

실시간 유비쿼터스 지능공간 모니터링 시스템을 위한 에이전트와 스마트 객체 간의 부하 분산 기법

(Load Balancing Scheme between Agents and Smart Objects for Real-Time Monitoring System of Ubiquitous Smart Space)

정 흥 규 [†] 이 동 욱 ^{**}
(Hong-kyu Chung) (Dong-Wook Lee)

김 재 훈 ^{***}
(Jai-Hoon Kim)

요약 유비쿼터스 지능공간 모니터링 시스템은 유비쿼터스 지능공간 내에서의 다양한 스마트 객체의 기능, 성능 및 상태를 모니터링하고 분석함으로써 사용자 지수, 서비스의 통합 동작 상황, 서비스의 상태를 판단할 수 있는 정보를 실시간으로 제공해준다. 또한 최적화 및 자기 관리를 통하여 서비스 성능을 향상시킬 수 있는 기능을 제공한다. 이러한 실시간 모니터링 시스템의 적용범위를 확장하기 위해

서는 유동적인 대량의 데이터 처리가 필요하다. 본 논문에서는 모니터링 시스템을 구성하는 스마트 객체의 데이터 발생에 따라 유발되는 에이전트들의 부하를 해결하기 위해 부하를 예측하여 미리 분산시키는 기법을 제안한다. 제안된 기법이 적용된 시스템은 실험을 통해 기존의 시스템에 비해 전체 데이터 마감시간 초과율이 매개변수가 1일 경우, 80%이상 감소됨을 보여준다.

키워드 : 유비쿼터스 시스템, 모니터링 시스템, 실시간 시스템

Abstract Monitoring and analyzing the status of smart objects, the ubiquitous smart monitoring system provides several information such as user-state index, service states and system operation among the services in real time. It also provides self-optimization and self-management for enhancing the performance of services. In order to expand the application scope of this real-time monitoring system, it is indispensable to process huge amount of stream data. In this paper, we propose a load balancing scheme to solve the overload of the monitoring agents. Our proposed scheme reduces deadline miss ratio of entire data by more than 80%.

Key words : Ubiquitous System, Monitoring System, Real-Time System

1. 서론

유비쿼터스 환경에는 사용자 요구를 만족시키기 위해 여러 종류의 스마트 객체들이 다양한 형태로 구성되어 있는 경우가 많다. 이러한 지능공간에서 서비스 기능 및 성능을 보장하고 최적화와 신뢰도를 높이기 위해서는 각각의 스마트 객체들이 정상적으로 동작하는지를 주기적으로 감시하고 관리하는 서비스가 실시간으로 제공되어야 한다. 이와 같은 서비스를 제공하기 위해 유비쿼터스 지능공간 내 다양한 스마트 객체들의 기능 및 상태를 실시간 모니터링 하는 기법들이 현재 연구 중에 있다[1,2].

그림 1은 모니터링 시스템의 전체 구조를 보여 준다.

- 본 연구는 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업에 의해 지원 받았음(No. 2009-0089958)
- 본 연구는 지식경제 프론티어 기술개발사업의 일환으로 추진되고 있는 지식경제부의 유비쿼터스컴퓨팅 및 네트워크원천기반기술개발사업에 의하여 수행됨
- 이 논문은 2009 한국컴퓨터종합학술대회에서 '실시간 유비쿼터스 지능공간 모니터링 시스템을 위한 에이전트와 스마트 객체 간의 부하 분산 기법'의 제목으로 발표된 논문을 확장한 것임

[†] 학생회원 : 아주대학교 컴퓨터공학과
mzaitn@ajou.ac.kr

^{**} 비회원 : 아주대학교 정보통신전문대학원
dwlee@ajou.ac.kr

^{***} 정회원 : 아주대학교 정보통신전문대학원 교수
jaikim@ajou.ac.kr

논문접수 : 2009년 8월 13일

심사완료 : 2010년 1월 10일

Copyright©2010 한국정보과학회 : 개인 목적이거나 교육 목적인 경우, 이 저작물의 전체 또는 일부에 대한 복사본 혹은 디지털 사본의 제작을 허가합니다. 이 때, 사본은 상업적 수단으로 사용할 수 없으며 첫 페이지에 본 문구와 출처를 반드시 명시해야 합니다. 이 외의 목적으로 복제, 배포, 출판, 전송 등 모든 유형의 사용행위를 하는 경우에 대하여는 사전에 허가를 얻고 비용을 지불해야 합니다.

