

이동 환경을 위한 비관적 메시지 로깅 기법

김치연*, 배석찬**

*목포해양대학교 해양전자통신공학부

**군산대학교 컴퓨터학과

e-mail : gegujang2@mmu.ac.kr

A Pessimistic Message Logging Protocol for Mobile Environments

Chi-yeon Kim*, Suk-chan Bae**

*Faculty of Maritime Electronic & Communications Engineering
Mokpo Maritime University

**Dept of Computer Information Science, Kunsan National University

요 약

이동 환경에서 회복 기법은 이동 환경을 구성하는 요소들의 낮은 신뢰성을 극복하기 위해 반드시 필요하다. 이 논문에서는 비관적 메시지 로깅 기법을 이용한 회복 프로토콜을 제안한다. 제안하는 방법은 이동 호스트와 MSS에 두 종류의 검사점을 유지하되, 이동 호스트의 부담을 줄이기 위하여 송신 메시지에 관한 정보만 이동 호스트에 로깅한다. 또한 검사점을 취할 때 시간 간격을 기준으로 사용하지 않고, 로깅된 메시지의 수에 따라 검사점을 결정함으로써 저장되는 검사점의 수를 줄일 수 있다. 또한, 제안하는 방법은 고아 프로세스를 발생시키지 않음으로써 회복 후 일관된 전역 상태를 유지할 수 있다.

1. 서 론

이동 컴퓨팅 환경은 이동 호스트와 고정 호스트로 구성되어 있다[1, 5]. 이동 호스트의 위치는 시간에 따라 변할 수 있으며 고정 호스트는 시스템 내에서 정적인 네트워크에 연결되어 있다. 이동 호스트는 무선 인터페이스를 갖춘 고정 호스트를 통하여 다른 호스트와 통신할 수 있다. 이동 환경이 좀 더 견고하게 작동되기 위해서는 이동 호스트와 무선 망의 낮은 신뢰성을 극복할 수 있는 방법을 필요로 한다.

분산 시스템에서 제안되었던 회복 기법[9]은, 무선망의 낮은 대역폭과 저장 공간의 한계, 호스트의 이동성, 그리고 단절과 같은 이동 환경의 고유한 특징들 때문에 그대로 적용할 수 없다.

분산 시스템에서 사용된 회복 기법은 검사점을 이용한 방법과 메시지 로깅을 이용한 방법으로 분류할 수 있다. 검사점 방법은 고장이 없는 동안 수행된 프로세스의 상태를 주기적으로 안정된 기억장치에 저장하여 고장이 발생하였을 때, 저장된 상태 중의 하나로 롤백함으로써 고장 전의 상태로 회복하는 방법이다. 검사점 방법은 조정된(coordinated) 검사점과 비조정(uncoordinated) 검사점, 그

리고 communication-induced 방법으로 분류할 수 있다 [10]. 조정된 검사점 방법은 모든 프로세스들이 하나의 일관성 있는 전역 상태를 형성하기 위해 협동하는 것을 요구하기 때문에 부가적 메시지 교환이 발생한다. 비조정 검사점 방법은 각 프로세스가 독립적으로 검사점을 취하게 하여 프로세스의 오버헤드를 줄일 수 있다. 하지만 도미노 효과의 발생가능성이 있고, 불필요한 검사점이 포함될 수 있다. communication-induced 방법은 프로세스가 응용 메시지에 piggyback된 정보를 이용하여 회복하는 방법이다.

메시지 로깅 방법은 검사점과 로그의 기록을 함께 병행하는 방법으로, 비관적, 낙관적, 그리고 causal 로깅 방법으로 분류할 수 있다. 비관적 방법은 메시지를 송신하거나 수신하기 전 로그에 먼저 기록하는 방법으로 고아 프로세스가 발생하지 않는다. 낙관적 방법은 일단 소멸성 기억장치에 로그를 유지한 후 주기적으로 안정된 기억장치에 기록하는 방법이다. causal 방법은 비관적 방법과 낙관적 방법을 혼합한 방법이다.

이동 환경은 분산 환경에 비하여 자원이 빈약하므로 새로운 회복 방법이 필요하다. [3]에서는 이동 환경에서 메시지 로깅이 적합함을 언급하였다. 이 논문에서는 비관적 메시지 로깅을 이용한 회복 알고리즘에 대하여 제안한다.

