분산환경에서 멀티에이전트 상호협력을 통한 신뢰성 있는 정보검색기법

Reliable Information Search mechanism through the cooperation of MultiAgent in Distributed Environment

박 민 기* 김 귀 태** 이 재 완***
Min-Gi Park Gui-Tae Kim Jae-Wan Lee

요 약

인터넷이 널리 보급되면서 지능형 검색 에이전트들이 사용자의 요구를 만족시키기 위해 일반화되어 사용되고 있다. 그러나 이러한 지능형 멀티에이전트들은 서로 독립적으로 사용되어 멀티에이전트들 간의 분산된 정보를 원활하고 효율적으로 처리하기 위한 상호 협력 작용이 부족해 정보의 신뢰성이 낮고 동적으로 변화하는 분산 환경에 대처하기가 어렵다.

이런 문제를 해결하기 위해 본 논문에서는 멀티에이전트간의 효율적인 상호 협력과 빠른 정보처리를 위해 브로커 에이전트에 에이전시를 생성하고 신경망을 이용해 멀티에이전트들의 에이전시들을 분류하여 더욱 신속·정확한 정보를 사용자에게 제공하도록 한다. 또한 정보의 신뢰성을 위해서 에이전트 관리기법을 제안하여 기존의 검색 시스템이 가지고 있는 정보갱신문제를 향상시키고, 시뮬레이션을 통해 본 연구의 성능을 평가한다.

Abstract

As the internet is widely distributed, the intelligent search agent is commonly used to meet the needs of user. But these Intelligent multi-agents are so independent each other that they can not give reliability of information and also have difficulty in coping with the dynamic distributed environments due to the short of cooperation abilities among multiagent.

To resolve these problems, this paper proposes the mechanism for efficient cooperation and information processing by creating agency within broker agent and clustering multiagent's agency using neural network. For reliability of information, we also propose the multiagent management mechanism that can improve the information update problems which are in existing search systems and evaluate the performance of this research through simulation.

Keyword : Distributed System, Multi-agent, Neural Network, Information Retrieval

1. 서 론

1990년대 중반 이후 지식의 홍수 속에서 지능형 에이전트간의 협력을 통해 좀 더 합리적으로 정보에 접근하기 위한 연구의 필요성이 높아짐에 따라 에이전트에 관한 연구가 활발하게 진행되고 있다. 그러나 하나의 에이전트가 해결할 수 있는 문제의 양이나 범위에는 한계가 있어 여러 에이전트들이 제한된 시간 내에 한정된 자원을 적절히 활용하고 배분해야 한다는 문제가 발생하며 이러한 문제를 해결하기 위하여 멀티 에이전트 시스템이 제안되었다[1,2].

에이전트는 다른 에이전트와 상호협력을 통하여 사용자에게 서비스를 제공하고 멀티 에이전트 시스템은 어떠한 상황 속에서도 사용자에게 지속적이고 일관된 서비스 지원을 제공할 수 있어야 한다. 더 나아가 멀티 에이전트 시스템들 간의 상호협력을 통해 사용자의 요구에 만족하는 신뢰성 높은 정보제공과 빠른 정보처리 서비스를 제공해 주

* 준 회 원 : (주)버추얼다임 연구원
sopiru@kunsan.ac.kr(제 1저자)
** 정 회 원 : 군산대학교 전자정보공학부 박사과정
gtkim@kunsan.ac.kr(공동저자)
*** 종신회원 : 군산대학교 전자정보공학부 교수
jwlee@kunsan.ac.kr(공동저자)

어야 한다[3].

멀티에이전트 시스템은 에이전트들간의 협력과 상호 보완적 관계를 통해 분산 환경의 정보 공유 및 통합을 용이하게 한다. 분산협동 에이전트 시스템 개발의 표준화 방향으로 가장 활발하게 진행되는 에이전트 연구단체는 FIPA로서 논리적으로 구성된 분산 협동 에이전트들의 관리를 지원하고 있다[4-6].

