

이동 에이전트 기반 실시간 네트워크 관리 모델

배인한* · 하숙정*

1. 서 론

최근 네트워크는 관련 기술의 급속한 발달로 다양한 기능 및 서비스를 제공하고 있으며 그 구성이 복잡해지며 지리적으로 분산화되는 경향이 있다. 그러므로 급변하는 네트워크 환경에서 신뢰성, 융통성, 확장성, 이·기종간의 공동 조작 등을 제공하며 네트워크 구성 요소를 효과적으로 감시하고 통제할 수 있는 네트워크 관리의 중요성이 점차 증가되고 있으며 이에 따라 효율적인 네트워크 관리 모델에 대한 많은 연구가 진행되고 있다.

기존의 TCP/IP의 네트워크 관리 프로토콜인 SNMP(Simple Network Management Protocol) [1-3]는 단순하며 구현이 쉬워 널리 사용되고 있다. 네트워크 관리자는 네트워크 관리 대상자에 대한 네트워크 관리 정보를 폴링 방식에 의해 수집하고 처리하는 클라이언트-서버 모델을 따르고 있다. 관리 작업의 집중화된 접근은 관리자의 처리 부하와 네트워크 트래픽을 증가시킨다[4]. 중앙 집중형 관리의 문제점을 보완하기 위해 분산 컴퓨팅 환경에서의 새로운 연구 분야인 이동 에이전트를 네트워크 관리에 사용하는 방법이 연구되고 있다[5,6].

이동 에이전트는 관리자로부터 관리 작업을 위임받아 관리 대상자간을 이동하면서 관리 작업을 수행 후 관리 작업 결과를 갖고서 관리자에게 복

귀한다. 그러나 다음 관리 대상자를 이동할 때마다 관리 작업 결과가 계속해서 누적되므로 관리 대상자의 수가 많아질수록 누적된 결과로 인한 전송 시간의 증가로 인해 전체 응답 시간이 증가될 수 있으며, 관리 작업 결과의 성능은 에이전트의 이동 방법에 따라 달라진다.

본 고에서는 네트워크 관리 프로토콜의 성능을 개선하기 위해 관리 대상자들을 클러스터링하고, 클러스터 별로 이동 에이전트를 병렬로 이동시키며, 클러스터별로 클러스터 내에 이동 에이전트들이 순회 계획에 따라 관리 대상자들을 이동한 후 관리자에게로 복귀함으로써 응답 시간과 트래픽 부하를 개선키는 이동 에이전트 기반 실시간 네트워크 관리 모델을 제시하고 SNMP와 Ku[6]의 모델과 응답 시간에 대해 성능을 비교한다.

본 고의 구성은 다음과 같다. 2장에서는 관련 연구로 이동 에이전트, 이동 에이전트 시스템, 네트워크 관리 시스템, 그리고 네트워크 관리 프로토콜 등에 대하여 살펴보고, 3장에서는 이동 에이전트 기반 실시간 네트워크 관리 모델을 제시하고, 4장에서는 제안하는 네트워크 관리 모델을 기존의 모델과 응답시간 측면에서 성능을 비교하고 평가한다. 그리고 5장에서 결론을 맺는다.

2. 관련 연구

2.1 이동 에이전트

이동 에이전트는 행위, 상태, 위치를 가지는 활

*대구효성가톨릭대학교 전자정보공학부

동적 객체로 정의된다. 이동 에이전트는 호출되어졌을 때 그 에이전트가 방문할 위치와 그 에이전트가 수행할 명령어를 자동적으로 결정하기 때문에 자치적(autonomous)이다. 이동 에이전트는 에이전트 실행을 위한 환경을 기본적으로 제공하는 위치들간에 이주할 수 있으므로 이동성이 있다. 이동 호스트로부터 보내진 이동 에이전트는 이동 호스트를 대신해서 정보를 모으기 위해 네트워크를 돌아다니고, 낮은 대역폭의 무선 연결을 통해 다수의 요청과 응답을 전송하는 대신에 각 호스트의 자원을 이용하여 원하는 작업을 수행한다. 그리고 이동 호스트와 계속적으로 통신하지 않기 때문에, 비록 이동 호스트의 전력이 꺼지거나 네트워크로부터 단절되더라도 계속해서 자신의 작업을 수행할 수 있다. 사용자가 다시 연결되었을 때, 그 에이전트는 작업한 결과를 가지고 이동 호스트로 돌아온다.

