분산 환경 하에서 지능형 에이전트 기반의 자동 스케줄링을 위한 제약만족 기법의 응용†

(Applying CSP techniques to automated scheduling with agents in distributed environment)

정 종 진*
(Jong-Jin Jung)

요 약 분산 인공지능 기법들은 기존의 처리되지 못했던 많은 복잡한 문제들을 모델링하여 해결할 수 있도록 하고 있다. 특히 지능형 에이전트 기법은 분산 처리 환경 하에서 프로세스들 간의 각기 다른 문제 해결 능력들을 공유하도록 하는 방법들을 제공함으로써 효율적인 방법론이 될 수 있음을 증명하고 있다. 본 논문에서는 분산 환경의 스케줄링 문제에 에이전트 기법을 적용하여 문제를 해결하는 멀티에이전트 기반의 스케줄링 시스템 모델을 제안하고자 한다. 특히 멀티에이전트 스케줄링 시스템에 CSP 기법을 응용하여 모델링함으로써 효율적인 스케줄링이 이루어지도록 한다. 제안하는 모델에서는 스케줄링 문제를 제약조건에 따라 부문제들로 분할하고 부문제들에 대해서 분산되어 있는 각각의 에이전트가 자체적인 CSP 해결기에 의해 해를 구하도록 한다. 본 논문에서는 제안하는 모델의 사례 연구로써 회의일정 스케줄링 문제를 통해 CSP의 모델링과 멀티에이전트 시스템을 이용한 문제해결 기법을 제시한다.

핵심주제어 : 멀티에이전트, 스케줄링, CSP

Abstract Many researchers have modeled complicated problems with distributed AI technologies in distributed environment. Especially, intelligent agent technology are often proposed among researchers as efficient method to solve the complicated problems because agent technology makes possibile to share different problem solving abilities among processors in distributed processing environment. In this paper, we propose multiagent system model to solve distributed scheduling problem. At this point, we apply CSP techniques to the multiagent model as individual problem solving ability of member agents. Scheduling problem is divided into subproblems according to constraints by distributed resources, then each agent solves its subproblem using CSP solver in the proposed model. This method improves scheduling efficiency. For meeting scheduling problem of case study, we show CSP modeling process and suggest problem solving procedure by multiagent system model.

Key Words : Multiagent, Scheduling, CSP

1. 서 론

인공지능에서 CSP(Constraint Satisfaction Problem)는 변수(Variable)의 집합과 각각의 변수가 취할 수 있는 값들의 집합인 도메인(Domain), 그리고 서로 다른 변수의 도메인들과의 관계를 나

† 이 논문은 2005학년도 대진대학교 학술연구비지원에 의한 것임.

* 대진대학교 컴퓨터공학과

타내는 제약조건(Constraint)의 집합으로 구성되는 문제로서, 각 변수가 가지는 유한 이산 도메인 내에서 일관성 검사기법이나 휴리스틱 탐색기법 등을 적용하여 모든 제약조건을 만족하는 해를 탐색함으로써 문제를 해결하는 기법이다. 이러한 CSP는 실세계에서 발생되는 스케줄링 문제, 자원할당 문제, 운송 문제, 인력 배치 문제 등과 같은 이산 복합 문제들(Discrete Combinatorial Problems)에 적용되어 그 탁월한 성능을 입증하여 왔다. 특히 스케줄링은 일련의 순서를 갖는 작업(Activity)을 수행하기 위하여 작업의 시간영역에 자원(Resource)을 할당하는 문제로서, 스케줄링 문제 해결에 있어서는 CSP 기법이 규칙 기반 기법에 비하여 문제에 대한 표현력이 뛰어나 제약조건의 변화에 쉽게 적용 가능하고, 제약조건 전파(Constraint Propagation)를 통해 탐색 공간을 축소하여 해를 찾는데 효과적이다[8]. 회의일정 스케줄링 문제 역시 참석자의 시간을 자원으로 고려하는 전형적인 스케줄링 문제이고, 또한 각 구성원은 자신의 일정을 따로 관리하기 때문에 회의일정 스케줄링에 필요한 정보나 지식이 자연적으로 분산되는 분산 시스템의 특성을 갖는다. 따라서 모든 구성원의 일정에 대한 정보를 갖는 하나의 데이터베이스와 중앙 제어 방식을 사용하여 회의일정을 스케줄링하는 것은 구성원의 사생활 침해와 같은 문제점을 안게 되므로 분산 시스템의 제어로 관련 정보만을 제공하여야 한다. 이렇듯 현실세계에는 중앙 집중 처리 방식에 의해 해결하기엔 부적절한, 분산된 환경에서의 처리가 요구되는 문제들이 존재한다. 분산 인공지능에서는 이러한 문제를 해결하기 위한 방법론으로 분산문제해결 기법(Distributed Problem Solving: DPS)과 멀티 에이전트 시스템(Multi-Agent System)에 대한 연구를 활발히 수행하고 있으며, 이러한 측면에서 회의일정 스케줄링 문제도 멀티 에이전트 시스템을 기반으로 연구되고 있다.

