論文
3-8-9

계층형 관리 도메인을 위한 프록시 기반의 이동 에이전트 모델의 성능 평가 (Performance Evaluation of Proxy-based Mobile Agent Model for Hierarchical Management Domains)

박 상 윤*
(Sang-Yun Park)

요 약

네트워크 상의 분산 자원이 급증하면서 분산 자원에 대한 접근이 활성화되고 있다. 특히, 이동 에이전트 기술을 이용한 분산 자원의 접근은 고정 네트워크 상의 자원들에 대한 동적인 접근 뿐 아니라 이동 네트워크 상의 사용자를 위한 이동성 지원 메커니즘을 제공한다. 프록시 기반의 이동 에이전트 모델은 동적으로 변화하는 분산 자원들에 대하여 계층적으로 도메인을 할당하고 도메인 별로 프록시 서버를 배치하여 이동 에이전트의 관리와 이동 에이전트간의 상호작용을 촉진시킬 수 있는 이동 에이전트 네트워크 모델이라고 할 수 있다. 본 논문에서는 계층적인 관리 분야에 적합한 프록시 기반의 이동 에이전트 모델의 구조 및 동작 시나리오를 소개하고 시뮬레이션을 통해서 프록시 서버의 경로 최적화 기능과 이동 에이전트의 실행 시간 단축 성능을 평가한다.

ABSTRACT

As the distributed resources in the networks have become increasingly popular, the accesses to these resources having been activated. Especially, the accesses to the distributed resources using the mobile agent technologies provide the mechanisms supporting mobility with mobile users as well as the dynamic accesses to the resources in the fixed networks. Proxy-based mobile agent model is defined as mobile agent network model which allocates the hierarchical domains to the distributed resources changed dynamically, assigns one proxy server for each domain, and promotes the management and the cooperation of the mobile agents. In this paper, we introduce the architecture and the execution scenario for proxy-based mobile agent model which is suitable for the hierarchical management domains. In simulation, we evaluate the proxy server's route optimization functionality and the performance reducing execution time of the mobile agents.

***Key words** : mobile agent, mobile agent systems*

* 준회원 : 성균관대학교 대학원 정보통신공학부

논문접수 : 2002. 7. 16.
심사완료 : 2002. 8. 12.

1. 서론

네트워크 상의 분산 자원이 급증하면서 분산 자원에 대한 접근이 활성화되고 있다. 현재까지 분산 자원의 접근 및 관리를 위해 다양한 네트워크 모델이 발표된 바 있는데 특히 이동 에이전트 기술을 이용한 분산 자원의 검색, 관리 및 공유는 새로운 패러다임으로 자리잡고 있다. 이동 에이전트 기술은 고정 네트워크 상의 자원들에 대한 동적인 접근 뿐 아니라 이동 네트워크 상의 사용자를 위한 이동성 지원 메커니즘을 제공한다. 현재까지 발표된 이동 에이전트 시스템들은 에이전트의 이주, 실행, 접근 제어, 에이전트간의 상호작용 및 안전한 통신 등을 위해 개별적인 구조와 구현 기술을 제시한 바 있다 [1][2].

프록시 기반의 이동 에이전트 모델은 동적으로 변화하는 분산 자원들에 대해 계층적인 도메인을 할당하고 도메인 별로 프록시 서버를 동적으로 이주시켜 이동 에이전트의 관리와 이동 에이전트간의 상호작용을 지원하도록 하는 네트워크 모델이라고 정의할 수 있다. 본 모델에서 프록시 서버는 이동 에이전트들의 도메인별 경로 최적화 기능을 제공하여 중복 검색을 예방하고 이동 에이전트 이주 및 실행 시간을 단축하며 대역폭 분산을 유도한다. 본 논문에서는 본 모델의 구조와 기능을 소개하고 시뮬레이션을 통해 본 모델의 효율성을 평가해 본다 [3].

본 논문의 2장에서는 기존 이동 에이전트 시스템들의 특성을 소개하고 3장에서는 본 모델의 구조, 구성 요소 및 동작 시나리오를 설명한다. 4장에서는 시뮬레이션을 통한 성능 평가 결과를 제시하며 5장에서는 요약 및 향 후 전망을 제시한다.

2. 기존 이동 에이전트 시스템

Telescript는 General Magic사에 의해 1990년대 초반 상업용 응용을 위해 구현된 최초의 이동 에이전트 시스템이다. 플레이스 (place)라고 하는 Telescript 서버는 고정 호스트에서 방문 에이전트에게 서비스를 제공하고, 이동 에이전트는 go 프리미티브를 통한 호스트간의 이동과 meet 프리미티브를 통한 에이전트간의 상호 작용을 할 수 있다. Telescript는 확장된 보안과 접근 제어 기능을 제공하는데, 플레이스는 방문 에이전트의 권한을 질의하여 접근을 불허하거나 자원에 대한 접근을 부분적으로 제한할 수 있다 [4].

Tacoma는 노르웨이 대학과 코넬 대학의 연합 프로젝트로서 에이전트는 Tcl로 작성되었으며 에이전트 상태 정보는 폴더 (folders) 에 저장되어 브리프케이스 (briefcases) 에서 통합된다. 프로그래머는 에이전트 작성을 위해 프로그램을 압축하여 'HOST' 폴더의 'CODE' 라는 하위 폴더에 저장하고 시스템은 에이전트 이주를 위해 'HOST', 'CODE' 및 응용 관련 정보를 포함하는 브리프케이스를 목적 호스트에서 실행 중인 관리 에이전트에게 전송한다. 에이전트들은 meet 프리미티브를 사용하여 상호 작용 및 브리프케이스 교환을 할 수 있다. Tacoma는 비동기 및 동기 통신 방식을 모두 지원하며, 통신의 대용으로서 고정 저장소인 캐비넷 (cabinets) 을 사용하여 에이전트간의 응용 관련 정보 공유를 지원한다. Tacoma는 보안 관련 기능을 제공하지 않으며 에이전트들이 이주 기록 (checkpoint) 을 남겨서 리어가드 (rearguard) 에이전트가 이를 추적할 수 있도록 한다 [5].

Dartmouth사에 의해 개발된 Agent Tcl은 tcl 스크립트를 사용한 에이전트의 이주, 통신 및 질의를 지원한다. 에이전트 이주 시에는 에이전트 소스, 데이터 및 실행 상태 등을 모두 이주하며 목적 호스트의 위치를 명시한 절대 이주만을 지원한다. Agent Tcl은 에이전트 복제를 허용하며 복제된 에이전트는 원하는 호스트에 이주되어 실행될 수 있다. 또한 에이전트간의 통신을 위해 메시지 교환 방식과 연결 설정 방식을 제공한다. Agent Tcl은 에이전트의 자원 접근을 제한하기 위해 'Safe Tcl' 라는 실행 환경을 제공하는데 에이전트가 적절한 보안 조치없이 위험한 연산을 수행하는 것을 예방한다. Agent Tcl은 동일한 호스트에서 발생한 모든 에이전트는 동일한 권한을 갖도록 설정된 접근 제어 목록을 관리하며, 'PGP' 프로그램을 사용하여 에이전트의 인증과 암호화된 정보 전송을 할 수 있다 [6][7].