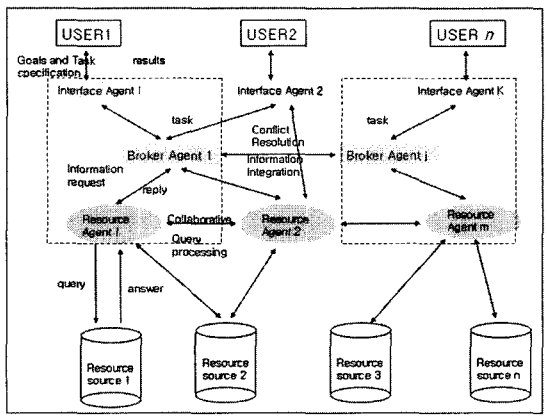
멀티에이전트 시스템에서 에이전트들 사이의 목표 충돌을 해결하기 위한 방안으로는 에이전트들이 서로 협상하여 문제를 해결하는 방법이 제안되었고 멀티 에이전트 시스템 상에서 에이전트 수행 종료에 의한 문제해결 방안도 제안되었다.[7,8]. 또한 유사도와 강화 학습을 사용하여 정보를 제공하는 에이전트와 정보를 요청하는 에이전트간의 연결을 매개하는 조정에이전트 구현방식이 제안되었다[9]. 이 방법은 정보를 요청하는 에이전트가 최대한 목적을 달성할 수 있도록 하는데 효과적이다. 이 방법은 정보에이전트가 자신의 처리 능력을 조정에이전트에 알려야하는 부담이 있으며 여러 에이전트가 동시에 등록할 경우에 생기는 부하문제를 해결할 수 없어 전체적인 시스템 마비를 불러일으킬 수 있고 정보에이전트의 정보가 갱신된 경우 연관된 에이전트들에게 갱신된 정보를 효율적으로 전달해줄 방법이 미흡하다.

따라서 본 연구는 이미 생성되어 운영중인 멀티에이전트들을 분산환경에서 단일 정보 시스템처럼 일괄되게 사용자에게 제공하기 위해 CORBA를 기반으로 하여 시스템을 설계하였고 멀티에이전트들간의 상호협력을 위해 사용자의 요구가 들어 왔을 때 멀티에이전트에 에이전시를 생성시키고 주변에 분산된 멀티에이전트와 통신하기 위해 패키지화한 메시지를 IIOP 통신프로토콜을 통해 다중 전송시킨다. 그리고 생성된 에이전시를 통해 가장 연관성이 높은 정보를 처리하는 멀티에이전트들을 클러스터링 시킨다. 클러스터링 기법은 기존의 코호넨의 자기구성 지도 모델인 SOM 신경망을 응용한 FSOM알고리즘을 통해 이루어지며 기존의 복잡한 학습과정에 따른 처리시간을 단축시켜 최종적으로 사용자에게 원하는 정보를 제공한다[10-13]. 또한 생성된 에이전시의 구성요소인 에이전시 팩토리에 참조된 리스트를 저장시킴으로써 캐쉬역할을 담당하도록 하여 동시에 많은 사용자로부터 유사정보 검색시 정보를 빨리 제공하도록 한다. 마지막으로 기존의 정보검색 시스템의 보완점인 정보갱신의 문제를 해결하고 동적인 네트워크 변화에 대응하기 위해 멀티에이전트 관리기법을 제안하여 정보의 신뢰성을 향상시킨다.

2. 정보검색에이전트 시스템

2.1 시스템 구조

이 연구에서는 정보검색에이전트(ISA) 시스템 구조를 제시하고 있으며 각각의 에이전트는 자율적이고 협력적으로 중재되고 지능적이며 합리적으로 다른 에이전트와 사용자의 욕구를 충족시키기 위해 서로 통신할 수 있다. 이 연구에서 제안한 시스템은 3단계 구조로 그림 1과 같이 구성된다.



〈그림 2〉 정보검색시스템 구조

(1) 인터페이스 에이전트

인터페이스 에이전트는 사용자에게 정보 시스템 환경과 상호 작용하게 하는 기관으로 볼 수 있다.

인터페이스 에이전트의 기능은 사용자들과 다른 인터페이스 에이전트들의 요청을 수행하기 위한 것이다. 인터페이스 에이전트는 관심 주제와 연관된 제약사항들을 포함하는 형태의 질의를 받아들인다. 사용자의 관심은 정보의 관련정도와 원하는 응답 시간을 포함한다.

(2) 자원 에이전트

자원에이전트는 정보 자원이나 정보 자원 표본을 직접 다루는 정보 제공자 역할을 수행한다. 자원 에이전트들은 관련된 정보를 검색하기 위해 검색 엔진을 이용한다. 동적인 인터넷 환경으로 인해 자원 에이전트는 정기적으로 미리 규정한 시간을 기반으로 한 정보 자원을 감시하며 군집(Cluster)을 찾기 위해 특수한 형태의 자기조직화 신경망(Self-Organizing Maps)을 사용하여 정보세분화를 지원한다.