DCSP(Distributed CSP)는 멀티 에이전트로 구성된 CSP로 정의되는데 CSP 문제가 변수나 도메인, 제약조건에 따라 논리적으로 혹은 지리적으로 분산되는 특성을 이용하여 다양한 DPS 문제가 DCSP로 정형화가 가능함이 밝혀졌다[1, 2]. 따라서 본 논문에서는 멀티 에이전트로 구성된 회의일정 스케줄링 시스템에 기존의 규칙 기반 기법이 아닌 제약 만족 기법을 적용하였다. 멀티 에이전트 시스템에 적합한 변수 기반의 DCSP 문제로 모델링하고, 문제를 해결하는데 있어서 변수로 사용될 회의일정에 대해서는 전역 정보가 제공되지 않으므로 분산 제어 방식을 사용하는 CP&CR (Constraint Partition & Coordinated Reaction)[4] 방법론을 적용함으로써 해결 가능하도록 하였다.

본 논문의 구성은 먼저 서론에 이어 2장에서 DCSP 문제 해결을 위한 접근 방법과 멀티 에이전트의 응용에 대해 설명한다. 이를 토대로 3장에서는 회의일정 스케줄링 문제를 DCSP로 모델링한다. 4장에서는 회의일정 스케줄링 문제에 대하여 멀티 에이전트 시스템을 설계하고 에이전트들 간의 조정을 통해 문제를 해결해 나가는 과정을 보인다. 마지막으로 5장에서는 본 논문의 결론을 맺기로 한다.

2. DCSP 모델링과 접근방법

2.1 DCSP 기반의 문제 해결기법

DCSP는 분산 AI의 한 연구 분야인 협동적 분산 문제 해결기법(Cooperative Distributed Problem Solving: CDPS)을 정형화하기 위한 방법으로서 멀티 에이전트 시스템의 구조를 갖는 확장된 CSP로 정의된다[2]. DCSP에서는 문제를 독립된 형태의 부문제들(Subproblems)로 분할하고, 각각의 부문제에 대하여 해당하는 에이전트가 지역 CSP해결기(Local CSP Solver)를 통하여 지역해를 구하고, 상호 간의 통신을 통해 공통해를 구함으로써 문제를 해결한다. 분산 환경에서의 처리가 요구되는 문제는 각 에이전트가 맡아 수행하는 부문제들이 어떠한 요소의 특성을 갖느냐에 따라 다음과 같은 문제 해결기법들이 적용된다[3].

2.1.1 도메인 기반의 해결기법

특정 변수가 갖는 도메인의 일부분, 즉 탐색 영역별로 문제가 분산된 경우, 에이전트들은 서로 독립된 각각의 탐색 영역을 맡아 지역적으로 각각의

부문제에 대한 해를 CSP 해결기법으로 구한다. 따라서 에이전트들 간의 통신이 거의 발생하지 않으므로, 해를 구하는데 있어서 시간적인 성능향상을 얻을 수 있으나, 분산된 탐색 영역 간에는 논리적 연결성이 부족하다. 과다한 제약조건을 갖는 문제의 경우, 문제 해결에 대한 실패 원인을 찾기 힘들기 때문에 단순히 크고 복잡한 문제 해결을 위해 멀티 에이전트 환경의 병렬성을 이용한 CSP 해결기법에 가깝다.

2.1.2 함수 기반의 해결기법

문제의 해를 찾기 위한 반복적인 탐색 작업의 일부분, 예를 들어 CSP의 Forward Checking 알고리즘에서의 변수마다 반복적으로 행해지는 도메인 여과와 같은 작업 단위로 분산되어 있는 경우, 에이전트는 마치 트리 탐색구조와 같이 루트 에이전트에 의해서 필요 시 생성 혹은 소멸되며, 공유메모리를 갖는 시스템의 경우에만 적용이 가능하다. 따라서 분산제어 방식을 사용하는 문제 해결기법으로는 적절하지 못하다.

