분산 환경에서의 이동 에이전트 핸드오프 처리

박 기 현†

요 약

본 논문은 단말의 이동성을 수용하기 위한 분산 이동 구조에 대하여 다루고 있다. 제안된 구조에서는 세가지 분산 객체에 대하여 소개하고 있는데, 무선 이동 사용자가 이용하는 이동 단말(MS : Mobile Station), 유선망에서 이동 단말을 논리적으로 대신하는 이동 에이전트(MA : Mobile Agent), 이동 단말에 서비스를 제공하는 유선용 응용 서버(AS : Application Server)를 이동 환경에서 수용하기 위한 추적 에이전트(TA : Trace Agent)가 그들이다. 이동 에이전트는 이동 단말과 밀접한 관련을 갖고, 추적 에이전트는 응용 서버와 밀접한 관련을 갖는다. 핸드오프 과정에서 패킷 손실이나 순서 변경과 같은 문제점을 처리하기 위한 흐름 제어 기법이 고려되었으며, 제시된 알고리즘을 사용하는 경우의 추가적인 지연 시간에 대한 산술적인 결과를 제시하고 있다. 시뮬레이션을 통하여 알고리즘의 검증과 함께 산술적인 결과와 동일한 지연 시간을 확인하였다.

Handoff Processing for Mobile Agent in Distributed Environment

kihyun Park†

ABSTRACT

This paper presents a distributed mobile architecture aimed to support the mobility of mobile station. The proposed architecture considers three significant objects ; mobile station (MS), mobile agent (MA) which logically represents mobile station in the wireline network, and trace agent (TA) to represent application server (AS) providing application service to mobile station. The mobile agent is made closely coupled with the mobile station while the trace agent is made closely coupled with the application server. During the handoff process, a flow control mechanism is considered to correct packet loss and out-of-sequence packets problem. The result of analytic additional time delay due to handoff processing is presented and simulation was performed to check the additional time delay as well as the correctness of handoff algorithm.

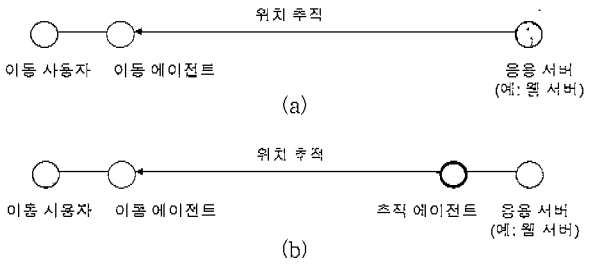
키워드 : 분산 컴퓨팅(Distributed Computing), 이동 컴퓨팅(Mobile Computing), 이동 에이전트(Mobile Agent)

1. 개 요

인터넷의 빠른 성장은 다중 지역에서 다중 사용자들이 정보를 빠르게 접근할 수 있는 길을 열어주고 있으며, 에이전트는 이 과정에서 중요하게 연구되는 분야의 하나이다[2, 3]. 이동 에이전트 (MA : Mobile Agent)는 스스로의 선택 혹은 외부적인 요인에 의하여 네트워크 상의 여러 호스트를 이동하는 분산 객체를 의미한다. 이동 에이전트의 장점은 클라이언트/서버 응용 환경에서 지연과 대역폭 관점에서의 향상과 네트워크 상의 연결 단절 현상을 감소시킬 수 있다는 점이다[2, 3, 5, 7]. 이동 단말 (MS : Mobile Station)의 급속한 보급에 따라서 인터넷 응용 환경에서 이들을 지원하는 것도 중요한 과제의 하나이다. 무선 환경의 상대적으로 열악한 통신 환경과 이동 단말 처리 능력의 한계를 극복하는 효과적인 대안으로 이동 에이전트를 이용하는 방안이 가능하다. 이동 에이전트는 무선 인터페이스 상의 통화량과 지연의 감소, 순간적인 연결 단절 문제의 극복 뿐만이 아니라, 협상의 대행, 단말기의 처리 능력 향상과 같은 기능도 수행한다[2, 4].

