이동통신망에서의 효율적인 인과순서 알고리즘의 설계 및 평가

장익현
동국대학교 정보통신공학과
e-mail:ihjang@dongguk.ac.kr

Design and Evaluation of Efficient Causal Order Algorithm in Mobile Network

Ik-Hyeon Jang
Dept. of Info. & Comm. Eng., Dongguk Univ.

요약

본 논문에서는 이동통신망에서의 효율적인 인과순서 알고리즘을 채널전환 알고리즘과 함께 제안한다. 인과순서를 유지하기 위해서는 순서를 유지하기 위한 제어정보를 교환하여야 하며, 송수신 양측은 메시지를 보내거나 받은 전후에 각각의 제어정보를 수정하여야 한다. 특히 이동유닛이 다른 셀로 이동할 경우, 채널전환 프로토콜은 기존의 기지국과 새로운 기지국에서 동시에 제어정보 관리를 위한 작업을 수행하여야 하며, 기존의 기지국에 있는 이동 유닛의 제어정보는 새로운 기지국으로 전달되어야 한다. 따라서 제어정보의 크기가 채널전환시간과 메시지 지연시간에 직접적인 영향을 주기 때문에, 전송되는 제어정보의 양을 최소화하여야 한다. 본 논문에서는 제어정보의 양을 줄이기 위하여 유효한 통신패턴을 분석하여 중복으로 교환되는 제어정보가 최소화되는 인과순서 알고리즘을 제안하였으며, 모의실험을 통하여 제안된 알고리즘이 기존의 알고리즘에 비해 효율적임을 보였다.

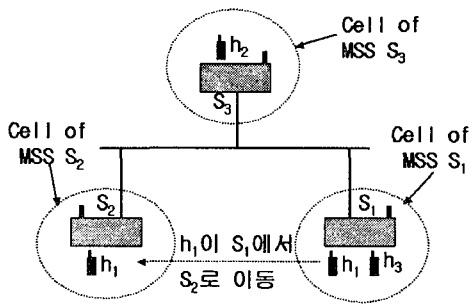
1. 서론

컴퓨터의 소형화 및 경량화가 진행됨에 따라 무선네트워크를 통하여 서비스를 제공할 수 있는 이동통신망 (mobile network)이 현실화되고 있다. 이동통신망은 [그림 1]과 같이 이동유닛(mobile host)과 고정유닛(fixed host)으로 구성된 시스템으로 셀(cell)이라는 범위로 구분되어 있고 각 셀에는 하나의 기지국(mobile support station, MSS)이 있어서 이동유닛과 네트워크 사이의 통신을 담당한다[1,2,5].

이동통신시스템은 분산시스템의 일종으로 여러 프로세스들이 서로 메시지를 교환하면서 공조하여 하나의 작업을 수행한다. 그러나 메시지 전송시간은 유한하지만 전달시간의 차이 때문에 송신하는 메시지와 전달되는 메시지들의 순서가 일치하지 않을 수 있다. 이런 메시지 순서에 관련된 불일치성을 해결하기 위해 인과순서(causal order)가 제안되었다[4].

분산시스템에서의 인과순서화는 Lamport가 제안한 *happened before* 관계에 기초를 두고 있으며, 두 개의 메시지가 서로 관련되어 있고 같은 목적지를 가진다면 송신한 순서와 같은 순서로 목적지에 전달할 것을 요구한다[4,7].

인과순서와 관련된 주요 이슈는 메시지 간의 인과관계를 파악하는 방법과 인과관계를 유지하기 위해 교환되는 제어정보의 양을 최소화하는 방법에 있으며 이와 관련된 많은 연구결과가 있다[4,6,8,9]. 본 논문에서는 통신패턴을 분석하여 중복되는 제어정보의 전송을 억제하여 인과순서를 유지하기 위하여 필요한 제어정보의 양을 최소화하는 알고리즘을 제안한다.



