이동 에이전트 컴퓨팅 환경에서 공간적 복제 기반 기법을 위한 이동 에이전트 위치관리 프로토콜

윤 준 원[†] · 최 성 진[††] · 안 진 호[†††]

요 약

다중 지역으로 구성된 이동 에이전트 컴퓨팅 환경에서 공간적 복제 기반 기법(SRBA: Spatial Replication-Based Approach)은 에이전트의 고장발생 시에도 그 에이전트의 수행이 대기 없이 계속적으로 진행될 수 있도록 하기 때문에, 대표적인 이동 에이전트 결함 포용 기법으로 사용될 수 있다. 그러나 이 기법을 실제의 이동 에이전트 기반 컴퓨팅 시스템에 적용하는데 있어서, 단계별로 복제된 이동 에이전트들에 대한 위치추적 및 관리 비용을 최소화시키는 것이 필수적이다. 본 논문에서는 이러한 문제점을 해결하는 새로운 이동 에이전트 위치 관리 프로토콜인 SRLM(Location Management protocol for Spatial Replication)을 제안한다. 제안된 프로토콜은 단계군 내의 복제된 이동 에이전트들 중 대표 작업자만이 자신의 지역서버에게 위치등록하게 함으로써 기존 프로토콜에 비해 위치갱신 및 메시지 전달 비용을 매우 줄인다. 또한, 이 프로토콜은 한 단계 군에서의 대표 작업자 실패 발생시 새로운 대표 작업자의 선출로 인한 위치 관리 문제를 해결한다.

키워드 : 이동 에이전트 시스템, 결함 포용성 공간적 복제 기반 기법, 위치 관리, 메시지 전달

Mobile Agent Location Management Protocol for Spatial Replication-based Approach in Mobile Agent Computing Environments

Junweon Yoon[†] · Sungjin Choi[††] · Jinho Ahn[†††]

ABSTRACT

In multi-regional mobile agent computing environments, spatial replication based approach may be used as a representative mobile agent fault-tolerance technique because it allows agent execution to make progress without blocking even in case of agent failures. However, to apply this approach to real mobile agent-based computing systems, it is essential to minimize the overhead of locating and managing mobile agents replicated on each stage. This paper presents a new mobile agent location management protocol SRLM to solve this problem. The proposed protocol allows only the primary among all the replicated workers of each stage to register with its regional server and then, significantly reduces its location updating and message delivery overheads compared with the previous protocols. Also, the protocol addresses the location management problem incurred by electing the new primary among the remaining workers at a stage in case of the primary worker's failure.

Key Words : Mobile Agent System, Fault-tolerance, Spatial Replication-Based Approach, Location Management, Message Delivery

1. 서 론

인터넷과 무선 기술이 급속히 확장됨에 따라 가용성 있는 온라인의 정보들이 증가하게 되었고, 이러한 정보를 효율적으로 탐색, 수집, 회수하기 위한 시스템들이 연구, 개발되어지고 있다[1, 2]. 이러한 인터넷에서 제공되는 많은 서비스들이 전통적이면서 보편적인 클라이언트-서버 모델에 기반을 두어 구축되었으며, 이 모델에서 서버는 일반적으로 클라이언트에게 제한적인 특정 서비스 또는 어플리케이션만을 제공하게 된다[3, 4]. 그러나 클라이언트 애플리케이션이 서버에 있는 매우 많은 데이터를 가져와서, 클라이언트 컴퓨터에서 그 데이터를 처리한다면, 비효율적인 네트워크 부하 및 긴 작업완료시간을 발생시킨다. 특히, 무선 네트워크 환경에서는 이러한 문제가 더욱더 커질 수가 있으며, 클라이언트의 배터리 파워 문제를 매우 악화시킬 수 있다. 따라서 이러한 문제점을 해결하기 위해 이동 에이전트 기술을 기존의 클라이언트-서버 모델의 대체방안으로 현재 많은 연구 및 개발 중이다.

이동 에이전트는 사용자를 대신하여 호스트1)와 호스트를

† 정 회 원 : 한국과학기술정보연구원 지식정보센터 지식포털팀
†† 준 회 원 : 고려대학교 컴퓨터학과 박사과정
††† 종신회원 : 경기대학교 정보과학부 전자계산학과 조교수(교신저자)
논문접수 : 2006년 4월 19일, 심사완료 : 2006년 9월 1일

자율적으로 이동하면서 연산을 수행하거나[5, 6], 사용자에 의해 할당된 작업을 수행하기 위한 컴퓨터 소프트웨어 프로그램이다[7]. 이러한 이동 에이전트들이 여러 지역으로 구성된 이동 에이전트 컴퓨팅 환경에서 원활하게 자신의 작업들을 처리하기 위해서는 여러 가지 고려해야 할 사항들이 있다. 그 중 이동 에이전트 수행 시 에이전트를 관리하기 위한 즉, 협업이나 통신, 취소 등과 같이 수행중인 이동 에이전트를 추적하여 다루기 위한 위치 관리 프로토콜은 이동 에이전트 시스템을 개발하는데 있어서 우선적으로 중요하게 고려할 사항이다[8, 9].

한편, 이동 에이전트 컴퓨팅 환경에서 이동 에이전트 수행 중 통신고장이나 호스트의 고장으로 인해 연산이 부분적 또는 전체적으로 손실되어 연산 수행이 정지 되는 현상이 발생한다[10]. 이동 에이전트에 대한 결함 포용적 연산 수행을 위해, 기존 연구에서는 이동 에이전트와 이동 에이전트의 실행장소(Place)를 복제하여 결함을 포용하는 기법이 사용된다. 복제를 기반으로 한 결함 포용 기법으로는 크게 시간적 복제 기반 기법(TRBA : Temporal-Replication-Based approach)과 공간적 복제 기반 기법(SRBA : Spatial-Replication-Based approach)으로 구분된다[10]. TRBA는 이전 단계에서 현재 진행 중인 단계의 에이전트를 안전한 저장소에 저장하여 현재 단계가 고장이 나더라도 이전 단계에서 미리 저장되어 있는 에이전트를 다시 보내어 연산을 수행하는 방법이다. SRBA는 한 단계에서 여러 실행장소들을 두어 결함 포용성을 제공하는 방법으로 현재 작업을 진행 중인 실행장소가 고장이 나더라도 현 단계를 이루는 다른 실행장소들 중 새로운 작업자[2]가 선출되어 작업을 진행하는 방법이다[10]. 또한 SRBA 기법은 이동 에이전트 수행 중, 중간 연산에 때한 빠른 결과를 제공하고 하나의 단계군(Stage) 내에서 하나의 작업을 실행 완료함으로써 중간 연산에 대한 단계군 단위의 결함 포용을 보장하기 때문에 많은 이동 에이전트 컴퓨팅 환경에서 사용되고 있다.

기존 이동 에이전트를 위한 위치 관리 프로토콜들은 TRBA기법에 적용 가능하나, 실행장소들을 공간적으로 구성하여 하나의 단계군으로 구성한 다음 결함을 포용하는 SRBA 기법에서는 적용 가능하지 않다. 특히, 기존의 위치 관리 기법을 SRBA에 적용했을 경우 높은 위치 관리 및 메시지 비용이 발생하며, 단계군 내에 작업자의 고장으로 새로운 작업자를 선출한 경우 메시지 전달을 보장하지 못하는 문제점이 발생한다.