이동 단말이 웹 서비스와 같은 인터넷 서비스를 받기 위하여 서비스를 제공하는 응용 서버 (AS : Application Server)와 연결할 때, 둘 사이의 연결은 이동 에이전트에 의하여 중개될 수 있다. 이때 단말이 이동하면 에이전트도 함께 이동하게 되어 (그림 1)의 (a)와 같이 응용 서버에서는 에이전트의 위치를 추적하는 기능을 가져야 한다. 이는 결국 기존의 유선망 환경에서 사용되던 응용 서버들이 새로운 이동 환경에서는 수정되어야 한다는 것을 의미하게 된다. 따라서 이와 같은 새로운 부담을 서버에게 전가시키는 방식보다는 (그림 1) (b)와 같이 추적 에이전트 (TA : Trace Agent)를 두는 것이 효과적인 접근 방법이라고 할 수 있다.

† 정 회 원 : 위덕대학교 정보통신과 교수
논문접수 : 2001년 5월 18일, 심사완료 : 2001년 10월 5일



(그림 1) 이동/추적 에이전트

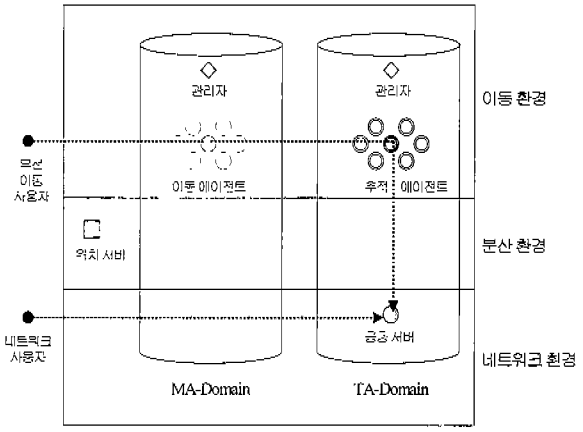
에이전트를 고정시키는 방식을 사용하고 있는 Infopad[8]와 PARCTAB[11]에서는 에이전트가 이동 단말의 위치를 추적하면서 적절한 경로로 데이터를 라우팅 해주는 역할을 수행해야 한다. 이들 시스템들은 서비스 영역이 제한된 지역에서만 동작하도록 가정되어 있으며, 그 영역이 커지게 되면 통신 환경이 상대적으로 나쁜 무선 시스템의 영향을 크게 받아서 에이전트의 효과가 상쇄 된다[1]. 실제로 Infopad의 경우는 학교 구내 영역을 가정하였으며, PARCTAB의 경우는 빌딩 내에서의 ubiquitous 컴퓨팅을 지원하기 위하여 설계되었다.

에이전트를 이동시키는 방식[4, 9, 10, 13, 14, 15]에서는 에이전트를 이동시키는 메커니즘에 관하여 많은 연구가 이루어 있으며, 서비스 제공자들이 에이전트의 위치를 추적해야 한다[12]. 이는 기존의 유선 환경에서 사용하던 모든 서비스 제공 객체들이 수정되어야 한다는 것을 의미한다. 즉, 서비스 영역의 크기에 제한을 받지 않기 위해서는 이동 에이전트가 적절하지만, 이동 에이전트만을 사용하는 경우는 기존의 유선 응용 환경인 응용 서버에게 에이전트의 위치를 추적해야 하는 부담을 주게 된다.

따라서 추적 에이전트의 존재로 인하여 응용 서버에게는 이동 환경이 투명하게 보이는 효과를 나타내게 된다. 본 논문에서는 이와 같은 모델을 지원하기 위한 새로운 분산 구조를 제시함으로써 다양한 하부 구조와 서비스 영역의 크기에 제한을 받지 않는 일반화된 구조를 제시한다.

전체 시스템은 위치 서버-이동 단말의 위치를 관리하는 MS 위치 서버(MS-Server : Mobile Station Location Server)와 응용 서버의 위치 정보를 제공하는 AS 위치 서버(AS-Server : Application Server Location Server)가 있다.- 의 관리 영역 단위인 도메인으로 구성된다. 각각의 서버들은 자기가 관리하는 도메인 내의 정보를 관리하고, 다른 도메인에 속한 동종의 서버들과 필요에 따라서 정보를 주고 받는 연합 구조를 갖게 된다.

도메인의 구성 요소에는 서버 이외에도 특정 응용 서버와 연결되는 추적 에이전트들을 관리하기 위한 도메인(TA-Domain : Trace Agent Domain)과 이동 단말이 접속 가능한 곳에서 이동 에이전트들을 관리하기 위한 도메인(MA-Domain : Mobile Agent Domain)이 있다. TA-Domain과 MA-Domain에는 도메인 관리자가 있어서 에이전트의 할당과 회수, 위치 정보 관리, 서버 및 다른 도메인 관리자와의 정보 교환 기능을 수행한다. MA-Domain 관리자는 자신의 관리 영역에 위치한 이동 단말의 위치가 변경될 때마다 이 정보를 MS-Server에게 전달하여 시스템 전체적으로 이동 단말의 위치를 관리할 수 있도록 한다.