정보과학회논문지: 컴퓨팅의 실제 및 레터 제16권 제4호(2010.4)

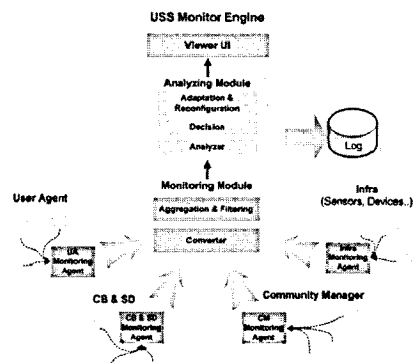


그림 1 모니터링 시스템 구조도[1]

모니터링 시스템은 유비쿼터스 지능공간 내의 배치되어 있는 에이전트들을 통해 스마트 객체(UA, CB, SD, CM, Infra 등)들의 상태 정보를 모니터링 모듈이 수집하고, 필터링하여, 분석 모듈로 전달하게 된다. 분석 모듈은 수집된 데이터를 가지고 스마트 객체의 상황을 분석하여 실시간으로 화면을 통해 사용자에게 보여준다. 이와 같이 모니터링 시스템은 실시간 시스템이며 데이터의 신뢰성을 보장하여야 한다.

현재 모니터링 환경에서는 데이터 발생량이 적기 때문에 데이터의 마감시간 초과가 발생되지 않는다. 하지만 모니터링 시스템을 실생활에 적용시키게 되면, 발생하는 데이터가 많아짐에 따라 부하가 발생하여 마감시간을 초과하는 데이터가 발생하게 된다. 이러한 문제점의 원인은 에이전트의 구조에서 볼 수 있다. 에이전트는 그림 2와 같이 작업 큐에 저장된 모니터링 데이터들을 일정한 간격에 따라 데이터를 처리하여, 모니터링 엔진으로 전송하는 구조로 되어 있다.

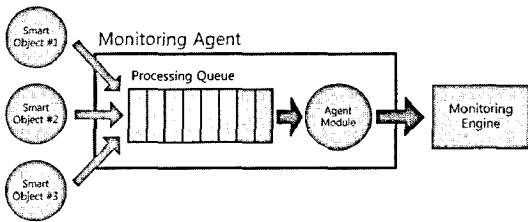


그림 2 에이전트의 데이터 처리

이런 구조의 에이전트는 시간 당 처리할 수 있는 데이터 양이 제한되어 있어 스마트 객체로부터 많은 양의 데이터를 전송 받으면 시간 당 들어오는 데이터 양이 데이터 처리량 보다 많아지게 되어 작업 큐에 쌓이는 데이터 양이 기하급수적으로 증가하는 문제점이 발생한다. 이에 따라 데이터의 실시간성을 보장 못해 실시간 시스템으로서의 역할을 상실하게 된다. 그러므로 본 논문에서는 이와 같은 문제점을 해결하기 위해서 작업 큐에 대기하는 데이터 량을 예측하여 필요 시 연결된 스마트 객체를 여유가 있는 다른 에이전트로 연결시키는 방법을 제안한다. 또한 작업 큐의 데이터 량을 완벽하게 예측하기에는 불가능하므로, 가변적인 매개 변수와 임계 범위를 두어 상황에 맞게 적용할 수 있도록 제안한다.

2. 관련 연구

2.1 부하 분산과 부하 이동의 관계

부하 분산은 모든 프로세서들의 작업 부하를 고르게 분산시키는데 중점을 둔다. 반면에 부하 이동은 발생된 부하를 다른 프로세서로 옮기는 방법에 중점을 둔다[3].

2.2 부하 분산 기법의 단계적 설계 방법

부하 분산 기법을 설계하기 위해서는 5단계의 과정을 따라 알고리즘 설계를 해야 한다[4]. 본 논문의 부하 분산 알고리즘은 다음과 같은 5단계에 따라 설계되었다.