본 연구에서는 공간적 복제 기반 기법(SRBA)에서 발생하는 불필요한 위치 관리 오버헤드를 줄이기 위해 이동 에이전트를 위한 새로운 위치 관리 기법인 SRLM(Location Management protocol for Spatial Replication) 프로토콜을 제안한다. 제안된 SRLM 프로토콜은 결함 포용을 위해 단계군 구성시 복제된 이동 에이전트에 대한 위치 등록비용 및 대표 작업자에게 메시지 전달을 보장하고 그 비용도 줄인다. SRLM 프로토콜은 단계군 내에서의 대표 작업자의 결함이 발생한 경우 새로운 대표 작업자의 위치 관리 문제 또한 해결한다.

본 논문의 이후 구성은 다음과 같다. 먼저, 2절과 3절에서는 기존의 관련연구와 본 논문에서 가정하는 이동 에이전트 시스템 모델을 기술한다. 4절에서는 공간적 복제 기반 기법에 효율적인 위치관리 프로토콜을 제안하고, 5절에서는 그 프로토콜의 성능을 분석하고, 6절에서는 결론을 제시한다.

2. 관련 연구

이동 에이전트 컴퓨팅 환경에서 이동 에이전트에 대한 위치 관리 및 메시지 전달 기법과 관련된 기존 연구들은 크게 홈 프락시(Home-Proxy), 추적 프락시(Follower-Proxy), 브로드캐스트(Broadcast), 블랙보드(Blackboard), 이메일(E-mail) 기법으로 분류된다[11-13]. 이런 기법들은 이동 에이전트의 결함 포용을 고려하지 않은 단순 위치 관리 기법으로 이동 에이전트 수행 중 고장이 발생할 경우 계속적인 연산을 보장하기 위한 방법들을 제시하지 않았다. 먼저, 홈 프락시 기법[14]을 살펴보면, 이동 에이전트의 위치 정보를 이동 에이전트가 생성된 홈 호스트 또는 특정 서버에 저장한다. 이동 에이전트 이주시마다 그 위치 정보를 홈 호스트의 위치 정보 저장소에 저장한다. 위치 탐색 시 이 정보를 이용하여 이동 에이전트를 찾아 메시지를 전달하는 기법이다. 그러나 홈 호스트와 같은 특정 서버가 이동 에이전트에 대한 위치 정보를 관리하기 때문에 만약 홈 호스트와 이동 에이전트의 거리가 멀어지면 이동 에이전트의 위치정보 변경시마다 홈 호스트로 갱신 메시지를 전송할 때나, 해당 이동 에이전트로 메시지를 전송할 때 통신비용이 증가하는 단점을 가지고 있다. 또한, 이동 에이전트의 수가 많아지고 이동 에이전트들 간의 통신과 이동 에이전트의 이주가 자주 일어나는 상황에서는 홈 호스트나 특정 서버는 병목현상(Bottleneck)을 겪게 된다.

추적 프락시 기법[15]은 이동 에이전트의 위치 정보를 이동 에이전트가 방문한 호스트들에 저장한 다음, 각 호스트에 저장된 위치 정보 경로 프락시(Path proxies)를 따라 이동 에이전트를 찾거나 계속해서 포워딩(Forwarding)하는 기법이다. 그런데, 이 기법은 홈 호스트나 특정 서버의 부하는 줄어드는 반면에 메시지의 전달을 위해서 이동 에이전트가 이동한 경로 상의 모든 호스트가 참가해야 하는 단점을 가지고 있다. 즉, 이동 에이전트가 방문한 모든 호스트에 위치 관련 정보를 저장해야 하고 메시지를 포워딩을 해주는 포워딩 데몬을 유지해야 하는 오버헤드가 있다. 또한, 이동 에이전트가 이동할 때마다 경로 프락시가 증가하게 되어 메시지 전달하는데 높은 통신비용이 발생하며, 경로 프락시를 이루는 중간 호스트가 고장이 나면 경로 프락시를 이루는 체인(Chain)이 끊어짐에 따라 이동 에이전트에게 메시지를 전달할 수 없는 상황이 발생한다.

브로드캐스트 기법[6, 14, 16]은 정해진 지역이나 이동계

1) 호스트는 컴퓨터, 서버 등 이동 에이전트가 실행되는 장소이다. 한 개의 호스트 내에는 한개 이상의 실행장소가 존재할 수 있다. 본 논문에서는 한 개의 호스트 내에 한 개의 실행장소가 존재한다고 가정한다.

2) SRBA에서는 결함 포용을 위해 단계군 내에 복수개의 실행장소가 존재하는데, 그 중 선출(election) 알고리즘에 의해 선택된 실제 이동 에이전트를 수행하는 실행장소를 말한다.

획(Itinerary)의 모든 호스트들에게 메시지를 보내어 위치를 찾거나 메시지를 전달하는 기법이다. 그러나 이 기법에서는 이동 에이전트 수행과정의 모든 호스트들에게 해당 메시지를 브로드캐스팅 해야 하기 때문에 높은 통신비용이 발생한다.

블랙보드 기법[17-20]은 각 호스트에 블랙보드를 두어 메시지를 교환하는 기법이다. 그러나 이 기법은 모든 호스트들이 메시지를 저장할 기억장소를 가지고 있어야 하는 단점을 가지고 있다. 또한, 이동 에이전트가 메시지를 전달받기 위해서는 직접 해당 호스트로 이동해야 하는데 메시지를 수신하는 이동 에이전트가 해당 호스트로 이동하는 패턴이 일정하지 않고 임의로 이루어지기 때문에 메시지 전달이 즉각적으로 이루어지지 않는 단점을 가지고 있다. 그리고 간단한 메시지의 경우, 직접 이동 에이전트가 해당 호스트로 이동하여 처리해야 하기 때문에 메시지를 처리하는 과정에서 불필요한 통신 부하가 발생한다.

이메일 기법[17, 21]은 이메일 서버와 유사한 방법으로 메시지를 전송하고자 하는 이동 에이전트는 메일 서버에 메시지를 남겨 놓으면 메시지를 수신하고자 하는 이동 에이전트가 서버에 접속하여 메시지를 얻는 기법이다. 그러나 이 기법에서 메시지의 전달이 즉각적으로 이루어지지 않는다는 단점을 가지고 있다.

그런데, 이러한 기법들은 단일 지역으로 구성된 이동 에이전트 컴퓨팅 환경에 기반하고 있다. 여러 지역으로 구성된 다중 지역 이동 에이전트 컴퓨팅 환경에서 위와 같은 기법들을 적용하였을 경우 많은 문제들이 발생한다. 예를 들어, 다중 지역 이동 에이전트 컴퓨팅 환경에서 기존 기법들을 적용하였을 때 이동 에이전트의 현재 호스트와 홈 호스트와의 거리에 따라 위치 갱신 비용 및 메시지 전달 비용이 증가하는 문제점과 이동 에이전트에게 해당 메시지를 전달하지 못하고 이동 에이전트가 방문한 호스트들만 계속해서 탐색하는 문제가 발생한다.

Stefano[22]는 SPC 프로토콜(Search-by-Path-Chase Protocol)을 통하여 이동 에이전트의 위치 관리 문제점을 해결하려고 하였으나, 경로 프락시(Path Proxies)를 사용함으로 인해 위치 관리 및 메시지 전달하는데 많은 통신비용이 발생하며 이동 에이전트가 지나온 경로만 계속 탐색하는 추적 문제가 발생한다.