(그림 2) 시스템 구조

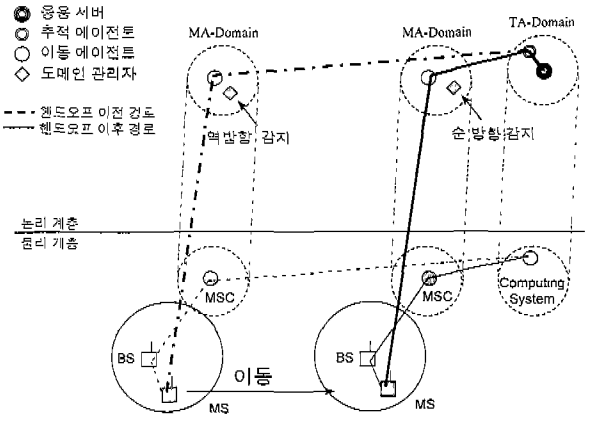
(그림 2)에서 제일 하부의 네트워크 환경은 고정 유선망 환경을 설명하고 있으며, 상부의 이동 환경에서는 이동하는 사용자를 지원하게 된다. 즉, 단말의 이동 과정에서 서비스를 제공하는 응용 서버와의 연결을 유지하기 위하여 이동 에이전트와 추적 에이전트를 통하게 된다.

2. 객체간의 동작 알고리즘

이동 단말이 서비스에 접속되는 과정은 망에 등록되는 과정과 등록된 단말이 응용 서버에게 서비스를 요구하는 두 단계로 구성된다. 위치 등록 요구는 이동 단말이 위치한 지역을 관리하는 MA-Domain 관리자에게 전달되며, 관리자는 이동 에이전트를 할당하여 단말과 연결 설정을 할 수 있도록 한다. 이때 단말이 위치한 도메인 지역의 MS-Server에게 위치 정보를 등록하게 된다. 위치 등록이 완료되면 응용 서버와 연결 설정을 시도할 수 있는데, AS-Server가 응용 서버의 위치 정보를 가지고 있으므로 이를 통하여 추적 에이전트의 주소를 얻을 수 있다. 이동 에이전트와 추적 에이전트의 주소를 모두 얻으면 하나의 연결 설정이 완성될 수 있다.

이동 단말, 기지국(BS), MSC (Mobile Switching Center), Computing System 들이 연결되어 동작하는 시스템에서 핸드오프 과정은 (그림 3)의 하단에 표시된 하부구조에서 발생하는 현상들이다. 이에 비하여 객체들의 동작은 상부 구조에서 이루어지므로 하부구조에서 발생하게 되는 다양한 조합의 무선 신호 측정/결정 단계는 두 가지로 추상화하여 에이전트에게 전달될 수 있다. 즉, 핸드오프 되기 이전 위

치의 MA-Domain 관리자에게 핸드오프 요구가 발생하는 역방향 감지의 경우와 새로운 위치의 관리자에게 핸드오프 요구가 발생하는 순방향 감지의 경우이다.



(그림 3) 물리적 환경과 에이전트 환경

핸드오프의 실행은 연결 제어 단계와 흐름 제어 단계로 구분되는데, 먼저 연결 제어 단계의 시작은 핸드오프 감지에 의하여 이루어지는데 가장 먼저 이루어지는 작업은 새로운 위치에서 이동 에이전트를 만드는 것이고, 이는 위치 등록 과정과 동일한 알고리즘을 갖는다. 이후 이동 단말, 이전 위치와 새로운 위치의 이동 에이전트, 추적 에이전트에게 핸드오프 사실이 통고되고, 각 객체들은 새로운 경로로의 연결을 설정한다.

흐름 제어 단계란 패킷 전달 경로의 순간적인 변경에 따라 패킷이 손실되거나, 패킷 순서가 변경되는 현상을 방지하기 위한 단계이다. 핸드오프 실행의 실행은 핸드오프 과정에서 이전 경로로의 패킷 전달이 가능한 역방향 핸드오프와 이전 경로를 이용한 패킷 전달이 불가능한 순방향 핸드오프의 두가지 경우가 가능하다.