(3) 브로커 에이전트

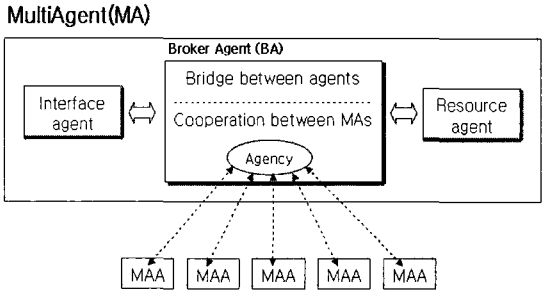
브로커 에이전트는 분산환경의 이질성을 해결하고 투명성을 높이기 위해 CORBA를 기반으로 하여 인터페이스 에이전트와 자원 에이전트들의 요청을 처리하는 중개자 역할을 한다. 또한 구현된 멀티에이전트들 간의 상호 협력을 통해 다양한 정보처리와 정보의 신뢰성을 높이기 위해 주변 멀티에이전트들 가운데 연관된 정보를 가진 멀티에이전트를 찾는 일이 중요하다. 이를 위해 브로커 에이전트 안에 에이전시를 생성하고 신경망을 이용한 데이터 마이닝을 수행하여 연관된 멀티에이전트들을 찾아내고 같은 호스트 내에 있는 멀티에이전트와 원격지에 있는 멀티에이전트들 간의 상호협력을 가능케 한다.

- 에이전시(Agency)

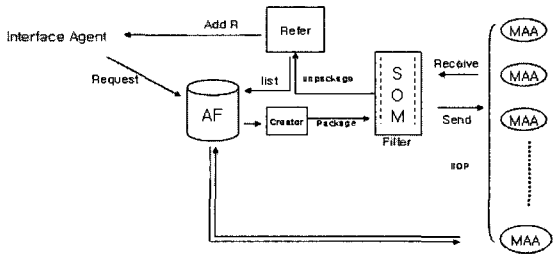
에이전시란 원격지 멀티에이전트와 상호협력을 위해 브로커에이전트에 생성되어 존재하는 기관으로써 에이전시 팩토리(Agency Factory), 에이전시 래퍼(Agency Refer), 에이전시 필터(Agency Filter)로 그림3과 같이 구성된다.

에이전시 팩토리는 새로운 요청이 들어올 때 하나의 에이전시가 멀티에이전트 쪽에 성공적으로 생성될 때까지 생성요청하며 에이전시 래퍼는 에이전시 필터로부터 가져온 정보(Resource_i)를 수신하고 참조된 에이전시의 리스트를 작성하여 참조 리스트를 생성하여 에이전시 펙토리에 저장시켜 관리 유지토록 한다. 에이전시 필터는 수신된 값에 따라 처리하는 분류기를 가지고 있으며 분산환경의 원격지 사용자로부터 요구 정보(Ui)가 수신되면 에이전시는 사용자가 요구하는 정보와 자신이 가지고 있는 정보를 비교하여 기준치 이상의 가중치 값을 처리하는 원격지 멀티에이전트들을 신경망을 이용해 클러스터링 한다.

에이전시의 메시지를 패키지화 및 언패키지화하여 수행하며 전송하고 수신한다. 또한 신경망을 응용해 사용자 정보요청에 대해 원격지 멀티에이



〈그림 2〉 브로커 에이전트 구조



〈그림 3〉 에이전시 구조

전트들을 클러스터링 하여 정보를 제공하는 기능과 원격지 멀티에이전트객체에 존재하는 특정한 자료나 정보를 찾았을 경우, 다음 사용자의 같은 정보를 클라이언트 멀티에이전트(CMA)공간에서 빠르게 접근하기 위해 그 정보를 에이전시 팩토리에 캐쉬(cache) 한다.

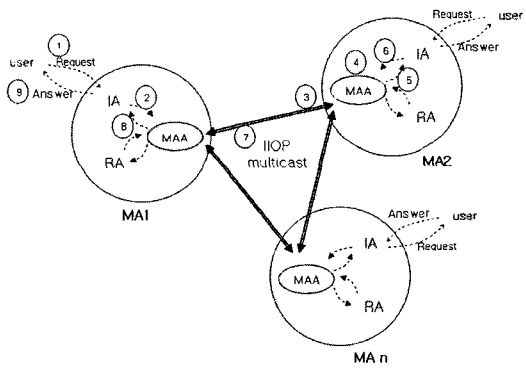
3. 멀티에이전트 상호협력

3.1 상호협력 시나리오

먼저 사용자로부터 정보요청이 들어오면 이 정보를 처리하는 에이전시들이 이미 존재하는지 검사하기 위해 에이전시의 팩토리에 있는 참조리스트를 접근한다. 만약 참조리스트에 이미 사용자가 요청한 정보를 처리하는 에이전시들의 리스트가 존재하면 자신의 에이전시가 처리한 정보를 바로 요청하여 사용자에게 전달하며 만약 리스트가 존재하지 않을 경우는 에이전시를 생성하고 다른 멀티에이전트들로부터 정보를 요청하는 작업을 수행한다. 다음 시나리오는 후자일 경우 시나리오를 작성한 것이다.