Aglets은 IBM에서 개발된 자바 기반의 이동 에이전트 시스템이다. Aglets이라 불리는 에이전트는 'aglet context' 라는 에이전트 서버간을 이주한다.

Aglets은 콜-백(call-back) 프로그래밍 모델을 사용하는데, 시스템은 에이전트의 라이프 사이클에 특정 이벤트가 발생하면 에이전트의 해당 메소드를 호출한다. 에이전트는 절대 이주 방식에 따라 지정된 URL의 목적 호스트로 이주하며, 이동성 지원을 위해 자바의 객체 직렬화 기술을 사용한다. 목적 호스트에서 에이전트는 재활성화되어 run 메소드를 실행하는데 프로그래머는 이 메소드의 내용을 구현해야 한다. 에이전트간의 유일한 통신 수단은 메시지 교환으로서 동기식, 단방향식, 향 후 응답식이 있으며 상대방의 메소드를 호출할 수는 없다. 시스템이 retact 프리미티브를 통해 aglets을 재호출하는 동안에는 에이전트에 대한 접근 제어를 하지 않으며 제한적인 보안 기능을 제공하고 있다 [8].

ObjectSpace사에 의해 개발된 자바 기반의 에이전트 시스템인 Voyager는 'vcc' 라는 유틸리트를 사용하여 자바 클래스로 원격 접속이 가능한 가상 클래스 (virtual class) 를 생성한다. Voyager는 가상 클래스 객체를 원격 호스트에 생성하여 위치 독립적인 접근을 허용하는 가상 참조 (virtual reference) 를 구성하며 프로그래머는 이 메소드를 사용하여 에이전트를 호출할 수 있다. Voyager는 에이전트에게 유일 식별자와 이름을 할당하여 식별자와 이름에 의한 에이전트 호출이 가능하다. 가상 클래스는 호스트 이름이나 식별자를 통해 에이전트 이주를 지원하는 moveTo 프리미티브를 제공한다. 전달자 (forwarder) 객체는 에이전트 이주 후에도 호스트에 남아 에이전트에 접속하려는 요구를 이주 위치로 전달하는 역할을 한다. 에이전트 통신은 가상 참조 상의 메소드 호출로 이루어지고 동기식, 단방향식, 향 후 응답식의 통신 방식을 제공한다. 에이전트들은 계층화된 그룹으로 구성되기 때문에 멀티캐스팅도 가능하며 간단한 이주 기록 메커니즘도 제공한다 [9].

Mitsubishi Electric사에 의해 개발된 Concordia는 자바 기반의 이동 에이전트 시스템으로서 객체 직렬화를 이용한 이동성과 클래스-적재 (class-load) 메커니즘을 지원한다. Concordia는 그룹 상호작용 메커니즘과 비동기식 이벤트 신호 처리를 통한 에이전트 통신을 제공하며 신뢰성 있는 에이전트 전송과 이주 기록을 통한 복구 등을 지원하는 객체 영속 (object-persistence) 메커니즘을 통해 장애 복구 (fault-tolerance) 요구를 해결하고 있다. Concordia는 암호 프로토콜을 사용하여 에이전트 전송 및 저장 시에 에이전트의 상태를 보호하고 있으며 사용자 신원 기반의 접근 제어를 통해 자원을 보호한다. 그러나, 단방향 해쉬 함수를 통해 생성된 사용자 패스워드 인증 코드의 안전성이 확인되지 않았으며 에이전트 패스워드 검사를 위해 전역 패스워드 파일에 접근해야 하므로 개방형 네트워크에는 적합하지 않다 [10].

미네소타 대학에서 개발된 자바 기반의 시스템인 Ajanta는 이동성을 위해 객체 직렬화 기술을 사용하며 에이전트 코드는 에이전트 서버로부터 요구에 의해 적재된다. Ajanta는 공개키 프로토콜을 사용하여 에이전트 코드 및 상태를 암호화 및 인증하고 네트워크 주소에 독립적인 이름을 사용한 에이전트 절대 및 상대 이주를 지원한다. 에이전트는 다른 에이전트의 간섭을 피하기 위해 고립된 도메인 내에서 실행되고, 서버는 프록시 객체 내에 자원을 캡슐화하여 공유 자원을 보호하며 이를 통해 에이전트간의 안전한 통신을 제공할 수 있다. 네트워크를 통한 통신은 자바의 RMI를 사용하며 인증된 제어 함수는 원격 에이전트의 재호출 및 종료를 지원한다. Ajanta는 암호 메커니즘을 통해 에이전트 소유자가 에이전트 상태 중 특정 부분을 보호할 수 있도록 하고 에이전트가 자신의 상태를 특정 서버로부터 숨실 수 있도록 지원한다 [11].

3. 프록시 기반 이동 에이전트 모델

본 논문에서 제안하는 프록시 기반 이동 에이전트 모델은 계층적 관리 분야를 위한 이동 에이전트 네트워크 모델로서 도메인 개념없이 단일 서버에 의해 중앙 집중식으로 관리되었던 이동 에이전트 플랫폼들에게 계층적인 도메인을 부여하고 프록시 서버가 도메인별로 이동 에이전트와 이동 에이전트 플랫폼에 대한 관리를 담당하는 구조를 갖는다. 특히 본 모델은 중앙 집중식 이동 에이전트 구조에서 발견된 이동 에이전트들과 이동 에이전트 플랫폼들의 상호 운용성 및 중앙 서버 과부하 문제 등의 개선을 목적으로 한다. 본 모델은 프록시 서버를 통하여 논리적인 도메인 계층을 구성하고 분산 자원에 대한 중복 검색을 방지하는 이동 에이전트들의 경로 최적화와 이동 에이전트간의 상호작용 활성화를 지향한다.

3.1 구조

[그림 1]에서 예시하는 프록시 기반 이동 에이전트 모델은 다음과 같은 구성 요소들을 갖는다. 마스터 서버(MS : Master Server)는 프록시 서버, 이동 에이전트 및 전역 데이터베이스 관리 등을 담당하고, 프록시 서버(PS : Proxy Server)는 MS에 의해 특정 도메인으로 이주되어 도메인 내의 이동 에이전트 플랫폼과 이동 에이전트 및 지역 데이터베이스 관리 등을 담당한다. 본 모델의 이동 에이전트는 초기 MS에 의해 PS로 이주된 후 PS에 의해 획득된 경로 정보를 바탕으로 도메인 내를 여행하면서 정보 검색을 실시하고 PS와 VPN을 구성하여 결과를 보고한 후 다른 도메인의 PS로 이주한다. PS는 이동 에이전트에 의해 축적된 정보를 필터링하고 가공된 정보를 VPN을 통해 MS에게 보고한다. 이 때 이동 에이전트는 MS로부터 방문할 도메인들의 PS 리스트를 할당받으며 각 도메인의 PS로부터 방문할 이동 에이전트 플랫폼들의 최적화된 경로 정보와 이질적인 이동 에이전트와의 상호작용에 필요한 실시간 위치 정보를 획득한다.