에이전트 모델은 그림 1과 같이 에이전트(Agent)와 플레이스(Place)로 구성된다. 에이전트 시스템은 에이전트에게 다양한 서비스를 제공하는 많은 플레이스들로 구성된다. 플레이스는 이동 에이전트를 받아들이고, 수행 환경을 지원하며, 호스트 자원을 활용하여 다양한 서비스를 제공한다. 에이전트는 크게 시스템 에이전트와 사용자 에이전트

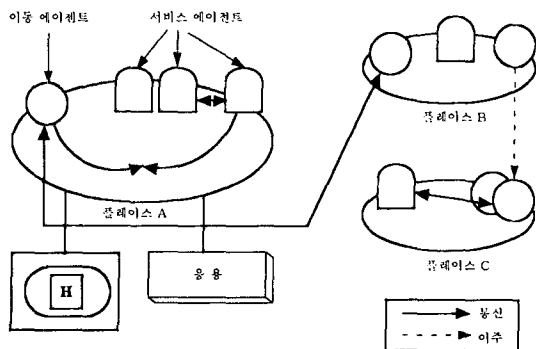


그림 1. 에이전트 모델

로 나누어진다. 시스템 에이전트는 특정 플레이스에 고정되어 있는 에이전트로 다른 에이전트와 통신하거나 자신의 플레이스로 이주해 오는 이동 에이전트에게 서비스를 제공한다. 사용자 에이전트는 플레이스가 제공하는 서비스를 요청하기 위하여 자신의 의지에 따라 플레이스간을 이주하는 이동 에이전트이다. 어떤 플레이스에서 다른 플레이스로의 여행(Travel)은 에이전트가 원격에서 제공되는 서비스를 받기 위하여 필요할 때 이동하고, 서비스를 받은 후 홈 플레이스로 돌아온다. 모임(Meeting)은 두 개 이상의 에이전트가 같은 플레이스에서 만나는 기능이다. 연결(Connection)은 서로 다른 플레이스에 있는 두 개의 에이전트가 통신할 수 있도록 선로를 열어준다. 연결은 주로 대화식 응용의 사용자가 자주 사용한다. 권한(Authorities)은 에이전트 또는 플레이스를 다른 것과 분별한다. 그리고 영역(Region)은 같은 권한에 의해 운영되는 플레이스들의 집합이다[8,9,11].

2.2 이동 에이전트 시스템

에이전트 시스템은 에이전트를 생성, 해석, 수행, 폐기 등의 생명 주기를 관리하는 플랫폼이다. 이동 에이전트 시스템은 에이전트 시스템에 추가적으로 에이전트의 이동성과 통신 기능, 허가 받지 않은 에이전트로부터의 접근 제한 등의 기능을 제공한다. 기본적인 이동 기능은 Jump 명령어를 사용한다. 이것은 하나의 플레이스에서 다른 플레이스로 에이전트를 이주시킨다. 이 때 에이전트의 내부 상태를 수집한다. 이것을 부호화하고, 목적지 플레이스 에이전트 서버에게 보낸다. 에이전트 서버는 에이전트를 받아들여 보안 검색을 한 다음, 인터프리터를 시작시킨다. 인터프리터는 상태 이미지를 복구하고 Jump 명령어 바로 다음에서부터 에이전트를 수행시킨다.

이동 에이전트 시스템의 주요 하부 기능은 첫째, 호스트 시스템의 자원을 안전하고 효율적으로 사용하기 위하여 호스트와 상호 동작을 해야 하고, 둘째, 사용자의 요구에 따라 서비스를 제공하기 위하여 다른 에이전트와 협동 작업을 할 수 있어야 하고, 셋째, 이질적인 망을 옮겨 다닐 수 있어야 하고, 넷째, 에이전트의 생명 주기를 관리할 수 있어야 한다[12].

