규칙 기반 BDI 에이전트 구조

A Rule-Based BDI Agent Architecture

손봉기, 이건명
충북대학교 전기전자컴퓨터공학부

Bong Ki Sohn, Keon Myung Lee
School of Electric and Computer Engineering, Chungbuk National University, Korea

요 약

이 논문에서는 규칙 기반 시스템의 문제에 대한 뛰어난 표현력과 빠른 추론 등의 장점을 BDI 에이전트 구조에 적절히 반영할 수 있는 규칙 기반 BDI 에이전트 구조를 제안한다. 제안하는 구조에서는 에이전트의 능력과 계획을 이해하기 쉬운 if-then 규칙으로 기술하고, 에이전트 상태를 믿음,목적,의도 집합으로 표현하여 이를 기반으로 어떤 규칙을 실행할 것인가를 결정한다. 절차적 지식에 해당되는 규칙계획의 실행을 독립적인 규칙 엔진이 담당하기 때문에 진행 중인 작업의 컨텍스트를 유지할 수 있고, 컨텍스트가 다른 여러 작업을 동시에 처리할 수 있다. 또한, STRIPS 연산자로 자연스럽게 변환 가능한 규칙을 이용하여 계획생성이 가능하고, 생성된 계획을 규칙으로 추가하여 점진적으로 에이전트 능력을 향상시킬 수 있다. 제안하는 에이전트 구조는 규칙 기반의 BDI 모델을 따르기 때문에 동적인 환경에서 반응성과 목표 지향성을 충족할 뿐만 아니라 에이전트의 지식 표현과 구축 및 제어 구조가 간단한 에이전트 구축이 가능하다.

1. 서 론

에이전트는 어떤 환경 내에서 작업하는 개체로서 환경 변화에 대해 적절한 시간 내에 반응하거나 내부 지식을 바탕으로 추론하여 수행할 행동을 결정하고 실행하여 환경에 영향을 미치며, 사용자 및 다른 에이전트와 상호 작용하는 소프트웨어라 할 수 있다[1,2].

에이전트 구조(agent architecture)는 이러한 에이전트를 구현하기 위한 특별한 방법론으로 이상적인 에이전트 구현을 위한 다양한 연구가 있었다[3]. 숙고형 구조(deliberative architecture)는 에이전트 자신과 환경에 대한 명시적인 기호모델(symbolic model)을 바탕으로 논리적 추론을 전개하여 현재 자신이 수행해야될 행동을 결정하는 구조이다. 반면에, 반응형 구조(reactive architecture)는 기호모델 대신에 환경으로부터 센서를 통해 감지한 신호를 처리하여 즉시 반응할 수 있는 구조이다. 그러나 숙고형 에이전트는 기호모델에 기초하기 때문에 지속적으로 변화는 환경에 적절하게 반응하기 어렵고, 반응형 에이전트는 구현하기 쉽지만 단순한 지능만을 나타낼 수 있다. 혼합형 구조(hybrid architecture)는 효과적인 에이전트 구축을 위해 반응형 구조와 숙고형 구조를 결합한 구조인데, 가장 대표적인 것이 BDI 모델에 의해 실행할 행동을 결정하는 BDI 에이전트 구조이다[3].

BDI 에이전트 구조는 내부 상태를 믿음(beliefs), 소망(desires), 의도(intentions), 계획(plans)으로 표현하고 이를 바탕으로 에이전트가 어떤 행동을 실행할 것인가를 추론한다. BDI 에이전트에 대한 연구는 인간의 실제 추론에 대한 이론인 BDI 모델이 이론적 토대를 제공하고, PRS[9]와 같은 성공적으로 구현된 많은 시스템과 다양한 응용 어플리케이션 등으로 인해 활발히 진행되고 있다[4]. 기존의 BDI 에이전트는 미리 정의된 계획에 의해 작업을 수행하기 때문에 작업 진행의 컨텍스트(context)를 유지할 수 있지만, 조건 분기가 많은 계획은 표현이 복잡하고 절차가 변경되는 경우 수정이 어렵다. 또한, 자동 계획 생성이나 학습을 통해 에이전트의 능력을 점진적으로 향상시키기 어렵다.