핸드오프 과정에서 문제가 될 수 있는 패킷은 이전 위치의 이동 에이전트에서 처리 중인 패킷들이다. 이를 해결하기 위하여 "LAST"라는 특수 패킷을 사용하는데, 예를 들어서 역방향 실행 방식에서 추적 에이전트로부터 이동 단말로 전송되는 패킷들을 고려해 본다. 패킷 전송 중에 핸드오프 실행이 시작되면 이전 경로로 "LAST" 패킷을 전송하여 더 이상의 패킷 전송이 없음을 통지하고, 이후의 패킷은 새로운 경로로 전달하게 된다. 이때 새로운 위치의 이동 에이전트는 이전 위치의 이동 에이전트로부터 "LAST" 패킷이 도착할 때까지 수신된 패킷을 보관한다.

이전 위치의 이동 에이전트에서는 "LAST" 패킷 전까지 수신된 패킷들을 이동 단말에 전송하고, "LAST" 패킷을 이동 단말과 새로운 위치의 이동 에이전트에게 전달한다. "LAST"를 수신한 새로운 위치의 이동 에이전트는 추적 에이전트로부터 수신한 패킷을 이동 단말에 전달할 수 있다. 이동 단말에서는 이전 경로로부터 "LAST"가 도착할 때까지 새로운 경로의 패킷 처리를 지연시킴으로써 패킷들의 흐름 제어가 제대로 처리될 수 있다. 이동 단말에서 추적 에이전트로 전송되는 패킷들도 같은 원리로 처리될 수 있다.

순방향 핸드오프의 경우는 이전 경로를 사용할 수가 없으므로 이전 이동 에이전트에서 대기중인 패킷들을 새로운 위치의 이동 에이전트로 전송하여 수신단으로 전송되도록 해야 한다. 이때 새로운 위치의 이동 에이전트에는 송신단으로부터 오는 패킷과 이전 이동 에이전트로부터 오는 패킷을 구분해서 보관할 수 있도록 버퍼 구조가 만들어져야 하며, 수신단으로의 전송 순서도 이전 이동 에이전트로부터 수신된 패킷부터 이루어져야 한다.

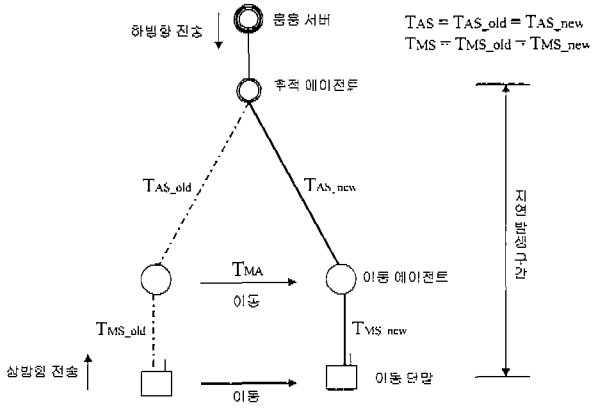
3. 핸드오프시의 추가 지연시간 분석

핸드오프 과정에서 성능에 영향을 줄 수 있는 요소는 여러 가지가 있는데, 연결 제어 단계에서는 이동 에이전트의 할당, 이전 에이전트 컨텍스트의 새로운 에이전트로의 복사와 연결 재조정에 걸리는 시간이 있을 수 있다. 이동 에이전트의 할당은 하나의 에이전트 프로세스를 탄생시키는 것인데 에이전트 풀을 이용하여 최소화 시킬 수 있으며 여기에서는 0으로 가정한다. 에이전트 컨텍스트 양의 경우에는 그 내용이 이동 단말의 특성 정보 등과 같은 기본적인 정보들을 포함하기 때문에 크지 않으므로 단순히 작은 패킷이 전달되는 시간으로 가정할 수 있다.

