

BDI 에이전트를 위한 논리기반 협상기법

이명진^o 김진상
계명대학교 컴퓨터공학과
mjlee@jinri.kmu.ac.kr, jsk@kmu.ac.kr

A Logic-based Negotiation Mechanism for BDI Agents

Myung-Jin Lee^o Jin-Sang Kim
Dept. of Computer Engineering, Keimyung University

요 약

멀티에이전트 시스템(MAS: Multi-Agent System)에서 에이전트들 사이의 목표 충돌은 일반적으로 발생 가능한 상황이고, 어떤 에이전트가 다른 에이전트에 관한 모든 지식을 가진다는 것은 불가능하기 때문에 상대방에 관한 부분적인 지식만을 가진 상황에서 목표 충돌을 해결할 수 있는 협상은 중요하다. 본 논문은 MAS에서 믿음(Belief), 소망(Desire), 의도(Intention)를 에이전트 구조의 핵심 요소로 가정하고 이러한 구조를 가지는 BDI 에이전트를 논리 프로그래밍의 입장에서 표현한다. 또한 서로 다른 목표를 가진 BDI 에이전트들이 서로 협상하여 문제를 해결하는 과정에서 발생하는 에이전트들 상호간의 목표 충돌을 해결하는 방법을 제시하며, 이 방법의 효과성을 검증하기 위하여 JAVA와 PROLOG를 결합시킨 InterPROLOG 프로그래밍 언어로 구현하여 시험한다.

1. 서론

에이전트들 사이에 존재하는 고유한 상호 의존성 때문에 자율적인 에이전트들은 서로 상호작용할 필요성이 있다. 이러한 상호작용을 다루는 기법 중에서 에이전트들이 어떤 문제 혹은 관점에 대해 충돌이 발생했을 때 서로 통신을 통해 상호 수용 가능한 일치 상황에 도달하는 과정을 협상이라 볼 수 있다. 협상의 일반적인 목적은 에이전트의 계획의 수정과 상호작용이 가능한 상황의 확인에 있다. 특히 협상은 작업과 자원의 할당, 충돌의 인식, 전자상거래 시스템에서의 가격 절충, 에이전트들 상호간의 일관성 유지를 위해 사용되고 있다.

본 논문에서는 다른 에이전트에 관한 불완전한 지식을 가지는 BDI 에이전트 구조를 사용하여 MAS에서의 협상 과정을 기술하고, BDI 에이전트들이 표현력이 높은 통신 언어를 사용하여 서로의 믿음과 의도를 변화시키면서 협상을 수행할 수 있게 하는 BDI 에이전트를 위한 협상 기법을 제시한다.

2. 협상을 위한 프로토콜들

협상 프로토콜은 협상 사이클을 시작하고 메시지에 응답할 가능성을 서술하는데, 아래와 같이 주어진 일련의 협

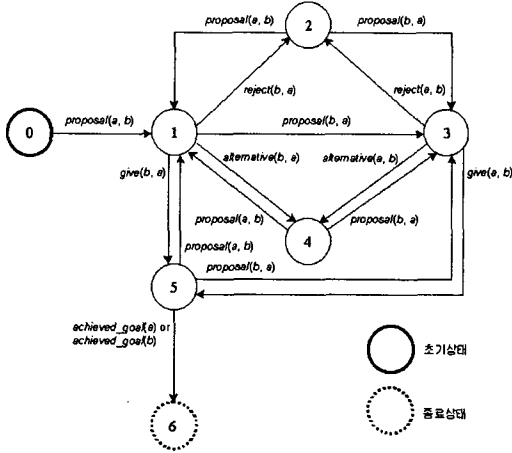
상 행위들로 정의할 수 있다.[2].

(*<negotiation primitive>*, *<message contents>*)
여기서 *negotiation primitive*는 proposal, give, inform, accept, alternative 등과 같이 에이전트의 의도를 나타내며 *message contents*는 구체적인 전송 내용을 나타낸다.