2.1.3 변수 기반의 해결기법

문제가 변수 기준으로 분산된 경우, 각 에이전트는 연관성을 갖는 하나 이상의 변수와 그들의 도메인을 제어하게 된다. 이 경우 탐색공간은 에이전트들 사이에서 공유되고, 변수간의 제약조건 때문에 한 에이전트의 행동은 다른 에이전트에게 직접적으로 또는 간접적으로 영향을 미치게 된다. 따라서 한 에이전트는 자신이 제어하고 있는 변수들로 구성된 지역 CSP를 해결하고, 여기서 구한 지역해를 가지고 제약조건을 공유하는 에이전트들과 통신을 통해 공통해를 찾게 된다. 멀티 에이전트 시스템 구조를 갖는 다양한 문제들이 변수 기반의 해결기법을 이용하여 해를 찾을 수 있으며, 일반적으로 DCSP는 변수 기준으로 분산된 문제를 말한다[2].

본 논문에서는 제시된 분산 문제 해결기법 중, 변수 기반의 해결 기법을 이용하여 기존의 멀티 에이전트 시스템의 구조로 표현되는 전형적인 문제를 DCSP로 정형화하여 해결한다.

2.2 멀티 에이전트 시스템의 응용

멀티 에이전트 시스템은 각각의 독립적인 기능을 갖고 있는 에이전트들이 분산 환경 하에 존재할 경우 이 에이전트들을 목적에 맞게 그룹화하고, 이들의 상호 협력을 통하여 단일 에이전트 시스템으로는 해결하기 어려운 복잡한 문제를 해결하는 시스템이다. 따라서 변수 기반의 DCSP 문제는 멀티 에이전트 시스템에서의 여러 에이전트가 상호 협력하여 문제를 해결해 나가는 것과 비교될 수 있다. CSP 모델링에 의해 변수로 표현되는 문제의 요소가 지리적으로 분산된 경우, 에이전트는 하나 이상의 변수를 맡아서 제약 만족 기법을 통해 그들의 도메인을 제어하여 지역해를 구한다. 제약조건을 공유하는 에이전트 간에는 통신을 통해 서로의 지역해가 제약조건을 위배하는지 검사하고, 변수가 모든 제약조건을 만족하도록 충돌을 해결하여 공통해를 찾게 된다.

중앙제어(Centralized Control) 방식은 전역 정보를 이용하는 것으로, 예를 들어 에이전트에 우선순위를 부여하여, 이 순서에 따라 에이전트들 간에 발생하는 충돌을 해결하도록 하는 제어 방식이다. 그러나 회의일정 스케줄링과 같이 전역 정보가 제공되지 않는 경우에는 분산 제어 방식을 사용하게 된다. 분산제어(Decentralized Control) 방식은 에이전트들 간의 전역적인 정보를 갖지 않으므로 탐색 작업 중에 발생한 충돌은 협상(Negotiation)이나 지역적으로 순서를 부여하여 전역적인 정보 없이 지역해를 전체적인 규모로 전파함으로써 해결하게 된다.

2.3 CP&CR 해결기법

CP&CR[4]은 변수 기반의 DCSP 해결기법 중 하나로 분산 환경에서의 처리가 요구되는 문제를 하나의 CSP로 정형화하여, 문제의 제약조건을 유형별로 분류하여 그룹화(Constraint Bunch)하고, 각 제약조건에 대해 서로 연결성이 있는 변수 단위로 분할하여 한 에이전트가 분할된 변수의 집합(Constraint Cluster)을 맡아 문제를 해결해 나가는 방법이다. 따라서 각 에이전트마다 변수 집합에 대한 제약조건의 정보가 지역적으로 분산되기 때

문에 전역적으로 제약조건을 분석하는 과정이 요구되지 않는다는 이점을 갖는다. 동일 그룹의 에이전트들은 연결성(Connectivity)이 있는 변수들의 집합으로 구성되어 있어, 자신이 제어하고 있는 변수에 해당하는 제약조건을 만족하도록 각각 독립적인 지역 CSP를 동시에 해결하게 된다. 반면 서로 다른 그룹의 에이전트들은 제약조건의 유형에 따라 그룹 단위로 분류되었기 때문에 하나 이상의 제약조건에 의해 제한을 받는 변수에 대해서는 공유된 탐색영역을 바탕으로 비동기적으로 수행된다. 탐색영역이 공유되는 변수는 서로 다른 에이전트에 의해 제어되므로 공통해를 찾는데 있어서 에이전트들 간의 충돌이 발생한다. 따라서 협상이나 제약조건 전파 혹은 지역해에 대한 정보교환을 통해 반응적 행동(Reaction)으로서 해결된다. 이러한 방법을 이용하여 [4]에서는 Job Shop Scheduling 문제를 작업 에이전트(Job Agents)와 자원 에이전트(Resource Agents)로 각각 모델링하고, 이를 바탕으로 효과적으로 해결할 수 있음을 입증하였다. 이러한 측면에서 본 논문에서는 회의일정 스케줄링 문제에 대하여 DCSP로 모델링하고 멀티 에이전트 시스템에 의한 해결 방법을 제시한다.