제안하는 방법은 비판적 메시지 로깅 방법을 사용하여 고아 프로세스가 발생하지 않고, 이동 호스트에서 검사점을 취할 때, 메시지 수에 기반하여 검사점을 취함으로써 잦은 검사점으로 인한 성능 저하를 막고, 이동 호스트에 필요한 정보만 로깅함으로써 저장 공간에 대한 부담을 줄였다.

이 논문의 구성은 다음과 같다. 2장에서는 회복을 다룬 기존의 연구와 회복 프로토콜의 일관성 조건에 대하여 기술하고, 3장에서 시스템 모델을 기술한다. 4장에서는 제안하는 회복 알고리즘을 기술하며, 5장에서는 결론을 기술한다.

2. 관련 연구

[2]에서는 이동 호스트에서 발생가능한 고장의 유형을 임시적인 소프트 고장과 복구할 수 없는 하드 고장으로 분류하였다. 소프트 고장으로부터 회복하기 위해 이동 호스트에는 지역적인 소프트 검사점을 저장하고, 하드 고장으로부터 회복을 위해 MSS의 안정된 기억장치에 하드 검사점을 저장하였다. [2]에서 제안한 방법은 조정된 검사점을 이용한 방법으로 모든 호스트의 동기화를 피하기 위해 지역 타이머를 사용하고 검사점 번호를 이용한 재동기화 메커니즘을 통해 호스트들의 클럭을 동기화하였다.

[3]에서는 프로세스의 상태 정보가 이동 호스트가 현재 연결되어 있는 컴퓨터 내에 저장되는 비조정 검사점 스킴을 제안하였다. 이 방법은 검사점 생성을 위한 프로세스들의 조정은 필요하지 않지만 상당히 많은 수의 검사점이 생성된다는 단점이 있다.

[4]에서는 2단계 규칙을 이용하여 언제 프로세스가 상태를 저장할 필요가 있는지를 결정하였다. 2단계 규칙은 프로세스가 메시지를 보낸 후에 메시지를 받을 때마다 새로운 검사점을 생성해야 한다. 프로세스들은 또한 이동 호스트가 쉘을 이동할 때마다 검사점을 생성해야 한다. 이 방법에서는 프로세스가 교환한 모든 메시지들을 기록되어야 하며, 이동 호스트의 이동에 따라 검사점은 분산되어 저장된다.

[7, 8]에서는 메시지 로깅을 이용한 회복 방법을 제안하였다. [7]에서는 메시지 로깅의 세 가지 기법의 특징에 대하여 기술하고, 오버헤드가 없는 최적의 로깅 방법에 대하여 기술하였다. [8]에서는 이동 환경에 적합한 수신자 기반의 비판적 로깅 프로토콜을 기술하였다.

회복 프로토콜은 일관성 있는 전역 상태를 보장할 수 있어야 한다. 이를 위해 가장 많이 사용되는 기준은 고아 프로세스의 존재 여부이다. 이를 위해 전역 상태와, in-transit 메시지, 고아 프로세스의 대한 정의를 기술한다.

(정의 1) 전역 상태 (Global State)

메시지 전송 시스템에서 전역 상태란 통신 채널의 상태와 수행중인 모든 프로세스의 개별 상태의 집합이다.

(정의 2) in-transit 메시지

전역 상태에 메시지가 보내진 것은 기록되었으나 아직 도착이 기록되지 않은 메시지를 말한다. in-transit 메시지는 전역 상태의 일부이며 어떤 비밀관성도 야기하지 않는다.

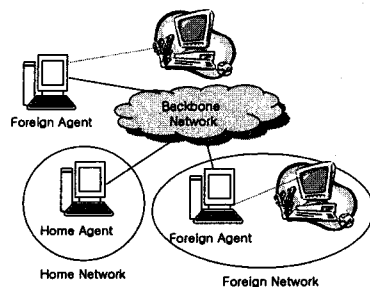
(정의 3) 고아 프로세스 (Orphan Process)

고장이 발생한 프로세스의 회복된 상태와 비밀관적인 상태를 가진 프로세스.

메시지 로깅 프로토콜은 반드시 회복 후 고아 프로세스가 없다는 것을 보장할 수 있어야 일관된 전역 상태를 유지할 수 있다.