지금까지 앞에서 언급하였던 모든 위치 관리 기법은 이동 에이전트의 연산 중 고장이 발생한 경우, 복제된 새로운 실행장소의 위치관리기법을 고려하지 않아 높은 위치 관리 비용 및 메시지 전달 비용이 발생한다. 반면에 논문에서 제안하는 SRLM 프로토콜은 공간적 복제 기반 기법에 적용할 경우 적은 위치관리 비용 및 메시지 전달 비용을 발생시키고, 또한 작업자의 결함이 발생하더라도 효율적이고 신뢰성 있게 메시지를 전달하도록 한다.

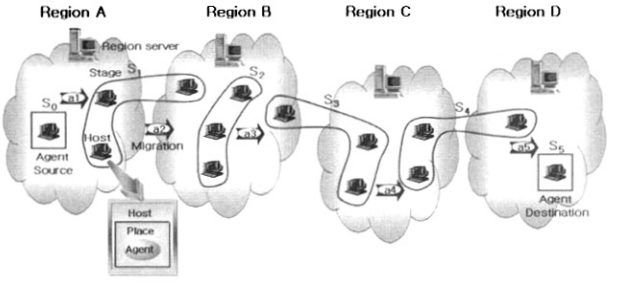
3. 이동 에이전트 시스템 모델

본 논문에서 가정하는 이동 에이전트 컴퓨팅 환경은 이동 에이전트 a_i, 실행장소 P_i, 단계군 S_i로 구성된다. 이동 에이전트 a_i는 각 호스트의 실행 결과에 따라 다음 호스트를 동적으로 선택하여 이동하거나, 이동계획(itinerary)에 따라 호스트와 호스트사이를 옮겨 다니며 작업을 수행한다. 이때 호스트에서는 이동 에이전트가 수행할 수 있는 실행장소 P_i를 제공한다[6, 22]. 실행장소는 이동 에이전트에게 특정 서비스를 제공한다.

이동 에이전트 시스템들 중에서 같은 권한을 가진 이동 에이전트 시스템들의 집합을 지역 R_i(Region)이라고 한다[6, 22]. 일반적으로 지역은 LAN(Local Area Network)으로 구성된 서브 네트워크 내의 이동 에이전트 시스템을 말한다. 본 논문에서는 지역이 여러 개 존재하는 다중 지역 이동 에이전트 컴퓨팅 환경을 가정한다. 지역에는 지역의 권한을 책임지는 지역 서버 RS_i(Region Server)가 존재한다[6]. 이동 에이전트의 결함 포용을 위해 실행장소들을 모아 단계군(Stage) 집합 $\{S_0, S_1, \ldots, S_{i-1}\}$을 형성하며, 에이전트 출발지 S_0와 목적지 S_{i-1}까지 동적으로 이주하며 작업을 수행하게 된다[10]. 전 단계에서 실행된 이동 에이전트는 결함 포용을 위해 다음 단계군 내의 모든 실행장소에 복제되고, 단지 하나의 실행장소에서만 에이전트가 수행되어진다. 만약, 에이전트 수행 중 통신고장이나 호스트의 고장이 생기면 단계군 내의 다른 실행장소(Place)들은 고장을 탐지한 후, 새로운 작업자를 선출하여 작업을 진행하게 된다[10][3]. 출발지 S_0와 목적지 S_{i-1}에서는 복제가 일어나지 않는다[10].

단계군을 구성할 때 이동 에이전트의 이동 계획과 단계군을 구성하는 실행장소의 수에 따라 다른 단계군이 구성된다[23]. 이때 단계군을 이루는 실행장소는 서로 같은 지역에 있을 수도 있고 서로 다른 지역에 있을 수도 있다. 본 논문에서는 이동 에이전트가 여러 지역으로 구성된 다중 지역 이동 에이전트 컴퓨팅 환경을 이동하면서 작업을 수행하는 것으로 가정한다. (그림 1)은 본 논문에서 가정하는 이동 에이전트 컴퓨팅 환경을 나타내고 있다.

마지막으로, 본 논문에서는 호스트 또는 실행장소의 고장을 간주하며, 결함 발생 시 다른 구성요소에 영향을 끼치지 않고 자신의 내부 수행 상태를 상실하는 고장 정지(Fail-stop)모델[10]을 전제한다.



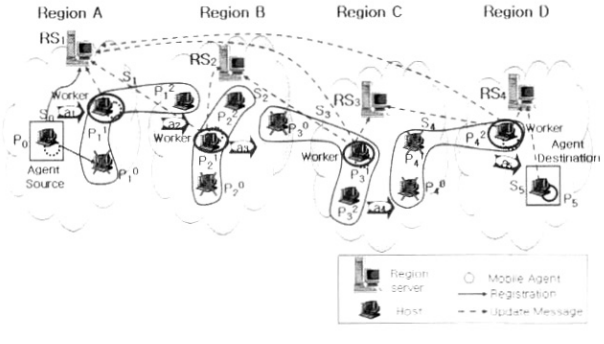
(그림 1) 단계군으로 구성된 이동 에이전트 컴퓨팅 환경

3) 단계군 내의 고장 탐지방법은 각 호스트들이 일정한 주기로 heartbeat 메시지를 주고받거나, 탐지 데몬을 통해 체크하는 기법들이 있으며, 그 외 다수의 논문과 연구가 진행되어져왔다[21, 22]. 또한, 고장 탐지 후 새로운 작업자를 선출하는 알고리즘에 대한 여러 방법과 연구가 있으나 본 연구에서는 단계군 내의 이동 에이전트가 실행되는 호스트들의 우선순위를 적용하여 실험결과를 측정하였다. 본 논문은 이동 에이전트의 위치관리비용을 다루고 있으며, 두 연구에 대해서는 관련 연구로 참조하기 바란다.

4. SRLM 프로토콜

공간적 복제 기반 기법(SRBA : Spatial-Replication-Based approach)은 한 단계에서 여러 실행장소를 두어 결함 포용을 해결하는 방법으로 중간 연산에 때한 빠른 결과를 제공하기 때문에 많은 이동 에이전트 컴퓨팅 환경에서 사용되고 있다. 그러나 기존의 위치 관리 프로토콜들이 SRBA에 적용될 경우 높은 위치 관리 비용 및 메시지 비용이 발생할 뿐만 아니라, 단계군 내 작업자의 고장으로 새로운 작업자가 선출될시 메시지 전달을 보장하지 못하는 문제점 또한 발생한다. 따라서 본 논문에서는 SRLM(Location Management protocol for Spatial Replication) 프로토콜을 제안하고, 다중 지역 이동 에이전트 컴퓨팅 환경에서 단계군으로 구성된 실행장소들에 대한 위치 관리시, 현재 작업 중인 에이전트의 실행장소만을 등록함으로써 위치 관리 비용을 감소시키고 작업자의 고장으로 새로운 작업자를 선출한 경우에도 메시지 전달을 보장하도록 설계한다. (그림 2)는 SRLM 프로토콜의 등록 및 이주과정을 설명한 전체적인 개요이다.

SRLM 프로토콜은 다음과 같이 크게 에이전트 네이밍 및 생성과정, 이주시 위치등록과정, 결함 포용적 위치 등록과정, 메시지 전달과정으로 구분된다.