1) 정보를 필요로 하는 클라이언트 멀티에이전트(CMA)는 기존의 멀티에이전트들에게 각각 에이전시를 생성해 아래 그림 4에서와 같이 MA1의 사용자로부터 정보요청이 들어오면 MA1의 에이전시는 이를 감지하여 MA1을 클라이언트 멀티에이전트로 여기고 연관된 멀티에이전트들에게 정보 요청메시지를 보낸다.
2) 이때 요청 메시지는 CMA와 다른 주소 공간에 있는 대상 멀티에이전트들에게 전달하기 위해 패키지화된다. 패키지화된 메시지는 상호호환성이 뛰어나 멀티에이전트들 간의 메시지전달을 빠르고 쉽게 해주는 장점이 있다.
3) 패키지화된 정보요청 메시지는 인접해 있는 멀티에이전트들 중 에이전시가 신경망을 통해 클러스터링 하여 선택한 가장 연관성이 있는 멀티에이전트들에게 전달된다.
4) 메시지를 받은 원격지 멀티에이전트의 에이전시들은 멀티에이전트 중 자원에이전트로 요청 메시지를 전달하고 자원에이전트로부터 전달받은 결과를 CMA로 되돌린다.
5) 사용자의 요구를 만족시킨 후 CMA 에이전시와 선택된 에이전시들간의 참조리스트를 작성하여 연관된 멀티에이전트패턴 경로를 에이전시 팩토리에 저장시켜 다른 멀티에이전트에 존재하는 값을 CMA에서 보유하고 운영시킬 수 있도록 한다.

이로써 신속한 정보를 제공할 수 있으며 참조리스트에 존재하는 에이전시들 중에 정보가 변경되면 이에 대한 신뢰성 있는 정보를 위한 방법으로 멀티에이전트 관리기법을 적용시킨다.



〈그림 4〉 멀티에이전트 상호협력 구조

3.2 에이전시 클러스터링

가장 연관성 있는 정보를 처리하는 멀티에이전트를 찾기 위해 에이전시 필터에서는 자기조직화 신경망을 사용하여 클러스터링 한다. 자기조직화 신경망은 경쟁적인 방법으로 입력에 대하여 승리한 출력층의 자원만이 자신과 자신의 이웃 입력층과의 연결강도를 조정할 수 있도록 하는 승자선택(Winner-Take-All)방법을 취하고 있다. 이러한 경쟁을 통하여 점차 신경망은 입력의 구조를 반영하

는 자기조직화 지도를 완성해 나가며 동적인 네트워크 환경에도 잘 대처한다. 그러나 기존의 자기구성 지도 알고리즘은 긴 훈련시간으로 인해 정보처리 시간이 길어지는 단점이 있다. 따라서 본 연구에서는 기존의 자기구성 지도알고리즘을 응용한 새로운 필터링 자기구성 지도 알고리즘을 통해 정보처리 시간과 시스템의 복잡도를 줄이도록 하였으며 그 절차는 아래와 같다.

Step 1. 입력층에 입력패턴(입력벡터)을 제시하여 연결강도를 초기화시킨다.

Step 2. 초기화된 연결강도와 입력패턴사이의 거리를 계산하기 전에 초기화된 연결강도를 필터링 한다.

Step 3. 최종 선택된 각 출력노드의 연결강도벡터와 사용자 요구 메시지인 입력패턴과의 거리를 구해서 가장 짧은 거리의 승자노드를 구한다.

Step 4. 승자노드와 이웃노드의 연결강도를 조절한다.

Step 5. 이웃의 범위와 학습율을 감소하면서 이웃의 범위가 자기 자신이 될 때까지 Step 1-4 과정을 반복한다.