흐름 제어 단계의 실행 과정에서도 지연이 발생될 수 있다. 순방향 실행의 경우는 이동 단말과 이전 이동 에이전트 사이의 연결 설정이 끊어지므로 이동 단말에서 고정 에이전트 방향의 전송에서는 새로운 경로가 만들어질 때까지 데이타 전송이 일시적으로 중단된다. 반대 방향의 경우는 이동 단말에서의 전송은 중단되지 않지만, 이 패킷들이 새로운 이동 에이전트에서 지연되는 현상이 발생한다. 순방향의 경우는 연결 제어 단계가 완료될 때까지 양방향의 패킷 전송이 중단되므로 역방향 실행에 비하여 연결 제어 단계의 실행 시간만큼의 추가 지연 시간이 늘어나게 된다. 역방향 실행에서는 핸드오프의 과정에서도 송신단에서의 패킷 송신은 중단되지 않는다.

핸드오프 과정에서의 지연 시간 분석을 위한 시뮬레이션은 UNIX 환경에서 이루어 졌으며, 4개의 객체인 추적 에이전트, 이전 위치의 이동 에이전트, 새로운 위치의 이동 에이전트, 이동 단말은 프로세스로 구현되었다. 이들 프로세스들간의 패킷 전송은 소켓을 이용한 가상 네트워크 형태로 이루어졌다. 가상 네트워크의 역할은 객체간의 패킷을 전송하는데 걸리는 시간을 조정하기 위한 것으로써 객체에서 전송된 패킷은 가상 네트워크에서 보관하고 있다가 각 객체별 전송 거리인 일정 시간이 경과하면 수신자 객체에게 전달하도록 작성되었다.

데이터 패킷과 프리미티브 패킷의 크기는 1K로 하였다. 이동 단말과 추적 에이전트에서 발신되는 패킷의 간격은 일정하며, 전달되는 패킷의 크기가 지연 시간에 미치는 영향은 크지 않다고 가정한다. (그림 4)에서 객체들 간의 거리는 패킷 전송 시간으로 정의한다. 핸드오프의 발생은 MS에서 발신된 5번 패킷이 이전 이동 에이전트에 도착할 때 통보가 되는 것으로 하였다. 따라서 이동 단말과 추적 에이전트에서 핸드오프의 발생 시점을 인지하는 시간은 다를 수 있다.

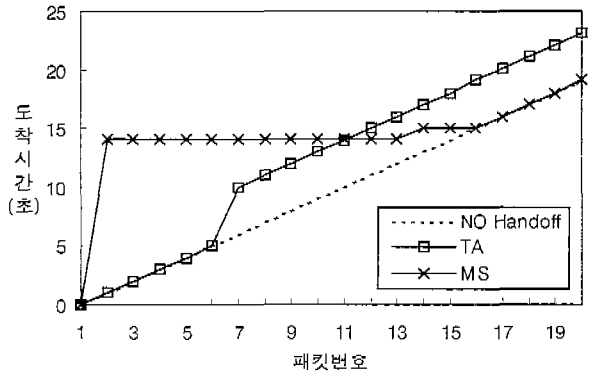


(그림 4) 패킷 전송 시간

본 논문에서는 순방향 핸드오프 실행에 대해서만 다루게 되는데, 역방향 실행의 경우 추가되는 지연 시간은 두 이동 에이전트 사이의 전송 시간인 T_{MA}에 의해서 영향을 받는다. 핸드오프의 발생은 이동 단말에서 발신된 5번 패킷이 이전 이동 에이전트에 도착할 때 이루어지는 것으로 가정했다.

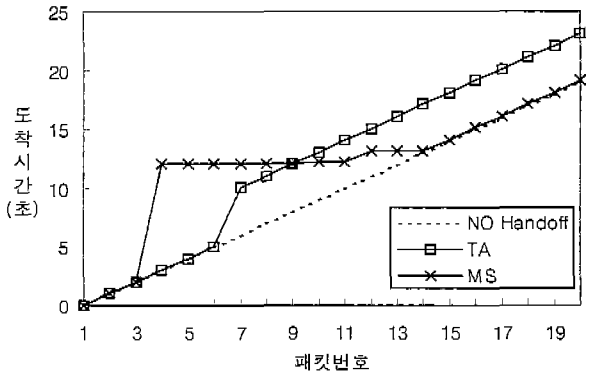
(그림 5)는 패킷의 발신 간격 I = 100ms, T_{AS} = 500ms, T_{MS} = 100ms, T_{MA} = 100ms인 환경에서 얻어낸 지연 시간을 나타내고 있다. 먼저 점선의 경우는 핸드오프가 발생하지 않았을 때의 패킷 도착 시간으로 일정한 수신 간격을 보이고 있다. 네모 표시가 된 실선은 추적 에이전트에서 수신된 패킷으로 이동 단말에서 전송한 패킷 6번 패킷의 처리가 완료된 후에 7번 패킷부터 이동 단말에서 전송되지 못하고 대기하다가 새로운 경로로 전송되면서 지연 현상이 발생되는 것을 알 수 있다. 이동 단말에서 수신한 패킷은 2번부터 12번 패킷까지 동일한 시간대에 도착하게 되는데, 이 패킷들은 이전 이동 에이전트에서 대기하고 있던 패킷들이기 때문에 이 패킷들에 한해서 발신 간격이 0ms이 되어 같은 시간대에 도착할 수가 있는 것이다. 또한 13번 패킷부터 지연 시간이 단축되는 현상이 발생하게 되는데, 이는 새로운 에이전트가 추적 에이전트로부터 미리 수신하고 있던 대기 패킷들이다.

