분산 제어 구조내의 로봇 작업 계획

김현식[0] 신행철 김만수 김인철
경기대학교 전자계산학과
{advance7, zest133, i3rew, kic}@kyonggi.ac.kr

Robot Task Planning within a Distributed Control Framework

Hyun-Sik Kim[0], Hang-Cheol Shin, Man-Soo Kim, In-Cheol Kim
Department of Computer Science, Kyonggi University

요 약

본 논문에서는 동작 모델과 작업 목표에 따라 지능 로봇 시스템의 작업 순서를 결정하는 작업 계획기의 설계와 데모시스템에 대해 설명한다. 블랙보드 중심의 분산 제어 구조에서 하나의 독립적인 지식원으로 동작하는 작업 계획기는 작업 관리기의 요청이 있을 때마다 지식베이스로부터 동작 모델과 월드 상태 정보를 제공받아 작업 목표 달성을 위한 작업 계획을 생성한다. 그리고 이렇게 생성된 작업 계획은 오류로 인해 재 계획이 필요할 때까지 작업 관리기를 통해 실행된다. 본 연구의 작업 계획기는 효율적인 작업 계획 생성을 위해 지역 상태공간 탐색법의 하나인 EHC+ 탐색법과 계획그래프-기반의 휴우리스틱 계산법을 적용한다. 본 논문에서는 작업 계획기의 효율성과 블랙보드와의 연계성을 시험하기 위한 데모 시스템에 대해 자세히 설명한다. 이를 통해 지식베이스, 작업 관리기, 컴포넌트 서비스기 등 로봇 제어 구조내의 다른 지식원들과의 인터페이스를 위한 메시지 설계도 소개한다.

1. 서론

목표 달성을 위해 스스로 작업 순서를 계획할 수 있는 작업 계획 능력은 가정 환경에서 인간과 더불어 활동하는 지능형 서비스 로봇들에게는 필수적으로 요구되는 능력이다. 본 논문에서는 지능 로봇 시스템 제어를 위한 작업 계획기의 설계와 데모시스템 구현에 대해 설명한다. 블랙보드 중심의 분산 제어 구조에서 하나의 독립적인 지식원(Knowledge Source, KS)으로 동작하는 작업 계획기는 작업 관리기의 요청이 있을 때마다 지식베이스로부터 동작 모델과 월드 상태 정보를 제공받아 작업 목표 달성을 위한 작업 계획을 생성한다. 그리고 이렇게 생성된 작업 계획은 오류로 인해 재 계획이 필요할 때까지 작업 관리기를 통해 실행된다. 본 연구의 작업 계획기는 효율적인 작업 계획 생성을 위해 지역 상태공간 탐색법 (local state-space search)의 하나인 EHC+ 탐색법과 계획그래프(planning graph)-기반의 휴우리스틱(heuristic) 계산법을 적용한다. 본 논문에서는 작업 계획기의 효율성과 블랙보드와의 연계성을 시험하기 위한 데모시스템에 대해 자세히 설명한다.

2. 블랙보드와의 연동

본 연구의 작업 계획기는 작업 관리기, 컴포넌트 서비스기, 인간-로봇 상호작용 관리기 등과 더불어 총 3계층(반응층(reactive layer), 순차층(sequencing layer), 숙고층(deliberative layer))으로 구성되는 로봇 제어 구조의 맨 상위층인 숙고층을 형성한다. 특히 숙고층은 하위 계층을 구성하는 컴포넌트들의 기능을 바탕으로 지식수준의 추론과 판단을 담당하는 독립적인 지식원(KS)들로 구성된다. 그리고 이들은 하나의 전역적 공유 작업 메모리(global shared working memory) 역할을 하는 블랙보드를 통해 협력적으로 문제해결에 참여한다. 블랙보드는 작업 계획기를 포함한 다양한 지식원들 사이에 사건(event) 기반의 상호작용을 지원하기 위해 사건 처리기(event handler) 등록과 사건 발생 통보를 위한 API를 제공한다. subscribe와 unsubscribe는 각각 사건 처리기를 등록/해제하는 메쏘드이고, 반면에 post는 사건 발생을 알리는 메쏘드이다. 따라서 실제로 기다리고 있던 사건을 한 지식원이 발생시키면, 블랙보드는 사건 처리기를 등록해둔 다른 지식원들에게 즉시 이 사건을 전달한다. 또한 블랙보드는 내부에 저장하고 있는 월드 상태 정보 관리와 연관된 API도 제공한다. assertFact, retractFact, update, match는 각각 릴레이션 형태로 표현되는 상태정보를 등록/제거/갱신/조회하는 메쏘드들이다.