Parsons가 제안한 argumentation 시스템[3]은 협상 중인 에이전트들의 추론 과정을 서술하는 프레임워크, BDI modality, argumentation 결과 관계(ACR)를 사용한 시스템이다. Maes, Guttman, 그리고 Moukas는 전자상거래에서 소비자의 구매 행위 모델에 기초하여 중개자로서의 에이전트의 역할에 관해 연구하였는데[1], 여기서는 협상을 트랜잭션의 조건과 가격을 결정하는 단계로 간주하였다.

본 논문에서는 BDI modality와 명확한 수용거부 조건을 가지는 시스템을 이용하고, 지식베이스와 규칙을 분리하고, 소켓을 통신수단으로 하는 멀티에이전트 모델을 사용한다. 수용거부의 조건으로 자신의 목표 달성을 위해 가지고 있어야 할 것을 상대방 에이전트가 요청한 경우와 아래와 같은 반대의 개념을 정의한다. 그리고 본 논문에서 사용하는 협상 프로토콜은 <그림 1>과 같다.

정의 1. s 와 s' 을 각각 $M^1(p_1) \wedge \dots \wedge M^n(p_n)$ 와 $M^1(q_1) \wedge \dots \wedge M^m(q_m)$ 형태의 문장들이라 하고, M^i 와 M^j ($1 \leq i \leq n, 1 \leq j \leq m$)을 믿음, 소망, 혹은 의도를 표현하는 modality 라고 할 때 어떤 p_i 와 q_j 에 대해 $p_i \equiv \neg q_j$ 이면 s 는 s' 을 반대한다.



<그림 1: BDI 에이전트의 협상 프로토콜>

3. BDI 에이전트의 협상 모델

3.1 BDI 에이전트의 구조

본 논문에서 사용하는 BDI 에이전트 구조는 다음의 네 가지의 요소들을 포함한다: 지식베이스, 계획, 모니터, 통신. BDI 에이전트 구조에서 실제로 협상이 어떻게 이루어지는가를 보이기 위해서 다른 목표들과 다른 자원들을 가진 집안 일을 도와주는 두 에이전트들을 가정한다. 에이전트 $a1$ 은 그림을 걸려고 하고 그림, 나사못, 스크루드라이버, 그리고 망치를 갖고있다. $a1$ 은 $a2$ 가 못을 갖고있다고 믿고있고, 그림을 거는 방법과 거울을 거는 방법을 알고 있다. 이와 같은 $a1$ 의 지식을 논리 프로그래밍으로 표현하면 <그림 2>와 같다.

```

b(my_name(a1)).          b(have(a1, picture)).
b(have(a1, screw)).     b(have(a1, hammer)).
b(have(a1, screwdriver)). b(a2, b(have(a2, nail))).
it(do(a1, hang_picture)).
rule(b(can(W, hang_picture)),
  [b(have(X,hammer)),b(have(Y,nail)),b(have(Z,picture))]).
rule(b(holdon(W, hammer)),
  [b(have(X,hammer)),b(have(Y,nail)),b(have(Z,picture))]).
rule(b(can(W, hang_mirror)),
  [b(have(X,screwdriver)),b(have(Y,screw)),b(have(Z,mirror))]).
rule(b(do(X, Y)), [it(do(X,Y)),b(can(X,Y))]).
    
```

<그림 2: 에이전트 $a1$ 의 지식베이스>

한편 $a2$ 는 거울을 걸려고 하고, 거울과 못을 갖고있고, 그리고 못을 사용하여 거울을 거는 방법을 알고 있다고 가정한다(<그림 3>).

```

b(my_name(a2)).          b(have(a2, mirror)).
b(have(a2, nail)).       b(a1, b(have(a1, hammer))).
b(a1, b(have(a1, screwdriver))).
b(a1, b(have(a1, screw))). it(do(a2, hang_mirror)).
rule(b(can(W, hang_mirror)),
  [b(have(X,hammer)),b(have(Y,nail)),b(have(Z, mirror))]).
rule(b(holdon(W, nail)),
  [b(have(X,hammer)),b(have(Y,nail)),b(have(Z,mirror))]).
rule(b(do(X, Y)), [it(do(X,Y)),b(can(X,Y))]).
    
```