참고로 두 객체에서 지연 현상의 발생 시점이 다른 이유는 T_{AS}와 T_{MS} 값 크기의 차이에 의한 실험의 특성에 의한 결과이다.

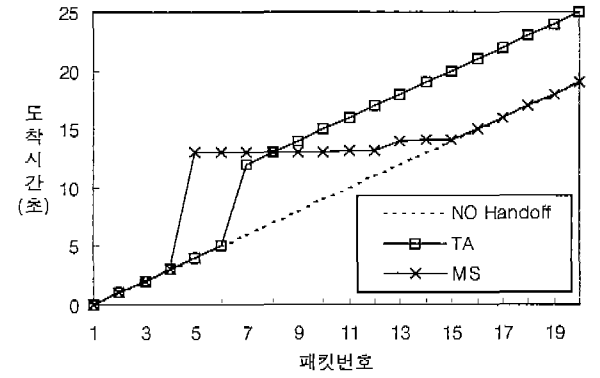


(그림 5) 실험 결과 (I = 100ms, T_{AS} = 500ms, T_{MS} = 100ms, T_{MA} = 100ms)

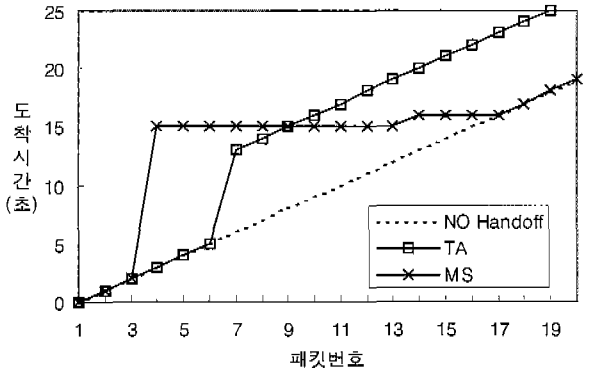
(그림 6)의 결과는 T_{AS}의 값을 300 ms으로 줄인 결과로써, 추적 에이전트의 수신 결과는 변화가 없고 이동 단말에서 수신한 결과는 4 초가 줄어들게 된다. (그림 7)은 (그림 6)에서 T_{MS} 값을 200 ms으로 크게 하여 얻은 결과로써, 추적 에이전트의 수신 결과는 2 초가 증가하게 되고 이동 단말에서 수신한 결과는 변화가 없게 된다. (그림 8)은 (그림 6)에서 T_{MA} 값을 200 ms으로 크게 하여 얻은 결과로써, 추적 에이전트의 수신 결과는 3초가 증가하게 되고 이동 단말에서 수신한 결과는 3초가 증가하게 된다.



(그림 6) T_{AS}의 영향 (I = 100ms, T_{AS} = 300ms, T_{MS} = 100ms, T_{MA} = 100ms)



(그림 7) T_{MS}의 영향 (I = 100ms, T_{AS} = 300ms, T_{MS} = 200ms, T_{MA} = 100ms)



(그림 8) T_{MA}의 영향(I = 100ms, T_{AS} = 300ms, T_{MS} = 100ms, T_{MA} = 200ms)

4. 결과 검토

4.1 결과 요약

<표 1>은 역방향/순방향 핸드오프 실행시의 상/하향 패킷의 추가 지연 시간에 대한 수식을 보이고 있는데, 앞 절의 그림에서 얻은 결과의 일반화된 표현이다. 이 수식은 알고리즘의 내용을 산술적으로 분석하여 얻어낸 것으로 여기에서는 그 결과만을 표시하였다.