3. 회의일정 스케줄링 문제의 DCSP 모델링

3.1 문제의 특성과 접근 방법

회의일정 스케줄링 문제는 스케줄링에 필요한 정보나 지식이 자연적으로 분산되는 특성을 포함하고 있다. 따라서, 모든 구성원의 일정에 대한 정보는 분산 시스템의 제어에 의해 관련 정보만을 서로 간에 제공하여야 한다. 이러한 특성을 고려하여 멀티 에이전트 시스템을 기반으로 회의일정 스케줄링 방법들이 연구되어 왔다. 우선 블랙보드(Blackboard)의 기능을 갖는 에이전트가 모든 스케줄링 작업을 전담하여 해결하는 방법은 각 사용자 에이전트가 스케줄링 작업의 기능이 요구되지 않으므로 시스템 구성이 단순하고, 전체적인 최적화를 위한 조정이 용이하지만, 스케줄링 작업을 전담하는 에이전트가 작업량에 따른 부하가 크다는 단점이 있다. 그리고 회의 주체자 측의 에이전트가 스케줄 작업을 하는 방법은 스케줄링만을 전담하는 특별한 에이전트가 불필요하지만, 비협조적인 에이전트들로 인해 스케줄링을 할 수 없게 되는 경우가 발생하므로 이를 해결하기 위한 에이전트들 간의 효율적인 협상 방법이 필요하다[5, 6].

본 논문에서는 회의일정 스케줄링 문제를 DCSP 문제로 모델링하여 스케줄링 작업에 있어서 고려되어야 하는 시간에 대한 제약조건들을 유형에 따라 분리함으로써 참석자 에이전트들과 회의 주체자 에이전트들을 그룹화하였다. 그룹별로 서로 다른 유형의 제약조건을 맡아 동시에 처리함으로써 한 에이전트가 모든 제약조건을 맡아 처리해야 하는 많은 양의 스케줄링 작업을 분산하였고, 스케줄링만을 전담하는 에이전트의 필요성을 없애고, 스케줄링 작업의 분산으로 인해 공통해 탐색도중 발생하는 지역해 간의 충돌은 그룹 간에 비동기적으로 지역 정보를 교환함으로써 해결하도록 하였다.

3.2 DCSP 모델링 과정

본 절에서는 회의일정 스케줄링 문제에 대하여 DCSP로 모델링하는 과정을 설명하도록 한다. 회의일정 스케줄링을 수행하기 위해서는 회의 주최자가 참석을 요하는 사람들에게 회의 예정 일자와 회의 시간 등을 알리면 참석자들은 자신의 개인적인 일정에 따라 참석여부 및 자신이 선호하는 회의 시간을 요구할 수 있어야 한다. 즉 스케줄링 작업을 하는데 있어서 주최자와 참석자 입장에서의 다양한 제약조건들이 만족되어야 한다. 예를 들어 다음과 같이 개최하고자 하는 회의 1, 2, 3이 있다고 하자.

회의 1 : 회의시간 2시간, 참석자 A, B, D
회의 2 : 회의시간 3시간, 참석자 A, C, D
회의 3 : 회의시간 1시간, 참석자 A, B, C

이 문제를 CSP로 정형화하면, 참석자 N이 참석해야 하는 회의 i의 일정 SNi는 회의시작시간(*start_t(SNi)*)과 회의시간(*dur(SNi)*)을 갖는 변수로 표현되고, 각각의 변수는 참석자 N의 개인 일정 중, 회의 예정 일자의 회의 참석이 가능한 시간을 매시간 단위로 표시한 이산 도메인(*DN*)을 갖게 된다. 그리고 변수들 간의 관계를 나타내는 제

약조건이 주어지는데, 스케줄링 문제에 있어서는 변수들 간에 만족해야 하는 다양한 제약조건 중에서도 반드시 시간에 대한 제약조건(Temporal Constraint)이 따르게 된다. 회의일정 스케줄링 문제에 있어서도 시간 제약조건은 참석자 측면에서 고려되어야 하는 참석자 제약조건과 주최자 측면에서 고려되어야 하는 주최자 제약조건으로 크게 두 가지로 볼 수 있다.

○ 참석자 제약조건

각 참석자의 회의일정은 다른 회의일정과 겹쳐서는 안 된다.