<그림 3: 에이전트 $a2$ 의 지식베이스>

3.2 BDI 에이전트의 통신언어

일반적으로 에이전트들의 협상은 메시지의 교환을 통해 달성되며, 메시지의 실제적인 교환은 참여하는 에이전트들의 개인적인 필요와 목표에 의해 유도된다. 협상에 관련된 에이전트들의 집합을 *Agents*로, 목표들의 집합을 *Goals*로, 자원들의 집합을 *Resources*로 표시하면 에이전트들을 위한 통신언어를 아래와 같이 정의할 수 있다:

정의 2. $a1, a2 \in Agents, g \in Goals, r \in Resources$ 라 할 때 통신언어 *CL*을 다음과 같이 정의한다:

- 1) $proposal(a1, a2, g, r) \in CL$
- 2) $give(a1, a2, r) \in CL$
- 3) $reject(a1, a2, g, r) \in CL$
- 4) $alternative(a1, a2, subgoals) \in CL$
- 5) $achieved_goal(a1, a2) \in CL$

3.3 BDI 에이전트의 협상

에이전트의 이론을 형식화할 때 요구되는 공리들은 일차 논리의 부분집합인 Horn 절들의 집합으로 나타내며, 추론은 SLDNF를 이용한 연역 추론으로 이루어진다. 새로운 지식의 추가 혹은 기존 지식의 변경이 발생하는 경우에 에이전트는 이전의 목표 달성을 멈추고 새롭게 변경된 지식베이스를 기초로 재계획하여 다시 목표를 달성하여야 한다. 본 논문에서 사용한 예제는 서로 다른 목표를 추구하는 두 명의 에이전트들을 가정하고 수용거부가 발생하는 경우 충돌이 생긴 것으로 간주한다.

에이전트의 계획 부분(<그림 4>)은 문제 혹은 작업을 어떻게 해결해야 할지를 결정한다.

```

plan(Task) :- Task.
plan(Task) :-
  find_agent(Task, Receiver, Tool),
  find_goal(Task, Goal), b(my_name(Sender)),
  send_message('proposal', Sender, Receiver, Goal, Tool), ...
plan(Task) :- rule(Task, Subtasks), execute1(Subtasks).
    
```

<그림 4: 에이전트의 계획>

에이전트의 모니터 부분(<그림 5>)은 작업 실행의 감시 그리고 메시지를 보낸 에이전트에게 결과를 보내주는 역할을 한다.

```
proposal(Sender, Goal, Tool) :-
    defeat(Tool), search_alternative(Goal, Subgoals), ...
proposal(Sender, Goal, Tool) :-
    defeat(Tool), it(do(X, Y)), call(solve(b(do(X, Y))))), !, ...
proposal(Sender, Goal, Tool) :- not(defeat(Tool)), ...
solve(Task) :- plan(Task), !.
solve(Task) :- write('Cannot do '), writeln(Task).
```

<그림 5: 에이전트의 모니터>

한편 에이전트의 통신 부분(<그림 6>)은 통신을 다루기 위해 소켓을 이용하여 다른 에이전트에게 메시지를 전송하고 수신된 메시지를 모니터 부분으로 보내는 역할을 한다.

```
send_message(Type, Sender, Receiver, Goal, Tool) :- ...,
    socket_send(Sock, Message, _).
wait_reply(Sock) :- ...,
    socket_rcv(Sock, Message, _), ...,
    proposal(Sender, Goal, Tool), ...,
    alternative(Sender, Receiver, Goal, Tool), ...,
    give(Sender, '_', Tool), ...,
    achieved_goal(Sender, '_', '_'), ...,
    write('Message type mismatched: ').
```

<그림 6: 에이전트의 통신>

수신한 메시지를 어떻게 해석하고 송신할 메시지를 어떻게 만들어낼 것인가를 결정하기 위해서 에이전트가 사용하는 수단이 수용거부 조건이다.