[그림 1] 이동통신망

한편 분산시스템을 위해 설계되었던 인과순서 알고리즘을 이동통신시스템에 적용하기 위해서는 고정유닛과 이동유닛 간의 통신대역폭, 이동유닛의 배터리의 용량 등을 고려해야 한다. 이동유닛은 셀과 셀

사이를 빈번하게 이동하므로 채널전환(handoff)을 효율적으로 처리하는 프로토콜이 필요하다[3]. 이동 통신시스템에서 이동유닛은 셀 사이를 이동함에 따라서 다양한 접속지점을 가지며 접속지점에 따라 서로 다른 대역폭을 가지는 통신 채널을 부여받게 된다. 일반적으로 이러한 통신채널의 대역폭은 유선통신 채널의 대역폭보다 적으므로 인과순서화를 위해 메시지와 함께 전송되는 제어정보의 양을 줄임으로써 주어진 적은 대역폭의 채널에서 통신지연시간을 줄일 수 있으며 전달되는 메시지의 양을 증대시킬 수 있다. 인과순서를 유지하기 위하여 필요한 제어정보를 이동유닛이 관리하는 경우에는 메시지를 보낼 때마다 이 정보를 기지국에 전달하여야 하므로 이동유닛의 대역폭이 낭비되고 부하가 커지게 된다. 본 논문에서는 제어정보를 각각의 이동 유닛이 속한 기지국에서 관리하도록 하는 방법을 제안한다.

본 논문의 구성은 먼저 인과순서의 개념을 설명한 후에 새로운 알고리즘을 제안한다. 그 후에 기존 알고리즘과 성능을 비교분석하며 마지막에 결론을 맺는다.

2. 인과순서

분산시스템은 여러 프로세스로 구성되어 있고 이 프로세스들은 메시지 송수신을 통해 통신한다. 분산시스템에서 프로세스의 실행은 메시지 전송 이벤트, 메시지 전달 이벤트, 내부 이벤트의 세 가지로 구성되나 내부 이벤트는 메시지 전송과는 관계없다. send(m)을 메시지 m을 전송하는 이벤트, deliver(m)을 m을 전달하는 이벤트라고 한다면 메시지 전송과 관련된 모든 이벤트는 *happened-before* 관계에 의하여 순서를 정할 수 있게 된다[7].

[정의] (인과순서화) : 메시지 m_1과 m_2가 같은 목적지를 가지고 있으며 send(m_1) → send(m_2)이면 deliver(m_1) → deliver(m_2)이다. (여기서 기호 → 은 *happened-before* 관계를 의미한다.)

메시지의 인과순서 전달은 Birman에 의해 ISIS 시스템을 위하여 처음 제안되었다[4].

3. 알고리즘

3.1 인과관계의 정리

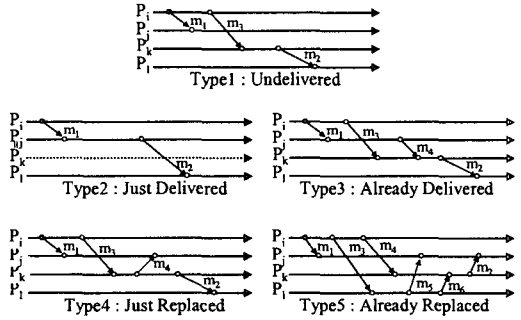
인과관계의 파악은 제어정보를 정리하는 방법에 따라 두 가지로 분류할 수 있다. 첫 번째 방법은 목적프로세스에 따라 인과관계를 정리하는 것으로 지금까지 대부분의 알고리즘은 이 부류에 속한다. 이 방법은 전달조건의 검사가 쉽다는 장점을 가지고 있으나 인과관계들 간의 우선 순위를 파악하기 위해서는 전체 제어정보를 확인해야 하는 어려움이 있다. 두 번째 방법은 송신프로세스를 기준으로 인과관계를 정리하는 것으로 전송되는 메시지와 그와 함께 전송되는 제어정보를 하나의 객체로 처리한다. 이 방법에서는 목적프로세스를 기준으로 인과관계를 검사해야 하는 전달조건의 검사는 상대적으로 어려우나 우선 순위 관계를 파악하는 것이 쉽다.

본 논문에서는 제어정보들 간의 우선 순위를 찾는 것이 중요하므로 두 번째 방법을 채택하며 전달조건의 확인을 빨리 하기 위하여 메시지 송신 시에 전달조건 확인을 위한 정보를 목적프로세스를 기준으로 별도로 추출하도록 한다.

3.2 추상통신패턴

전송되는 메시지 m에 대한 정보를 가지는 상태를 송신프로세스와 목적프로세스를 기준으로 분류하면 모든 프로세스는 다음 4가지 중 한 상태에 속한다.