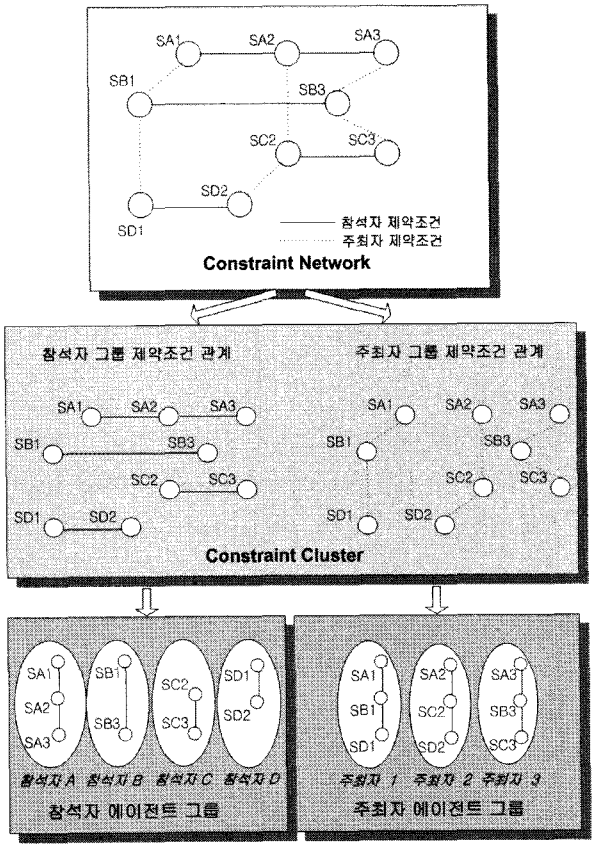
참석자 N이 참석하는 회의 i, j에 대해서,
start_t(SNi) + dur(SNi) start_t(SNj)
or start_t(SNi) dur(SNj) + start_t(SNj)

○ 주최자 제약조건

회의 i에 대해서는 참석이 요구되는 참석자 모두의 일정이 일치되어야 한다.

회의 i는 참석이 요구되는 참석자 N, M에 대해,
start_t(SNi) = start_t(SMi) (for N, M)

이상의 제약조건들을 모두 만족하도록 회의일정을 참석자의 개인 일정에 배정함으로써 문제는 해결된다. 이와 같이 변수, 도메인, 제약조건을 갖는 CSP 문제는 멀티 에이전트 시스템으로 구성하기 위해 시간 제약조건에 따라 문제를 분할하게 되는데, 그 결과는 그림 1과 같다. 우선 CSP로 정형화된 문제는 변수를 노드로, 제약조건은 에지로 표현되는 제약조건 네트워크(Constraint Network)를 통해 변수들이 어떠한 제약조건으로 연결되어 있는지를 쉽게 알아볼 수 있다. 이 문제는 에지의 타입, 즉 제약조건의 유형에 따라 참석자 제약조건에 의해 분류된 참석자 그룹과 주최자 제약조건에 의해 분류된 주최자 그룹으로 나뉘게 되고, 각각의 그룹은 에지로 서로 연결되어 있는 변수들을 구성요소로 하는 부분 집합들로 다시 분할된다. 이러한 문제 분할 과정을 통해 에이전트는 변수의 집합을 맡아 참석자 에이전트와 주최자 에이전트로써 각각의 시간 제약조건 및 개인 일정에 대한 제약조건들을 만족하도록 지역적으로 스케줄링하여 지역해를 구하고, 공통해를 구하기 위해 서로 다른 그룹의 에이전트들 간에 지역해에 대한 정보의 교환으로 문제를 해결하게 된다.