(그림 2) SRLM 프로토콜에 대한 전반적 개요

4.1 에이전트 네이밍 및 생성과정

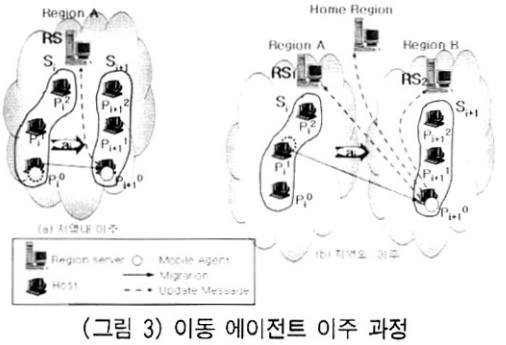
이동 에이전트가 생성되면 위치 관리를 위해 그 정보를 등록하게 되는데, 우선 생성된 홈 지역 서버(RS_{Home} : Home Region Server)에 이동 에이전트의 식별자와 생성된 호스트의 주소({AgentID, HostLocation})를 등록하게 된다.

4.2 이주시 위치등록과정

다중 지역에서 이주 위치등록과정은 크게 같은 지역으로 이주하는 지역내 이주(Intra Migration)와 다른 지역으로 이주하는 지역외 이주(Inter Migration)로 나뉜다. 지역내 이주시 갱신정보는 현재 지역의 지역서버($RS_{current}$)와 이전 호스트에게 알리게 되고, 다른 지역으로 이주하는 경우, 이동 에이전트는 생성된 홈 지역서버(RS_{Home}), 이주 전 지역서버(RS_{before}), 이주 후 현재 지역서버($RS_{current}$) 그리고 이전 실행장소에 갱신 메시지를 전달하게 된다. <표 1>과 (그림 3)은 이동 에이전트의 메시지 갱신 내용과 이주에 따른 위치

<표 1> 이동 에이전트 이주시 메시지 갱신 정보

지역내(Intra) 이주	$RS_{current}$	{AgentID, Location of P_i^{prim}}
지역외(Inter) 이주	RS_{Home}	{AgentID, Location of $RS_{current}$}
	$RS_{current}$	{AgentID, Location of P_i^{prim}}
	RS_{before}	{AgentID, Location of $RS_{current}$}

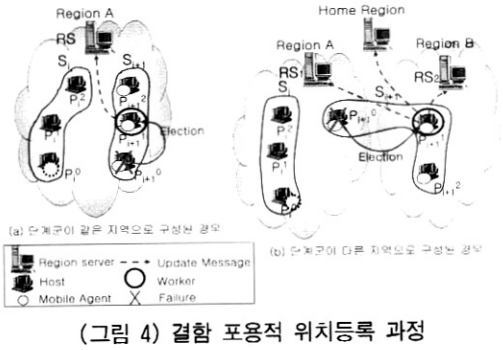


(그림 3) 이동 에이전트 이주 과정

등록 과정을 나타내고 있다(단, <표 1>에서 P_i^{prim}는 이주 후 수행할 단계군에서 해당 에이전트 a_i가 수행할 대표 작업자 호스트 임). 또한, 이러한 위치등록과정 알고리즘은 (그림 5)의 상단부에 제시되고 있다.

4.3 결함 포용적 위치 등록 과정

SRLM 프로토콜에서 에이전트 수행 중 고장이 생기면 단계군 내의 다른 실행장 소(place)들은 고장을 탐지한 후, 새로운 대표 작업자(worker)를 선출하여 작업을 진행하게 되고[10], 이동 에이전트의 위치 정보는 단계군 내에서 새로 선출된 실행장소의 위치에 따라 표 1과 같이 지역내 이주 또는 지역외 이주 위치등록방법에 의해 지역서버에 위치 메시지를 갱신하게 된다. (그림 4)와 같이 SRLM 프로토콜에서 고장이 발생하여 새로운 대표 작업자의 위치 정보를 갱신할 때, 이전 단계군의 실행장소가 위치한 지역과 상관없이 이전 작업자의 위치 정보를 따르게 된다. 즉, 단계군 내에서 고장이 발생한 후, 새로 선출된 작업자의 위치가 같은 지역에 있다면 이전 단계군의 실행장소와 상관없이 새로운 작업자에 대한 위치 갱신 메시지는 (그림 4(a))와 같이 지역내 이주 위치등록방법과 유사하게 전달되며, 새로 선출된 작업자의 위치가 다른 지역에 있다면 (그림 4(b))와 같이 지역외 이주 위치등록방법과 유사하게 전달된다. 이러한 결함 포용적 위치등록알고리즘은 (그림 5)의 하단부에 제시된다.



(그림 4) 결함 포용적 위치등록 과정

```
// --- 이주 과정 ---
SendAgent(a_i);                    // 에이전트 이주
UpdateMsg(RS_current, Loc of P_i^prim);    // 에이전트 이주 지역의 지역서버 등록(지역
                                           내 이주)
if compare(Loc of P_i^prim, Loc of P_i^prim) > 0 then    지역외 이주시 추가 등록
    UpdateMsg(RS_before, Loc of RS_current);    이주전 지역의 지역서버 등록
    UpdateMsg(RS_Home, Loc of RS_current);      홈 지역서버 등록
fi;
if(P_i^prim=P_destination^prim) then   // 에이전트 연산 종료
    Termination();
fi;

//    결함 포용적 위치 등록 과정
if P_i^prim = failure then  // 단계군(stage M_i)내에 작업자가 고장 난 경우
    Election();             // 새로운 작업자 선출
    UpdateMsg(RS_current, Loc of P_i^NewWorker);   // 새로운 작업자의 위치를 현재 지역
                                                      서버에 등록
        if compare(Loc of P_i^prim, Loc of P_i^NewWorker) > 0 then
        // 이전 작업자와 새로운 작업자가 서로 다른 지역에 있는 경우 추가 등록
            UpdateMsg(RS_before, Loc of RS_current);    이주전 지역서버에 위치 갱신
                                                        메시지 전달
            UpdateMsg(RS_Home, Loc of RS_current);      홈 지역서버에 위치 갱신 메
                                                        시지 전달
        fi;
    prim = NewWorker;
fi;
```

(그림 5) 이주시 및 결함 포용적 위치 등록 과정

4.4 메시지 전달 과정

공간적 복제 기반 기법은 하나의 단계군 내에서 하나의 작업을 실행 완료함으로써 중간 연산에 대한 단계군 단위의 작업을 보장한다. 즉, 단계군의 동작은 각 단계군의 수행종료시점에서 커밋(commit)된다[10]. 현재 지역서버는 이동 에이전트에게 메시지 전달이 완료되어 단계군 내에 작업이 종료되면 커밋되고, 그 지역서버에서 단계군에 관련된 모든 메시지를 삭제한다. (그림 6)은 메시지 전달과정에 대한 알고리즘이다. 본 논문의 기법은 작업자가 고장이 난 경우에도 메시지 전달을 보장한다.

① 메시지를 전달하고자 하는 통신자는 이동 에이전트가 생성된 지역서버(RS_{Home})에 질의를 한다.

② RS_{Home}는 이동 에이전트가 있는 지역서버를 찾는다. 이동 에이전트가 RS_{Home}에 있는 경우, 해당 호스트로 메시지를 전달하게 되고, 그렇지 않은 경우 이동 에이전트가 있는 지역서버로 메시지를 보낸다.

③ 지역서버에 있는 이동 에이전트의 위치 정보를 이용하여 이동 에이전트가 있는 호스트로 메시지를 전달한다.

④ 이동 에이전트가 고장이 난 경우, 지역서버는 고장 난 이동 에이전트에게 보냈던 전달되지 않은 메시지를 버퍼에 저장한다.

⑤ 새로운 작업자가 선출되면, 그 실행장소를 결함 포용적 위치 등록과정에 따라 지역서버에 갱신 메시지를 전달하고, 새로운 작업자는 현재 지역서버 버퍼에 저장되어 있는 전달되지 못한 메시지를 가져온다.