에이전트의 행동과 지식을 규칙으로 표현하고 이를 바탕으로 추론하는 규칙 기반 시스템(rule-based system)은 계획과 같은 절차적 지식을 표현하기 어렵고, 진행되고 있는 작업의 컨텍스트를 유지하기 어려운 단점이 있다. 그러나 이해하기 쉬운 if-then 규칙으로 문제를 기술할 수 있기 때문에 표현력이 뛰어나고, 에이전트의 목적, 습관, 선호도 등을 규칙에 반영할 수 있는 장점이 있다[5]. 또한, 규칙 집합을 정의하고 적재함으로써 에이전트 구축이 가능하고, 전반향(forward chaining) 추론과 같이 매우 빠른 추론 방법을 지원하기 때문은 문제에 대한 빠른 인식과 응답이 요구되는 실시간 감시나 원격제어 시스템 등에도 사용될 수 있다.

이 논문에서는 규칙 기반 시스템의 뛰어난 표현력과 빠른 추론의 장점을 BDI 에이전트에 결합한 규칙 기반 BDI 에이전트 구조를 제안한다. 제안하는 구조는 에이전트의 능력과 규칙계획(ruleplan)을 if-then 규칙으로 기술하고, BDI 모델에 의해 추론하여 적절한 규칙을 실행한다. 절차적 지식에 해당되는 계획규칙의

본 연구는 첨단정보기술연구센터(AITrc)를 통해서 과학재단 지원으로 수행된 것임.

실행은 새로운 규칙 엔진(rule-engine)이 담당하여 진행 중인 작업의 컨텍스트를 유지하고, 컨텍스트가 다른 여러 작업을 동시에 처리할 수 있다. 또한 자동계획생성을 통해 생성된 계획을 규칙으로 변환함으로써 에이전트의 기능을 점진적으로 향상시킨다.

2. 규칙 기반 시스템의 BDI 개념

인간의 실제 추론에 대한 이론인 BDI(Belief-Desire-Intention) 모델은 Bratman에 의해 창안되었고, 이를 기반으로 많은 시스템들이 개발되었다[4]. 이러한 시스템에서는 소망과 의도의 추상적 개념을 목적(goal)과 계획(plan)의 구체적인 개념으로 사용하였다. 이 장에서는 제안하는 규칙 기반 BDI 에이전트 구조에서의 BDI 개념 표현 방법을 설명한다.

믿음은 에이전트 내부 상태와 환경에 대한 정보를 나타내는 것으로 일차술어논리의 정형식(well formed formula)으로 사실(facts)을 표현한다. 예를들어, "우수고객의 할인율이 10%이다."와 같은 사실은 (discount-rate 10%)로 표현할 수 있다. 사실을 표현할 때 AND나 OR 연산자를 포함하는 복합식과 변수를 포함하는 리터럴(literal)은 허용하지 않는다.

목적은 에이전트의 동기를 나타내는 것으로 BDI 구조의 중요한 개념이다. 목적은 달성해야할 상태나 규칙 및 서비스를 명시적으로 표현 것으로 에이전트가 목표 지향적으로 동작하게 한다. 목적은 정형식으로 표현되고, achive, maintain, perform, utility-maximize로 분류한다.

achive는 목적 달성 방법을 명시하지 않고 에이전트가 도달해야할 특정 상태를 기술한다. 예를 들어, " 주문한 제품 A를 고객 B에게 배송하라." 라고 하는 목적은 (achieve (delivered A B))로 표현할 수 있다. 이 목적은 기차,비행기,배 등의 교통 수단을 통해 달성할 수 있는 여러 가지 방법이 있다. 이 목적을 달성할 수 없다면 자동계획생성(planning) 방법을 적용하여 연속된 규칙들로 가능한 지를 검사한다.

maintain은 에이전트가 지속해서 유지해야할 상태를 기술한 목적이다. 예를 들어, "제품 A의 재고율을 30%으로 유지하라."라고 하는 목적은 (maintain (stock-rate keep-charge-over(30))으로 표현할 수 있다. 이러한 목적을 달성하기 위해서는 감시 조건 (batter charge-state-under(30))을 에이전트에 등록하여 이 조건이 위배될 때마다 대응되는 규칙을 실행하여 달성한다.

perform은 특정 규칙이나 외부에 공개된 서비스 실행을 기술한 목적으로, 기술된 규칙이나 서비스에 대응되는 규칙을 직접적으로 실행한다.

utility-maximize는 특정 유틸리티 함수를 최대화하는 규칙 실행을 기술한 목적이다. 예를 들어, "목적지까지 가장 빠르고 안전하게 이동하라"라고 하는 목적은 "가장 빠르고"와 "가장 안전하게"라는 서로 상충되는 목적이 있는 경우이다. 이런 경우는 규칙의 우선순위에 의해 실행할 규칙을 선택하는 것보다는 상충되는 목적을 적절히 절충(trade-off)할 수 있는 유틸리티 함수를 이용해 시뮬레이션한 후 가장 좋은 결과를 갖는 규칙을 실행하는 것이 타당하다.