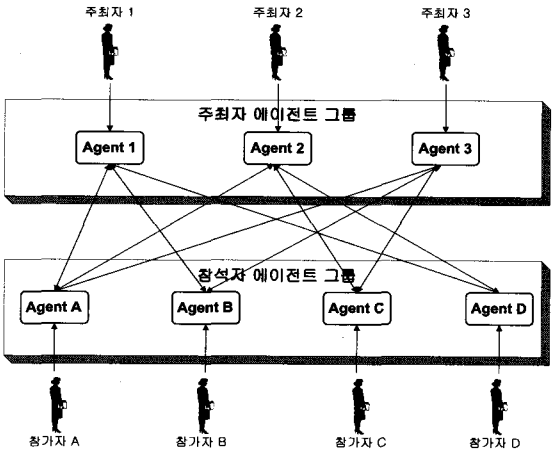


<그림 1> 회의일정 문제의 분할

4. 회의일정 스케줄링을 위한 멀티에이전트 시스템 설계

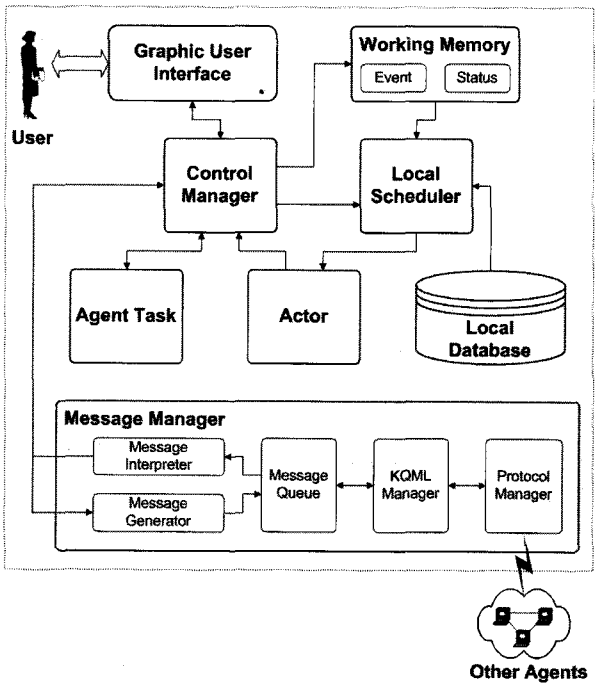
4.1 시스템 구성 및 에이전트의 구조

본 연구에서 제안하고자 하는 멀티 에이전트 시스템은 그림 2와 같이 문제 분할을 통해 분류된 참석자 그룹과 주최자 그룹, 각각의 그룹에는 변수의 집합을 맡고 있는 참석자 에이전트들과 주최자 에이전트들로 구성된다.



<그림 2> 에이전트 구성도

변수를 공유하는 다른 그룹의 에이전트들은 point-to-point 방식의 메시지 교환을 통해 지역 정보를 제공한다.



<그림 3> 에이전트 구조

각 참석자, 주최자 에이전트의 구조는 지역적으로 각자의 일정을 스케줄링하는 지역 스케줄러(*Local Scheduler*)와 다른 그룹의 에이전트들과의 메시지 교환 방식의 통신을 위한 메시지 관리자(*Message Manager*), 그리고 이 사이에 상호 작용을 위한 인터페이스와 같은 역할을 하는 제어 관리자(*Control Manager*)로 구분되어지고, 에이전트의 세부 구조는 그림 3과 같다.

그림 3에서 지역 스케줄러는 사용자의 개별적인 조건 혹은 다른 에이전트들의 일정 변경에 대한 요구에 따라 독립적으로 스케줄링 작업을 하고, 작업 결과에 대한 정보는 지역 데이터베이스(*Local Database*)에 저장한다. 제어 관리자는 에이전트의 전반적인 행동을 제어하게 된다. 사용자 인터페이스를 통해 입력된 제약조건과 다른 에이전트로부터 수신한 메시지의 내용을 지역 스케줄러에 전달하고, 스케줄링된 결과에 대한 정보를 사용자 및 관련 에이전트에게 전달할 수 있도록 메시지 관리자에게 전달한다. 작업 메모리(*Working Memory*)는 발생되는 사건에 대한 정보와 현재 에이전트의 상태에 대한 정보를 가지고 있다. 에이전트 작업(*Agent Task*)은 에이전트들마다 다르게 구성되어 에이전트의 역할을 결정지어주는 요소이다. 즉, 에이전트의 작업을 다르게 구현하고 지역 데이터베이스의 내용을 다르게 함으로써 에이전트는 고유한 기능을 갖는 에이전트로 특성화될 수 있다. 메시지 관리자는 지역 스케줄러에 의해 구해진 지역해에 대한 정보를 메시지화하고 제어 관리자를 통해 다른 에이전트와 상호 교환하는 역할을 담당한다. 이 때 메시지 관리자가 교환하는 메시지는 KQML을 사용한다. 메시지 관리자는 메시지의 물리적인 송수신을 담당하는 프로토콜 관리자(*Protocol Manager*), KQML 메시지의 송수신을 담당하는 KQML 관리자(*KQML Manager*), 보낼 메시지를 생성하는 메시지 생성기(*Message Generator*), 다른 에이전트로부터 수신된 메시지를 일시적으로 저장하는 메시지 큐(*Message Queue*), 다른 에이전트로부터 받은 메시지를 해석하여 스케줄러에 전달하는 메시지 해석기(*Message Interpreter*)로 구성된다.

4.2 회의일정 스케줄링 알고리즘

회의일정 스케줄링과 같은 멀티 에이전트 시스템을 기반으로 하는 문제는 다음과 같이 크게 3단계로 나누어지는 문제 해결 알고리즘을 적용할 수 있다. 각각의 에이전트들은 그룹별로 동시에 지

역 CSP를 통해 지역해를 구하고, 비동기적으로 그룹간의 지역 정보를 교환함으로써 에이전트들 간의 충돌을 해결하여 공통해를 찾는 문제 해결 과정을 갖는다.