3. 시스템 모델

이 논문에서 사용하는 시스템 모델은 (그림 1)과 같이 [11]에서 정의된 Mobile IP 구조를 사용한다. 이 구조에는 무선 인터페이스를 사용하는 이동 호스트(MH : Mobile Host)와 유선 망에 연결되어 있는 고정 호스트(FH : Fixed Host)가 있다. 고정 호스트 중 이동 호스트와 통신할 수 있는 무선 인터페이스를 갖춘 호스트를 이동 기지국(MSS : Mobile Support Station)이라 한다. 각 이동 호스트는 홈 주소를 할당받고 홈 주소는 이동 호스트가 이동하더라도 변하지 않는다. 하나의 MSS에 의해 제어되는 지리적인 영역은 셀이라 한다. 이동 호스트의 홈 에이전트(HA : Home Agent)는 이동 호스트의 홈 네트워크 상에서 수행된다. 홈 에이전트는 이동 호스트의 위치 정보를 관리하며, 이동 호스트가 이동할 때 현재의 셀에 있는 MSS를 위해 에이전트(FA : Foreign Agent)라 한다. 이동 호스트로 보내지는 정보들은 이동 호스트의 홈 에이전트에 일단 보내지고, 홈 에이전트는 현재의 이동 호스트의 위치를 파악하여 외래 에이전트에 전달한다.



(그림 1 : Mobile IP 환경)

이 논문에서는 다음과 같은 가정을 사용한다.

- 프로세스들은 PWD(Piece-wise Deterministic) 가정에 따른다. : 메시지 로깅 프로토콜은 프로세스가 수행한 모든 비결정적(nondeterministic) 이벤트를 식별할 수 있고 회복 동안 각 이벤트들을 재수행할 때 필요한 정보를 로깅한다. 이 정보를 결정자(determinant)라 한다. 메시지 로

킹 프로토콜에서는 비결정적 이벤트들을 고장 전과 동일한 순서로 재수행함으로써 프로세스의 상태를 결정적으로 재생성할 수 있다. 비결정적 이벤트의 예로는 프로세스가 입력이나 메시지, 또는 인터럽트를 받는 것 등을 들 수 있다. 메시지의 송신은 비결정적 이벤트가 아니다.

- 프로세스들은 fail-stop 모델에 따른다. : 고장이 발생하면 프로세스는 정지된다.

- 이동 호스트는 MSS와 통신할 때 FIFO 링크를 사용한다. : 이동 환경의 특성상 통신 채널의 지연은 존재한다. 하지만 이 가정으로 MH와 MSS는 동일한 순서로 메시지를 주고 받을 수 있다.

- MSS와 홈 에이전트는 이동 호스트보다 신뢰성이 있다. : 이동 호스트에서 자주 발생하는 고장은 고정 호스트에서는 드물게 발생할 수 있고, 이동 호스트의 고장에 초점에 두기 위하여 고정 호스트의 신뢰성을 가정하였다.

4. 회복 프로토콜

제안하는 회복 프로토콜은 이동 호스트와 MSS에서 수행되는 알고리즘으로 구성되어 있다. 이동 환경에서 바람직한 회복 알고리즘은 검사점의 수행 동안 다른 작업의 수행을 방해하지 않아야 하며, 최소의 비용으로 일관성 있는 상태를 수집할 수 있어야 한다. 또한 회복시 재수행되는 작업의 양을 줄일 수 있어야 한다. 이 논문에서는 메시지 수에 기반하여 검사점을 취함으로써 저장되는 정보의 양을 줄이고, 롤백되는 프로세스의 양에 한계를 줄 수 있다.

4.1 이동 호스트 알고리즘

각 이동 호스트에는 하나나 그 이상의 프로세스가 동시에 수행될 수 있고, 이동 호스트의 고장은 물리적 고장이나 예상된 단절을 포함한다. 이동 호스트는 제한된 배터리 소모를 줄이기 위해 연산하지 않는 동안 MSS와의 연결을 끊을 수 있는데, 이를 단절이라 한다[6]. 이동 호스트에서는 독립적으로 지역 검사점을 저장하는데, 언제 검사점을 취할 것인지는 이동 호스트의 특징에 따라 결정된다. 즉, 이동 호스트가 보낸 메시지 수를 기준으로 일정 개수가 넘으면 지역 검사점을 취한다. 검사점을 취할 메시지의 한계는 각 이동 호스트별로 결정할 수 있다. 이렇게 함으로써 시간을 기준으로 검사점을 취한 방법들에 비해 검사점 수행으로 연산이 불특되는 경우를 줄일 수 있다. 즉, 일정 시간 간격으로 검사점을 취하게 되면 작업의 양과는 무관하게 검사점을 취하게 되나, 메시지 수를 기반으로 검사점을 취하게 되면 메시지 전송이 드문 경우, 검사점 수행 횟수는 감소한다. 지역 검사점에 저장되는 정보는 이동 호스트가 다른 호스트에게 전송하려고 하는 메시지의 결정자이다. 결정자는 (송신 프로세스 번호, 메시지 번호, 수신 프로세스 번호)로 구성된다. 다른 호스트에 의해 이동 호스트로 전송되는 메시지는 해당 MSS를 통해 홈 에이전트