⑥ 실행장소에게 메시지가 전달된 후, 단계군 내의 작업이 끝나면 버퍼에 있는 메시지를 삭제한다.

```
CN.SendMsg(Msg, a_i, RS_Home); // 통신자는 에이전트가 생성된 지역서버에 메시지 전달
RSAddr = FindAddr(a_i);        // 에이전트가 위치한 지역서버 주소를 가져온다.
if compare(Addr of RS_Home , RSAddr) = 1 // 에이전트가 생성된 지역서버(RS_Home)에
                                            있는 경우
    SendMsg(Msg, a_i, HostAddr);          // 해당 호스트에 메시지 전달
    if HostAddr = failure then            // 해당 호스트가 고장 난 경우
        SendMsgFailure(a_i);              // 메시지가 전달되지 않은 경우
    fi;
else                                  // 에이전트가 다른 지역으로 이주 한 경우
    SendMsg(Msg, a_i, RSAddr);    // 해당 지역서버로 메시지 전달
    SendMsg(Msg, a_i, Hostaddr);  // 해당 호스트에 메시지 전달
    if HostAddr = failure then    // 해당 호스트가 고장 난 경우
        SendMsgFailure(a_i);      // 메시지가 전달되지 않은 경우
    fi;
fi;

Fucnction SendMsgFailure(a_i)         // 이동 에이전트가 고장 난 경우
    Buf.StoreMsg(Msg, a_i, RSAddr) // 전달 메시지를 지역서버의 버퍼에 저장한다.

    RS.receive(NewHostAddr);      // 지역서버에 새 작업자의 주소를 받는다.
    if compare(Addr of NewHost, RSAddr);    // 새로운 작업자가 다른 지역에 있
                                               는 경우
        SendMsg(Msg, a_i, RSAddr);          // 먼저 해당 지역서버로 메시지 전달
    fi;
    SendMsg(Msg, a_i, NewHostAddr);         // 새로운 작업자에게 메시지 전달
    StageCommit(S_i);
End Function;
```

(그림 6) 이동 에이전트 메시지 전달 과정

5. 성능평가

본 논문에서 기존 다중 지역을 다루었던 대표적인 기법인 SPC 프로토콜에 SRBA가 적용됐을 경우와 본 논문의 제안한 기법인 SRLM 프로토콜을 이주 과정에 따른 메시지 갱신 비용과 전달 비용을 수치적 분석(numerical analysis)에 의해 성능평가 하였다. SPC 프로토콜에서는 이동 에이전트가 이주한 단계군 내의 모든 실행장소에 대해 위치 갱신 메시지를 전달하게 된다. SRLM 프로토콜에서는 단계군 내의 대표 작업자에 대해서만 위치 갱신 메시지를 전달하게 되고, 이동 에이전트가 실행 중 오류가 발생했을 경우에 대해서만 추가적인 위치 갱신 메시지가 발생한다. 위치 갱신 메시지 수는 다음과 같이 나타낼 수 있다.

- SPC 프로토콜: (단계군 개수-2) × 단계군 크기 + 1
- SRLM 프로토콜: 단계군 개수 + 이동에이전트 실행 오류수 - 1

이주 과정에 대한 위치 갱신 메시지 비용은 다음과 같다.

- SPC 프로토콜 : 2×지역내 이주수 + 4×지역외 이주수
- SRLM 프로토콜 : 지역내 이주수 + 3×지역외 이주수

따라서, 위치 갱신 메시지가 지역내 이주에 적용되는지 또는 지역외 이주에 적용되는 지에 따라 메시지 비용을 적용하면 된다. SPC 프로토콜에서 메시지 비용은 이전 단계군(S_{i-1})의 작업자와 다음 단계군(S_i)의 실행장소들이 같은 지역 또는 다른 지역에 존재하는지에 따라 달라진다. SRLM 프로토콜에서 결함 포용 위치 갱신 메시지 비용은 이전 작업자와 새로운 작업자가 같은 지역으로 구성된 단계군인지

〈표 2〉 SPC 프로토콜과 SRLM 프로토콜에서 위치 갱신 메시지 비용

구분	SPC 프로토콜	SRLM 프로토콜
위치 갱신 메시지 비용	n(2a+4b)	$a+3b+\sum_{S_i=2}^{m-1} k_{S_i}$

〈표 3〉 지역에 따른 단계군 구성시 위치 갱신 메시지 비용

구분	단일 지역내에 모든 단계군 구성	서로 다른 지역으로 모든 단계군이 구성
SPC 프로토콜	2an(최소비용)	4bn(최대비용)
SRLM 프로토콜	$a+\sum_{S_i=2}^{m-1} k_{S_i}$	$3b+\sum_{S_i=2}^{m-1} k_{S_i}$

에 따라 결정된다. 즉, 단계군의 실행장소가 지역외에 구성되어 있다 하더라도 이전 작업자와 새 작업자가 같은 지역으로 구성된 경우 지역내 이주 비용이 적용된다.

단계군의 개수가 m개($S_0, S_1, \ldots, S_{m-1}$), 단계군의 크기는 n개(일반적으로 3, 5, 7개로 구성)가 있고, 이동 에이전트가 단계군을 이동할 때, 지역내 이주 수를 a, 지역외 이주 수를 b라 하자. 각 단계군에서 실행장소의 결함 횟수는 도착지와 목적지 그리고 각 단계군에서 첫 번째로 실행되는 작업자를 제외한 모든 단계군에서 발생한 고장 호스트를 나타내므로 $\sum_{S_i=2}^{m-1} k_{S_i}(0 \leq k_{S_i} < n)$와 같이 표기된다.

SPC 프로토콜에서 위치 갱신 메시지 비용과 SRLM프로토콜에서 이주 과정 및 결함 포용적인 과정을 포함한 위치 갱신 메시지 비용을 살펴보자.

우선, 각 단계군의 구성이 같은 지역에 구성되어 있는 경우를 가정하고 메시지 비용을 측정해보면 <표 2>와 같다.

<표 2>에서 최선의 경우, 단계군이 하나의 지역내에 모두 구성되어 있다면 SPC 프로토콜의 경우는 2an, SRLM 프로토콜에서는 $a+\sum_{S_i=2}^{m-1} k_{S_i}$의 위치 갱신 메시지 비용이 발생한다. 모든 단계군 각각이 서로 다른 지역으로 구성된 경우 SPC 프로토콜의 경우는 4bn, SRLM 프로토콜에서는 $3b+\sum_{S_i=2}^{m-1} k_{S_i}$의 위치 갱신 메시지 비용이 발생하며, SPC 프로토콜에서는 위치 갱신 메시지 비용이 최악의 경우가 된다. <표 3>은 지역에 따른 단계군 구성시 위치 갱신 메시지 비용을 나타낸다.

같은 지역내에 호스트들 사이에서의 연결은 다른 지역 사이의 호스트들 사이보다 더 빠르고 신뢰성이 있다고 가정한다[8, 11]. 이 가정은 분산 환경에 제한 없이 적용된다. 일반적으로 같은 지역내에 호스트들은 LAN(10~100 MB/s)에 의해 연결되어져 있고 지역과 지역사이의 연결은 WAN으로 더 낮은 속도(64Kb/s~2Mb/s)로 동작한다[14]. LAN에서의 지연시간은 1~10ms를 WAN에서는 300~500ms를 갖는다. 본 연구에서는 결함 포용을 위해 단계군으로 구성된 이동에이전트 컴퓨팅 환경에서 위치 관리시 발생되는 지연시간 성능 측정을 목적으로 하고 있으며, 이동 에이전트 연산 수행 중 오류로 인해 이동 후에 다시 작업을 수행함으로써 발생하는 지연 시간에 대해서 다루지는 않겠다. LAN에서의 지연시간을 ℓ, WAN에서의 지연시간을 ω라 하자. 위의 <표 4>에 있는 위치 갱신 메시지 비용의 지연시간은 다음과 같이 측정된다.