① 회의일정 공시 단계

주최자 에이전트는 회의를 갖고자 하는 회의 예정일자 및 회의 진행시간, 스케줄링 처리기한(deadline)에 대한 정보를 참석이 요구되는 참석자 에이전트에게 보낸다. 참석자 에이전트는 개인 스케줄을 탐색하여 참석 가능 여부 및 회의 예정일자 중, 참석 가능한 시간 등을 주최자 에이전트에게 보낸다. 이 단계는 각 그룹의 에이전트들이 지역적으로 하위문제를 해결하는 첫 단계로, 각자가 맡은 변수 혹은 변수의 집합에 대해 제약조건을 만족하는 값을 찾아 지역 해에 대한 정보를 다른 그룹 내에 변수를 공유하는 에이전트에게 전달한다.

② 회의 시간 조정 단계

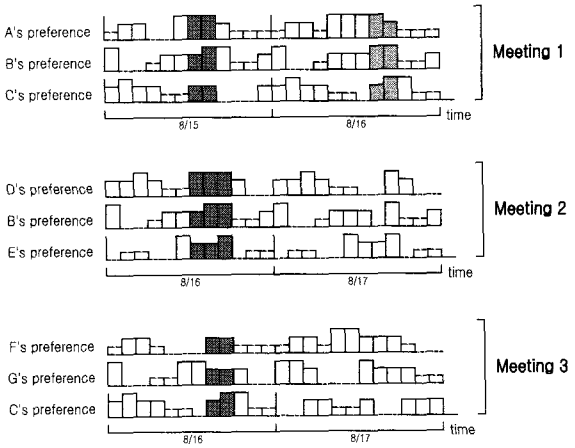
주최자 에이전트는 받은 지역 정보를 토대로 회의에 대한 참석자의 일정이 모두 일치하여야 한다는 제약조건을 만족하도록 스케줄링하여, 그 결과를 다시 참석자 에이전트로 전송하면, 참석자 에이전트 역시 전담한 변수들에 대한 제약조건을 만족하는 해를 찾는다. 이 단계는 에이전트들이 공통해 탐색 과정에서 서로의 모든 제약조건을 공유하지 않기 때문에 발생하는 충돌을 지역해의 정보를 통해 제약조건을 전파함으로써 해결한다. 지역해에 대한 충돌 해결은 검사 해결 공시 과정을 주기로 반복하여 검사 과정에서 더 이상 제약조건을 위배하는 변수가 발견되지 않으면 공통해가 구해지는 것이다. 단, 이 조정 과정을 최소화하기 위한 다양한 전략이 추가로 고려되어야 하지만, 본 논문에서는 이 부분에 대해서는 다루지 않도록 한다.

③ 회의일정 확정 단계

스케줄링 처리기한이 되면 주최자 에이전트는 조정단계에 의해 공통해를 찾은 경우 회의일정은 확정이 되고, 그렇지 못한 경우에는 다양한 전략을 통해 부분해를 구함으로써 문제 해결과정을 종료하게 된다. 멀티 에이전트 시스템과 같은 분산 처리 과정에서는 전역적인 정보가 제공되지 않으므로 문제 해결의 종료 지점을 파악하여야 한다. 따라서 지역해간의 더 이상의 충돌이 발생하지 않는 경우, 공통해로 인정하고 문제 해결이 종료되지만, 공통해가 존재하지 않는 경우는 처리기한을 두거나, 순환되는 조정 작업을 예방하는 전략을 적용한다.

4.3 회의일정 스케줄링 예시

본 논문에서는 전형적인 멀티 에이전트 시스템 구조로 표현되는 회의일정 스케줄링 문제에 대하여 DCSP의 모델링 과정을 통해 시간적 제약조건이 유형에 따라 변수의 집합들로 분할되면, 각각의 에이전트가 분담하여 회의일정 스케줄링 알고리즘을 통해 해결하도록 하였다. 이로써 회의일정 스케줄링과 같이 문제에 포함된 정보나 지식이 지리적으로 분산되어, 에이전트들 간의 협력을 통해 해결하는 전형적인 멀티 에이전트 구조의 문제가 DCSP 해결기법을 이용하여 해결 가능하도록 설계하여 보았다.



