비동기 에이전트 복제를 이용한 효율적인 이동 에이전트 복구 기법

변일수,° 강수석, 박태순
세종대학교 컴퓨터공학과
{widepis°, sskang76, tspark}@sejong.ac.kr

An Efficient Mobile Agent Recovery Scheme based on Asynchronous Agent Replication

Ilsoo Byun,° Sooseok Kang and Taesoon Park
Dept. of Computer Engineering, Sejong University

요 약

신뢰할 만한 이동 에이전트 시스템을 구축하기 위해서는 이동 에이전트의 결함 내성 기능이 중요하다. 지금까지 여러 결함 내성 기법이 제안되었는데, 그 중의 하나가 에이전트 복제 기법이다. 에이전트 복제는 검사점 기법에 비해 훨씬 나은 신뢰성을 제공하는 반면, 복제 에이전트의 이동과 동의 과정에 수반되는 부하가 시스템의 성능을 저하시킨다. 그러므로 본 논문에서는 비동기 에이전트 복제 기법을 제안한다. 이 기법에서는 에이전트를 비동기적으로 전송하며 동의 과정을 위해 뒤늦게 전송되는 복제 에이전트를 기다리는 것이 아니라 고정된 동의 전용 에이전트를 사용하여 동의 과정의 비용을 감소시킨다. 제안된 기법은 Aglet 시스템 상에 구현되었으며, 실험 결과 제안된 기법을 사용할 경우 35%-69%의 성능 향상을 얻을 수 있었다.

1. 서 론

이동 에이전트란 사용자가 부여한 임무를 수행하기 위해 여러 사이트를 이동해 다니며 수행되는 프로그램이다. 이동 에이전트는 자율적으로, 그리고 비동기적으로 수행된다. 에이전트가 일단 실행되면 목적지를 스스로 설정할 수 있으며 사용자와의 상호 반응 없이 수행된다. 그러기 위해 에이전트는 프로그램 코드와 각 사이트들을 돌아다니며 수집한 자료들을 가지고 다닌다. 또한 에이전트 수행의 연속성을 위해 에이전트의 중간 상태도 같이 이동한다.

이동 에이전트가 좀더 다양한 분야에 적용되기 위해서는 신뢰할만한 에이전트의 수행이 요구된다. 에이전트는 이동 중이나 수행 중에 시스템 오류로 손실될 수 있다. 반대로, 잘못된 오류 복구는 에이전트를 중복 실행시킬 수도 있다. 신뢰할 만한 에이전트 시스템은 에이전트가 아무 오류 없이 정확히 한번(exactly-once) 수행되는 것을 보장해야한다.

신뢰할 만한 에이전트 시스템을 위해 제안된 결함 내성 기법들은 크게 복제 기법[1,2,3]과 검사점 기법[4,5,6]으로 분류된다. 복제 기법에서는 각 단계마다 에이전트의 복제 본을 여러 사이트에 전송하며 중복 수행을 막기 위해 분산 트랜잭션[7], 리더 선출 절차[3], 혹은 동의 절차[1,2]를 거치게 된다.

복제 기법의 단점은 에이전트의 사이즈가 크거나 네트워크 비용이 비싼 환경에서는 비효율적이라는 점이다. 이에 반해, 검사점 기법에서는 에이전트를 하나의 사이트에만 보내며 미래의 오류를 대비해 에이전트의 중간 상태를 방문한 사이트에 저장한다. 오류가 발생할 경우 저장된 상태로부터 에이전트를 복구시킨다. 이 기법은 오류가 발생한 사이트가 복구될 때까지 에이전트가 봉쇄된다는 단점을 가지고 있다.

복제 기법은 2k+1개의 에이전트 복제 본을 전송하였을 경우, k개의 오류를 견딜 수 있는 반면, 검사점 기법에서는 하나의 오류가 심각한 지연을 발생시킬 수 있다. 하지만 오류가 없을 경우, 전체 에이전트의 실행 시간을 고려해 볼 때 검사점 기법이 복제 기법에 비해 훨씬 나은 성능을 보인다. 복제 기법은 에이전트의 복제와 동의 과정에 따르는 부하가 존재하기 때문이다. 본 논문에서 제안되는 비동기 복제는 기존 복제 기법에서 발생하는 부하를 줄임으로 높은 결함 내성 기능을 지니면서 동시에 성능을 향상시킬 수 있는 기법이다.