1단계는 각 프로세서로부터 부하 량에 대한 정보를 수집하는 단계이다. 2단계는 어떤 프로세서가 부하 분산을 시작할 것인가를 결정하는 단계이다. 3단계는 부하 정보를 바탕으로 부하를 재배치할 시점을 결정하는 단계이다. 본 논문에서는 데이터 량의 예측을 통해 분산 시점을 결정한다. 4단계는 부하 분산 시 재배치할 작업을 결정하는 단계이다. 5단계는 작업을 재배치 할 때 재배치되는 작업을 처리할 프로세서를 선택하는 단계이다.

2.3 부하 분산 기법 종류

부하 분산 기법은 중앙 집중형 부하 분산 기법과 분산형 부하 분산 기법 등이 있다[5]. 중앙 집중형 부하 분산 기법은 시스템 중앙에 있는 조정자가 모든 프로세서의 상태 정보를 통해 작업을 분배하는 기법이다. 이 기법은 최적의 부하 분산을 할 수 있으나, 조정자의 결함 발생 시 시스템 전체의 영향을 미치게 된다. 분산형 부하 분산 기법은 중앙 집중형 부하 분산 기법의 단점을 줄일 수 있지만, 모든 프로세서가 부하 분산을 처리함에 따라 오히려 성능 저하를 일으킬 수 있다[6].

본 논문에서 제시하는 알고리즘은 에이전트의 부하를 줄이는 데 목적이 있다. 또한 모니터링 모듈은 실시간으로 에이전트의 상태 정보를 전송 받는다. 그러므로 본 논문에서는 집중형 부하 분산 기법에 따라 모니터링 모듈 속에 부하 분산 모듈을 두어 각 에이전트의 정보를 수집하고, 부하를 분산시킨다. 또한 본 논문에서의 부하 분산 알고리즘은 데이터의 실시간성과 에이전트의 구조와 같은 모니터링 시스템만의 특징을 고려하여 설계되었다.

3. 본 론

3.1 기본 원리 및 시스템 구성도

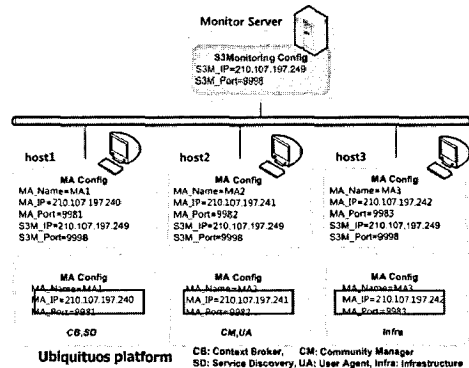


그림 3 에이전트와 스마트 객체의 연결 설정[1]

모니터링 시스템에서 스마트 객체와 에이전트는 데이터 전송은 환경설정 형태로 저장된 상위 모듈의 위치 주소에 따라 이루어진다. 그림 3는 각 스마트 객체 내부의 미리 정의된 환경설정에 따라 에이전트와의 연결을 보여준다. 그림 3과 같이 스마트 객체인 CB와 SD는 Host1 에이전트로, CM과 UA는 Host2 에이전트로, Infra는 Host3 에이전트로 자신의 모니터링 정보를 전송한다. 즉, 각각의 스마트 객체는 환경설정에 설정된 에이전트의 네트워크 IP주소로 데이터를 전송하게 된다.

이러한 모니터링 시스템의 특징 때문에 그림 4와 같이 부하 분산 모듈을 통해 간단히 스마트 객체의 환경 설정만을 바꿔 줌으로써 부하를 이동시킬 수 있다.

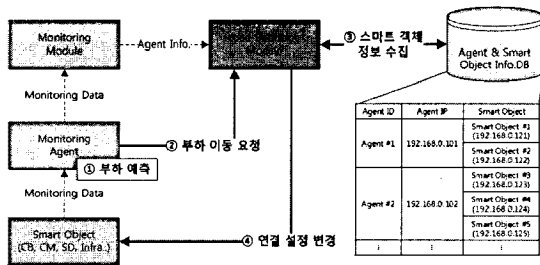


그림 4 시스템 구성도

그림 4는 부하 분산을 위한 시스템 구성을 보여준다. 부하 분산 모듈은 모니터링 모듈과 같은 레벨에 존재함으로써 에이전트의 상태를 모니터링 모듈을 통해 접근한다. 또한 부하 분산 모듈은 별도의 데이터베이스를 통해 에이전트와 스마트 객체의 연결 상태를 관리한다.