2.3 네트워크 관리 시스템

네트워크 모니터링과 제어를 위한 통합 도구인 네트워크 관리 시스템은 다음과 같은 요소들로 이루어져 있다[10].

- 관리자 또는 관리 스테이션: 인간인 네트워크 관리자가 네트워크 관리 시스템의 한 요소로서 작업할 수 있는 인터페이스 역할을 수행한다. 관리자는 데이터 분석, 오류 복구 등을 위한 네트워크 관리 응용들의 집합, 네트워크를 모니터링하고 제어하는데 사용하는 인터페이스, 네트워크 관리자의 요구를 네트워크의 원격요소들에 대한 실제 모니터링과 제어로 변환하는 능력, 그리고 관리되는 모든 엔티티의 데이터베이스로부터 추출된 네트워크 관리 정보 데이터베이스를 가지고 있다.

- 에이전트: 네트워크 관리 시스템의 능동 요소로 호스트, 브리지, 라우터, 허브와 같은 핵심 플랫폼들은 그들이 관리자로부터 관리될 수 있도록 에이전트 소프트웨어를 장착하고 있다. 에이전트는 관리자로부터의 정보 요구에 응답하고, 관리자로부터의 동작 요구에 대해 응답하고, 관리 시스템에 중요하고 자발적인 정보를 비동기적으로 제공한다.

- 관리 정보 베이스: 네트워크에서 자원을 관리하는 수단은 이 자원을 객체로 표현하는 것이다. 각 객체들은 관리되는 에이전트의 한 단면을

표현하는 데이터 변수이다. 객체들의 집합을 관리 정보 베이스(Management Information Base, MIB)라 한다. MIB는 관리자를 위한 에이전트의 접근점 집합으로써 기능을 한다.

- 네트워크 관리 프로토콜: 관리자와 에이전트는 네트워크 관리 프로토콜에 의해 링크된다. 네트워크 관리 프로토콜은 다음 핵심 기능을 포함한다.

- ① Get: 관리자가 에이전트의 객체들의 값을 검색할 수 있도록 한다.

- ② Set: 관리자가 에이전트의 객체들의 값을 설정할 수 있도록 한다.

- ③ Notify: 에이전트가 중대한 사건을 관리자에게 알려줄 수 있도록 한다.

2.4 네트워크 관리 프로토콜

TCP/IP의 네트워크 관리 프로토콜인 SNMP는 관리자가 호스트, 게이트웨이, 단말기 서버, 허브와 같은 네트워크 구성 요소들을 감시하고 통제하는 관리 응용을 수행하는 것으로 구현이 쉽고 간단하여 네트워크 관리 프로토콜로서 널리 사용되고 있다. 관리 대상자에는 관리자로부터 요청된 네트워크 관리 기능을 수행하는데 책임이 있는 관리 에이전트가 있으며, SNMP는 관리자와 관리 대상자에 있는 에이전트간의 관리 정보를 교환하고 변경하는데 사용된다. 관리자는 그림 2와 같이 관리 대상자에 있는 에이전트에게 폴링 방식으로 메시지를 전송하여 응답을 받는다. 메시지에는 관리 정보를 읽기 위한 GetRequest, 다음 관리 정보를 읽기 위한 GetNextRequest, 관리 정보를 변경하기 위한 SetRequest, 관리자의 요구에 따라 관리 대상자가 응답을 주기 위한 GetResponse, 관리 대상자에 특정 상황이 발생했음을 관리자에게 알리기 위한 Trap이 있다(그림 3). 네트워크