비동기 복제 기법에서는 에이전트의 수행과 복제가 비동기적으로 수행될 수 있도록 제일 먼저 이동한 에이전트 복제본이 주 에이전트가 되어 임무를 수행하며 그 사이에 나머지 복제 본들이 이동한다. 이 경우, 주 에이전트의 실행이 끝날 때까지 복제 본들의 이동이 완료되지 못하는 상황도 발생한다. 이에 따른 지연을 줄이기 위해 우리는 고정된 동의 전용 에이전트를 사용하였다.

2k+1개의 복제본들 중 k+1개만이 실질적인 오류의 대비책이며 나머지 k개의 복제 본들은 단순히 동의 과정에만 참여한다는 점에 기인하여 제안된 기법에서는 k개의 고정된 동의 전용 에이전트를 사용하도록 작성하였다.

따라서 오류가 발생하지 않은 상황에서는 에이전트의 수행이 다른 복제 본들이 전송되기 이전에 끝나더라도 아무 지연 없이 동의 과정을 완료할 수 있다.

제안된 기법의 정확성과 성능을 증명하기 위해 Aglet 시스템 상에 기존의 복제 기법과 비동기 복제 기법을 구현하고 그 성능을 측정하였다. 그 결과 비동기 복제 기법을 사용할 경우 35%-69%의 성능 향상을 얻을 수 있었다.

2. 시스템 모델

이동 에이전트 시스템은 네트워크로 연결된 여러 개의 사이트로 구성되어 있다. 각 사이트는 에이전트의 실행 환경을 제공하는 하나 이상의 플레이스(place)를 제공한다. 에이전트는 플레이스들 사이를 이동해 다닐 수 있다. 한 플레이스에서의 수행과 다음 플레이스로의 이동을 하나의 스테이지(stage)라고 부른다. 따라서 에이전트의 수행은 일련의 스테이지라고도 볼 수 있다.

에이전트 시스템에서 발생할 수 있는 오류에는 다음과 같은 세 가지가 있다

- 에이전트 오류: 하나의 에이전트에 오류가 발생했으며 수행을 중단 한다
- 플레이스 오류: 하나의 플레이스에 오류가 발생했으며 수행을 중단한다. 플레이스에서 수행중인 모든 에이전트가 중단된다.
- 시스템 오류: 시스템에 오류가 발생한 경우로, 시스템 상에서 수행중인 모든 플레이스가 중단된다.

복제 기법을 사용하는 결함 내성 시스템에서는 위와 같은 오류가 발생하더라도, 초기 상태에서 다시 시작할 필요가 없다. 전송된 복제 본 중에 하나가 결함을 발견하고, 수행을 재개한다.

3. 비동기 에이전트 복제 기법

3.1 에이전트 복제 및 동의 과정

복제 기법에서는 k개의 결함을 견디기 위해 2k+1개의 복제 본을 서로 다른 사이트에 존재하는 플레이스에 전송한다. 그 중에 수행을 책임진 복제 본을 주(primary) 에이전트라고 부르며 주 에이전트가 실패했을 경우 다른 복제 본들이 수행을 대신한다. 대체 수행의 경우, [3]에서처럼 주 에이전트의 실패를 제일 먼저 감지한 복제본이 수행을 담당할 수도 있고 [2]에서처럼 각 복제 본에 우선 순위를 할당하여, 우선순위에 따라서 수행을 담당할 수도 있다. 결함의 감지를 위해서는 타임아웃을 사용한다. 주 에이전트가 일정 시간동안 응답이 없는 경우 다음 우선순위를 지닌 복제본이 주 에이전트의 결함을 각 복제 본들에게 알린 후 실행을 시작한다. 이때, 잘못된 결함 발견에 의한 중복 수행의 가능성을 막기 위해, 수행이 끝난 에이전트는 다음 플레이스로 이동하기 전에 다른 복제 본들로부터 과반수이상의 동의를 얻어야만 한다.