의도(intention) 집합은 에이전트가 실행하기 위해 선택한 규칙들로 구성되지만, 반드시 실행되어야 하는 것이 아니라 환경에 변화에 따라 취소될 수도 있다. 제안하는 에이전트가 여러 작업을 동시에 처리하고, 여러 에이전트와 동시에 대화할 수 있기 위해서 세션(session) 개념을 이용한다. 세션은 진행되고 있는 에이전트 행위에 대한 컨텍스트를 유지하기 위한 것으로 다른 에이전트와의 대화(conversation), 새로운 목적, 기존의 세션과 관계없는 이벤트에 대해 세션이 할당된다. 모든 이벤트는 세션 번호가 부여되는데, 이를 이용해 컨텍스트를 유지한다. 그림 4는 에이전트 해석기와 의도 집합을 나타낸 것이다. 에이전트는 의도 집합 중에서 실행할 규칙을 다음과 같은 방법으로 선택한다. 먼저 각 세션 별로 우선순위가 가장 높은 하나의 규칙을 선택한다. 그런 다음 이들 중 우선순위가 가장 높은 규칙을 선택하여 준비 리스트(ready list)에 포함된 규칙의 우선순위와 비교하여 높으면 교체하고, 같으면 추가하고 낮으면 세션 어젠더에 그대로 둔다. 준비 리스트에 포함된 규칙들은 RR(round robin)방식에 의해 차례대로 실행된다. 실행되는 규칙에서 계획규칙을 실행하는 경우는 독립적인 규칙엔진이 생성되어 실행을 담당하는데, 이 때 생성되는 의도가 계획규칙 어젠더이다.

계획(plan)은 에이전트의 숙고적인 구성 요소로서 기존의 BDI 에이전트들은 미리 정의된 규칙을 사용한다. 계획은 계획의 활성 조건, 문맥, 효과, 목적, 몸체 부분으로 구성되고, 계획의 활성 조건이 만족되면 목적 달성을 위한 절차인 계획의 몸체 부분이 실행된다.

제안하는 에이전트 구조에서는 계획을 if-then 규칙과 실제 절차적 지식인 규칙 계획(ruleplan)으로 분리하여 표현한다. 그림 1은 규칙계획을 나타낸 것으로 규칙계획은 실행할 타스크들의 집합과 이들의 제어 흐름을 기술한 것으로 규칙계획 라이브러리에 저장된다. product-deliver 규칙계획의 실행은 타스크 A를 실행하고 조건에 따라 다음 타스크 B, C 중 하나를 실행한다. 타스크 A의 실행은 규칙 R1, R2를 대상으로 추론하여 규칙계획 어젠더에 더 이상 실행할 규칙이 없을 때까지 지속하면 완료된다. 이러한 규칙계획은 단순히 새로운 규칙을 추가,삭제하여 계획을 쉽게 변경하고 복잡한 계획을 간단히 표현할 수 있다. 그림 2는 규칙계획을 if-then 규칙의 결론부(action part)에서 실행하는 규칙의 예이다. 규칙은 규칙의 소속 규칙집합, 규칙 이름, 우선순위, 조건부, 결론부 및 효과(effect)부분으로 구성된다. 효과 부분은 자동계획생성을 수행할 때 STRIPS 연산자로 사용하기 위해 기술한다.

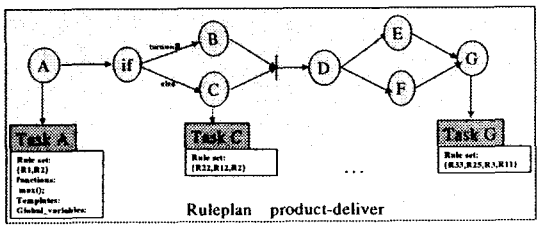

그림 1. 규칙 계획의 예

제안하는 에이전트 구조는 에이전트 내에서 발생하는 모든 것이 이벤트로 표현되는 이벤트 기반 시스템(event-based system)이다. goal 이벤트는 목표의 생성과 완료시 발생하고, message 이벤트는 메시지 송.수신할 때 발생된다. internal 이벤트는 믿음베이스의 사실 변경, 타임아웃, 예외 등에 대해 발생된다. 이벤트의 종류에 따라 에이전트 해석기의 추론 대상 규칙 집합이 결정되어 빠른 추론이 가능하다.

```
(Deliver ::product-deliver (99)
  IF (deliver-distance ok)(deliver-product TV)
  THEN (call ruleplan-executer(product-deliver))
  EFFECTS (deliver-man out-of-office)
```