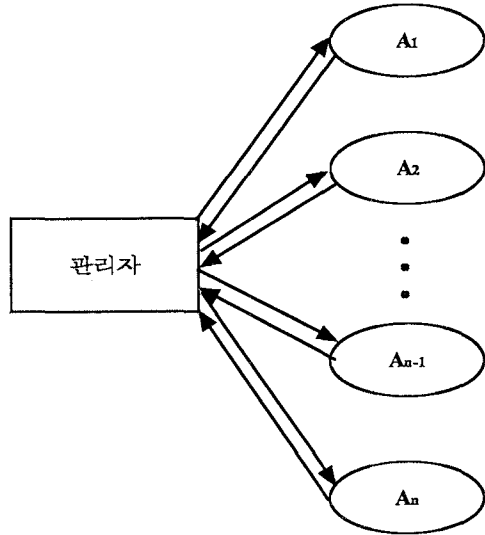


그림 2. SNMP에서의 폴링

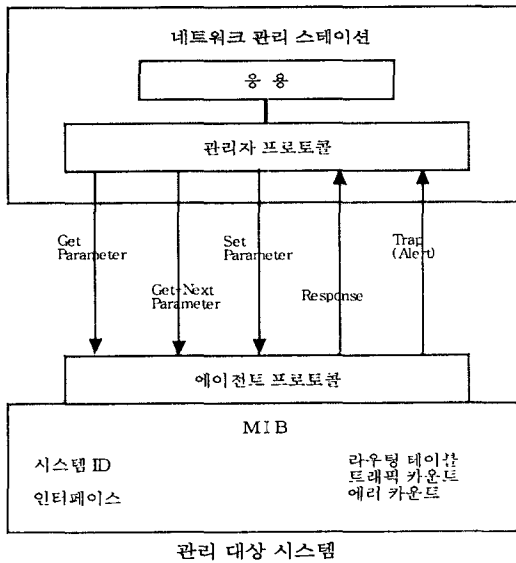


그림 3. 관리자와 에이전트간에 전송되는 메시지들

규모가 커질수록 관리자의 처리 부하가 집중되며 네트워크 트래픽이 증가하게 된다[1-3].

Ku는 중앙집중형 관리의 문제점을 보완하기 위해 이동 에이전트를 이용하는 네트워크 관리 모델을 제안했다[6]. 이동 에이전트는 사용자가 명

시한 액션을 수행할 수 있도록 관리 대상자에서 원격으로 실행되는 코드와 데이터를 포함하고 있으며, 관리자에 의해 관리 대상자에게 전송된다. 관리 대상자에 전송된 이동 에이전트는 관리 대상자의 에이전트 호스트와 상호 작용을 하여 관리 자원을 수집하며, 명시된 모든 작업을 완수한 후에는 그 결과를 자신의 컨테이너 안에 저장한다. 관리자는 그림 4와 같이 이동 에이전트를 관리 대상자에게 전송하며 이동 에이전트는 정해진 순회 계획에 따라 관리 대상자들로 이동하여 작업을 수행 후 마지막 관리 대상자의 방문을 마치면 관리자에게로 복귀한다. 관리 대상자는 이동 에이전트를 수신하고 인증하며 다음 관리 대상으로 전송하며, 이동 에이전트가 로컬 자원을 액세스 할 수 있는 메커니즘을 제공하는 에이전트 호스트를 가지고 있다.