3.2 비동기 에이전트 복제

[9]에서 언급된 것처럼 에이전트의 복제와 이동에 2-단계 완료 형식이 적용되는 것은 에이전트의 총 실행시간에 심각한 지연을 초래한다. i 번째 스테이지의 에이전트는 i-1 번째 스테이지의 주 에이전트로부터 모든 에이전트가 안전히 전송되었음을 확인하는 메시지를 받기 전까지 수행을 시작할 수 없다. 만약 i-1 번째 스테이지의 주 에이전트가 결함으로 인해 실행 중지가 되었을 경우 i-1 번째 스테이지의 복제 에이전트들이 대신 수행될 것이고 그렇게 된다면 i 번째 스테이지에 이미 도착해있는 에이전트들은 취소되어야 하기 때문이다. 그러므로 i 번째 스테이지의 에이전트는 i-1 번째 스테이지로부터 전송 확인 메시지를 받기 전까지 시작되어서는 안 된다.

이로 인해 발생되는 실행 지연은 복제 기법의 성능 저하를 초래한다. 이러한 성능상의 문제를 해결하기 위해 비동기 복제 기법에서는 제일 처음에 전송된 복제 에이전트가 i 번째 스테이지의 주 에이전트의 역할을 수행하도록 한다. 그 동안 i-1 번째 스테이지의 주 에이전트는 나머지 복제 에이전트들의 전송을 담당하게 하여 결과적으로는 i 번째 스테이지의 주 에이전트의 수행과 i-1 번째 스테이지의 복제 에이전트 전송 과정이 동시에 처리될 수 있다.

이에 더해 동의 전용 에이전트를 사용하여 성능을 향상시킬 수 있다. 동의 전용 에이전트는 아무 기능도 갖추지 않고 아무 자료도 지니지 않은 채 동의 과정에만 참여할 수 있는 에이전트다. 위에서 말했듯이 2k+1개의 에이전트 중 실제 오류가 발생했을 경우 대신 임무를 수행하게 될 가능성이 있는 에이전트는 k+1개뿐이다. 나머지 k개는 단순히 동의 과정에만 참여할 뿐이다. k개의 복제 에이전트 대신 고정된 사이트에 존재하는 k개의 동의 전용 에이전트를 사용함으로 i 번째 스테이지의 에이전트가 이미 임무를 끝낸 뒤에는 i-1 번째 스테이지로부터 뒤늦게 전송돼오는 복제 에이전트를 기다리는 것이 아니라 고정된 동의 전용 에이전트로부터 동의를 얻어 바로 다음 사이트로 이동을 시작할 수 있다. 물론 오류가 발생하게 될 경우에는 그에 따르는 지연이 발생하겠지만 오류가 발생하지 않을 경우에는 지연을 최소화할 수 있다.

4. 실험 환경 및 결과

우리는 실험을 위해 기존의 동기적인 복제 기법과 비동기적 복제 기법을 Aglet SDK 1.1b2를 이용하여 구현하였다. 에이전트의 크기를 조정하기 위해 NxN 배열을 사용하였고 실험 결과 값은 총 10번 실험한 후 최고 값과 최저 값을 제외한 8개의 평균값을 사용하였다. 실험 결과 우리는 비동기 복제 기법이 동기적인 복제기법에 비해 최고 69%정도의 실행시간 단축을 보인 것을 알 수 있었다.

그림 1과 2는 에이전트 복제본의 수에 따를 실행시간

과 이동 시간의 변화를 나타낸다. 비동기 복제 기법이 Lazy-Replication으로, 기존의 동기적 기법이 Full-Replication으로 표현되며, No-Replication은 복제 기법이 적용되지 않은 단일 에이전트의 실행 시간 및 이동 시간을 표현 한다. 그림 1과 2에서 볼 수 있듯이 비동기 복제 기법은 복제 에이전트의 수가 많아질수록 성능상의 이익을 보인다.