를 거쳐 전달되므로 이동 호스트의 홈 에이전트에 로깅될 수 있다.

이동 호스트는 주기적으로 홈 에이전트의 기억 장치에 지역 검사점을 저장한다. 이를 전역 검사점이라 한다. 전역 검사점을 취하는 이유는 회복시 롤백되는 양을 줄이기 위해서이다. 이동 호스트가 홈 네트워크에 없는 경우에는 외래 에이전트에 임시로 복사한 후, 외래 에이전트에 의해 홈 에이전트로 전달된다. 이로써 이동 호스트의 검사점 정보는 분산되지 않고 홈 에이전트에만 저장된다.

프로세스에서 고장이 발생하면, 이동 호스트는 현재 위치한 셸의 MSS에게 고장을 알린다. 이동 호스트가 현재 홈 네트워크에 있지 않다면 외래 에이전트에 의해 홈 에이전트로 고장 발생이 알려진다. 이동 호스트의 고장으로 이동 호스트에 지역적으로 저장된 정보는 이용할 수 없게 되고, 홈 에이전트에 저장된 로그를 찾아 재수행한다. 이동 호스트의 의도된 단절의 경우에는 단절되기 전 검사점을 홈 에이전트에 저장한다. 단절 후에는 지역적인 검사점을 계속 생성하고, 재연결시 홈 에이전트에 검사점 정보를 전송한다. 홈 에이전트에는 각 이동 호스트를 위한 마지막 검사점만 유지되므로 저장 공간에 대한 부담이 없다. (그림 2)는 이동 호스트 알고리즘을 기술한 것이다.

```
// 메시지 로깅
// msgLimit 설정
msgCnt=0
When MH is to send a message {
    SAVE (srcPID, msgID, destPID) to local disk;
    send a message to MSS;
    msgCnt++;
    Periodically, send a local checkpoint to HA;
}
// 지역 검사점 수행
if (msgCnt == msgLimit) {
    take a checkpoint CPI;
    msgCnt=0; //초기화
}
// 고장의 발생
send a failMesg to MSS; // 고장을 알림
if (current MSS is not HA of MH)
    forward failMesg to HA;
receive a last checkpoint record from HA;
replay last checkpoint record;
// 예상된 단절
take a global checkpoint;
```

(그림 2) 이동 호스트 알고리즘

4.2 MSS 알고리즘

회복과 관련하여 MSS의 주요한 기능은 이동 호스트를 대신하여 안정된 검사점 정보를 저장하는 것이다. 여기에 추가하여, 외래 에이전트로서 홈 에이전트로부터 이동 호스트에 전달될 메시지를 전달한다. 고장을 홈 에이전트에

알리며, 현재 셀의 이동 호스트의 검사점 정보를 임시 저장한 후 홈 에이전트에게 전달한다. (그림 3)은 MSS 알고리즘을 기술한 것이다.

```
// 전역 검사점 수행
When MH requests global checkpoint,
  take a global checkpoint;
// 고장발생의 탐지
When FA receives a failMesg from MH,
  forward the failMesg to MH's HA;
When HA receive a failMesg {
  send a last checkpoint to MH's FA;
}
// MH로 보내질 메시지의 포워드
When HA receives a msg destined for MH {
  logging determinant;
  forward MH's FA;
}
```

(그림 3) MSS 알고리즘

5. 결론

이 논문에서는 이동 환경을 위한 비관적 메시지 로깅 기법을 제안하였다. 이동 환경은 분산 환경보다 유연성이 뛰어나지만 구성 요소의 비신뢰성은 고장에 대비한 회복 프로토콜을 요구하고 있다.