결과에서 보는 것처럼 역방향 핸드오프의 경우는 이동 에이전트 사이의 거리에만 영향을 받는다. 순방향 핸드오프의 경우에는 세 종류의 값이 모두 영향을 미치는데, 이중에서도 T_{MA}값이 상대적으로 큰 영향을 주고 있다. 특히 상향 패킷의 경우는 T_{AS}값에 전혀 영향을 받지 않게 된다. 순방향의 결과는 이동 단말에서 추적 에이전트 쪽으로 데이타 전달이 많이 발생하는 응용 환경에서는 새로운 이동 에이전트의 위치가 이동 단말에 가까운 쪽으로 할당하는 것이 유리하다는 것을 설명하고 있다.

〈표 1〉 추가 지연 시간 요약

항 목	추가 지연	방 향
역방향 핸드오프 실행	$T_{back\_down} = T_{MA}$	AS -> MS
	$T_{back\_up} = T_{MA}$	AS <- MS
순방향 핸드오프 실행	$T_{forw\_down} = 2T_{AS} + T_{MS} + 9T_{MA}$	AS -> MS
	$T_{forw\_up} = 3T_{MS} + 9T_{MA}$	AS <- MS

4.2 역방향 실행중 순방향 핸드오프 발생

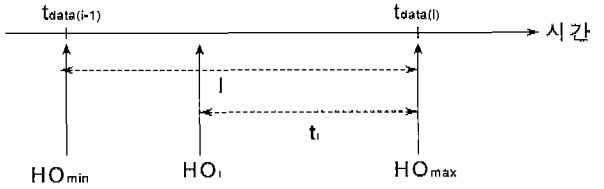
역방향 핸드오프의 처리 과정에서 이전 위치의 이동 에이전트와 이동 단말 사이에 단절이 발생할 수 있다. 이는 역방향 핸드오프 실행 과정에서 순방향 핸드오프가 발생하게 되는 경우이다. 역방향 핸드오프의 처리가 시작된 시점부터 단절이 발생하기까지 경과된 시간을 $T_{back_to_forw}$라고 하면, 개념적으로 다음과 같은 식을 얻게 된다. 식 (1)은 하향 패킷의 경우이며, 식 (2)는 상향 패킷의 결과이다.

$$T_{back\_to\_forw\_down} = T_{forw\_down} - T_{back\_to\_forw} \quad (1)$$

$$T_{back\_to\_forw\_up} = T_{forw\_up} - T_{back\_to\_forw} \quad (2)$$

4.3 발신 간격의 영향

(그림 9)는 이동 단말 혹은 추적 에이전트에서 패킷을 전송할 때의 발신 간격 I가 추가 지연에 미치는 영향을 보이고 있다. 그림은 패킷 data(i-1)을 $t_{data(i-1)}$ 시간에 전송하고 data(i)를 $t_{data(i)}$ 시간에 전송하는 경우로써 이들의 시간 간격은 I가 된다.



(그림 9) 발신 간격의 영향

앞 절에서의 성능 결과는 핸드오프의 발생 시점이 $t_{data(i)}$ 직전에 발생하는 것을 가정하였다. 만일 핸드오프의 발생이 $t_{data(i-1)}$ 시간 직후에 발생하면 다음 패킷 data(i)가 전달될 때까지의 시간 I 동안 핸드오프가 진행될 수 있기 때문에 이 시간 동안은 추가 지연에서 제외되게 된다. 일반적인 경우로써 중간에서 핸드오프가 발생하면 t_i 시간만큼의 추가 지연 감소효과가 발생하게 되므로 역방향과 순방향에서의 추가 지연 시간은 다음과 같이 된다.

$$T_{forw\_I} = T_{forw} - t_i \quad (3)$$

$$T_{back\_I} = T_{back} - t_{iv} \quad (4)$$

5. 결 론

무선 시스템을 이용하는 통신 수요의 증대와 기술의 발전에 따라서 기존의 유선 응용 환경과 무선 사용자 환경이 통합되는 새로운 응용 서비스 환경 지원에 대한 요구가 점점 증가하고 있다. 유/무선 통합 환경에서 이들 사이의 불균형적인 신뢰성과 대역폭의 차이를 극복하기 위한 방안의 하나로 에이전트 모델이 연구되고 있다.