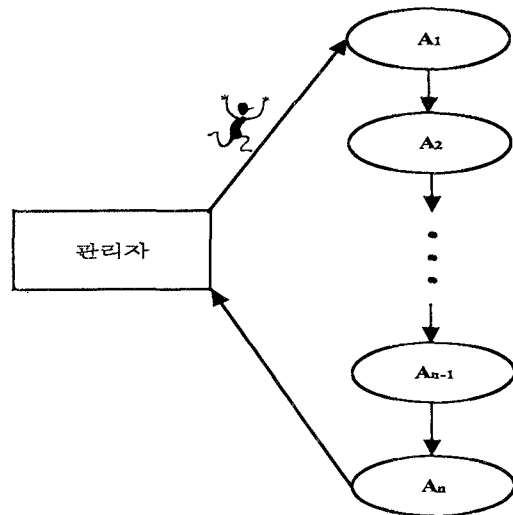


그림 4. Ku의 이동 에이전트를 이용한 네트워크 관리 모델

3. 에이전트 기반 실시간 네트워크 관리 모델

Ku는 이동 에이전트를 관리 대상자에게 전송

시킴으로써 SNMP에서의 관리자에게 집중된 처리 부하와 트래픽 증가로 인한 문제점을 개선하고자 했다. 그러나 Ku에서 측정된 성능 척도인 응답 시간에서는 이동 에이전트가 관리 대상자에서 다른 관리 대상자로 이동할 때마다 누적되는 처리 결과는 고려하지 않고 있다. 네트워크 규모가 커질수록 관리 대상자의 수가 증가하므로 이동 에이전트가 순회 계획대로 다음 관리 대상자로 이동해감에 따라 처리 결과의 누적으로 인해 두 관리 대상자간의 이동 에이전트의 이동시간은 점점 더 증가되므로 이를 고려할 때 큰 규모의 네트워크에서는 오히려 Ku의 모델이 SNMP보다 응답시간이 더 커질 수도 있다.

본 고에서는 관리 대상자들을 클러스터링하고 클러스터별로 이동 에이전트를 생성하여 병렬로 전송하여 응답을 받음으로써 SNMP의 트래픽 부하와 관리자의 처리 부하를 개선시킬 뿐만 아니라 Ku의 전체 관리 대상자의 순차적 순회와 처리 결과 누적으로 인한 응답 시간의 증가 문제를 개선하기 위한 이동 에이전트를 기반으로 하는 실시간 네트워크 관리 모델을 제시한다. 제안한 모델은 이동 에이전트의 홈인 관리자, 이동 에이전트 그리고 에이전트 호스트로 구성된다. 관리자는 사용자로 하여금 이동 에이전트를 위한 정책을 명시하고 이동 에이전트를 디스패치하기 위한 인터페이스를 제공하며, 사용자가 제공한 정보에 기인하여 이동 에이전트를 생성할 수 있는 능력을 가지고 있다. 관리자는 이동 에이전트 안에 순회 계획, 액션 계획, 관리 정보의 보안과 같은 이동 에이전트의 특징들을 명시하여 이동 에이전트를 출발시키고, 이동 에이전트가 관리 대상자들을 순회하면서 수집한 관리 정보를 가지고 복귀하면, 이 관리 정보를 처리하여 사용자에게 제시한다.

이동 에이전트는 관리자가 미리 명시한 관리 업무를 관리 대상자의 에이전트 호스트에서 수행

하기 위한 행위를 명시한 코드, 새로운 에이전트 호스트로 이동한 후 이동 에이전트가 활성화되는데 필요한 상태 정보, 그리고 이동 에이전트의 개시자, 이동 히스토리, 인증 키와 같은 이동 에이전트의 정보가 포함된 속성들을 포함하고 있다. 에이전트 호스트는 이동 에이전트를 수신하고 송신하는데 책임을 지고 있으며 에이전트 호스트상에서 이동 에이전트가 활성화되면 로컬 자원들과 상호 작용하여 관리 정보를 수집한다. Ku의 모델에서와 같이 이동 에이전트는 지능을 포함하여 초기의 관리자에서 결정된 순회 계획을 관리 대상자들의 상황에 따라 순회 경로의 최적화, 다음 목적지 발견 그리고 비정상적인 상황의 발견 등에 사용될 수 있다.