지금까지 이동 환경에서 회복을 위한 방법으로 검사점을 이용한 방법과 메시지 로깅을 이용한 방법들이 제안되었으나 이동 환경의 특성상 메시지 로깅 기법이 더 적합하다. 이에 이 논문에서는 메시지 로깅 기법의 세 부류 중 비관적 메시지 로깅 기법을 이용한 회복 알고리즘을 제안하였다. 제안하는 방법의 장점은, 비관적 프로토콜을 사용하여 알고리즘이 간단하고, 이동 호스트의 지역 메모리에 송신 정보만 로깅함으로써 공간 부담을 줄였다. 또한 모든 메시지를 보내기 전에 먼저 로깅하므로 고아 프로세스가 발생하지 않는다. 또 검사점을 취할 때 메시지 수에 기반하여 검사점을 수행하여 일정 시간 간격으로 검사점을 취하는 것보다 효율적이다. 이동 호스트의 이동이 잦을 때도 이동하면서 수행한 작업에 대한 정보가 분산되지 않고 모두 홈 에이전트에 저장되므로, 빠른 회복을 할 수 있다.

제안하는 알고리즘에서 이동 호스트가 송신하는 메시지는 이동 호스트에 로깅되고, 수신하는 메시지는 홈 에이전트에 로깅된다. 홈 에이전트의 메모리는 안정된 기억장치라 할 수 있기 때문에 고장이 발생하더라도 메시지의 목록을 유지할 수 있다. 문제는 지역 검사점과 전역 검사점 사이에서 고장이 발생했을 때, 이동 호스트의 소멸성 메모리에 저장된 내용이 상실되는 경우인데, 이 때 상실된 정보는 이동 호스트가 전송한 메시지이다. 이 문제는 홈 에이전트에 저장된 수신 메시지 목록을 이용하여 해당 이동 호스트에게 요청함으로써 일관된 상태를 유지할 수 있다.

여기에 추가하여 알고리즘의 효율성을 위하여 이동 호

스트의 전력 소모를 줄일 수 있는 방법과 검사점 정보를 줄일 수 있는 방법에 대하여 연구되어야 할 것이다.

참고문헌

- [1] V. Bharghavan, "Challenges and Solutions to Adaptive Computing and Seamless Mobility over Heterogeneous Wireless Networks," International Journal on Wireless Personal Communications, 1996.
- [2] N. Neves, W. K. Fuchs, "Adaptive Recovery for Mobile Environments," Communications of the ACM, vol. 40, no. 1, pp. 68-74, Jan., 1997.
- [3] D. K. Pradhan, P. Krishna, N. H. Vaidya, "Recovery in Mobile Environments : Design and Trade-off Analysis," Proceedings of the 26th International Symposium on Fault-Tolerant Computing, pp. 16-25, June 1996.
- [4] A. Acharya, B. Badrinath, "Checkpointing Distributed Applications on Mobile Computers," Proceedings of the 3rd International Conference on Parallel and Distributed Information Systems, pp. 73-80, Sep., 1994.
- [5] G. H. Forman, J. Zahorjan, "The Challenges of Mobile Computing," IEEE Computer, vol. 27. no. 6, Apr., 1994.
- [6] J. Kistler, M. Satyanarayanan, "Disconnected Operating in the CODA File System," Proceedings of the 13th SOSP, Oct., 1991.
- [7] L. Alvisi and K. Marzullo, "Message Logging: Pessimistic, Optimistic, and Causal," International Conference on Distributed Computing Systems, pp. 229-236, 1995.
- [8] B. Yao, K. Ssu, W. K. Fuchs, "Message Logging in Mobile Computing," in Proceedings of the 29th Symposium on Fault Tolerant Computing Systems, pp. 294-301, June 1999.
- [9] 박윤용, 전성익, 조주현, "분산 트랜잭션 처리 시스템에서 2-단계 확인 프로토콜을 근거로 하는 검사점 설정 및 오류 복구 알고리즘", 한국정보처리학회 논문지 3권 2호 pp. 327-338, Mar., 1996.
- [10] M. Elnozahy, L. Alvisi, Y.-M. Wang, D. Johnson, "A survey of rollback-recovery protocols in message passing systems," Tech. Rep. CMU-CS-96-181", School of Computer Science, Carnegie Mellon University, Oct., 1996.
- [11] C.P. "IPv4 Mobility Support," RFC 2002, Oct., 1996.