3.2 적응 알고리즘

부하 분산을 위해 작업 큐의 데이터 량을 예측하여 식 (1)과 같은 부하 조건을 만족하면 해당 에이전트에 연결된 스마트 객체 중 적절한 스마트 객체를 다른 에이전트로 연결시켜준다.

$$P_{i,t} > D_{i,max} \times DL \text{ or } Q_{i,t} > D_{i,max} \times DL \quad (1)$$

$P_{i,t}$: 시간 t 일 때 i 번째 에이전트의 다음 작업 큐의 예측 데이터 량

$Q_{i,t}$: 시간 t 일 때 i 번째 에이전트의 현재 작업 큐의 데이터 량

$D_{i,max}$: i 번째 에이전트의 시간 당 최대 데이터 처리 량
 DL : 데이터의 마감시간

만약 다음 작업 큐의 예측 데이터 량($P_{i,t}$)과 현재 작업 큐의 대기하는 데이터 량($Q_{i,t}$) 중 하나라도 임계값을 초과하면 부하 분산을 시작한다. 여기서 에이전트의 시간 당 최대 데이터 처리 량($D_{i,max}$)과 마감시간(DL)의 곱으로 임계값을 구할 수 있다.

예를 들어 1초 동안 에이전트의 최대 데이터 처리 량이 50개이고, 데이터의 마감시간이 2초라면 임계값은 100개가 된다. 즉, 작업 큐의 데이터가 100개 이상 쌓이면, 이후 들어오는 데이터는 모두 마감시간을 초과한다.

식 (1)은 다음 작업 큐의 예측 데이터 량($P_{i,t}$)과 현재 작업 큐의 데이터 량($Q_{i,t}$)을 이용한다. 먼저 현재 작업 큐의 데이터 량($Q_{i,t}$)은 식 (2)와 같이 계산한다.

$$Q_{i,t} = \max(Q_{i,t-1} + \sum S_{i,t} - D_{i,max}, 0) \quad (2)$$

$Q_{i,t}$: 시간 t 일 때 i 번째 에이전트의 현재 작업 큐의 데이터 량

$S_{i,t}$: 시간 t 일 때 i 번째 스마트 객체로부터 입력되는 데이터 량

$D_{i,max}$: i 번째 에이전트의 시간 당 최대 데이터 처리 량
 현재 작업 큐에 대기하고 있는 데이터 량($Q_{i,t}$)은 식 (2)와 같이 이전 작업 큐에 대기하고 있는 데이터 량($Q_{i,t-1}$)과 현재 스마트 객체로부터 입력되는 총 데이터 량($\sum S_{i,t}$)의 합에서 현재 처리되는 데이터 량($D_{i,max}$)의 차로 나타낼 수 있다. 이 때 작업 큐에 대기할 수 있는 데이터 량은 음수가 될 수 없으므로 최소값이 0이 된다.

다음 작업 큐의 예측 데이터 량($P_{i,t}$)을 계산을 위해 먼저 작업 큐에 대기하는 데이터의 변화 량($\Delta Q_{i,t}$)과 작업 큐의 데이터 변화가속 량($A_{i,t}$)을 다음과 같이 계산한다.