그림 5와 같이 관리자는 정해진 응답 마감 시간을 만족시킬 수 있도록 관리 대상자들을 클러스터링하여, 각 클러스터 별로 이동 에이전트를 생성

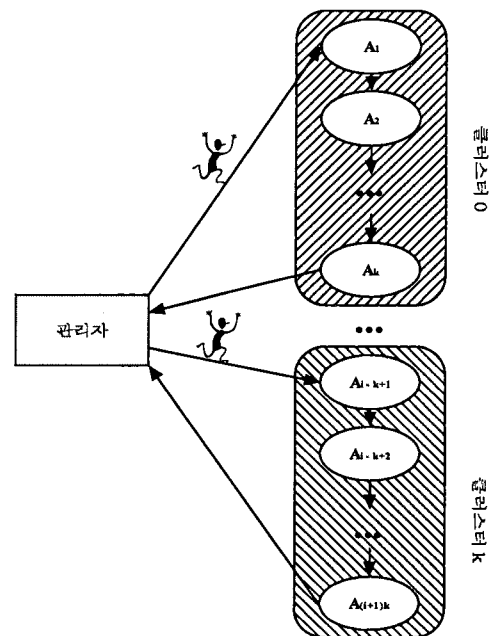


그림 5. 이동 에이전트 기반 실시간 네트워크 관리 모델

하여 전송한다. 각 클러스터에 전송된 이동 에이전트는 클러스터 내 관리 대상자들을 순회 계획에 따라 순차적으로 방문하며 관리 작업을 수행한다. 클러스터 별로 한 개의 이동 에이전트를 생성, 전송하여 그룹별로 동시에 해당 에이전트에 의해 원격으로 관리 작업이 수행됨으로써 SNMP에서의 폴링으로 인한 관리자로의 처리 부하와 메시지 트래픽 부하를 감소시킬 뿐만 아니라 응답 시간도 감소된다. 클러스터 내의 관리 대상자의 개수는 시스템에서 정해진 응답 마감 시간을 만족시킬 수 있도록 적절히 조절할 수 있으므로 실시간 네트워크 관리 환경에 적합하다.

4. 성능 평가

네트워크 관리 모델의 성능 평가의 한가지 척도로 응답 시간을 들 수 있다. 응답 시간은 관리자가 관리 작업을 시작한 시점부터 관리자에게 관리 정보 결과가 반환되는 시점까지의 시간이다. [7]에 따른 성능 척도 모델을 사용하여 응답 시간을 측정하여 제안한 모델과 SNMP 그리고 Ku의 모델을 비교한다. 이동 에이전트의 크기 B_{Agent} 는 다음과 같다.

$$B_{Agent} = B_{Code} + B_{Data} + B_{State}$$

여기서 B_{Code} 는 이동 에이전트의 코드 크기, B_{Data} 는 이동 에이전트의 데이터 크기, B_{State} 는 이동 에이전트의 상태 정보의 크기이다. 임의의 위치에 있는 두 노드 A_1, A_2 간에 이동 에이전트가 이동할 때 네트워크 부하 B_{Mobile} 과 이동 시간 T_{Mobile} 은 다음과 같다.

$$B_{Mobile}(A_1, A_2, B_{Agent}) = \begin{cases} 0, & \text{if } A_1 = A_2 \\ B_{Agent}, & \text{else} \end{cases}$$

$$+ \frac{T_{Mobile}(A_1, A_2, B_{Agent})}{\tau(A_1, A_2)} + \begin{cases} 0, & \text{if } A_1 = A_2 \\ 2\mu B_{Agent}, & \text{else} \end{cases}$$

여기서 δ 는 두 노드간의 네트워크 지연 시간, τ 는 두 노드간의 네트워크 처리율, 그리고 μ 는 마살링 처리력이다. 이동 에이전트가 다음 노드로 이동할 때 이전 노드에서 구한 관리 작업 결과의 평균 크기를 B_{Result} 라 하자. 작업 결과를 다음 노드로 이동하는데 걸리는 평균 전송 시간 T_{Result} 는 다음과 같다.

$$T_{Result} = \frac{B_{Result}}{\tau}$$

k 개의 관리 대상으로 구성된 한 클러스터에 대해 이동 에이전트를 생성하여 전송하고 클러스터의 마지막 관리 대상자로부터 이동 에이전트가 누적된 처리 결과를 가지고 관리자에게 복귀하는데 걸리는 시간은 다음과 같다. 여기서 k 를 클러스터 크기라 한다.

$$T_{total} = t_m + \sum_{i=0}^k (T_{Mobile} + i T_{Result}) + k T_{Comp}$$

여기서 t_m 은 이동 에이전트 생성 시간이며, T_{Comp} 는 이동 에이전트의 계산 시간이다. 주어진 이동 에이전트의 응답 마감 시간을 D 라 할 때 D 를 만족시킬 수 있는 클러스터 크기 k 는 다음과 같은 식으로 구할 수 있다.