〈표 4〉 SPC 프로토콜과 SRLM 프로토콜에서 지연시간

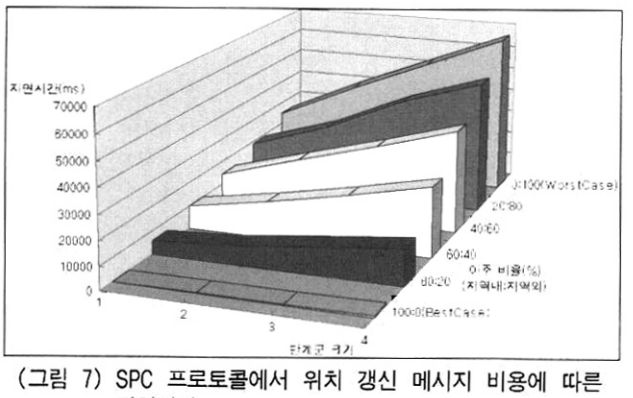
구분	지연시간
SPC 프로토콜	$(2a+b)\ell+3b\omega$
SRLM 프로토콜	$(a+b)\ell+2b\omega+(\sum_{S_i=2}^{m-1} k_{S_i}\times$지연시간$)$

SPC 프로토콜에서는 지역내 이주시, 단계군 내 지역서버와 호스트에 위치 정보를 알리게 되므로 2개의 메시지는 2ℓ의 지연시간을 갖게 되고, 지역외 이주시에는 홈 지역서버, 이전 지역서버, 이전 호스트, 현재 지역서버에 위치 정보를 알리게 되므로 4개의 메시지는 $3\omega+\ell$의 지연시간을 갖게 된다. 따라서 $(2a+4b)n \Rightarrow (2\ell a+(3\omega+\ell)b)n$의 지연 시간이 측정된다.

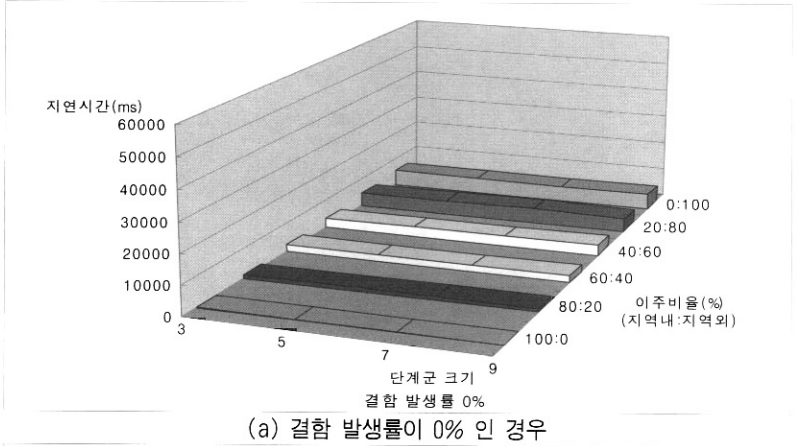
SRLM 프로토콜에서는 지역내 이주시 현재 지역서버에만 위치 정보를 알리게 되므로 1ℓ의 지연시간을, 지역외 이주시에는 홈 지역서버, 이전 지역서버, 현재 지역서버에 위치 정보를 알리게 되므로 $2\omega+\ell$의 지연시간을 갖게 된다. 여기에 결함이 발생한 호스트들에 대한 메시지 갱신 지연시간 $(\sum_{S_i=2}^{m-1} k_{S_i}\times$지연시간$)$을 추가하면, $a+3b=1\ell a+(2\omega+\ell)b+(\sum_{S_i=2}^{m-1} k_{S_i}\times$지연시간$)$의 지연 시간이 측정된다.

(그림 7)은 SPC 프로토콜 경우, 위치 갱신 메시지 비용에 따른 지연시간을 측정한 그래프이다. 이 경우 이동 에이전트가 복제된 모든 단계군 내의 실행장소에 대해 위치 갱신 메시지를 보내게 되므로 높은 메시지 비용이 발생하게 된다.

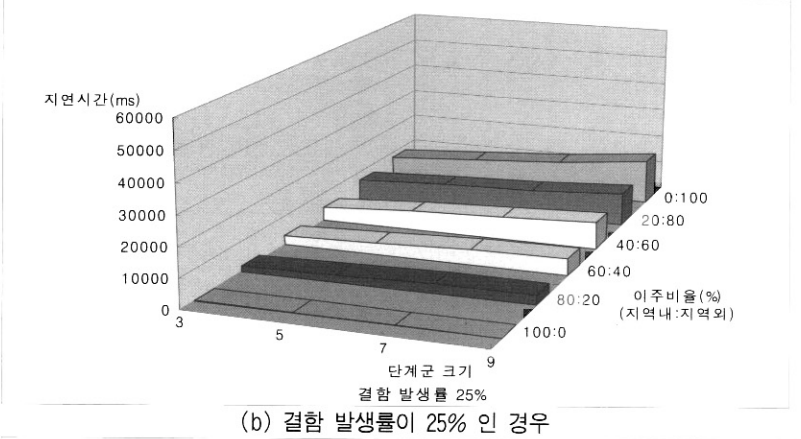
(그림 7)의 SPC 프로토콜에서 이주 비율이 100:0(지역내:지역외)인 경우 동일 지역내에 모든 단계군이 구성된 경우이므로 최소 메시지 비용이, 이주 비율이 0:100(지역내:지역외)인 경우 서로 다른 지역으로 모든 단계군이 각각 구성된