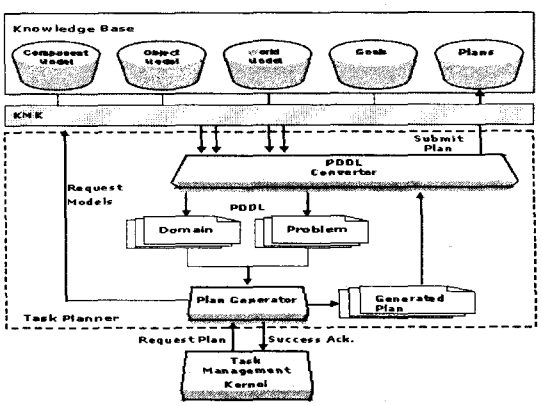

그림 1 작업 계획기와 블랙보드의 연동

[그림 1]은 이와 같은 블랙보드 중심의 분산 제어 구조 체제에서 본 연구의 작업 계획기가 블랙보드와 혹은 블랙보드를 통해 다른 지식원들과 상호 공유하게 될 지식들을 나타내고 있다. 작업 계획기는 동작 모델의 기초가 되는 컴포넌트 모델과 오브젝트 모델, 그리고 계획문제를 구성하기 위한 월드 모델과 작업 목표 등을 블랙보드를 통해 가져와서 새로운 작업 계획을 생성하여 반환하는 방식으로 블랙보드와 연동한다. 이와 같은 모든 상호작용은 앞서 소개된 블랙보드 API 메쏘드들의 호출로 구현된다.

3. 작업 계획 알고리즘

본 연구의 작업 계획기는 효율적인 작업 계획 생성을 위해 휴우리스틱에 기초한 지역 상태공간 탐색 (local state-space search)을 전개한다. 잘 알려진 A* 탐색은 탐색의 완전성(completeness)과 허용성(admissibility) 을 보장할 수 있으나, 전역적 탐색의 하나로서 복잡도가 높은 로봇 작업 계획을 위해서는 메모리 요구량이 매우 크고 탐색 시간이 길다는 단점이 있다. 따라서 본 연구에서는 휴우리스틱 계산에 기초하여 목표 도달 가능성이 낮게 평가되는 상태들에 대해서는 과감히 가지치기(pruning)를 하는 언덕오르기(Hill-Climbing, HC)와 같은 지역 탐색을 적용한다. [그림 2]는 기존의 언덕오르기 탐색의 문제점을 개선한 EHC(Enforced Hill-Climbing) 탐색을 나타낸 것으로서, 현재 상태노드 s1에서 시작하여 넓이-우선 탐색(Breadth-First Search)을 전개하여 최초로 현재 상태노드보다 낮은 휴우리스틱 값을 갖는 후손노드를 찾는다. 그리고 이 상태노드를 시작노드로 삼아 다시 이와 같은 탐색과정을 반복함으로써 목표상태를 찾아간다.