$$T_{total} \leq D$$

$$t_m + \sum_{i=0}^k (T_{Mobile} + i T_{Result}) + k T_{Comp} \leq D$$

위의 부등식을 다음과 같이 k 에 대해서 전개할 수 있다.

$$t_m + (k+1) T_{Mobile} + \frac{k(k+1)}{2} T_{Result} + k T_{Comp} \leq D$$

$$T_{Result} k^2 + (2 T_{Mobile} + T_{Result} + 2 T_{Comp})k + (2 t_m + 2 T_{Mobile} - 2 D) \leq 0$$

위의 식은 k 에 대한 2차 방정식이며 $k > 0$ 이므로 k 는 다음과 같이 구해진다.

$$k = \frac{-(2 T_{Mobile} + T_{Result} + 2 T_{Comp})}{2 T_{Result}} + \frac{\sqrt{(2 T_{Mobile} + T_{Result} + 2 T_{Comp})^2 - 4 T_{Result} (2 t_m + 2 T_{Mobile} - 2 D)}}{2 T_{Result}}$$

그림 6은 노드간의 네트워크 지연이 동일하고, n 이 k 의 배수인 것으로 가정할 때, $t_m=150ms$, $B_{Result}=0.5KB$, $\mu=0.0885 ms/KB$, $\tau=400KB/sec$, $\delta=15ms$, $B_{Code}=10KB$, $B_{Data}=1KB$, $B_{State}=1KB$ 인 경우 D 에 대한 클러스터 크기 k 를 보여준다.

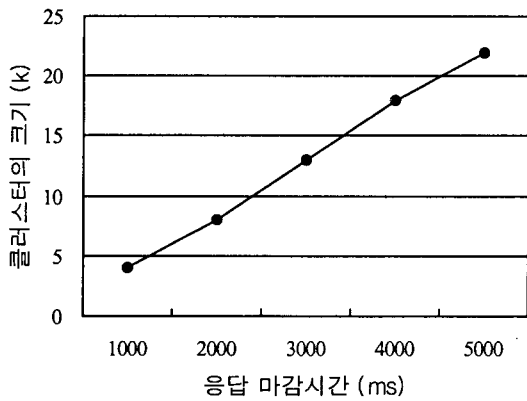


그림 6. 응답 마감시간에 따른 클러스터의 크기

그림 7은 $k=10$ 일 때의 제안한 모델을 Ku 의 모델과 SNMP에 대해 응답 시간 측면에서 비교한 결과를 보여준다. 제안한 모델은 Ku 와 SNMP보다 응답 시간이 작음을 알 수 있다. Ku 의 모델의 경우는 모든 노드를 순회한 후에야 관리 정보를 얻게 되나 제안한 모델에서는 클러스터 내 마지막

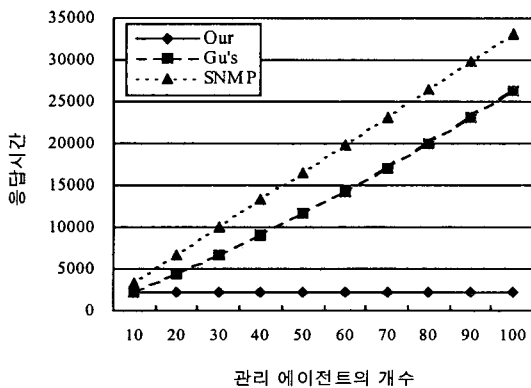


그림 7. 관리 에이전트 개수에 따른 평균 응답 시간

관리 대상자를 방문한 후에는 관리자로서 처리 결과가 전송되며 클러스터별로 동시에 수행되므로 응답 시간이 작아지고, 관리자가 원하는 응답 마감 시간에 따라 k 를 결정할 수 있음으로써 실시간 네트워크 관리에 적합하다.

5. 결론

네트워크 관리를 위한 SNMP에서는 폴링으로 인한 관리자로서의 처리 부하 집중, 네트워크 트래픽 증가 등의 문제가 있으며, Ku 의 이동 에이전트를 이용한 모델에서는 새로운 관리 대상자로 이동할 때마다 관리 작업 처리 결과의 누적과 전체 관리 대상의 순차적 순회 후 관리자로서 이동 에이전트가 복귀함으로써 응답 시간이 증가되는 문제가 있었다.