그림 2. 규칙계획 호출 규칙

3. 규칙 기반 BDI 에이전트 구조

제안하는 규칙 기반의 BDI 에이전트 구조는 그림3과 같다. 에이전트의 상태를 나타내는 믿음베이스, 목적, 의도, 규칙계획 등의 자료 집합과 이들을 바탕으로 추론하여 적절한 행동을 수행하는 에이전트 해석기, 규칙의 결론부에서 실행 가능한 부시스템(subsystem) 및 믿음베이스를 감시하여 이벤트를 발생시키는 믿음 감시기로 구성된다.

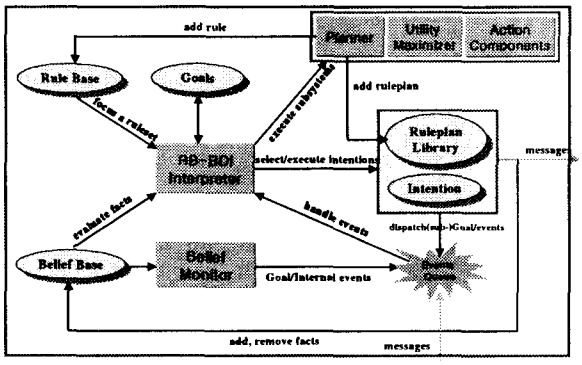

그림 3. 규칙 기반 BDI 에이전트 구조

규칙베이스(rule base)는 에이전트가 어떤 상황에서 어떤 행동을 할 것인가를 기술한 if-then 규칙들로 이루어지며, 그림 4와 같이 이벤트와 타스크에 따라 기능별로 모듈화되어 있다. 믿음베이스에 대한 사실의 추가,삭제는 이벤트나 목적을 처리하는 규칙 실행을 통해 이루어진다. 외부 메시지나 규칙 실행에 의해 주어지는 목적은 목적이 달성될 때까지 유지된다. 의도 집합은 에이전트가 실행할 규칙들로 구성된다.

에이전트 해석기(RB-BDI interpreter)는 규칙 기반 에이전트 구조에서 가장 핵심적인 역할을 한다. 그림 4와 같이 에이전트 해석기는 이벤트에 대한 추론 대상 규칙 집합을 결정하고, 의도 집합을 관리하는 추론 제어기(Inference Controller)와 전방향 추론을 수행하는 규칙 엔진으로 구성된다. 규칙 엔진은 선택된 규칙 집합과 믿음베이스의 사실에 대해 패턴매칭하고 만족되는 규칙들을 추론 제어기로 전송한다. 추론 제어기가 실행할 규칙을 결정하면 규칙엔진이 이를 실행한다.

규칙은 특수한 목적의 부시스템을 실행할 수 있다. 계획기(planner)는 achieve 목적을 달성하기 위해 자동 계획생성을 지원한다. 규칙에 대응하는 STRIPS 연산자를 사용하는 Graphplan 계획기[6]는 계획을 생성하고, 이를 규칙계획 및 if-then 규칙으로 변환하여 규칙 집합과 규칙계획 라이브러리에 추가하여 에이전트의 기능을 점진적으로 향상시킨다. 유틸리티 최적화기(utility-maximizer)는 utility-maximize 목적 달성에 사용되는 부시스템이다. 유틸리티 최적화기는 에이전트가 작업 처리하는데 필요한 유틸리티 함수를 이용하여 현재 상태에서 최적화된 행동을 선택할 수 있게 한다. 행위 구성요소(action component)는 에이전트 내부 함수에 해당된다.

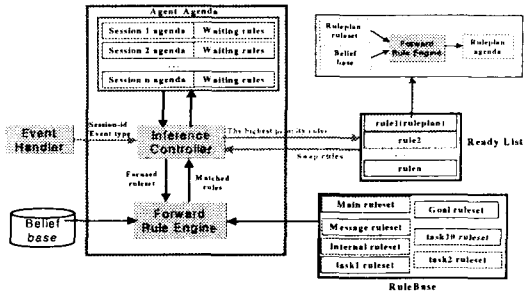

그림 4. 에이전트 해석기 구조

믿음 감시기(belief monitor)는 믿음 베이스를 감시하여 믿음베이스에 사실의 추가.삭제될 때나 maintain 목적에 대한 조건을 검사하여 이벤트를 발생하는 역할을 한다.