<그림 4> 회의일정 스케줄링의 예

그림 4는 제안하는 모델을 기반으로 하는 회의일정 스케줄링 예를 보여준다. 그림에서 회의일정 1(*Meeting 1*)의 회의 가능 시간 중 참석자들이 가장 선호하는 회의시간은 8월 16일의 4시부터 6시이고, 두 번째로 선호하는 시간은 8월 15일의 3시부터 5시이다. 그러나 회의일정 1을 참석자들이 가장 선호하는 시간으로 스케줄링을 수행하면 참석자 B가 참석하는 회의일정 2(*Meeting 2*)와 참석자 C가 참석하

는 회의일정 3(*Meeting 3*)은 스케줄링을 할 수 없게 된다. 본 모델에서는 회의일정 1을 8월 15일의 3시부터 5시로 스케줄링하여 확정짓고, 회의일정 2, 회의일정 3에서 발생할 수 있는 충돌을 피할 수 있었다. 앞으로 요청될 회의일정들이 잠정적으로 결정한 회의일정 2와 회의일정 3에 영향을 크게 미치지 않을 경우, 제안하는 회의일정 스케줄링의 조정 기법은 효과적인 결과를 보임을 알 수 있었다.

5. 결 론

본 논문에서는 분산처리 문제에 대하여 인공지능의 DCSP 기법을 적용하는 방법과 이에 대한 예시로서 회의일정 스케줄링 문제를 변수 기반의 DCSP로 모델링하여 멀티 에이전트 시스템으로 구현하는 방법을 설계하였다. 제안한 방법에서는 분산된 멀티 에이전트 환경에서의 문제를 DCSP로 모델링함에 따라 문제에 대한 표현력이 뛰어나 제약조건의 추가나 수정에 따른 변화에 쉽게 적응하고, 지역 정보의 교환을 통해 제약조건의 일관성을 유지함으로써 문제의 탐색 공간을 크게 축소시킨다. 또한 CP&CR 방법론을 사용하여 제약조건의 유형 및 연결성에 따라 변수의 집합 단위로 분할하여, 전역적인 정보가 공유되지 않아도 지역해에 대한 정보 교환으로 제약조건이 전파되어 다른 그룹의 에이전트들 간의 충돌을 해결하고, 보다 신속하게 공통해를 구할 수 있다.

본 연구에서는 향후에 에이전트들 간의 조정 작업을 최소화하기 위한 전략과 동적인 문제 변화에 따른 해결기법을 고려한 시스템의 구현을 통해 본 논문에서 제안한 문제 해결 기법의 효율성을 입증하고, 발생 가능한 문제점을 해결해 나갈 것이다.

참 고 문 헌

[1] Makoto Yokoo, Toru Ishida and K. Kuwabara (1990), Distributed Constraint Satisfaction for DAI Problems, Proceeding of the 10th International Worshop on DAI, p20-24, Oct.

[2] Q. Y. Luo, P. G. Hendry and J. T. Buchanan (1993), Comparison of Different Approached for Distributed Constraint Satisfaction Problem, Technical Report No. RR-93-74, University of Strathclyde, Scotland, UK.

[3] JyShane Lui and Katia Sycara(1993), Distributed Constraint Satisfaction through Constraint Partition and Coordinated Reaction, Proceeding of the 12th International Workshop on DAI, May 1993.

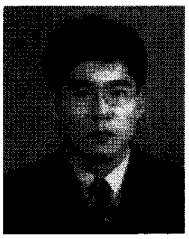
[4] JyShane Lui and Katia Sycara(1994), Distributed Problem Solving through Coordination in a Society of Agents, Proceeding of the 13th International Workshop on DAI, July.

[5] Steven Schmeier and Achim Schupeta(1996), PASHAII - Personal Assistant for Scheduling Appointments, PAAM'96, pp.523-542, Apr.

[6] Rodolfo Brancaleoni, Amedeo Cesta and Daniela D'Aloisi(1997), MASMA : A Multi-Agent System for Scheduling Meetings, PAAM'97, pp.31-49, Apr.

[7] Gerhard Weiss(1999), Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, The MIT Press.

[8] 양종윤, 조근식(1998), 스케줄링 문제 해결을 위한 지식기반 기법과 제약만족 기법의 비교연구, 한국전문가시스템학회지, 4(2), pp45-59.



정 종 진 (Jong-Jin Jung)

- 1992년 2월 : 인하대학교 전자계산공학과 (공학학사)
- 1995년 2월 : 인하대학교 전자계산공학과 (공학석사)
- 2000년 2월 : 인하대학교 전자계산공학과 (공학박사)
- 1998년 3월 ~ 2002년 8월 : 경문대학 인터넷미디어정보과 조교수
- 2002년 9월 ~ 현재 : 대진대학교 컴퓨터공학과 부교수
- 관심분야 : 전문가시스템, 지능형 에이전트, 스케줄링