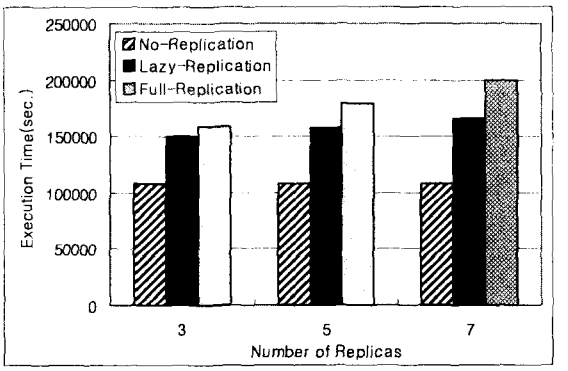

그림 1. 에이전트 복제 기법간의 실행시간 비교

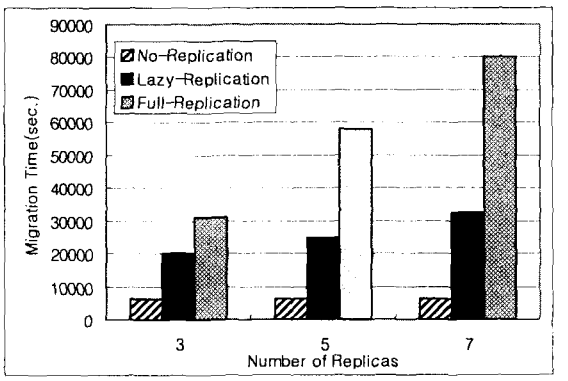

그림 2. 에이전트 복제 기법간의 이동 시간 비교

그림 3은 에이전트의 크기에 따른 실행시간의 변화를 보여준다. 동기적인 에이전트 복제기법이 에이전트의 크기에 민감한 반면 비동기적 기법은 에이전트의 크기가 성능에 그다지 큰 영향을 미치지 않음을 알 수 있다.

5. 결 론

본 논문에서 우리는 비동기적 에이전트 복제 기법을 제안하였다. 주 에이전트는 비동기적으로 전송될 수 있도록 하였고 k+1개의 플레이스에는 복제 에이전트를 보내고 k개는 고정된 동의 전용 에이전트를 사용함으로써 복제 기법의 오버헤드를 최소화할 수 있었다. 성능 측정

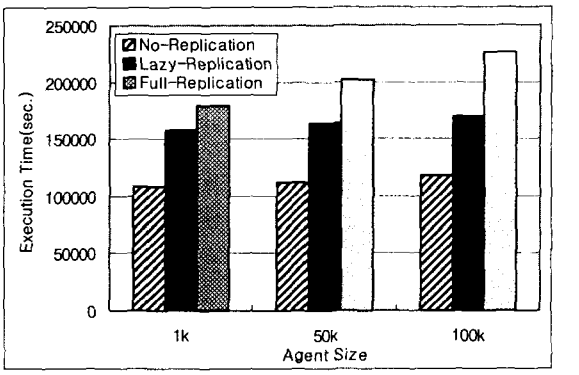

그림 3. 에이전트의 크기에 따른 실행시간 비교

을 위해 두개의 결함 내성 기법을 구현하였고, 실험 결과 비동기 복제 기법이 동기적 복제 기법보다 35%-69% 정도의 성능 향상을 보이는 것을 확인할 수 있었다.

Acknowledgements

본 연구는 한국과학재단 목적기초연구 지원으로 수행되었음 (과제번호: R04-2002-000-20102-0).

6. 참고 문헌

[1] S. Pleisch and A. Schiper, "Modeling Fault-Tolerant Mobile Agent Execution as a Sequence of Agreement Problems," *Proc. of the 19th Symp. on Reliable Distributed Systems,* pp. 11-20, 2000.

[2] S. Pleisch and A. Schiper, "FATOMAS-A Fault Tolerant Mobile Agent System Based on the Agent Dependent Approach," *Proc. of the Int'l Conf. on Dependable Systems and Networks,* pp. 215-224, 2001.

[3] M. Strasser and K. Rothermel, "Reliability Concepts for Mobile Agents," *International Journal of Cooperative Information Systems,* Vol. 7, No. 4, pp. 255-282, 1998.

[4] M. Dalmeijer, E. Rietjens, D. Hammer, A. Aerts and M. Soede, "A Reliable Mobile Agents Architecture" *Proc. of the Int'l Symp on Object-Oriented Real time Distributed Computing,* 1998.

[5] E. Gendelman, L.F. Bic, and M.B. Dillencourt, "An Application-Transparent, Platform-Independent Approach to Rollback-recovery for Mobile Agent Systems," *Proc. of the 20th Int'l Conf. on Distributed Computing Systems,* 2000.

[6] M. Strasser and K. Rothermel, "System Mechanism for Partial Rollback of Mobile Agent Execution," *Proc. of the 20th Int'l Conf. on Distributed Computing Systems,* 2000.

[7] H. Vogler, T. Kunkelmann and M.L. Moschgath, " An Approach for Mobile Agent Security and Fault tolerance using Distributed Transaction," *Proc. of the Int'l Conf. on Parallel and Distributed Systems,* pp. 268-274, 1997.