본 고에서는 이를 개선하기 위해 관리 대상자들을 클러스터링하고, 클러스터별로 이동 에이전트를 생성하고 순회시켜, 처리 결과가 관리자에게로 반환되는 이동 에이전트 기반 실시간 네트워크 관리 모델을 제시하였다. 응답 시간이 SNMP와 Ku 의 모델보다도 우수하여 실시간 처리에 보다 적합함을 알 수 있었다. 응답 마감 시간이 주어졌을 때 이를 만족시킬 수 있는 클러스터 크기는 이미 알려진 관련 매개변수들로부터 쉽게 구할 수 있었다.

향후 연구 과제로 제안한 모델의 구현을 통한 실제적 검증, 계층적 클러스터링을 통한 이동 에이전트의 병렬성 증가, 그리고 관리자의 트래픽 부하를 고려한 클러스터 개수 결정 등이다.

참고 문헌

[1] J. Case, K. McCloghrie, M. Rose and S. Waldbusser, "Introduction to version 2 of the

Internet-standard Network Management Framework," Internet RFC 1441, April 1998.

[2] J. Case, M. Fedor, M. Schoffstall and J. Davin, "A Simple Network Management Protocol," Internet RFC 1067, August 1988.

[3] William Stallings, *Network Management*, Los Alamitos, 1993.

[4] Divakara K. Udupa, *Network Management System Essentials*, McGraw-Hill, 1996.

[5] H. Ku et al., "Network Management Agents Supported By Java Environment," IFIP/IEEE International Symposium on Integrated Network Management (INM'97), San Diego CA, May 1997.

[6] H. Ku et al., "An Intelligent Mobile Agent Framework for Distributed Network Management," Globecom'97 Phoenix AZ, pp.160-164, Nov. 1997.

[7] Markus Straßer and Markus Schwehm, "A Performance Model for Mobile Agent Systems," Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications PDPTA'97, 1997.

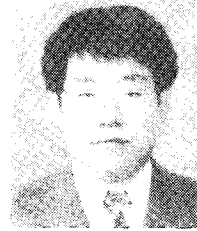
[8] W. R. Cockayne, M. Zyda, *Mobile Agents*, Manning Pub. Co., 1998.

[9] R. Gray, D. Kotz, S. Nog, D. Rus, G. Cybenko, "Mobile Agents for Mobile Computing," Technical Report PCS-TR96-285, Dept. of Computer Science, Dartmouth College, May, 1996.

[10] W. Stallings, *Data and Computer Communications*, Macmillian Pub. Co., 1994.

[11] 이재경, 김운정, 배인한, "근사 최적 이동 에이전트 이주 정책을 찾기 위한 클러스터 기반 방법," 한국멀티미디어학회 추계학술발표논문집, pp. 353-358, 1998.

[12] 김평준, 윤석환, "이동 컴퓨팅을 위한 이동 에이전트 시스템," 정보처리학회지, 4권 5호, pp.67-75, 1998.



배 인 한

- 1984년 경남대학교 전자계산학과 (학사)
- 1986년 중앙대학교 대학원 전자계산학과 (석사)
- 1990년 중앙대학교 대학원 전자계산학과 (박사)
- 1996년~1997년 Dept. of Computer and Information Science, The Ohio State University (Postdoc)
- 1989년~현재 대구효성가톨릭대학교 전자정보공학부 부교수
- 관심분야 : 운영체제, 분산 시스템, 멀티미디어 시스템, 이동 컴퓨팅 시스템, 이동 에이전트



하 속 정

- 1988년 계명대학교 전자계산학과 졸업 (학사)
- 1990년 중앙대학교 대학원 전자계산학과 졸업 (석사)
- 1998년 대구효성가톨릭대학교 대학원 전산통계학과 (박사)
- 관심분야 : 분산 시스템, 멀티미디어 시스템, 이동 컴퓨팅 시스템, 이동 에이전트