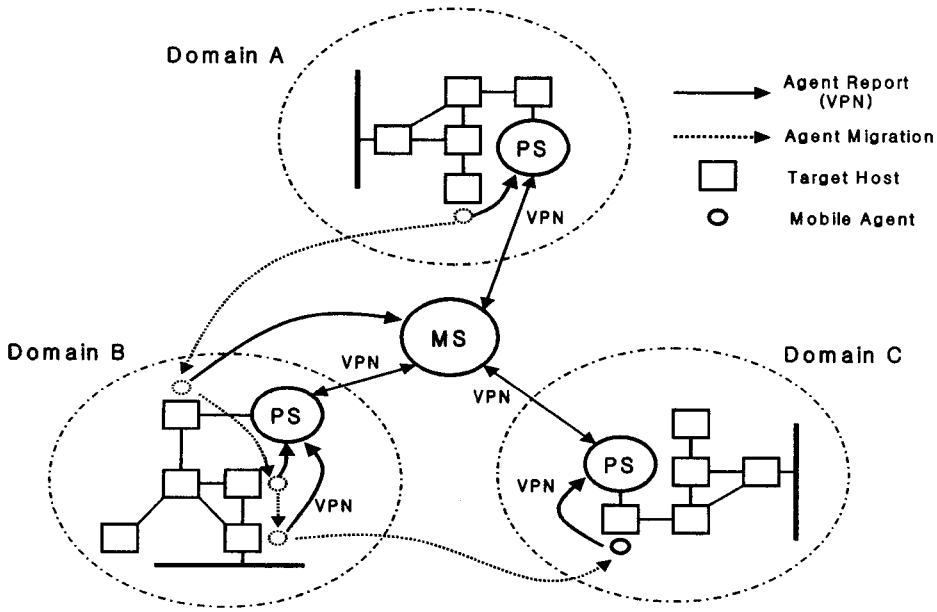
(1) 마스터 서버(MS : Master Server)

MS는 관리 대상 도메인들을 총괄하는 중앙 서버로서 도메인 리스트를 유지하고 각 도메인의 이동 에이전트 플랫폼들에 대한 테이블을 관리한다. MS는 각 도메인별로 PS를 이주시켜 지역 도메인을 관리하도록 하고 PS로부터 가공된 에이전트 보고를 주기적으로 수신한다. 수신된 에이전트 보고는 MS에 의해 2차 필터링되고 분류되어 전역 데이터베이스에 적재된다.

MS와 PS는 가공된 정보의 유출을 방지하기 위해 VPN으로 연결되는데 방화벽 등 각 도메인의 네트워크 특성을 고려하여 VPN 구성에 필요한 보안 속성들이 PS 이주시에 포함된다. MS는 이동 에이전트가 방문할 도메인들의 경로 지정을 위하여 이동 에이전트에게 순서화된 PS들의 리스트를 전달하며 PS 및 이동 에이전트의 송·수신을 위하여 객체 직렬화 기술을 사용한다.

(2) 프록시 서버(PS : Proxy Server)

PS는 지역 도메인을 총괄하는 지역 서버로서 이동 에이전트 플랫폼을 탑재한 도메인 내의 호스트들



[그림 1] 프록시 기반 이동 에이전트 모델

[Fig. 1] Proxy based mobile agent model

에 대한 테이블을 관리한다. PS는 MS에 의해 PS로 이주된 이동 에이전트를 이동 에이전트 플랫폼들에게 이주시켜 정보를 처리하도록 하고 이동 에이전트로부터 가공된 에이전트 보고를 수신한다. 수신된 에이전트 보고는 PS에 의해 1차 필터링되고 분류되어 지역 데이터베이스에 적재된다.

PS는 여러 도메인으로부터 다양한 경로를 통해 진입하는 이동 에이전트들의 도메인 내의 경로와 정보 검색의 중복을 예방하기 위하여 경로 최적화 과정을 수행한다. 즉, 이동 에이전트가 대상 자원에 대한 구체적인 정보를 갖지 못한 경우 과거 방문한 이동 에이전트들로부터 획득한 축적된 정보를 바탕으로 최적화된 경로 정보를 제공하고 최근 다른 에이전트에 의해 이미 동일한 정보가 검색된 경우 지역 데이터베이스에 적재된 정보를 MS로 전송하도록 한다. 또한 경로상의 특정 호스트가 과부하 중에 있을 경우 검색 대상 호스트의 순서를 변경하여 지연을 피하도록 한다. PS는 이동 에이전트의 경로 최적화를 위해 에이전트 라우팅 테이블을 운용하는데 라우팅 테이블에는 대상 자원의 위치, 최근 대상 자원에 접근한 이동 에이전트 정보 및 접근 시간 등이 유지되며 대상 정보의 속성에 따라 범주별로 분류되어 관리된다. 방문한 이동 에이전트는 먼저 접근 예정 정보와 이미 접근한 자원에 대한 정보를 자신의 정보와 함께 PS에게 등록하고 PS로부터 대상 자원에 대한 경로 정보를 획득한다.

PS는 이질적인 이동 에이전트간의 상호작용을 지원하기 위해 최근에 방문한 이동 에이전트의 현재의 예상 위치 정보를 유지하고 다른 이동 에이전트의 요청에 따라 이동 에이전트간의 상호운용을 주선한다. PS는 기존 이동 에이전트의 위치를 추적하기 위해 검색을 실시하고 필요에 따라 다른 PS와도 정보를 교환한다. 기존 에이전트의 위치가 감지되면 이질적인 에이전트들간의 상호운용을 주선하고 필요한 공통 인터페이스를 조율한다.

(3) 이동 에이전트(MA : Mobile Agent)

이동 에이전트는 검색 대상 호스트로 이주되어 이동 호스트 실행 플랫폼 상에서 제시된 항목에 대한 정보 검색을 수행한다. 이동 호스트는 PS로부터 전달받은 이동 에이전트 플랫폼들의 리스트에 따라 도메인 내의 호스트를 방문하며 검색된 정보는 VPN 연결을 통해 PS에 전송한다. 도메인 내에서의 여행이 끝나면 PS에게 도메인 이주를 통보하고 MS로부터 전달받은 다음 도메인의 PS에게로 이주한다. 리스트 상의 모든 도메인에 대한 방문이 끝나면 이동 에이전트는 소멸된다. 이동 에이전트의 이동과 실행을 위해서는 객체 직렬화 기술과 활성화 플랫폼이 요구되며 다중 이동 에이전트와 PS간의 다중 세션 제어에 의한 부하 감소를 위해 이벤트 또는 콜백 방식 등에 의한 단방향 메시지 전송 방식을 사용한다.