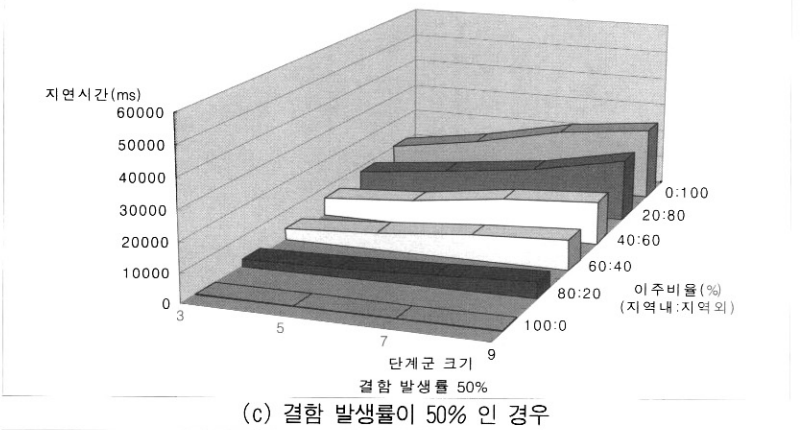
(그림 7) SPC 프로토콜에서 위치 갱신 메시지 비용에 따른 지연시간



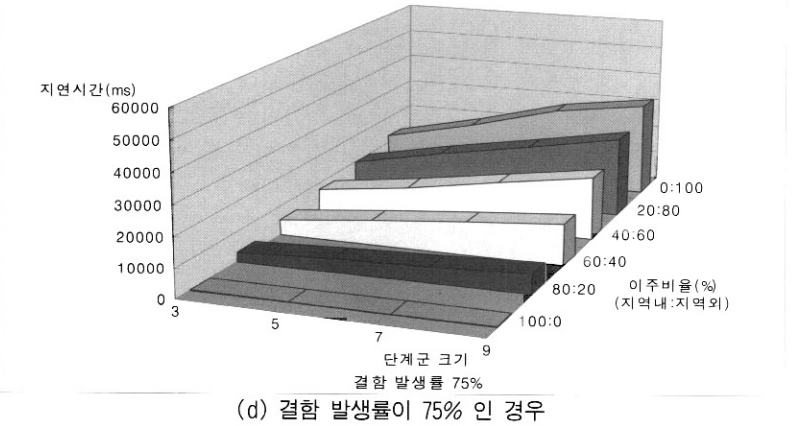
(a) 결함 발생률이 0% 인 경우



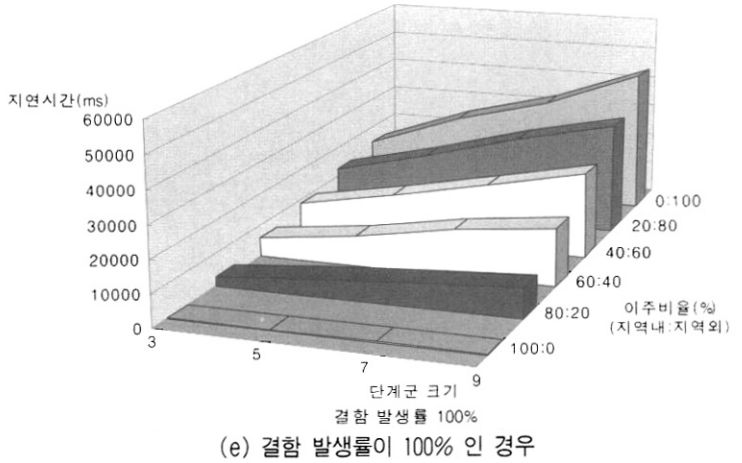
(b) 결함 발생률이 25% 인 경우



(c) 결함 발생률이 50% 인 경우



(d) 결함 발생률이 75% 인 경우



(e) 결함 발생률이 100% 인 경우

(그림 8) SRLM 프로토콜에서 이주 및 결함 포용 위치 갱신 메시지 비용에 따른 지연 시간

경우 이므로 최대의 메시지 비용이 발생한다.

(그림 8)의 (a)~(e)까지의 표는 SRLM 프로토콜에서 결함 횟수($\sum_{S_i=2}^{m-1} k_{S_i}$)을 증가시키면서 지연시간을 측정하였다.

(a)는 하나의 단계군에서 고장 호스트가 발생하지 않고 단지 하나의 대표 작업자만이 작업을 수행하는 경우이다. 앞에서 언급하였듯이 SRLM 프로토콜에서는 단계군 내에 대표 작업자에 대해서만 위치 관리 메시지를 보내게 되므로 SPC 프로토콜에 비해 메시지 비용이 줄어들게 된다. (e)의 경우는 실행되는 하나의 호스트를 제외한 나머지 결함포용 호스트 모두가 고장이 난 최악의 경우이다. 따라서 출발지부터 목적지 사이에서 실행 호스트를 제외한 단계군 내에 있는 모든 결함포용 호스트가 고장이 발생하여 위치 갱신 메시지를 전달하는 경우로, 그 비용은 (그림 7)의 SPC 프로토콜에서 메시지 비용과 비슷하게 측정된다. 그러나 (e)이와 같은 경우는 확률적으로 매우 미미하다.

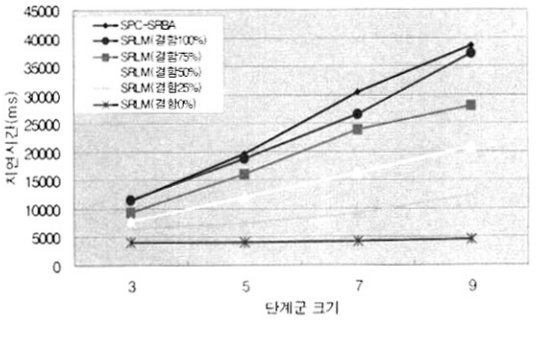
결함 포용적 위치 등록 과정에 대한 메시지 비용을 계산해보자. 이전 기법에서는 작업자의 고장이 난 경우 새로운 작업자에 대한 위치 관리 기법이 없었다. 본 SRLM 프로토콜에서는 대표 작업자가 고장이 난 경우 새로운 작업자에 대한 위치 정보를 등록함으로써 메시지 전달을 보장한다.

이동 에이전트가 생성된 출발지(S_0)만이 홈 지역내에 있고 다른 단계군들은 다른 지역에 구성되어 있다고 가정 하자. 메시지를 전달하고 하는 통신자가 홈 지역외 있을 때 메시지 전달 비용을 계산해보자. 메시지 전달 과정에 따른 지연시간을 살펴보면, 우선 홈 지역서버에게 메시지를 전달하는 시간은 WAN에서의 지연시간을, 다른 지역서버 메시지 전달시간 또한 WAN을, 지역서버가 지역내에 있는 호스트에 전달하는 시간은 LAN에서의 지연시간이 적용된다. (1 Message 지연시간 = $2\omega+\ell$)

<표 5>와 (그림 9)는 SPC 프로토콜과 SRLM 프로토콜에서 위치 메시지 전달 비용과 지연시간을 나타내고 있다.

<표 5> SPC 프로토콜과 SRLM 프로토콜 메시지 전달 비용

구 분	메시지 전달 수	지연시간
SPC 프로토콜	$3\times(m-2)\times n$	$(2\omega+\ell)\times(m-2)\times n$
SRLM 프로토콜	$3\times(m-2)\times\sum_{S_i=2}^{m-1}k_{S_i}$	$(2\omega+\ell)\times(m-2+\sum_{S_i=2}^{m-1}k_{S_i})$



(그림 8) SPC 프로토콜과 SRLM 프로토콜 메시지 전달 지연시간

<표 5>의 메시지 전달 비용을 살펴보면 SPC 프로토콜에서는 단계군 내의 모든 실행장소에 메시지를 전달하게 되고, SRLM 프로토콜에서는 단계군 내에 실행장소가 고장 난 경우 다시 메시지를 전달하게 된다. (그림 9)는 SPC 프로토콜과 SRLM 프로토콜 메시지 전달 지연시간을 나타낸 그래프로 최악의 경우로 메시지 전달이 마지막 호스트까지 전달되는 경우는 SPC 프로토콜에서에서 발생하는 메시지 전달 지연시간과 비슷하다. 그러나 SRBA 기법에서 단계군 내에서 결함으로 모든 호스트를 사용하는 경우는 매우 확률이 낮다. 지역 서버는 메시지가 전달된 후 단계군의 작업이 끝날 때까지 메시지를 버퍼에 저장한다. 새로운 작업자는 지역서버에 위치 갱신 메시지를 등록하고, 현재 지역서버 버퍼에 저장되어 있는 메시지를 가져온다.

한편, 기존 위치 관리 기법에서는 메시지 전달시 대표 작

업자가 고장이 난 경우 새로운 작업자에 대한 위치 관리가 없어 메시지 전달을 보장하지 못한다. 본 논문의 기법에서는 대표 작업자가 고장 난 경우 작업자를 선출한 후 새로운 실행장소의 위치 정보를 갱신함으로써 이전 위치 관리 기법에서 다루지 못한 고장 난 호스트들에 대한 메시지 전달을 보장한다.