본 논문은 분산 환경에서 단말기의 이동성을 지원하기 위한 방안에 관한 연구로써, 에이전트 프로세스를 이동 에이전트와 추적 에이전트로 분리한 새로운 모델을 제시하고 있다. 본 논문의 2단계 에이전트 모델은 서비스 영역의 제한을 받지 않으면서도 기존의 유선 환경 응용 프로그램들을 수용할 수 있다는 장점을 가지고 있다. 제안된 에이전트 모델을 지원하기 위한 분산 시스템의 구조는 확장성이 용이하도록 설계되었다. 제시된 모델은 TINA 구조와 같이 기존에 연구된 분산 모델과 쉽게 통합될 수 있도록 기능에 대한 제약을 최소화하여 설계되었다.

한편 기존의 연구 내용에 비하여 통신 과정에서 하나의 객체가 추가됨으로써 패킷 전달에 소요되는 지연 시간과 전체적인 시스템 부하의 증가가 예상되지만, 이는 본 논문에서 강조하는 서비스 영역의 확장성과 기존의 유선 환경에서 제공하는 응용환경의 수용이라는 측면과는 비교될 수 있는 요소는 될 수 없을 것이다.

제안된 모델은 특정한 시스템 환경을 가정하지 않고, 다양한 환경에 적용이 가능하도록 모델 설정이 이루어 졌다. 이동 시스템의 발전은 앞으로도 계속될 것이며, 이러한 이동 시스템이 기존 시스템과 연동되어 하나의 통합된 환경으로 진화될 것이라는 측면에서 본 논문에서 제시한 모델은 이러한 환경에 대한 하나의 토대가 될 수 있으리라고 생각된다.

참 고 문 헌

[1] Ramachandran Ramjee *et al.*, "The Use of Network-Based Migrating User Agents for Personal Communication Services," IEEE Personal Commun., Dec. 1995.

[2] Gian Pietro Picco, "Mobile agents : an introduction," Microprocessors and Microsystems, Vol.25, Issue2, 2001.

[3] David Kotz, Robert S. Gray, "Mobile Agents and the Future of the Internet," Department of Computer Science, Dartmouth College, 1999.

[4] L. A. Guedes, et al, "An agent-based approach for supporting quality of service in distributed multimedia systems," Computer Communications, Vol.21, Issue 14. 1998.

[5] Prithviraj, Nitya Narasimhan. Louise E.Moser. P.M. Melliar-Smith, "MAgNET : Mobile Agents for Networked Electronic trading," IEEE Trans. On Knowledge and Data Processing, Vol.11, No.4, 1999.

[6] Larry T. Chen, Tatsuya Suda, "Designing Mobile Computing Systems using Distributed Objects," IEEE Commun. Mag., 1997.

[7] Dejan Milojicic. "Mobile Agent Applications," IEEE Concurrency, 1999.

[8] M. T. Le *et al.*, "InfoNet : The networking infrastructure of Infopad," Proc. Compcon, California, Mar. 1995.

[9] L. M. Silva, et al, "Comparing the performance of mobile agent systems : a study of benchmarking," Computer Communications. Vol.23, Issue 8, 2000.

[10] Johnny Wong, et al, "SMART mobile agent facility," Journal of Systems and Software, Vol.56, Issue 1, 2001.

[11] Roy Want *et al.*, "An Overview of the PARCTAB Ubiquitous Computing Experiment," IEEE Personal Commun., Dec. 1995.

[12] Wen-Shyen E. Chen, et al, "A novel mobile agent search algorithm," Information Sciences, Vol.122, Issue 2-4, 2000.

[13] P. Farjami, et al, "Advanced service provisioning based on mobile agents," Computer Communications, Vol.23, Issue 8, 2000.

[14] N, Kawaguchi, et al, "MAGNET : ad hoc network system based on mobile agents," Computer Communications, Vol. 23. Issue 8. 2000.

[15] Claudia Raibulet, Claudio Demartini, "Mobile agent technology for the management of distributed systems a case study," Computer Networks, Vol.34, Issue 6, 2000.

박 기 현

e-mail : kihyun@mail.uiduk.ac.kr

1985년 고려대학교 전자공학과 졸업(학사)

1987년 고려대학교 대학원 전자공학과 (석사)

1998년 고려대학교 대학원 전자공학과 (박사)

1987년~1994년 삼성전자 선임 연구원

1999년~현재 위덕대학교 정보통신과 조교수

관심분야 : 이동 컴퓨팅, 운영 체제