$$\Delta Q_{i,t} = Q_{i,t} - Q_{i,t-1} \quad (3)$$

$$A_{i,t} = \Delta Q_{i,t} - \Delta Q_{i,t-1} \quad (4)$$

$$P_{i,t} = Q_{i,t} + (\Delta Q_{i,t} + A_{i,t}) \times \alpha \quad (5)$$

$\Delta Q_{i,t}$: 시간 t 일 때 i 번째 에이전트의 작업 큐의 데이터 변화 량

$A_{i,t}$: 시간 t 일 때 i 번째 에이전트의 작업 큐의 데이터 변화가속 량

$P_{i,t}$: 시간 t 일 때 i 번째 에이전트의 다음 작업 큐의 예측 데이터 량

α : 매개 변수

작업 큐의 데이터 변화 량($\Delta Q_{i,t}$)은 식 (3)과 같이 현재 작업 큐의 데이터 량($Q_{i,t}$)과 이전 작업 큐의 데이터 량($Q_{i,t-1}$)의 차로 계산한다. 또한 작업 큐의 데이터의 변화가속 량($A_{i,t}$)은 식 (4)와 같이 시간 별 작업 큐의 데이터 량에 따라 계산된 현재 작업 큐의 데이터 변화 량($\Delta Q_{i,t}$)과 이전 작업 큐의 데이터 변화 량($\Delta Q_{i,t-1}$)의 차로 계산한다.

다음 작업 큐의 예측 데이터 량($P_{i,t}$)은 식 (5)에 따라 현재 작업 큐의 데이터 량($Q_{i,t}$)과 현재 작업 큐의 데이터 변화 량 ($\Delta Q_{i,t}$), 현재 데이터 변화가속 량($A_{i,t}$)들을 통해 계산할 수 있다. 이렇게 계산된 예측 데이터 량을 임계 범위와 비교하여 초과 시 부하 분산 모듈을 통해 부하 이동 작업을 수행하게 된다. 이 때 수식의 복잡성을

줄이기 위해 하나의 매개 변수(α)만을 가변 인자로 두어 매개 변수에 따른 오차율과 데이터의 마감시간 초과율을 분석할 수 있게 하였다. 이와 같이 부하 분산 기법의 영향을 고려함으로써 모니터링 시스템의 상황에 맞게 매개 변수 α 값을 설정하여 예측 성능을 향상시킬 수 있다.

부하 분산은 분산 조건이 만족되었을 때 에이전트가 부하 분산 모듈로 부하 이동 요청으로부터 시작된다. 부하 이동 요청을 받은 부하 분산 모듈은 해당 에이전트의 연결된 스마트 객체들 중 데이터 발생량이 많은 스마트 객체를 선출하여 여유 있는 에이전트로 연결함으로써 부하 분산을 마친다. 이후 여러 번의 부하 분산으로 스마트 객체들이 섞이는 것을 방지하기 위해 부하 이동을 요청한 에이전트의 상태가 호전되면 이동했던 스마트 객체를 원상태로 복귀시킨다.

4. 성능 및 평가

먼저 제안된 예측 알고리즘의 정확도를 평가하기 위해 실제 데이터를 바탕으로 한 실험에서 다음과 같은 결과를 얻었다.

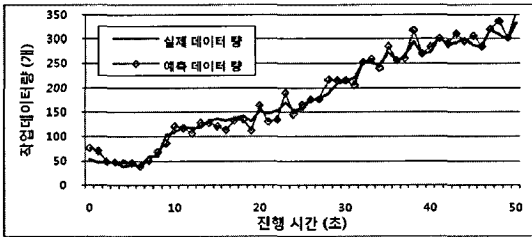


그림 5 실제 데이터 량과 예측 데이터 량의 관계

그림 5는 예측되는 데이터 량이 실제 데이터 량과 차이가 근소함을 보여준다.

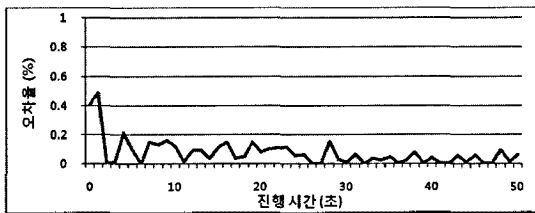


그림 6 실제 데이터 량과 예측 데이터 량의 오차율

그림 6과 같이 초기에는 작업 큐의 데이터 변화 량이 크기 때문에 오차율이 크지만 작업 큐의 쌓이는 데이터 량이 증가함에 따라 비례적으로 변화 량이 작아지므로 오차율이 줄게 된다. 제안된 기법은 과부하를 예측하는 알고리즘이므로 실제 과부하를 예측 시에는 작업 큐에

많은 데이터들이 쌓여 있게 된다. 그러므로 실제 과부하 예측 시에는 오차율이 작게 된다. 현재 모니터링 시스템은 아직 실험 단계 이므로 스마트 객체의 발생하는 데이터 량이 부하를 일으킬 정도로 크지 않다. 하지만 실제 적용 단계에서는 모니터링 요소인 스마트 객체의 수가 많아지기 때문에 데이터 량도 증가하게 될 것이다.