6. 결 론

본 논문은 다중 지역 이동 에이전트 컴퓨팅 환경에서 작업자가 고장 난 경우에 단계군 내에서 새로운 작업자가 선출되어 계속적인 연산을 보장할 수 있는 공간적 복제 기반 기법(SRBA)이 적용된 위치 관리 프로토콜 SRLM을 제안하였다. 이 프로토콜은 다중 지역에서 실행장소를 단계군으로 구성하여 대표 작업자의 고장이 발생한 경우 새로운 대표 작업자를 선출하고, 그 실행장소만을 등록함으로써 위치 등록비용을 감소시키고, 새로운 대표 작업자에게 메시지 전달을 보장한다. 성능평가 결과에서도 기존 다중 지역 이동 에이전트 컴퓨팅에서 위치관리를 다루었던 SPC 프로토콜에 비해 제안된 SRLM 프로토콜이 이주 과정에 따른 메시지 갱신 비용과 전달 비용을 매우 적게 발생시킴을 보여준다.

향후 연구로써는 단계군 내 실행장소들의 재구성 및 동적인 구성으로 이동 에이전트의 컴퓨팅 환경의 전체적인 부하를 감소시킬 수 있는 위치 관리 기법에 대한 연구를 진행하고자 한다.

참 고 문 헌

[1] S. Adnan, J. Datuin and P. Yalamanchili, "A Survey of Mobile Agent Systems", CSE 221 Final Project, June 13, 2000.

[2] D. Kotz and R. S. Gray, "Mobile Agents and the Future of the Internet", ACM Operating Systems Review, Vol.33, No.3, pp.7-13, August, 1999.

[3] R. S. Gray, "Agent Tcl: A transportable agent system", Proceedings of the CIKM Workshop on Intelligent Information Agents, Fourth International Conference on Information and Knowledge Management (CIKM 95), Baltimore, Maryland, December, 1995.

[4] R. Tahboub, V. Lazarescu, "Novel Approach for Remote Energy Meter Reading Using Mobile Agents", Information Technology: New Generations, 2006. ITNG 2006. Third International Conference on10-12, pp.84-89, April, 2006. Digital Object Identifier 10.1109/ITNG.2006.100

[5] N. Karnik and Anand Tripathi, "Design issues in Mobile Agent Programing Systems", IEEE Concurrency, pp.52-61, July Sep., 1998.

[6] D. Milojicic, M. Breugst, I. Busse, J. Campbell, S. Covaci, B. Friedman, K. Kosaka, D. Lange, K. Ono, M. Oshima, C. Tham, S. Virdhagriswaran1 and J. White, "MASIF : The OMG Mobile Agent System Interoperability Facility", In Proceedings of the Second International Workshop on Mobile Agents (MA'98), LNCS 1477, pp.14-15. springer Verlag, September, 1998.

[7] K. Mohammadi and H. Hamidi, "Modeling of fault-tolerant mobile agents execution in distributed systems", Systems Communications, 2005. Proceedings 14-17 Aug., 2005 pp.56-60, Digital Object Identifier 10.1109/ICW.2005.57

[8] A. Di Stefano, L. Lo Bello, C. Santoro, "Naming and Locating Mobile Agents in an Internet Environment", Enterprise Distributed Object Computing Conference, 1999. EDOC '99. Proceedings. Third International , 27-30, pp.153-161 9. 1999

[9] T. Y Yeh and T. I. Wang, "A mechanism for tracking mobile agents in a cluster topology", Parallel and Distributed Systems, 2005. Proceedings. 11th International Conference on Vol.1, pp.320-327, 20-22 July, 2005. Digital Object Identifier 10.1109/ICPADS.2005.33

[10] S. Pleisch and A. Schiper, "Fault-Tolerant Mobile Agent Execution", IEEE Transactions on Computers, Vol.52, No.2, pp.209-222, 2003.

[11] D. Deugo, "Mobile Agent Messaging Models", In Proc. 5th International Symposium on Autonomous Decentralized Systems, pp.278-286, 2001.

[12] J. Baumann, "A Comparison of Mechanisms for Locating Mobile Agents", IBM Research Report 3333, 1999.

[13] P. T. Wojciechowski, "Algorithms for Location Independent Communication between Mobile Agents" Technical Report DSC-2001/13, Departement Systemes de Communication, EPFL, 2001.

[14] D. Lange and M. Oshima, "Programming and Deploying Java Mobile Agents with Aglets," Addison Wesly, 1998.

[15] Object Space Inc., "The ObjectSpace Voyager Universal ORB", Technical Report, 1999.

[16] A. L. Murphy and G. P. Picco, "Reliable Communication for Highly Mobile Agents", Autonomous Agents and Multi-Agent Systems, Vol.5, 81-100, 2002.

[17] A. Lingnau and O. Drobnik, "Agent-User Communications: Requests, Results, Interaction", MA'98, LNCS 1477, pp.209-221, 1998.

[18] G. Cabri, L. Leonardi and Franco Zambonelli, "Mobile-Agent Coordination Models for Internet Applications", IEEE Computer. Vol.33, No.2, pp.82-89, 2000.

[19] G. Cabri, L. Leonardi, and F. Zambonelli, Reactive Tuple Spaces for Mobile Agent Coordination, Proc. 2nd Int'l Workshop Mobile Agents, Lecture Notes in Computer Science, No.1, 477, Springer-Verlag, pp.237-248, 1998.

[20] P. Domel, A. Lingnau, and O. Drobnik, "Mobile Agent Interaction in Heterogeneous Environments", Proc. 1st Int

l Workshop Mobile Agents, Lecture Notes in Computer Science, No.1219, Springer Verlag, pp.136-148, 1997.

[21] J. Cao, X. Feng, J. Lu, and S.K. Das, "Mailbox-based scheme for mobile agent communications Computer , Vol.35 Issue: 9, pp.54-60, Sept., 2002.

[22] A. Di Stefano and C. Santoro, "Locating Mobile Agents in a Wide Distributed Environment", IEEE Transactions on Parallel and Distributed Systems, Vol.13, No.8, 2002.

[23] M. Straßer, K. Rothermel, "Reliability Concepts For Mobile Agents", International Journal of Cooperative Information Systems(IJCIS), Vol.7, No.4, pp.355-382, 1998.

윤 준 원

e-mail : jwyoon@kisti.re.kr

2004년 고려대학교 컴퓨터학과(이학석사)

2005년 한국과학기술정보연구원 정보시스템부 표준화기술지원실

2006년~현재 한국과학기술정보연구원 지식정보센터 지식포털팀

관심분야 : 이동에이전트, 결함포용, 분산컴퓨팅, P2P 컴퓨팅, 그리드 컴퓨팅, 유비쿼터스 서비스

최 성 진

e-mail : lotieye@disys.korea.ac.kr

2003년 고려대학교 컴퓨터학과(이학석사)

2004년~현재 고려대학교 컴퓨터학과 박사과정

관심분야 : 이동에이전트, P2P 컴퓨팅, 그리드 컴퓨팅, 분산시스템, 결함포용

안 진 호

e-mail : jhahn@kyonggi.ac.kr

1997년 고려대학교 컴퓨터학과(이학학사)

1999년 고려대학교 컴퓨터학과(이학석사)

2003년 고려대학교 컴퓨터학과(이학박사)

2003년~현재 경기대학교 정보과학부 전자계산학과 조교수

관심분야 : 결함포용 분산시스템, 이동에이전트 시스템, 그룹통신, p2p 컴퓨팅