정의 3. 에이전트 $a1$ 으로부터 메시지를 받은 에이전트 $a2$ 를 가정한다. 통신언어가 주어지면 에이전트 $a2$ 의 메시지의 해석과 생성을 다음과 같이 정의할 수 있다:

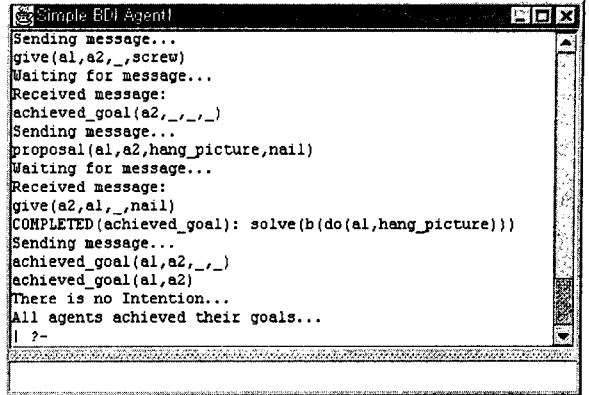
- 1) 수용거부 조건이 만족하고 목표를 달성할 대안이 있는 경우
 $G(\text{proposal}(a1, \text{goal}, \text{tool})) = \text{alternative}(a1, \text{goal}, \text{subgoals})$
- 2) 수용거부 조건이 만족하고 목표를 달성할 대안이 없는 경우
 $G(\text{proposal}(a1, \text{goal}, \text{tool})) = \text{reject}(a2, a1, \text{goal}', \text{tool})$ or
 $\text{make counterproposal}$
- 3) 수용거부 조건이 만족하지 않는 경우
 $G(\text{proposal}(a1, \text{goal}, \text{tool})) = \text{give}(a1, \text{tool})$
 ...중략...

위와 같은 메시지의 해석과 생성을 포함하는 BDI 에이전트 구조에서 <그림 7> 및 <그림 8>과 같은 협상을 통해 서로 수용가능한 일치에 도달하게 되고 각자의 목표를 달성한다.

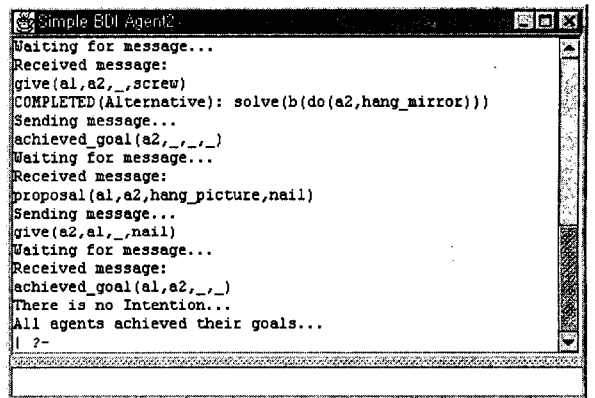
4. 비교 및 평가

Rao와 Georgeff의 BDI 모델[4]에서는 에이전트가 포함될 수 있는 가능한 세상들의 집합을 믿음으로 표현하기 때문에 세상에 관한 에이전트의 모델은 비현실적이다. 본 연구에서는 에이전트들이 상호작용하여 그들의 믿음에 대해 일관성이 없음을 알았을 때 그들의 믿음을 변경할 수 있게 하여 가능한 한 믿음을 한정지으려 하였고, 논리 프로그래밍 환경에서 작업 수행을 보다 정교하게 만

들려고 시도하였다.



<그림 7: 에이전트 a1의 협상과정>



<그림 8: 에이전트 a2의 협상과정>

5. 결론

본 논문에서는 MAS을 위한 BDI 에이전트의 믿음, 소망, 그리고 의도와 이를 바탕으로 한 협상 기법을 논리 프로그래밍의 입장에서 표현하고, 협상을 통한 목표 충돌이 어떻게 해결되는가를 보였다. 에이전트는 자신의 목표 달성에 방해되는 제안을 받은 경우에는 수용거부 조건을 검사해야 하는데, 본 논문에서 사용한 반대의 개념이 이러한 조건에 적합하였다.

참고문헌

- [1] Maes, P., Guttman, R. H. and Moukas, A., Agents that Buy and Sell: Transforming Commerce as we Know It, CACM, 1999.
- [2] Muller, H. J., Negotiation Principles, Foundations of DAI, Wiley & Sons, 1996.
- [3] Parson, S., Sierra, C. and Jennings, N. R., Agents that reason and negotiate by arguing, J. of Logic and Computation 8(3), 1998.
- [4] Rao, A. S., and Georgeff M. P., BDI Agents: From Theory to Practice, ICMAS, 1995.