결정을 하고 토큰을 수신하기 위해 기다리는 지연시간이 필수적으로 발생한다. 하지만, Totem이 논리적 링을 기반으로 하기 때문에 방송도메인을 구성하는 노드들의 수가 증가하면 토큰 수신을 위한 지연시간도 비례적으로 증가한다는 단점을 갖게 된다. 이 문제는 노드의 고장이 발생하여 새로운 링을 생성할 경우도 작용하여 확장성에 큰 문제가 있다.

본 연구에서는 Totem의 확장성 문제를 보완하는 새로운 알고리즘을 제안하고 실험하였다. 본 연구에서 제안한 BToken 방송통신 알고리즘은 방송도메인을 구성하는 노드들을 논리적 링으로 구성하지 않고 토큰 또한 순차적으로 노드들을 회전하지 않는다. 대신에 방송결정을 한 노드가 토큰을 소유하고 있는 노드로 토큰 요청 메시지를 보내어 토큰 전송메시지로 토큰을 소유하도록 하였다.

시뮬레이션을 통해 BToken은 방송도메인을 구성하는 노드 수와 무관한 순서화 방송메시지 전송시간을 갖고 있음을 보였고, 방송결정 상황에도 Totem에 비해 민감하지 않음을 알 수 있었다.

향후 연구과제는 BToken이 방송도메인을 구성하는 모든 노드가 동시에 방송 결정을 하는 특수한 상황에서는 전달 메시지의 길이가 너무 커지고, 비교 대상인 Totem보다 더 많은 오버헤드를 보이기 때문에, 동시에 토큰을 요청하는 노드의 수가 임계값 이상이 될 경우의 처리 방안을 연구하는 것과 시뮬레이션된 BToken 방송

통신 알고리즘을 실제 이더넷과 같은 환경에서 적용할 수 있도록 라이브러리를 만드는 작업이 진행되어야 한다. 현재 인터넷과 분산 응용 프로그램에서 널리 사용되고 있는 JAVA 프로그래밍 언어로 Btoken 방송통신 프로토콜을 구현함으로써 분산 응용 프로그램 작성에 기여하게 될 것이다.

참 고 문 헌

[1] L.C.N Tseung and K.-C. Yu, "Guaranteed Reliable, Secure Broadcast Networks", the 9th International Phoenix Conference on Computer and Communication, pp.576~583, 1990

[2] Jo-Mei Chang and N. F. maxemchuk, "Reliable Broadcast Protocols", ACM Transactions on Computer Systems, Vol.2, No.3, pp.251~273, 1984

[3] S. Navaratnam, S. chanson, and, G. Neufeld, "Reliable Group Communication in Distrinuted Systems", IEEE, pp.439~446, 1998

[4] M. Frans Kaashoek, Andrew S. Tanenbum, et. al., "An Efficient Reliable Broadcast Protocol", ACM Operating System Review, pp.5~19, 1989

[5] M. Frans Kaashoek, Andrew S. Tanenbum, "Group Communication in the Amoeba Distributed Operating System", the 11th International conference on Distributed Computing System, pp.222~230, 1991

[6] Andrew S. Tanenbaum, M. F. Kaashoek, and H. E. Bal, "Using Broadcasting to implement Distributed Shared Memory Efficiently", Reading in Distributed Computer Systems, IEEE, Computer Society Press, pp.387~408, 1994

[7] Y. Amir, L. E. Moser, P. M. Melliar-Smith, D. A. Agarwal, and P. Ciarfella, "The Totem Single-Ring Ordering and Membership Protocol", ACM Transactions on Computer Systems, pp.311~342, 1995

[8] Y. Amir, L. E. Moser, P. M. Melliar-Smith, D. A. Agarwal and P.Ciarfella, "Fast Message Ordering and Membership Using a Logical Token-Passing Ring", Proceedings of the 13th IEEE International Conference on Distributed Computing Systems, 551-560, 1993.

[9] WJ Jia, "Implementation of a Reliable Multicast Ptorocol" , Software-Practice & Experience , V.27 N.7 , 1997.

[10] WJ Jia, JN Cao, XH Jia, CH Lee, "Design and Analysis of an Efficient and Reliable Atomic Multicast Protocol", Computer Communications, V.21 N.1 , 1998

[11] R. Aiello, E. Pagani, and G.P. Roiss, "An Efficient Algorithm for Group Communication", Proceeding of the fifth IEEE Symposium on Parallel and Distributed Processing, pp.226~232, 1993

[12] K.P Birman and T.A. Joseph, "Reliable Communication in the Presence of Failure", ACM Transaction on Computer Systems, Vol.5, No.1, pp.47~76, 1987

한 인

1994년 동국대학교 전자계산학과(공학사). 1996년 동국대학교 대학원 컴퓨터공학과(공학석사). 2000년 동국대학교 대학원 컴퓨터공학과(공학박사). 2001년 ~ 현재 ㈜뱅크타운 기술연구소 시스템체계 연구실장. 관심분야는 분산시스템, 고장감내 시스템, 웹 환경에서의 클러스터링 시스템.

홍 영 식

1973년 서울대학교 응용수학과(공학사). 1975년 한국과학원 수학및 물리학과(이학석사). 1986년 서울대학교 대학원 컴퓨터공학과(공학박사). 1976년 ~ 현재 동국대학교 컴퓨터.멀티미디어공학과 교수. 1983년 ~ 1984년 미국 Rensselaer Polytechnic Institute 방문교수/강사. 1989년~1990년 미국 University of North Carolina-Charlotte 방문교수. 관심분야는 분산처리, 객체지향 분산 실시간처리, 고장감내시스템, 암호알고리즘