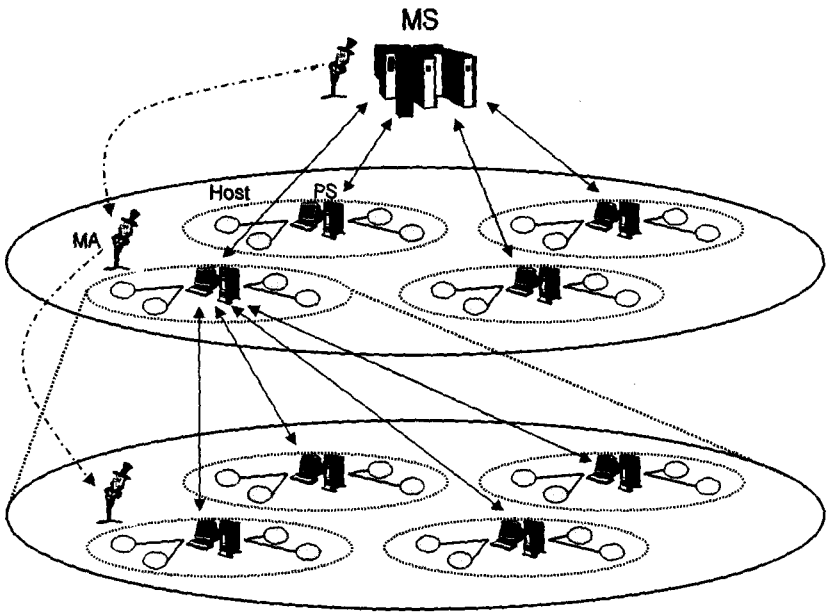
(4) VPN(Virtual Private Network)

MS와 PS간의 연결과 이동 에이전트와 PS간의 연결을 안전하게 수립하기 위해 VPN(Virtual Private Network) 기술을 사용한다. VPN은 IPSec 기술에 의한 터널링(tunnelling)을 수행하므로 MS, PS 및 이동 에이전트는 관련 보안 기능을 필요로 한다. 즉, 공개키 기반 구조(PKI : Public Key Infrastructure)에 의한 인증을 위해서는 인증서 관리 메커니즘이 필요하고, 암호 통신을 위해서는 키 관리 및 암호 프로토콜의 선정이 필요하다.

특히 이동 에이전트의 접근 제어를 위해서 PS와 대상 호스트는 이동 호스트 등급에 따라 접근 제어 매트릭스를 관리 및 적용해야 한다. 이동 에이전트의 등급은 MS에 의해 할당되며 등급 할당의 기준은 이동 에이전트의 역할에 따라 호스트 정보의 관리, 검색에 대한 다단계 등급으로 수립한다.

(5) 계층형 구조

[그림 2]에서 예시하는 바와 같이 프록시 기반 이동 에이전트 모델은 도메인간의 계층 구조에 의해 PS가 하위 PS들을 관리하도록 하여 도메인 범주 또는 네트워크 구성 등에 따라 논리적으로 광범위하게 구성된 도메인 트리를 MS가 총괄할 수 있도록 한다. MS는 가장 높은 단계의 PS들을 관리하며 PS는 직접 연결된 하위 PS들을 관리한다.



[그림 2] 프록시 기반 계층형 이동 에이전트 모델

[Fig. 2] Proxy based hierarchical mobile agent model

3.2 특징

(1) 부하 배분

프록시 기반 이동 에이전트 모델은 CPU 부하, 서버의 제한된 대역폭에 의한 전송 지연 및 다중 이동 에이전트와의 다중 세션 제어 부하 등과 같은 중앙 집중식 이동 에이전트 구조에서의 중앙 서버의 부하를 PS로 분할 할당함으로서 부하 배분 기능을 제공한다. 또한 도메인을 계층적으로 분할함으로서 대규모 호스트 그룹에 대한 관리가 가능하다.

(2) 경로 최적화

프록시 기반 이동 에이전트 모델은 도메인들에 대한 경로 설정을 MS 또는 상위 PS에서 수행함으로서 네트워크 상태나 호스트 부하 등의 실시간 정보에 기초한 동적인 경로 최적화 기능을 제공한다. 즉, MS나 상위 PS는 하위 PS 또는 호스트들의 상태를 모니터링하고 이동 에이전트 경로 설정시에 이를 반영하여 검색 대상 및 순서를 결정한다.

(3) 중복 검색 예방

프록시 기반 이동 에이전트 모델은 이동 에이전트의 중복 검색을 예방하기 위해 검색 대상 정보가 이미 다른 에이전트에 의해 검색되었을 경우 대상 호스트를 방문 경로에서 제외하여 자원 낭비를 최소화한다.

많은 이동 에이전트에 의해 수집된 정보들은 유효하지 않거나 높은 중복 가능성을 내포하고 있다. 데이터베이스를 관리하는 서버는 이러한 정보의 유효성 및 중복성을 필터링을 통해 검증해야 하는데, 필터링 대상 정보의 증가에 따른 유효성 및 중복성 검증은 급격히 증가할 수 있다. 프록시 기반 이동 에이전트 모델은 PS에서의 전단계 필터링을 통해 MS에 의한 최종 필터링 시간을 상당 부분 감소시킨다.

(4) 상호운용에 의한 정보 공유

프록시 기반 이동 에이전트 모델은 이질적인 이동 에이전트간의 상호운용을 지원하고 방문한 이동 에이전트들의 자원 접근 정보를 유지함으로서 이동 에이전트간의 정보 공유를 지원한다.

(5) 전송 지연 최소화

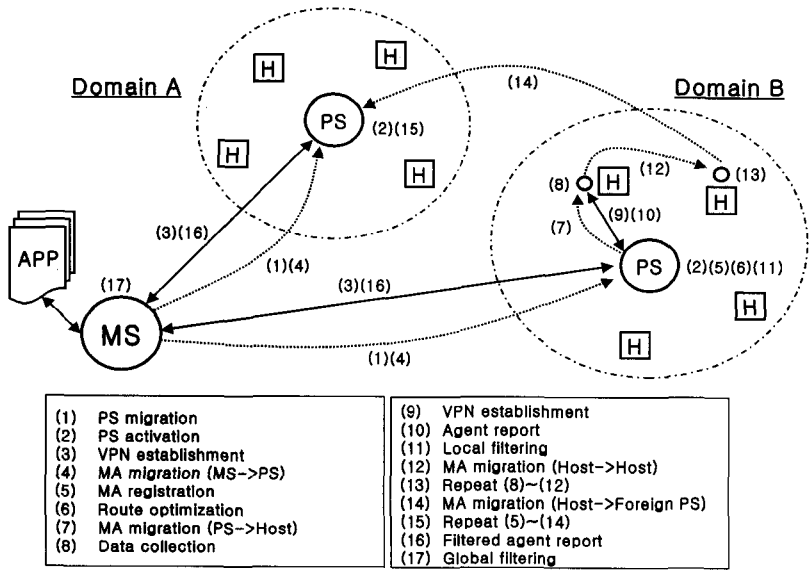
중앙 집중식 이동 에이전트 구조에서 중앙 서버는 다중 이동 에이전트와 다중 세션에 의한 검색 과정을 수행해야 한다. 이 때, 중앙 서버의 네트워크 대역폭은 제한적이므로 이동 에이전트 수에 비례한 다중 세션이 늘어날수록 전체적인 통신 지연은 급격히 증가하고 기존 이동 에이전트의 실행과 새로운 이동 에이전트의 생성에 영향을 줄 수 있다.