그리고 표 1에 나타난 알고리즘은 위에서 제시한 절차를 표현한 것이며 네트워크 상에 수많은 노드들이 존재할 경우 초기화된 노드들의 가중치 값(Wi)과 기준 가중치 값(Ws)을 비교하여 필터링 과정을 수행하며 필터링 된 노드들의 가중치 값(Wf)이 (+)값이 될 경우에만 노드들을 선택하고 선택된 노드들의 가중치 값(Wf)과 입력패턴(Ii) 사이의 거리를 계산하여 최종의 승자노드를 선택하였고 선택된 노드(멀티에이전트)로부터 넘겨받은 정보를 사용자에게 제공하도록 하였다.

이 방법은 요청한 키워드에 대한 요구사항(Ui)에 대해 필터링 과정을 거쳐 기준 가중치(Ws) 이상의 가중치 값(Wf)만을 클러스터링하여 선택된 원격지 에이전트들로부터 정보를 클라이언트 에이전트에게 전달하므로 신경망의 단점인 긴 훈련시간을 단축시킬 수 있고 정보 처리시간을 단축시킬 수 있다.

〈표 1〉 Filtering SOM 알고리즘

```
Algorithm : Filtering SOM
Input : the training samples
Output : Clustring after selecting
  distance(d)
Method
 Compute the net input of user request
  message I;
     I=SUM(Ii*Mp);
     broadcast I through IIOP;
  while terminating condition is not
    satisfied {
     for each receiving Ui from MAs
Step 1: Initialize all weights in
  network
               Wi <- wi=0 ~ 1;
Step 2: Filter initialized weights
               Set Ws=∀value;
               Wf = AgencyFilter(Wi
                 <> Ws);
Step 3: Calculate the distance(dj)
     if Wf > 0 then Select Wf;
     dj = Σ_{i=0}^{N-1} (Ξ(t) - Wij(t))^2;
     if OS ← dj then /* select MA in
       Output Selector */
      if dj < dj+1 then select(MA);
            send(Pi);
            record(reference list);
      end if
     end if
    end if
Step 4: Coordinate the new weight
     If WC ← result then /* send to
       Weight Coordinator */
           Wij(t+1) = Wij(t) + α(Xi(t) -
             Wij(t));
           Wnew ← Wij(t+1); //weight
             update
     end if
     if WI ← Wnew then /* coordinate new
       weight */
              coordinate(Wnew);
     end if
}
```

3.3 멀티에이전트 관리

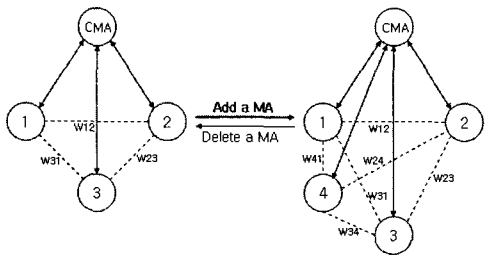
(1) 정적 멀티에이전트 관리

새로운 멀티에이전트가 네트워크에 추가되지 않고 분산환경에 존재하던 기존의 멀티에이전트가 네트워크로부터 분리 또는 삭제될 경우에는 사용자가 찾은 정보가 이미 에이전시 팩토리에 존재하고 있으므로 기존의 정보에는 영향을 미치는 요소가 되지 않는다. 그러나 기존 멀티에이전트의 정보가 새로 갱신된 경우에는 에이전시 팩토리에 존재하는 정보에 영향을 미치게 되므로 에이전시 팩토리 안에 존재하는 리스트를 참조하여 목적지 멀티에이전트로부터 갱신된 정보의 내용을 기존의 정보에 반영시킨다.

(2) 동적 멀티에이전트 관리

새로운 멀티에이전트가 네트워크에 추가되면 기존 네트워크에 존재하는 표준 가중치 값과 자신이 처리하는 정보의 가중치 값을 비교하기 위해 새로운 멀티에이전트에 가중치 값을 부여한다. 만약 새로운 멀티에이전트 가중치 값이 기존 네트워크의 표준 가중치 값보다 작을 경우 이를 네트워크에 반영하지 않고 큰 경우에는 에이전시 클러스터링 기법 중 Step 4를 수행한 것과 같이 가중치 값을 재조정하여 항상 새로운 가중치 값을 유지하도록 한다. 이렇게 새로운 가중치 값을 항상 보유하고 있다는 것은 동적으로 변화하는 분산환경의 멀티에이전트들을 동적으로 대처할 수 있는 능력을 제공한다.