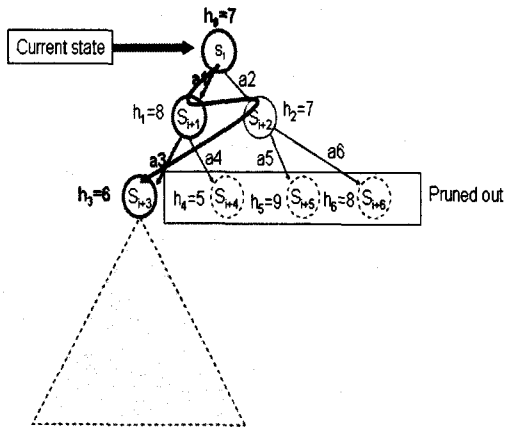

그림 2 Enforced Hill-Climbing(EHC) 탐색

이 탐색방법은 그림에서 보듯이 후손노드 s3 보다 이후에 나타나는 모든 형제노드들에 대해서는 가지치기가 이루어지고, 다음 탐색 주기에서는 바로 상태노드 s3를 시작노드로 하는 탐색트리가 펼쳐지는 현상을 보인다. 만약 상태노드 s3 이후에 더 우수한 휴우리스틱 값을 갖는 형제노드들이 있다고 하더라도 모두 무시되는 결과를 가져와서 탐색의 효율성과 계획의 최적성을 얻기 어렵다. 이와 같은 문제점을 극복하고자 본 연구에서는 [그림 3]과 같은 EHC+ 탐색법을 개발하였다. EHC+ 탐색은 기존의 EHC 탐색을 확장한 것으로서, 넓이-우선 탐색을 통해 현재 상태노드보다 휴우리스틱 값이 우수한 최초의 후손노드를 찾았으면 탐색을 남아 있는 형제노드들까지 탐색을 연장하여 그 레벨의 최우수 상태노드인 s4를 찾아 이것을 다음 탐색주기에서 시작노드로 삼는 탐색방법이다.

이 방법은 EHC 탐색에 비해 소량의 추가적인 탐색을 통해 작업 계획을 위한 전체적인 탐색의 효율성을 높일 수 있고 양질의 작업 계획을 보장할 수 있는 특징을 가지고 있다.

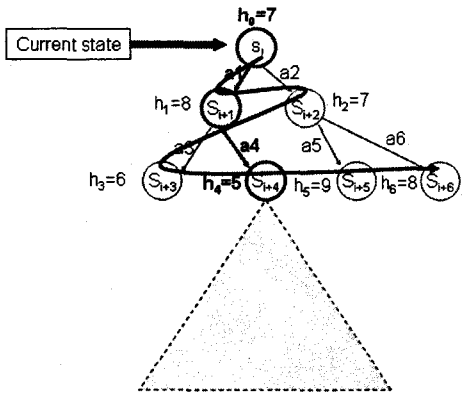

그림 3 Enforced Hill-Climbing Plus(EHC+) 탐색

또한, 본 연구의 작업 계획기에서는 탐색에 보다 효과적인 각 상태노드들의 휴우리스틱 값을 얻기 위해 계획그래프(planning graph) 기반의 휴우리스틱 계산법을 적용하였다. Graphplan 시스템에서 처음 개발된 계획그래프 표현법은 이후 많은 상태공간 계획 알고리즘과 계획기에서 휴우리스틱 계산에 널리 이용되고 있다. [그림 4]에서 보듯, 계획그래프는 일반적으로 여러 레벨로 구성되며, 각 레벨은 다시 하나의 동작층(action layer)과 사실층(proposition layer)으로 구성된다. 동작층에는 이전 레벨의 사실들에 의해 적용 전조건(precondition)이 만족되는 동작들이 놓여지고, 사실층에는 이러한 동작들의 효과들 중 추가 조건(add list)에 속하는 사실들만 놓여진다. 따라서 하나의 계획그래프는 일반적으로 원래 문제보다 간략화된 계획문제를 나타내고, 이를 이용해 [그림 4]와 같은 다양한 유형의 휴우리스틱을 계산해낼 수 있다. [그림 4]는 하나의 계획그래프를 바탕으로, 3 가지 다른 유형의 레벨 휴우리스틱들을 계산하는 방법(set-level, max, sum)을 나타내고 있다. 이들 중에 set-level와 max는 현재 상태를 출발점으로 삼아 목표 상태에 도달할 때까지 계획그래프를 펼친 다음, 그때 최대 레벨 수(maximal level number)를 현재 상태노드의 휴우리스틱 예측치로 삼는 방법이다. 반면에 sum은 각 단위 목표가 처음

달성되는 레벨들을 모두 합산하여 휴우리스틱 예측치로 삼는 방법이다.