프록시 기반 이동 에이전트 모델에서 MS와 PS 또는 PS와 하위 PS간의 연결의 수는 고정되어 대역폭의 예측이 가능하고 이동 에이전트들은 각 PS에 분할 실행되므로 대역폭 부하의 분할 효과를 획득할 수 있다. 또한 하위 PS 또는 호스트에 대한 부하가 지속적으로 모니터링되어 경로 최적화에 반영되므로 대역폭 과부하를 예방한다.

3.3 동작 시나리오

[그림 3]에서 예시하는 바와 같이 프록시 기반 이동 에이전트 모델에서 MS, PS 및 이동 에이전트간의 동작 과정은 다음과 같다.

① MS는 지역 도메인들에 PS를 이주한다.
② 지역 도메인의 이동 에이전트 실행 플랫폼은 이주된 PS를 활성화한다.
③ MS와 PS는 안전한 통신을 위해 VPN을 구성한다.
④ 새로 생성된 이동 에이전트는 방문할 PS들의 목록을 할당받고 객체 직렬화 과정 후에 MS에서 PS로 이주된다.
⑤⑥ 이주된 이동 에이전트는 PS에 등록 후 도메인 내에서 방문할 경로 최적화된 호스트들의 목록을 할당받는다.
⑦⑧ 이동 에이전트는 대상 호스트로 이주된 후 필요한 정보를 검색한다.
⑨⑩ 정보 검색이 끝나면 PS와 VPN을 수립한 후 에이전트 보고를 전송한다.



[그림 3] 프록시 기반 이동 에이전트 모델의 동작 시나리오

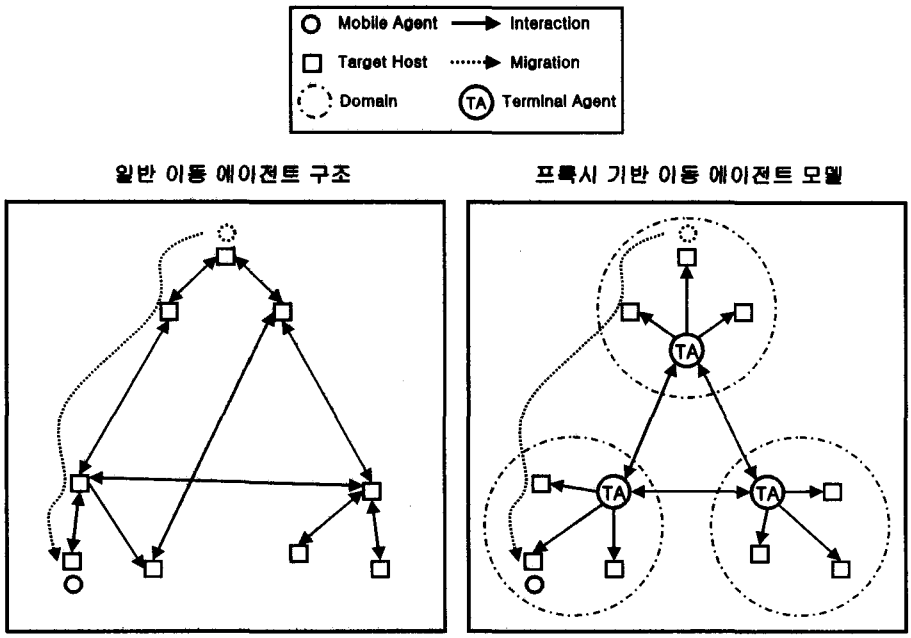
[Fig. 3] Scenario of the proxy based mobile agent model

⑪ PS는 수신된 에이전트 보고를 지역 필터링 한 후 지역 데이터베이스에 저장한다.
⑫ 이동 에이전트는 다음 호스트로 이주한다.
⑬ ⑧ ~ ⑫ 과정을 호스트 목록이 끝날 때까지 반복한다.
⑭ 이동 에이전트는 PS에게 타 도메인으로의 이주를 알리고 이를 실행한다.
⑮ MS가 전달한 PS들의 목록이 끝날 때까지 ⑤ ~ ⑭ 과정을 반복한다.
⑯ 각 도메인의 PS는 지역 필터링된 에이전트 보고를 VPN을 통해 MS에게 전달하고 MS는 이를 전역 필터링해서 전역 데이터베이스에 분류 및 적재한다.

4. 평가

4.1 시뮬레이션 모델

프록시 기반 이동 에이전트 모델을 평가하기 위하여 본 모델과 별도의 관리 서버가 없는 일반 이동 에이전트 구조를 비교하였다. [그림 4]에서 예시하는 바와 같이 일반 이동 에이전트 구조는 도메인 개념 없이 이동 에이전트 플랫폼을 탑재한 호스트들로 구성되며 이동 에이전트는 정해진 호스트 범위내에서 이주하며 자원에 대한 접근을 시도한다. 본 모델은 정해진 호스트 범위에 도메인을 부여하고 각 도메인에 프록시 서버를 할당하여 도메인별로 이주하는 이동 에이전트를 지원한다. 시뮬레이션은 임의의 수 생성 알고리즘과 이벤트 중심 시뮬레이션 모델 등을 제공하는 시뮬레이션 라이브러리인 SIMLIB C++을 사용하여 상태 변수에 변화를 기하면서 반복 실시하였다.



[그림 4] 시뮬레이션 모델
[Fig. 4] Simulation model

(1) 가 정

이동 에이전트 모델들의 시뮬레이션을 위하여 다음과 같은 가정을 수반한다.

① 각 모델은 동일한 개수의 호스트를 갖는다.
② 이동 에이전트는 경로 최적화에 의해 생략된 호스트를 제외한 모든 호스트를 방문한다.
③ 동일한 이동 에이전트는 이미 방문한 경로를 다시 방문하지 않는다.
④ 각 호스트 및 프록시 서버는 동시에 다중 이동 에이전트 검색을 지원한다.
⑤ 시뮬레이션에서 사용된 상태 변수에 대한 정의는 <표 1>과 같다.