새로운 가중치 값은 여러 사용자가 같은 정보를 요청하였을 경우 기존의 가중치 값이면 에이전시 팩토리에 저장된 정보를 참조하여 사용자에게 정보를 전송하며 새로운 가중치 값으로 대치된 경우에는 훈련을 통해 분산된 멀티에이전트들로부터 정보를 다시 찾는다. 이렇게 함으로써 사용자는 최신의 정보를 제공받을 수 있고 기존의 네트워크와는 달리 모든 노드에 대한 훈련을 수행하지 않고 가장 밀접한 정보를 처리하는 멀티에이전트들로부터 정보를 가져오기 위해 훈련을 수행하므로 기존 SOM의 단점인 긴 훈련시간을 극복하여 정보처리 시간을 크게 단축시키고 비감독 학습(unsupervised learning)으로 정확한 훈련 예가 없어도 높은 성능의 클러스터링을 제공한다.



〈그림 5〉 네트워크에 새로운 멀티에이전트 추가하는 경우

〈표 2〉 새로운 멀티에이전트 추가 알고리즘

```
Algorithm for Adding MA

On add new MA into existing network
        decide(weight of MA);
        if Wn < Ws then
                disregard(Wn);
        else
        for each weight Wij in network {
                Wij(t+1) = Wij(t) + α(Xi(t)
                  - Wij(t));
                Wnew ← Wij(t+1);
        end if

On request the same Information from
  several users
        if Wnew ≠ Wold then
                search(I) from Agency
                  Factory; //I::Information
                respond(I);
        else
                 training(I) with Wnew;
                 respond(I);
        end if
```

4. 성능평가

4.1. 시뮬레이션 환경

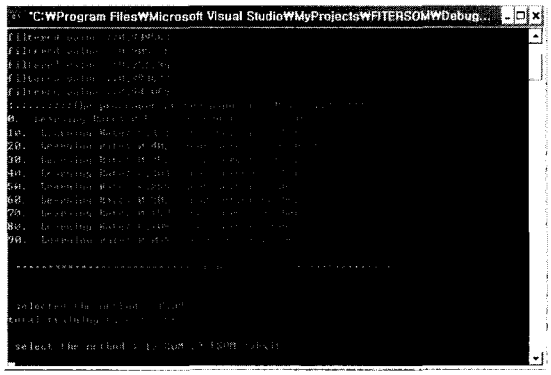
본 연구에서 구현한 정보검색 시스템 환경은 정

보를 요청하는 클라이언트 멀티에이전트와 상호 협력을 통해 정보를 처리하는 멀티 에이전트들을 각각 객체모듈을 가지는 시스템들로 구성하였다. 구현 환경은 Window 2000 server를 사용하였으며, 이들 시스템간의 정보처리를 위해 OMG CORBA 규격에 따라 상용화된 제품인 IONA사의 Orbix 3.01 for Windows를 미들웨어로 두었고, 개발 언어는 IDL과 C++언어를 사용하여 Orbix 상에서 구현하였으며 C++컴파일러는 Microsoft사의 Visual C++6.0을 사용하였다.

분산 환경에서 멀티에이전트 시스템 사이의 상호 협력기법으로 제시한 FSOM알고리즘과 기존의 코호넨의 자기구성 지도 모델인 SOM알고리즘을 비교하였다. 하나의 클라이언트에 에이전시로부터 IIOP를 통하여 멀티케스트 된 기사를 각각의 서버 에이전트들이 수신하였을 경우 그 입력에 가장 밀접한 정보를 처리하는 노드를 선택할 때 SOM과 FSOM 알고리즘을 통해 각각 클러스터링이 시작된 시각과 끝난 시각의 차를 계산하여 처리시간을 산출하였다.

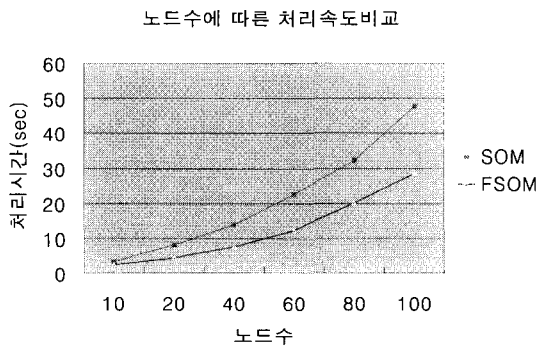
4.2 결과 및 분석

기사 검색 시 이루어지는 학습 과정은 사용자로부터 정보를 입력받아 이 입력 데이터패턴을 보내는 클라이언트와 입력 데이터와 연관된 정보를 처리하는 100개의 원격지 멀티에이전트를 가정하였다. 학습율은 0.5에서 시작하여 0.01에서 끝나고 훈련집합은 플러스 기호(+)의 모양으로 분포되는 1000개의 2차원적 좌표들로 구성되고 훈련은 훈련집합의 100회의 반복으로 수행되며 승자 노드 선택은 매 10회 실행마다 이루어진다. 그리고 아래 그림 6에서와 같이 근접한 서버 개수를 경쟁학습에 의해 줄여가며 최종 서버개수가 1이 되었을 때 프로그램은 종료된다.