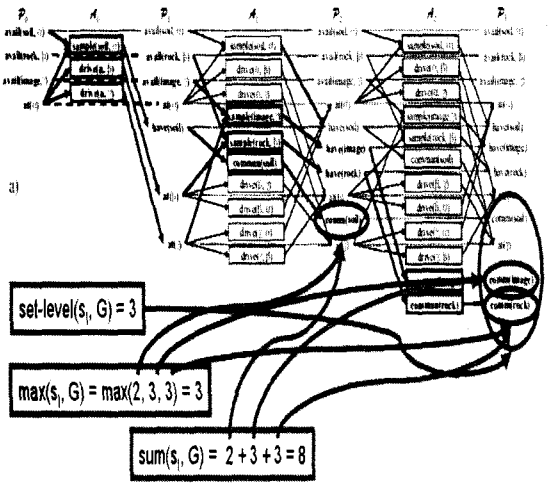

그림 4 계획그래프를 이용한 heuristic 계산

4. 작업 계획기 구조

작업 계획기는 [그림 5]와 같이 Parser, Grounding, State Manager, Heuristic, Search, Plan Synthesizer 등의 주요 구성요소들과 State Queue, Planning Graph 등의 기타 자료 구조들로 구성된다. Parser는 PDDL로 기술된 동작모델과 계획문제에 대한 구문분석을 통해 Domain, problem, Predicate, Action 등의 내부 자료구조들을 만드는 역할을 하며, Grounding은 술어논리(predicate logic) 형태의 상태정보와 동작모델을 명제논리(propositional logic)로 변환하는 역할을 한다.

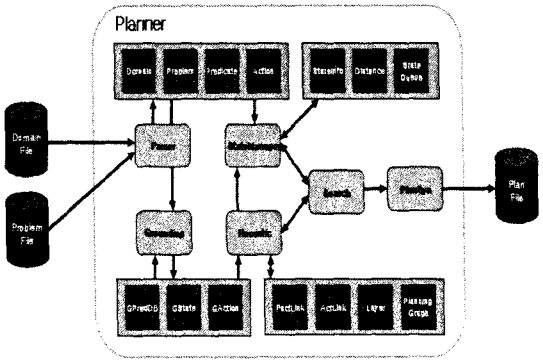

그림 5 작업 계획기의 내부 구성도

Search는 상태 공간 탐색을 통해 해 계획을 찾는 역할을 수행하며, StateManager와 Heuristic은 각각 이를 위해 탐색과정에 발생하는 상태 정보를 관리하고 이들에 대한 휴우리스틱을 계산하는 역할을 한다. 앞서 설명한대로 각 상태의 휴우리스틱은 계획그래프를 이용하여 계산된다. 마지막으로 PlanSyn은 탐색과정을 통해 구해진 해를 하나의 작업 계획으로 합성하고 변환하는 역할을 수행한다.

4. 데모 시스템

4.1 시나리오

블랙보드 중심의 분산 로봇 제어 구조에서 작업 계획기의 적용 가능성과 효율성 알아보기 위해 시뮬레이션 기반의 간단한 데모시스템을 개발하였다. 데모 시나리오는 [그림 6]과 같이 가정에서 뜻하지 않은 화재가 발생했을 때 사용자와 함께 생활하는 서비스 로봇이 스스로 작업 계획을 세워 우선 사용자에게 화재 발생 사실을 알린 다음, 귀중품을 찾아 안전한 곳으로 옮기는 지능행위를 보여주는 줄거리이다. 이 밖에도 이 시나리오는 작업 계획 후에 사용자가 위치 이동을 함으로써 로봇이 다시 재 계획을 시도해야 하는 상황과, 닫힌 문을 열거나 불을 끄는 등 작업에 필요한 새로운 기능 컴포넌트를 로봇 스스로 찾아 설치해야 하는 상황 등을 포함한다.

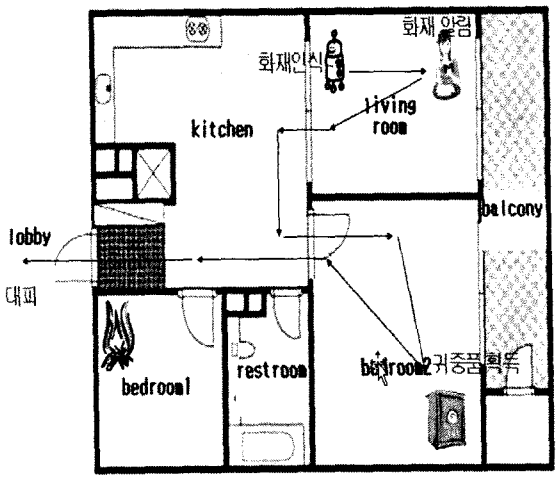

그림 6 데모 작업 환경

따라서 이 시나리오는 작업 계획기가 블랙보드를 중심으로 다른 지식원들과의 유기적인 상호작용을 통해 작업 계획에 필요한 지식을 공유하는 능력, 실행환경의 가변적 상황에 맞추어 작업을 재 계획할 수 있는 능력, 작업을 위해 어떤 새로운 기능의 컴포넌트가 필요한지를 알아내는 능력, 마지막으로 이러한 새로운 기능 컴포넌트들을 포함한 작업 계획을 생성할 수 있는 능력 등을 점검할 수 있는 기회를 제공한다.