<표 1> 시뮬레이션 상태 변수

<Table 1> Definition of simulation state variables

상태 변수	값
시뮬레이션 횟수	10회
시간 단위	분
시뮬레이션 시간	1,440분(1일)
이동 에이전트의 생성 간격	지수 분포(평균 : 10분)
이동 에이전트의 처리 데이터량	지수 분포(평균 : 100 kbyte)
이동 에이전트 이주 대역폭	지수 분포 (평균 : 40 kbyte / 초)
이동 에이전트의 디스크 검색 속도	지수 분포 (평균 : 40 kbyte / 초)
경로 최적화 유효 시간 간격	1분

(2) 평가 요소

완료 이동 에이전트 수(M_{cnt})

생성 후 모든 경로의 접근을 마치고 종료한 이동 에이전트의 수는 각 모델에서의 이동 에이전트 총 처리량의 지표를 제공한다. 이동 에이전트 도착 간격(AR_{exp}), 처리 데이터량(MSG_k) 및 호스트 수(H_{cnt}) 등의 상태 변수의 변화에 대하여 시뮬레이션 실행 시간 동안에 생성된 후 모든 경로의 접근을 마치고 종료한 이동 에이전트의 수의 변화를 관찰한다.

완료 이동 에이전트의 평균 응답 시간(MR_{ovg})

완료한 이동 에이전트의 생성 시간과 종료 시간의 차는 해당 이동 에이전트의 서비스 시간을 의미한다. 따라서 완료한 모든 이동 에이전트들의 총 서비스 시간(MR_{tot})의 평균은 사용자에 대한 완료 이동 에이전트의 평균 응답 시간을 의미한다. 이동 에이전트 도착 간격, 처리 데이터량 및 호스트 수 등의 상태 변수의 변화에 대하여 완료 이동 에이전트의 평균 응답 시간의 변화를 관찰한다.

<표 2> 완료 이동 에이전트의 평균 응답 시간

<Table 2> Average response time of departure mobile agents

$$MS_k = \frac{MSG_k}{DISK_{speed}} + \frac{MSG_k}{TRANS_{speed}}$$

$$MR_i = \sum_{k=1}^{H_{cnt}} MS_k$$

$$MR_{tot} = \sum_{i=1}^{M_{cnt}} MR_i$$

$$MR_{avg} = \frac{MR_{tot}}{M_{cnt}}$$

<표 2>에서 예시하는 바와 같이 이동 에이전트의 k번째 접근 시 처리 데이터량(MSG_k)는 지수 분포 (Exp($MeanMSGSize$))를 따르고 이동 에이전트 접근 시점의 대역폭($BR_{current}$)을 접근 시점의 이동 에이전트의 수($M_{current}$)로 나누면 이동 에이전트 별 대역폭($TRANS_{speed}$)를 구할 수 있다. 접근 시점의 디스크 검색 속도($DISK_{speed}$)는 지수 분포($Exp(MeanDisk-Speed)$))를 따르므로 k번째 접근의 접근 시간 및 이주 시간(MS_k)는 MSG_k를 $DISK_{speed}$와 $TRANS_{speed}$로 각각 나눈 값의 합으로 구할 수 있다. 따라서 완료 이동 에이전트 i 의 총 실행 시간(MI_i)는 검색 경로 상의 호스트들의 수(H_{cnt})까지 MS_k를 누적한 결과이며 완료 이동 에이전트들의 총 실행 시간(MR_{tot})는 MR_i를 완료 이동 에이전트 수(M_{cnt})까지 누적하면 구할 수 있다. 따라서 시뮬레이션 시간 동안 발생하여 모든 경로에 대한 검색을 마치고 종료한 이동 에이전트들의 평균 응답 시간(MR_{avg})은 MR_{tot}를 M_{cnt}로 나누면 구할 수 있다.

4.2 시뮬레이션 결과

(1) 이동 에이전트 도착 간격의 변화

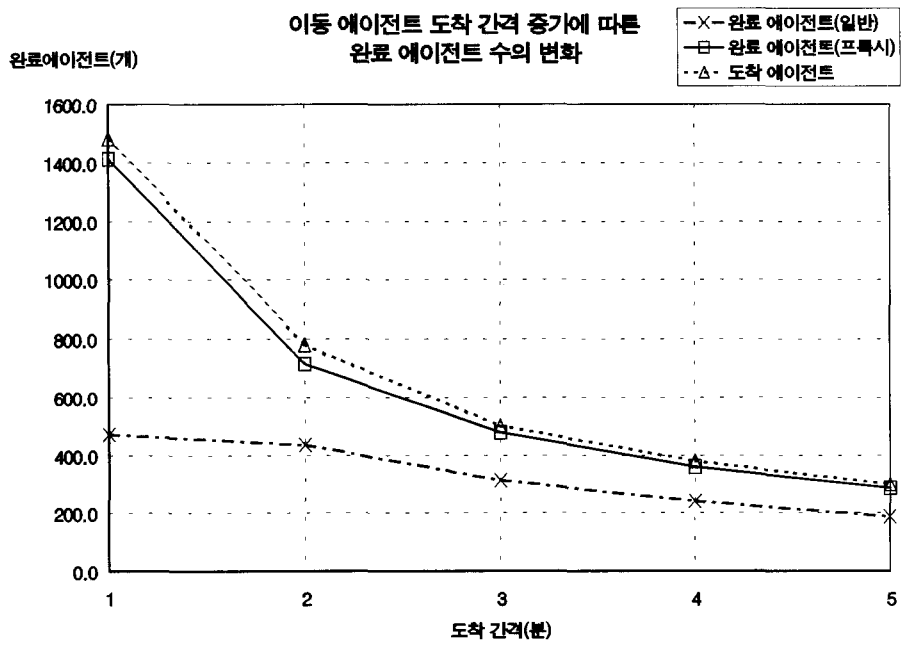
[그림 5]에서 예시하는 바와 같이 이동 에이전트의 도착 간격이 증가함에 따라 도착 이동 에이전트 수와 각 모델의 완료 이동 에이전트의 수는 감소함을 알 수 있다. 본 모델은 감소하는 도착 이동 에이전트 수에 대해 평균 94.5%의 상대적인 처리율을 나타내었으며, 일반 모델은 평균 47.9%의 처리율을 나타내었다. 따라서 1분 ~ 5분 구간의 이동 에이전트 도착 간격의 변화는 본 모델에 적은 영향을 미친 것을 알 수 있다.

[그림 6]에서 예시하는 바와 같이 이동 에이전트 도착 간격 증가에 대하여 본 모델은 평균 76.4분의 완료 이동 에이전트 평균 응답 시간을 나타내었으며 일반 모델은 평균 350.5분의 평균 응답 시간을 나타내었다.