4. RB-BDI 에이전트 해석기 알고리즘

규칙 기반 BDI 에이전트 구조의 핵심적인 역할을 담당하는 에이전트 해석기의 작업 처리 알고리즘은 다음과 같다.

```
RB-BDI Interpreter
 Initialize-State();
Repeat
  an-event := dispatch(Event-Queue);
  focus-ruleset :=interprete(an-event);
  rules :=match(Beliefbase, focus-ruleset);
  update-agent-agenda(rules);
  agenda-rules:=resolve(agent-agenda);
  update-ready-list(agenda-rules);
  a rule := select(ready-list);
  execute(a rule);
  drop-successful-attitudes();
  drop-impossible-attitudes();
End Repeat
```

에이전트 해석기는 지속적으로 외부로부터의 메시지를 수신하면서 위와 같은 사이클을 반복하여 작업처리를 수행한다. 먼저 이벤트를 디스패치하여 추론 대상 규칙집합을 선택하여 믿음베이스와 패턴 매칭을 수행한다. 만족된 규칙들을 해당 세션 어젠더에 추가하고, 에이전트 어젠더에서 우선순위가 가장 높은 실행할 규칙들을 선택하여 준비 리스트를 갱신한다. 준비 리스트에 포함된 규칙들 중 하나를 선택하여 실행한다. 규칙의 결론부에서 부시스템을 실행하는 경우는 어느 정도의 실행 시간이 필요하기 때문에 규칙이 중지되어 부시스템의 완료 이벤트를 대기하고, 다른 규칙에 실행 제어권을 넘긴다. 계획규칙 실행은 추론 엔진이 한 사이클에 완료할 수 없기 때문에 다음과 같은 프로시져로 처리한다.

```
ruleplan_executer(rule_engine, ruleplan)
   task_list:=rule_engine.loading(ruleplan);
   Repeat until empty(task_list)
     a-task := dispatch(task_list);
     focus-ruleset :=focus(a-task);
     Repeat until empty(ruleplan-agenda)
       rules :=match(Beliefbase, focus-ruleset);
```

add-ruleplan-agenda(rules);
rule:=resolve(ruleplan-agenda);
execute(a rule);
End-Repeat
End-Repeat

준비 리스트의 규칙이 규칙계획을 실행할 경우에는 새로운 규칙 엔진을 생성하여 이를 실행하고, 이미 실행되고 있는 규칙계획에서 또 다시 규칙계획을 호출하는 것이면 기존의 규칙 엔진이 실행한다. 규칙 계획의 실행은 먼저 규칙계획에 대한 정보를 규칙 엔진에 적재하여 제어 정보인 task-list를 구하고 이를 기반으로 진행한다. task-list의 모든 타스크가 완료되면 그 규칙계획의 실행이 완료된다.

5. 구 현

제안하는 에이전트 구조는 JADE 에이전트 구조를 확장하여 구현 가능하다. JADE[7]는 멀티 에이전트 시스템 응용 개발을 지원하는 에이전트 플랫폼 미들웨어로써 FIPA 에이전트 표준을 따른다. JADE는 FIPA-ACL에 의한 통신 기반구조와 분산 멀티에이전트 시스템을 지원한다. JADE 에이전트는 JADE Agent 클래스와 필요할 때마다 행동을 동적으로 생성할 수 있는 JADE Behaviour들로 구성된다. 에이전트는 여러 Behaviour를 가질 수 있고, 이들은 RR 방식에 의해 비선점적(nonpreemptive)으로 처리된다.

제안하는 에이전트 구조는 JADE Agent 클래스 상속받아 확장하고 특정 기능을 수행하는 Behaviour들을 구현하여 추가한다. 그림 5는 제안한 에이전트의 실행 모델로서, 하나의 에이전트는 timer, message handler, RR-schdeuler, RB-BDI interpreter, Belief Monitor의 JADE Behaviour를 기본적으로 가지게 된다. 이들 Behaviour는 에이전트 내부의 자료 구조 집합을 접근하면서 동시에 실행된다. Message Handler Behaviour는 다른 에이전트로부터 ACL 메시지 수신에 대한 message 이벤트를 발생하고, Timer Behaviour는 timetable에서 처리할 시간에 도달한 이벤트를 이벤트 큐에 추가한다. RB-BDI interpreter behvaiour는 내부적으로 JESS[8] 규칙 엔진을 포함하고 있어 전방향 추론에 의해 규칙을 실행한다. RR-scheduler Behaviour는 준비 리스트에서 실행할 하나의 규칙을 RR 방식으로 선택한다. Belief Monitor Behaviour는 믿음베이스를 감시하여 사실의 추가.삭제 및 Maintain 목적의 조건을 체크하여 이에 해당되는 이벤트를 발생한다.