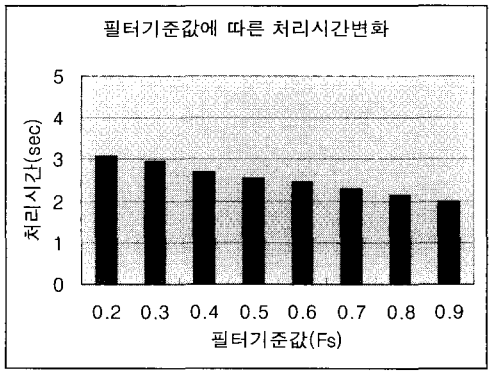


〈그림 6〉 승자 노드 검색을 위한 훈련과정

실험 결과는 다음 그림 7과 같다. 필터링 기준값을 0.5로 고정하고 학습률은 0.5 반복회수는 100회일 때 정보를 처리하는 노드의 수에 따른 신경학습방법인 SOM과 FSOM 알고리즘의 성능을 보



〈그림 7〉 시뮬레이션 결과1



〈그림 8〉 시뮬레이션 결과2

여주고 있다. 결과에서 보는 바와 같이 처리속도에서 FSOM방법을 적용한 경우가 SOM방법을 적용하여 클러스터링 한 경우에 비해 더 빠른 클러스터링을 수행하였고 노드의 수가 증가하면 할수록 더 큰 차이의 처리속도를 보였다.

그림 8은 정보를 처리하는 노드의 수를 10으로 고정하였을 때 필터 기준값에 따른 처리시간 변화에 대한 결과이다. 필터 기준값이 클수록 처리시간이 단축되는 결과를 보였고 이는 SOM에 적용한 필터링 부분이 복잡한 훈련시간을 단축시켜서 전체적으로 처리시간을 1.2초 단축 시켜 주었고 노드의 수가 증가함에 따라 반영되는 비율도 커지므로 시스템 전체에 미치는 영향은 커지게 되었다.

5. 결 론

본 연구에서는 이미 생성되어 운영 중인 멀티에이전트들을 단일 정보 시스템처럼 일괄되게 사용자에게 제공하기 위해 CORBA를 기반으로 하였다. 그리고 멀티에이전트들 간의 상호협력을 위해 사용자의 요구가 들어 왔을 때 멀티에이전트에 에이전시를 생성시키고 주변 멀티에이전트와 통신하기 위해 패키지화한 메시지를 IIOP 통신프로토콜을 통해 다중 전송시킨다. 그 이후에 기존의 멀티에이전트에서 상호 협력을 위해 중재에이전트의 등록 시 문제점인 복잡도와 여러 에이전트가 동시에 접속할 경우 피할 수 없는 서버 부하 문제를 해결하기 위한 방안으로, 신경망을 이용한 FSOM 기법을 제안하였다. 이 방법은 요청한 키워드에 대한 요구사항에 대해 기준치 이상의 가중치 값을 보이는 노드들을 선택하여 선택된 노드들로부터 넘겨받은 정보를 클라이언트 에이전트에게 전달하므로 훈련시간을 단축시켜 정보 처리시간을 빠르게 하였으며 에이전시를 통해 독립된 다른 멀티에이전트와 상호 협력을 하여 더 정확한 정보를 제공 할 수 있도록 하였다. 또한 동적으로 변화하는 멀티에이전트 노드들의 추가로 인한 정보 갱신 문제와 기존의 노드들 가운데 정보를 갱신한 노드로부터 정보 갱신문제를 해결하기 위해 각각 동적 멀티에이전트 관리 기법과 정적 멀티에이전트 관리 기법을 제안하였으며 이를 통해 정보의 신뢰성을 높였고 에이전시 팩토리에 참조 리스트를 저장하여 캐쉬역할을 담당케 함으로써 빠른 정보 처리 서비스를 제공하도록 하였다.

성능평가에서는 노드의 수에 따른 SOM과 FSOM 알고리즘의 처리속도를 비교하였고 FSOM 방법을 적용한 경우가 SOM 방법을 적용한 경우보다 전체적으로 더 빠른 클러스터링을 수행하였고 노드의 수가 증가함에 따라 점점 더 큰 차이의 처리속도를 보여 처리속도에서 우수한 성능을 보였다.