(2) 이동 에이전트 데이터 처리량의 변화

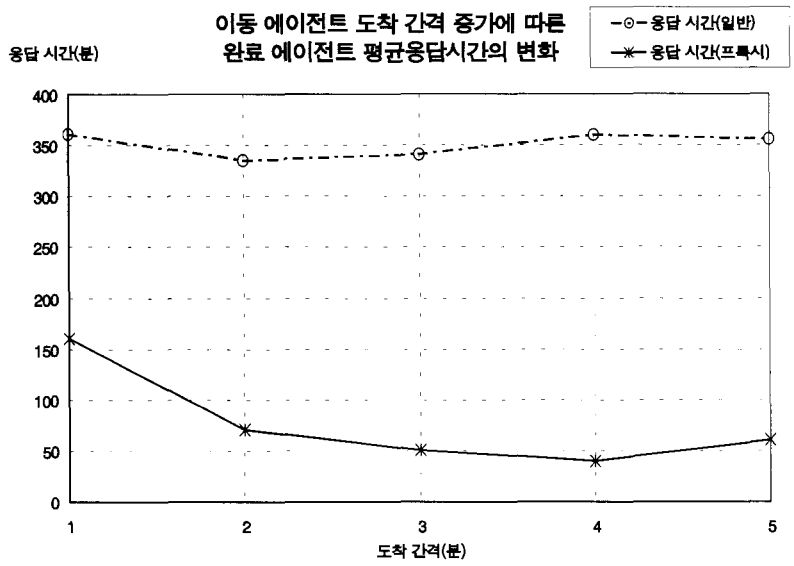
[그림 7]에서 예시하는 바와 같이 이동 에이전트 데이터 처리량이 증가함에 따라 각 모델의 완료 이동 에이전트 수는 감소함을 알 수 있다. 데이터 처리량의 증가는 각 호스트의 디스크 접근 부하를 증가시키고 이동 에이전트 이주 시의 전송 지연을 증가시키므로 완료 이동 에이전트의 수는 감소하게 된다. 평균 도착 에이전트 수가 1440개일 때 100,000byte ~ 500,000byte 구간의 데이터 처리량의 증가에 대해 본 모델과 일반 모델은 각각 평균 855.2개와 161.6개의 완료 이동 에이전트의 수를 나타내었다.

[그림 8]에서 예시하는 바와 같이 에이전트 데이터 처리량의 증가에 대해 완료 이동 에이전트의 평균 응답 시간은 증가함을 알 수 있다. 본 모델은 210.3분의 평균 응답 시간을 나타내었으며 일반 모델은 716.2분의 평균 응답 시간을 나타내었다.

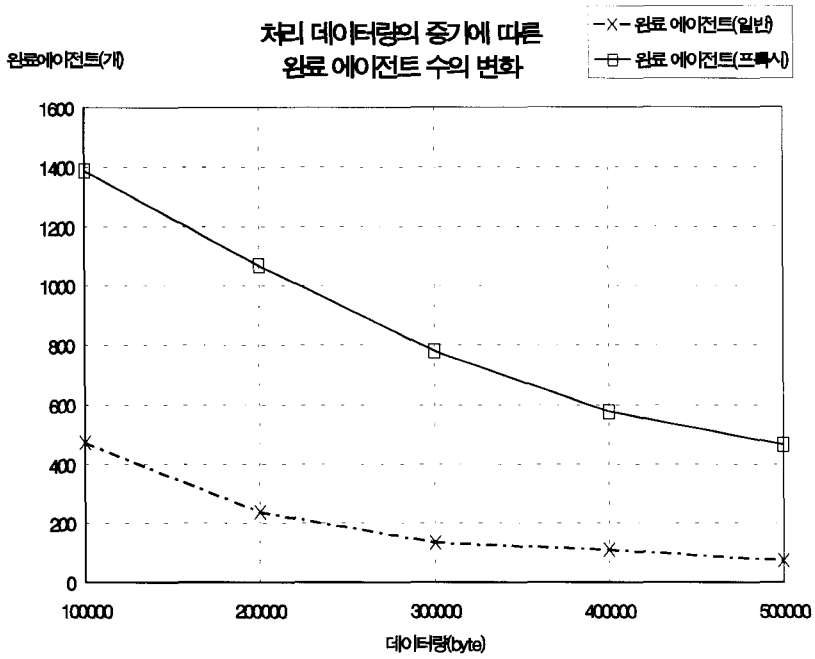


[그림 5] 이동 에이전트 도착 간격 증가에 따른 완료 에이전트 수의 변화

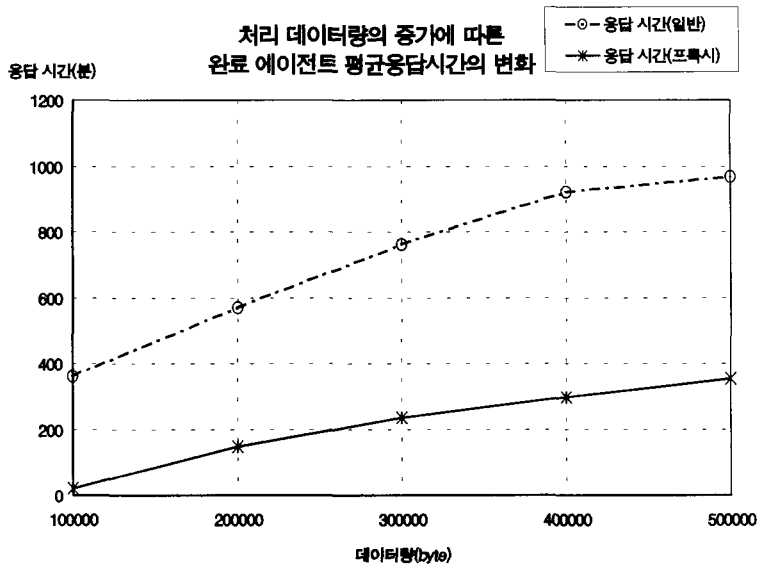
[Fig. 5] The number of departure agents according to arrival intervals



[그림 6] 이동 에이전트 도착 간격 증가에 따른 완료 에이전트 평균 응답 시간의 변화
[Fig. 6] Average response time of departure agents according to arrival intervals)

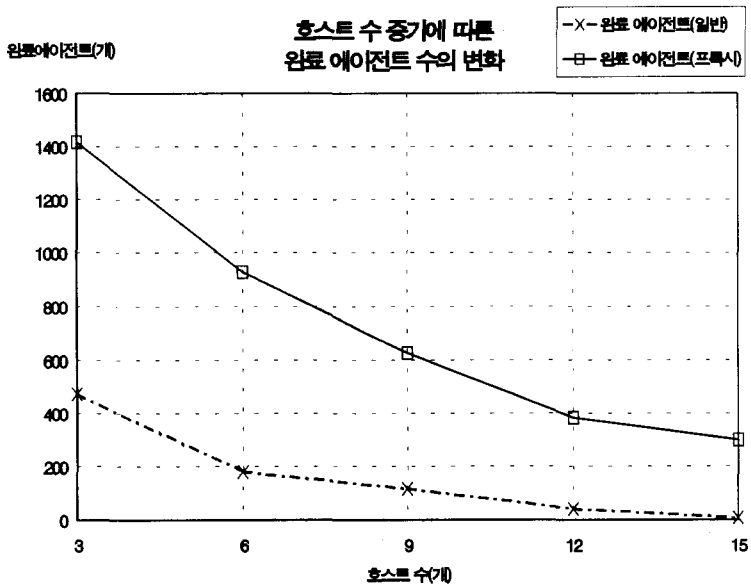


[그림 7] 처리 데이터량의 증가에 따른 완료 에이전트 수의 변화
[Fig. 7] The number of departure agents according to the amount of data