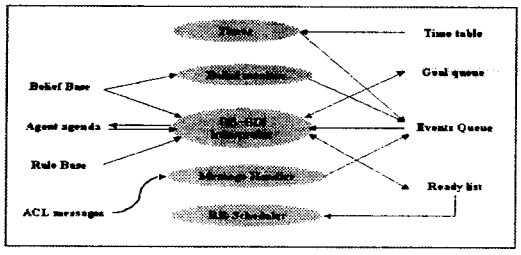

그림 5. 규칙 기반 BDI 에이전트 실행 모델

6. 결 론

BDI 에이전트 구조는 확고한 이론적 토대와 성공적인 구현 사례 및 다양한 응용 어플리케이션으로 인해 많은 연구가 있었다. 기존의 BDI 에이전트 구조는 미리 정의된 계획에 의해 작업을 수행하기 때문에 작업 진행의 컨텍스트를 유지할 수 있는 장점이 있지만, 조건 분기가 많은 계획은 표현이 복잡하고 절차가 변경되는 경우 수정이 어렵다. 또한, 자동 계획 생성이나 학습을 통해 에이전트의 능력을 점진적으로 향상시키기 어렵다.

이 논문에서는 규칙 기반 시스템의 뛰어난 표현력과 빠른 추론의 장점을 BDI 에이전트에 결합한 규칙 기반 BDI 에이전트 구조를 제안하였다. 제안하는 구조는 에이전트의 능력과 규칙계획을 if-then 규칙으로 기술하고, BDI 모델에 의해 추론하여 적절한 규칙을 실행한다. 절차적 지식에 해당되는 규칙집합의 실행은 새로운 규칙 엔진이 담당하여 진행 중인 작업의 컨텍스트를 유지하고, 컨텍스트가 다른 여러 작업을 동시에 처리할 수 있다. 또한, 제안하는 에이전트 구조는 JADE 에이전트 구조와 JESS 규칙 엔진을 결합하여 구현하는 방법을 제시하였다.

제안하는 에이전트 구조는 규칙 기반의 BDI 모델을 따르기 때문에 동적인 환경에서 반응성과 목표 지향성을 충족할 뿐만 아니라 에이전트의 지식 표현과 구축 및 제어 구조가 간단한 에이전트 구축이 가능하다.

참고 문헌

[1] S. R. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice-Hall, pp. 44-45, 1995.
[2] N. Jennings, K. Sycara, M. Wooldridge, Roadmap of Agent Research and Development, Autonomous Agents and Multi-Agent Systems, vol. 1, pp. 7-38, 1998.
[3] M. Wooldridge, N.R. Jennings, Intelligent Agents: Theory and Practice, The Knowledge Engineering Review, Vol. 10(2), pp. 115-152, 1995.
[4] A. Rao, M. Georgeff, BDI agents: From theory to practice, In Proceedings of the First International Conference on Multi-Agent System-ICMAS95, San Francisco, USA, 1995.
[5] E. P. Katz, A Multiple Rule Engine-Based Agent Control Architecture, *IEEE 6th International Conference on Intelligent Engineering Systems*, May 2002.
[6] A.Blum, M.First, Fast planning through planning graph analysis, Artificial Intelligence, Vol. 90(1-2), pp. 281-300, 1997.
[7] F. Bellifemine, G. Caire, A.Poggi, G.Rimassa, JADE-A FIPA-Compliant Agent Framework, 4th International Conference and Exhibition on The Practical Application of Intelligent Agent and Multi-Agents(London, UK, April 1999), pp. 97-108, 1997.
[8] Fridman-Hill, E.J, Jess, the Java Expert System Shell, Sandia National Laboratories, http://herzberg.ca.sandia.gov/jess/.
[9] M.P. Georgeff, A.L. Lansky, Reactive reasoning and planning, In Proceedings of AAAI-87, pp. 677-682, 1987.