본 연구는 멀티에이전트의 상호협력 및 이를 통한 정보검색 시스템에 활용될 수 있으며, 메시지 패키지화기법, 효율적인 정보제공을 위한 참조리스트 저장구조에 대한 세부적인 연구 및 브로커 에이전트의 브리지 기법에 대한 향후 연구가 필요하다.

참 고 문 헌

[1] 최중민 "에이전트의 개요와 연구 방향", 정보과학학회지, 제15권, 제3호, pp. 79-92, 1997.

[2] O. Etzioni. "The world wide web:Quagmire or gold mine", Communication of ACM. Vol.39 No.11, pp.1-5, March 1996.

[3] 김태훈, 최중민 "사용자 편의의 인터넷 정보검색을 위한 지능형 웹 브라우징 에이전트", 정보과학회논문지, 제25권, 제7호, pp. 1064-1078, 1998.

[4] FIPA. "Agent Management", FIPA, http://www.fipa.org/spec/FIPA9 8.html, Ver. 1.0, 1998.

[5] In Michael J. Wooldridge and Nicholas R. Jennings "Intelligent Agents", pp. 1-39, Sringer-Verlag, Germany, 1995.

[6] T. Witting, N. R. Jennings and E. H. Mamdani "ARCHON-A Framework for Intelligent Co-operation", IEEBCS Journal of Intelligent Systems Engineering-Special Issue on Real-time Intelligent Systems in ESPRIT, 1994.

[7] 이명진, 김진상 "BDI 에이전트 구조에서 충돌 해결을 위한 논리기반 협상 기법의 연구", 한국 퍼지 및 지능시스템학회 논문지, Vol. 10 No, pp. 548-556, 2000.

[8] 장명욱, 박상규, 이광로, 민병의 "다중 에이전트 시스템 상에서 에이전트 수행종료에 의한 문제해결", 한국정보처리학회 논문지, 제4권, 제1호, pp. 118-136, 1997.

[9] 양소진, 이현숙, 오경환 "유사정보 추출에 기반 한 조정 에이전트 모델", 한국 인지과학회 논문지, 제 12권, 제 1 · 2호, pp. 55-64, 2001.

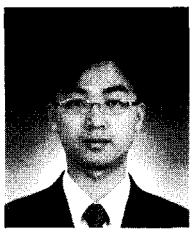
[10] 금영욱, 장연세, "코바3프로그래밍바이블", pp. 433-453, 도서출판 그린, 2000.

[11] 김대수, "신경망 이론과 응용", pp. 169-177, 하이테크정보 출판사, 1999.

[12] J. J. Hopfield, Neurons with graded response have collective computational properties like those of two-state neurons, Proc. of the National Academy Science 81, pp. 3088-3092, 1984.

[13] Jiawei Han and Micheline Kamber, "Data Mining : Concepts and Techniques", pp. 379-381, Morgan Kaufmann, 2001.

◐ 저 자 소 개 ◑



박 민 기
2002년 군산대학교 정보통신공학과 졸업(학사)
2004년 군산대학교 대학원 정보통신전파공학과 졸업(석사)
2004년 ~ 현재 (주)버추얼다임 연구원
관심분야 : 분산시스템, 데이터 마이닝, 멀티에이전트, CDMA이동통신
E-mail : sopiru@kunsan.ac.kr

김 귀 태
1989년 계명대학교 사회과학대학 무역학과 졸업(학사)
1995년 계명대학교 교육대학원 전산교육학과 졸업(교육학 석사)
1993 ~ 현재 대한상공회의소 충남인력개발원 교수
2002년 ~ 현재 군산대학교 전자정보공학부 박사과정
관심분야 : 분산시스템, 네트워크, 데이터 마이닝, 멀티에이전트
E-mail : gtkim@kunsan.ac.kr

이 재 완
1984년 중앙대학교 전자계산학과 졸업(학사)
1987년 중앙대학교 대학원 전자계산학과 졸업(석사)
1992년 중앙대학교 대학원 컴퓨터공학과 졸업(박사)
1996년 3월 ~ 1998년 1월 한국학술진흥재단 전문위원
1992년 ~ 현재 : 군산대학교 전자정보공학부 교수
관심분야 : 분산시스템, 운영체제, 데이터베이스, 컴퓨터 네트워크 등
E-mail : jwlee@kunsan.ac.kr