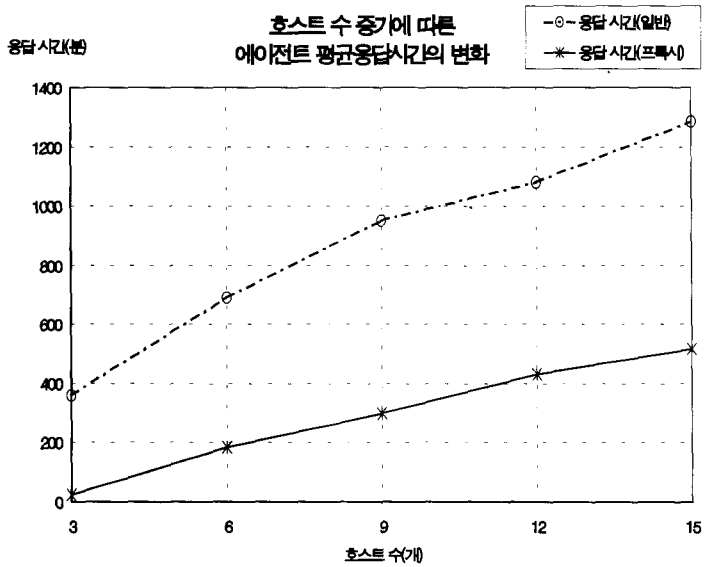
[그림 8] 처리 데이터량의 증가에 따른 완료 에이전트 평균 응답 시간의 변화

[Fig. 8] Average response time of departure agents according to the amount of data



[그림 9] 호스트 수 증가에 따른 완료 에이전트 수의 변화

[Fig. 9] The number of departure agents according to the number of hosts



[그림 10] 호스트 수 증가에 따른 완료 에이전트 평균 응답 시간의 변화

[Fig. 10] Average response time of departure agents according to the number of hosts

(3) 호스트 수의 변화

[그림 9]에서 예시하는 바와 같이 각 모델의 호스트 수가 증가함에 따라 각 모델의 완료 이동 에이전트 수는 감소함을 알 수 있다. 호스트 수의 증가는 이동 에이전트의 경로를 확장시키고 이에 따라 에이전트 접근 및 이주 시간을 증가시키므로 완료 에이전트 수를 감소시킨다. 평균 도착 에이전트 수가 1440개일 때 9개 ~ 45개 구간의 호스트 수의 증가에 대해 본 모델과 일반 모델은 각각 평균 730.2개와 164.2개의 완료 이동 에이전트의 수를 나타내었다.

[그림 10]에서 예시하는 바와 같이 호스트 수의 증가에 대해 완료 이동 에이전트의 평균 응답 시간은 증가함을 알 수 있다. 본 모델은 291.1분의 평균 응답 시간을 나타내었으며 일반 모델은 873.8분의 평균 응답 시간을 나타내었다.

5. 결론

이동 에이전트에 의한 분산 자원의 관리가 활성화 되고 있다. 이동 에이전트 기술은 분산 자원에 대한 동적인 접근 뿐만 아니라 이동 컴퓨팅 환경 하의 사용자들에게 이동성 지원 서비스도 제공하고 있다. 본 논문에서는 이동 에이전트 네트워크 상에 계층적인 프록시 서버를 배치하여 이동 에이전트 플랫폼과 이동 에이전트를 효율적으로 관리할 수 있는 프록시 기반 이동 에이전트 모델을 소개하고 시뮬레이션을 통해 본 모델의 성능을 평가하였다.

본 모델에서 프록시 서버는 이동 에이전트들의 경로 최적화 기능을 제공하여 중복 검색을 예방하고 이동 에이전트 이주 및 실행 시간을 단축하여 이동 에이전트의 효율적인 자원 접근을 지원한다. 또한, 이동 에이전트 네트워크 상에서 정보 공유 서버로의 역할을 수행하여 이동 에이전트간의 상호운용을 통한 정보 교환을 활성화시킨다.

빠르게 변화하는 분산 자원과 동적으로 진화하는 서비스 네트워크 환경에서 이동 에이전트 기술의 활

용도는 더욱 높아지고 있다. 프록시 기반의 이동 에이전트 모델은 이러한 요구를 만족시킬 수 있다.

※ 참고문헌

[1] Neeran Karnik and Anand Tripathi, "Design Issues in Mobile Agent Programming Systems," IEEE Concurrency, pp 52-61, July-Sep 1998.

[2] D. Milojicic, M. Breugst, I. Busse, J. Campbell, S. Covaci, B. Friedman, K. Kosaka, D. Lange, K. Ono, M. Oshima, C. Tham, S. Virdhagriswaran, 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.

[3] David Kotz, Guofei Jiang, Robert Gray, George Cybenko, and Ronald A. Peterson, "Performance Analysis of Mobile Agents for Filtering Data Streams on Wireless Networks," In Proceedings of the Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'2000), pp 85-94, August, 2000.

[4] J. E. White, Telescript technology : An introduction to the language, White Paper, General Magic, 1995.

[5] Dag Johansen, Nils P. Sudmann, and Robbert van Renesse, "Performance Issues in TACOMA," International 3rd. Workshop on Mobile Object Systems, 11th European Conference on Object-Oriented Programming, Jyvaskyla, Finland, June 1997.

[6] Robert S. Gray, "Agent Tcl : A flexible and secure mobile agent system," In Proceedings of the 4th Annual Tcl/Tk Workshop (TCL'96), July 1996.

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

[8] Gunter Karjoth, Danny Lange, and Mitsuru Oshima, "A Security Model for Aglets," IEEE Internet Computing, pp 68-77, July-August 1997.

[9] ObjectSpace, ObjectSpace Voyager Core Package Technical Overview, Technical report, ObjectSpace, Inc., July 1997.

[10] D. Wong, N. Paciorek, T. Walsh, J. DiCelie, M. Young, and B. Peet, "Concordia : An infrastructure for collaborating mobile agents," Proceedings of the First International Workshop on Mobile Agents (MA'97), LNCS 1219, pp 86-98. Springer Verlag, April 1997.

[11] Neeran Karnik and Anand Tripathi, "Agent Server Architecture for the Ajanta Mobile Agent System," In Proceedings of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98), July 1998.

박 상 윤

1997.2: 동국대학교 전자계산학과 학사
1999.2: 성균관대학교 전기전자 및 컴퓨터공학부 석사
2001.2: 성균관대학교 전기전자 및 컴퓨터공학부 박사과정 수료
관심분야 :
이동 컴퓨팅 시스템,
이동 에이전트,
멀티미디어 통신,
분산 시스템
E-mail :
bronson@ece.skku.ac.kr