1) 송신프로세스의 상태
2) 목적프로세스의 상태
3) 송신과 목적프로세스 사이의 프로세스의 상태
4) 목적프로세스 이후에 있는 프로세스의 상태



[그림 2] 추상통신패턴 (P_k의 m_2 전송 시점에서 보는 m_1에 대한 정보의 흐름을 기준으로 분류)

만약 두 개의 프로세스가 동일한 상태 정보를 가지고 있다면 그 두 프로세스는 메시지 m에 관한 정보에 대해서는 하나의 프로세스로 볼 수 있다. 따라서 분산시스템은 4개의 프로세스로 이루어진 시스템으로 변환이 가능하며, 어떤 메시지에 관한 정보의 흐름 관점에서 보면 다중전송(multicast)도 단일전송(unicast)으로 변환되므로 인과순서화를 제공하는 유효한 모든 통신패턴은 [그림 2]에 있는 추상통신패턴 중의 하나로 변환이 가능하다[6].

3.3 중복정보의 제거

만약 [그림 2]의 Type2 프로세스들이 어떤 이유로 메시지 m_1에 대한 정보를 가지고 있어야 한다면 그 정보는 인과순서를 유지하는데 필수적이지 않으므로 중복정보로 취급된다. 즉 중복정보는 Type2부터 Type5 사이에 해당하는 프로세스들이 인과순서를 유지하는데 필수적이지 않은 정보를 가지고 있을 경우로 이러한 정보의 파악과 제거는 인과순서를 이용한다.

동일한 프로세스에서 전송된 메시지들 간의 인과순서는 송신순서를 이용하면 쉽게 파악된다. 하나의 프로세스에는 동일한 프로세스에서 전송한 메시지에 대해서 최종 메시지에 대한 정보만 유지하면 되므로

나머지 메시지에 대한 정보는 중복정보가 된다.

중복정보는 전송된 메시지에 부속된 제어정보와 각 노드에 있는 제어정보를 비교하여 파악하며 인과순서 상 가장 뒤에 있는 메시지에 대한 정보만 유지하도록 하여 중복정보를 제거한다. 단 이 경우 모든 송신 노드에 대해 최소한 하나의 메시지에 대한 정보는 남겨 두어서 이후에 사용할 수 있도록 한다.

3.4 시스템 모델과 자료구조

이동통신시스템은 n_h개의 이동유닛의 집합 H={h_1, h_2, ..., h_{nh}}와 n_s 개의 기지국 집합 S = {s_1, s_2, ..., s_n}로 이루어진다고 가정한다. 이동유닛 h_i가 기지국 s_j의 셀안에 있을 때 이동유닛의 통신 및 처리에 대한 부하를 줄이기 위하여 인과순서화에 관한 제어정보는 s_j가 관리하며 h_i가 다른 이동유닛으로 메시지를 보내거나 받을 때의 인과순서화 알고리즘은 기지국 s_j에서 수행된다.

제어정보는 (k, τ)의 형태로 송신프로세스 별로 분류하여 노드 i에서는 $CI_i[j]$로 관리하며 메시지를 보낼 때마다 함께 보낸다. 이때 목적지에서의 전달조건을 확인하기 위한 제어정보는 분리하여 DCm[j]를 만들어 메시지와 함께 보낸다. 여기서 (k, τ)는 프로세스 k가 τ시간에 보낸 메시지이고 DCm[j]는 CI 중에서 목적지가 j인 제어정보를 별도로 추출한 것이다. 또 각 노드에는 프로세스 k로부터 프로세스 i에 전달된 최종 메시지의 송신시간을 나타내는 $DLV_i[k]$도 유지한다. 한편 MH_SNUM_i은 기지국이 이동유닛과의 메시지 송수신을 관리하기 위한 것이고 MH_RNUM_i 는 h_i가 기지국과의 메시지 송수신을 관리하기 위한 변수이다. MH_PEND_i는 전달조건을 만족시키지 못한 메시지들을 저장하는 큐이며 $PEND_ACK_i$는 기지국이 hi로 데이터를 전송한 후 ACK가 올 동안 메시지를 저장하는 큐이다.

어떤 프로세스 P_i에 도착한 메시지는 메시지와 함께 전송된 제어정보에 포함된 인과관계 중에서 P_i를 목적지로 하는 메시지들이 모두 P_i에 전달되었다는 것이 확인되면 P_i에 전달될 수 있다. 이 전달조건이 만족되지 않으면 해당 메시지는 조건이 만족될 때까지 목적프로세스에 전달되지 않고 대기하여 인과순서를 유지하도록 하며, 이는 다음 식으로 표시된다.

$$\forall (k, \tau) \in DC_m[i] : \tau \leq DLV_i[k]$$

3.5 인과순서 알고리즘

이동유닛 h_i가 기지국 s_j의 셀에 있다고 가정하며 송신알고리즘은 이동유닛 h_i로부터 메시지 m을 받은 기지국 s_j가 메시지를 목적지 노드로 보내기 위하여 수행하며, 수신알고리즘은 이동유닛 h_j로부터 h_i로 전송된 메시지 m을 받은 기지국 s_j가 수행한다.