그러므로 실제 상황을 바탕으로 표 1과 같이 시뮬레이션 환경을 가정하였다.

표 1 시뮬레이션 환경

| 분류 | 내용 |
|-----------------|--|
| 전체 데이터 발생 분포 | 많은 데이터 발생 구간: 초당 0~1200개의 데이터가 균등분포로 발생 적은 데이터 발생 구간: 초당 0~1000개의 데이터가 균등분포로 발생 (10분 주기로 많은 데이터가 5분 동안 발생하고, 나머지 5분 동안은 적은 데이터가 발생한다.) |
| 스마트 객체 수 | 10개 |
| 시뮬레이션 작동 시간 | 1시간(3600초) |
| 에이전트의 데이터 처리 속도 | 500개/1초 (1초 동안 500개의 데이터를 처리한다.) |
| 데이터의 마감시간 | 2초 |
| 재배치 시간 | 0.5초 (부하 분산으로 인한 스마트 객체의 재배치 시 에이전트와의 연결이 0.5초 걸린다.) |
| 재분산 한계시간 | 0.5초 (부하 분산 후 0.5초 이내로 부하가 완화되지 않으면 재 부하 분산을 한다.) |
| 복귀 임계 값 | 70% (부하 분산 후 부하가 임계 값의 70%이내로 감소하면 재배치된 스마트 객체를 복귀 시킨다.) |

표 1과 같은 시뮬레이션 환경에 따라 시뮬레이션을 하면 다음과 같이 매개 변수 값에 따른 데이터 마감시간 초과 데이터 량을 볼 수 있다.

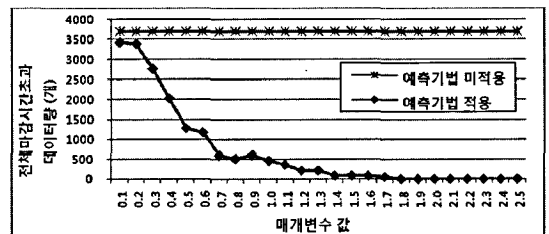


그림 7 매개변수 α 값에 따른 전체 마감시간 초과 데이터 량

그림 7은 부하 분산 기법을 미적용한 경우와 적용 시 매개변수에 따른 전체 마감시간 초과 데이터 량을 보여준다. 부하 분산 기법을 미적용 시 1,787,714개의 입력

데이터 중 3,696개의 데이터가 마감시간을 초과하였지만, 매개변수를 1.8로 설정하여 부하 분산 기법을 적용하면 마감시간을 초과하는 데이터가 전혀 발생하지 않았다.

그 이유는 예측 알고리즘으로 인해 마감시간을 초과하는 데이터가 발생하기 전 부하를 분산시키기 때문이다. 하지만 부하 분산 시 에이전트와 부하 분산 모듈간의 메시지 교환 등으로 인해 부하 분산 예측 시점과 실제 부하 분산 시점 사이의 지연 시간이 생긴다. 이 시간 동안 에이전트에서 마감시간을 초과하는 데이터가 발생하게 된다. 그래서 매개변수 α 값의 증가로 데이터 변화의 민감성을 증가시켜 과부하 예측을 부하 분산 처리 시간보다 먼저 할 수 있게 함으로서 마감시간 초과 데이터량을 줄일 수 있게 된다. 그래서 그림 7와 같이 매개변수 α 값이 증가할수록 마감시간 초과 데이터량이 감소됨을 볼 수 있다. 하지만 매개변수 α 값이 증가시키면 데이터량의 작은 변화에도 예측 데이터량이 크게 변하므로 다음과 같이 부하 분산 횟수가 증가하게 된다.