가. 송신알고리즘

프로세스 i가 j로 메시지 m과 CIi, DC_i를 송신

1) $\tau_i = \tau_i + 1$;
2) CI_i에서 DCm[j]를 분리
3) CI_i에서 송신지별 최종메시지 정보만 남김
4) τ_i, CI, DC를 메시지와 함께 j로 송신
5) (τ_i, j)를 CI_i에 포함시킴
6) $MH\_SNUM_i = MH\_SNUM_i + 1$

나. 수신알고리즘

프로세스 i가 j로부터 m, CI_m, DC_m 수신

1) 전달조건이 만족될 때까지 대기
   전달조건이 만족되면 메시지 전달
2) 동일 송신지에서 보낸 메시지에 대한 중복정보 제거
   - CI_m과 CI_i에 공존하는 정보는 남김
3) CI_m과 CI_i를 비교하여 이미 전달 완료된 메시지에 대한 정보 삭제
4) $CI_i = CI_i \cup CI_m$
5) 동일목적지에 대한 정보가 중복으로 있을 경우 최종 메시지에 대한 정보만 남김
6) CI_i에서 송신지별 최종메시지 정보만 남김

3.6 채널전환 알고리즘

채널 전환시에 기지국들 사이에서 이동되는 메시지들은 다음과 같으며, 이동유닛 h_i가 기지국 s_j로부터 새로운 기지국 s_k로 이동해 왔다고 가정한다.

. register(h_i,s_j,MH_RNUM_i) : 이동유닛 h_i가 기지국 s_j의 셀을 벗어나 새로운 셀에 도착해서 그 셀의 기지국에 자신이 들어왔음을 알리는 메시지

. begin_handoff(h_i) : 새로운 셀의 기지국이 기지국 s_j에 채널 전환 시작을 알리는 메시지

. end_handoff(h_i,MH_SNUM_i) : 기지국 s_j가 새로운 셀의 기지국에게 채널전환 종료를 알리는 메시지

가. 기지국 s_k에서 수행되는 작업

1) h_i에서 register(h_i,s_j,MH_RNUM_i) 수신시 s_j에게 begin_handoff(h_i) 메시지를 보냄
2) s_j에서 DLV_i,CI_i,MH_PEND_i,$PEND\_ACK_i$ 수신
3) end_handoff(h_i,MH_SNUM_i) 메시지 수신 시 $PEND_ACK_i$내의 메시지 번호가 MH_RNUM_i 보다 큰 메시지를 h_i에게 전송
4) h_i가 보낸 메시지 중 메시지번호가 MH_SNUM_i보다 큰 메시지는 목적지로 전송

나. 기지국 s_j에서 수행되는 작업

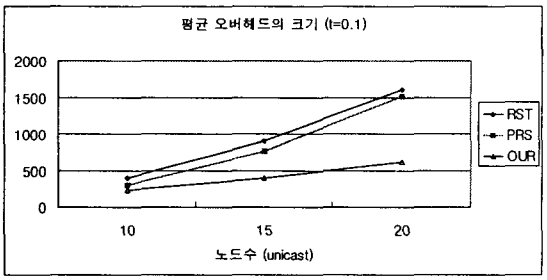
1) s_k로부터 begin_handoff(h_i) 수신시 DLV_i, CI_i, MH_PEND_i, $PEND_ACK_i$를 s_k에게 보낸 후 end_handoff(h_i,MH_SNUM_i) 메시지 전송

이동유닛 h_i가 다른 셀로 옮겨가는 경우 채널전환하는 동안에 $PEND_ACK_i$에 저장된 메시지들은 h_i에게 보내지고 h_i도 이전의 기지국으로부터 ACK를 받지 못한 메시지들을 새로운 기지국에 다시 보낸다.

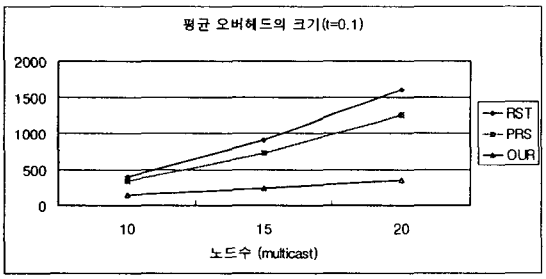
4. 성능 평가

시뮬레이션에서는 시스템 전체의 구성이 이동유닛과 기지국으로만 구성되어 있다고 가정하며 유선통신 채널의 대역폭은 100Mbps이고 무선통신 채널의 대역폭은 1Mbps라고 가정하였으며 두 개의 메시지 전송 이벤트 사이의 시간 간격은 평균을 t로 하는 지수분포를 따른다고 가정하였다.

시뮬레이션에서는 t와 이동유닛의 수를 10, 15, 20 으로 변화시키고 t도 0.1, 0.5, 1.0, 1.5초 등의 여러 경우에 대하여 오버헤드의 크기, 전송지연시간 등에 대해 기존의 RST 알고리즘[2,9], PRS 알고리즘[8]과 비교하였다. 그 중 오버헤드의 크기에 대한 결과의 일부를 [그림 3]과 [그림 4]에서 보여주고 있다. [그림 3]과 [그림 4]의 결과를 통해 제안한 알고리즘이 기존의 RST 알고리즘이나 PRS 알고리즘에 비해 훨씬 적은 오버헤드를 가진다는 것을 볼 수 있다. 이것은 유니캐스트 상황에서만이 아니라 멀티캐스트가 되면 더욱 우수한 결과를 보이는 것도 알 수 있다.



[그림 3] 유니캐스팅에서의 오버헤드 크기



[그림 4] 멀티캐스팅에서의 오버헤드 크기

5. 결론

이동통신망에서 효율적인 인과순서 알고리즘은 이동유닛의 제한된 자원 상황을 고려할 경우 가능하면 적은 오버헤드를 가지도록 하는 것이 바람직하다. 본 논문에서는 추상통신패턴의 개념을 도입하여 중복되는 제어정보의 보유를 최소화하였다. 이렇게 설계된 알고리즘은 인과순서를 유지하기 위해 최소한의 제어정보만을 가지게 되므로 이동통신망에서 효율적으로 동작할 수 있다. 채널전환 알고리즘도 이동유닛 대신에 기지국에서 자원을 관리하도록 하여 이동유닛의 한정된 자원 사용을 반영할 수 있도록 하였다.

제안한 알고리즘의 성능을 시뮬레이션을 통해 기존의 다른 방법과 비교한 결과 유니캐스트와 멀티캐스트 통신 모두에서 월등히 적은 오버헤드를 가짐을 알 수 있었다.

참고문헌

[1] A. Acharya and B. Badrinath, "Delivering Multicast Messages in Networks with Mobile Hosts," Proc. 13th Int'l Conf. Distributed Computing Systems, pp.292-299, Pittsburgh, 1993.

[2] S. Alagar and S. Venkatesan, "Causal Ordering in Distributed Mobile Systems," IEEE Trans. on Computer, vol.46, no.3, pp.353-361, Mar. 1997.

[3] B. Badrinath, A. Acharya, and T. Imielinski, "Impact of Mobility on Distributed Computations," ACM Operating System Review, vol.27, no.2, pp.15-20, 1993.

[4] K. Birman and T. Joseph, "Reliable Communication in the Presence of Failure," ACM Trans. Comp. Syst., pp.47-76, 1987.

[5] J. Ioannidis, D.Duchamp, and G. Maguire, "IP-Based Protocols for Mobile Internetworking," proc. ACM SIGCOMM Symp. Comm. Architecture and Protocols, pp.235-245, Zurich, 1991.

[6] I. Jang, J. Cho, and H. Yoon, "An Efficient Causal Multicast Algorithm for Distributed System," IEICE Trans. on Info. & Syst., vol.E81-D, no.1, pp.27-36, 1998.

[7] L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," Comm. of the ACM, vol.21, no.7, pp.558-564, Jul. 1978.

[8] R. Prakash, M. Raynal, and M. Singhal, "An Adaptive Causal Ordering Algorithm Suited to Mobile Computing Environments," Journal of Parallel and Distributed Computing, vol.41, no.2, pp.190-204, Mar. 1997.

[9] M. Raynal, A. Schiper, and S. Toueg, "The Causal Ordering Abstraction and a Simple Way to Implement It," Information Processing Letters, vol.39, no.6, pp.343-350, 1991.