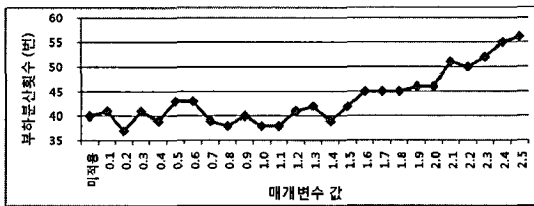


그림 8 매개변수 α 값에 따른 부하분산 횟수(비용)

그림 8은 시뮬레이션 환경에서 매개변수 α 값에 따라 부하 분산을 수행한 전체 횟수를 보여준다. 부하 분산을 수행하는 동안 메시지 교환 및 처리 등으로 인해 자원을 사용함으로 비용이 발생한다. 그러므로 부하 분산을 수행한 횟수가 많아질수록 비용도 비례적으로 많이 소요된다. 하지만 매개변수 α 값이 증가할수록 다음과 같이 부하 분산 성공률이 증가하게 된다.



그림 9 매개변수 α 값에 따른 부하 분산 예측 성공률

그림 9는 매개변수 α 값에 따른 부하 분산 시 예측 성공률을 보여준다. 전체적으로 매개변수 α 값이 증가하면 성공률이 증가하는 성향을 볼 수 있다. 그러므로 이러한

성향을 활용하여 실제 부하 분산 시 실패하는 경우, 매개변수 α 값을 증가시켜 성공률을 높일 수 있다. 즉, 상황에 따라 매개변수 α 값을 변화시켜 데이터의 마감시간 초과율을 효율적으로 줄일 수 있다.

5. 결론

현재 연구 중인 모니터링 시스템은 소규모 단위의 유비쿼터스 공간 내에서는 데이터가 많이 발생되지 않아 문제 없이 제 기능을 수행할 수 있다. 하지만 앞으로 아파트와 같은 대규모의 공간에서 시스템을 적용하면 많은 데이터 량에 따른 부하를 고려해야 한다. 그러므로 본 논문은 대규모의 공간에서 스마트 객체의 많은 데이터 발생에 따른 에이전트의 과부하를 예측하여 부하를 사전에 분산시킴으로써 에이전트의 효율을 높이고, 스마트 객체의 확장성을 고려한 실시간성을 유지하도록 제안하였다. 하지만 부하를 예측하는 알고리즘은 완전한 정확성과 신뢰성을 보장하지 못하므로 많은 연구를 통해 정확성과 신뢰성을 높여야 한다. 또한 부하 분산 시점을 결정 시 단순히 예측 방법만을 이용하여 결정하는 것보다는 다른 요소들을 병행하여 부하 분산 시점을 결정함으로써 보다 효율적인 시스템을 구현해야 한다.

참고 문헌

- [1] Sung-Hyun Kwon, Byoung-Hoon Lee and Jai-Hoon Kim, We-Duke Cho, "Real-Time Monitoring Agent for Ubiquitous System", *Journal of KIISE : Computing Practices and Letters*, vol.14, no.8, pp.803-837, 2008. (in Korean)
- [2] Hyo-Nam Lee, Sung-Hwa Lim, Byoung-Hoon Lee, Dong-Wook Lee, Jai-Hoon Kim and We-Duke Cho, "Monitoring System in Ubiquitous Smart Space," *2008 International Workshop on Context-Aware Pervasive Communities*, pp.261-264, 2008.
- [3] Linda F. Wilson and Wei Shen, "Experiments in load migration and dynamic load balancing in SPEEDES," *Simulation Conference Proceedings*, vol.1, pp.483-490, 1998.
- [4] Cho Cho Myint and Khin Mar Lar Tun, "A Framework of Using Mobile Agent to Achieve Efficient Load Balancing in Cluster," *APSITT 2005*, pp.66-70, 2005.
- [5] Chang-Min Kim, "Hierarchical Load Sharing Polices for Distributed Systems", *The Journal of Information Industry and Technology*, vol.4, pp.15-38, 1999. (in Korean)
- [6] Derek L. Eager, Edward D. Lazowska and John Zahorjan, "A comparison of receiver-initiated and sender-initiated adaptive load sharing," *Performance Evaluation*, vol.6, pp.53-68, 1986.