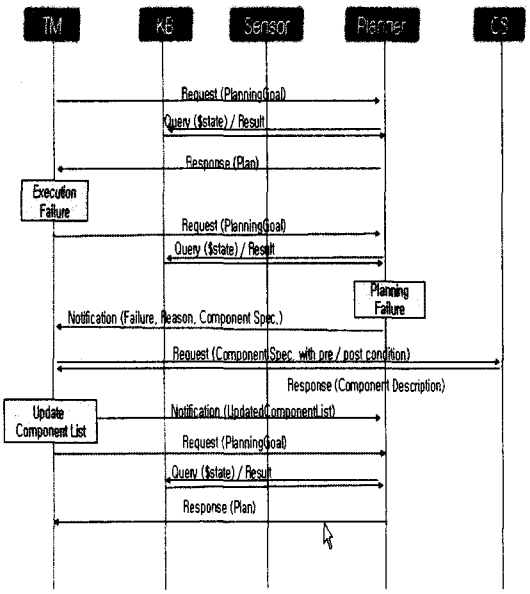

그림 7 상호작용 다이어그램

[그림 7]은 이 시나리오의 실현을 위해 필요한 작업 계획기와 관련 지식원들(예컨대, 작업 관리기(TM), 컴포넌트 서비스기(CS), 지식베이스(KB) 등)간의 상호작용을 예시한 것이다.

4.2 동작모델

작업 계획기 내부에는 이미 로봇에 설치되어 실행중인 기능 컴포넌트들을 기초로 [그림 8]과 같은 동작모델을 저장하고 있다고 가정한다. [그림 8]은 서비스 로봇의 동작들을 정의한 한 예로서, 각 방들 사이를 오가는 서비스 로봇의 자율 이동 동작을 표준 계획영역기술언어인 PDDL로 정의한 것이다. 이러한 동작모델은 대응되는 컴포넌트의 기능명세로부터 직접 자동 추출할 수 있다고 가정하였다. 동작모델 이외에 계획에 필요한 작업 목표와 상태 정보는 작업 관리기 등 다른 지식원과의 상호작용을 통해 입수한다.

```
(:action automove
        :parameters    ( ?R - robot ?R1 - room  ?R2 - room )
        :precondition  (and
                            (robot_in_room ?R ?R1)
                            (released_cashbox ?R)
                            (not (= ?R1 ?R2))
                            (connect ?R1 ?R2 )
                            (known_user_location)
                       )
        :effect        (and
                            (not (robot_in_room ?R ?R1))
                            (robot_in_room ?R ?R2)
                       )
)
```

그림 8 동작모델

4.3 TM 및 CS 와 인터페이스

화재발생을 감지하면 작업 관리자는 블랙보드를 통해[그림 9]와 같은 XML 기반의 계획 요청 메시지를 작업 계획기에 전달한다.

```
- <request-planning id="12345">
    <goal>(cashbox_in_room cashbox1 lobby)(known_about_fire
      user1)(gasrange_off gasrange1)</goal>
  </request-planning>
```

그림 9 계획 요청 메시지

```
- <result>
  - <plan>
    - <step number="1" name="goto_user">
        <parameter>robot1</parameter>
        <parameter>user1</parameter>
        <parameter>livingroom</parameter>
      </step>
    - <step number="2" name="notify_of_fire">
        <parameter>robot1</parameter>
        <parameter>user1</parameter>
        <parameter>livingroom</parameter>
        <parameter>fire1</parameter>
        <parameter>bedroom1</parameter>
      </step>
    - <step number="3" name="leave_user">
        <parameter>user1</parameter>
        <parameter>livingroom</parameter>
        <parameter>robot1</parameter>
      </step>
    - <step number="4" name="go_to_livingRoom_kitchen_door">
        <parameter>robot1</parameter>
        <parameter>kitchen_livingroom_door</parameter>
        <parameter>livingroom</parameter>
        <parameter>kitchen</parameter>
      </step>
    - <step number="5" name="go_thru_livingRoom_kitchen_door">
        <parameter>robot1</parameter>
        <parameter>kitchen_livingroom_door</parameter>
        <parameter>livingroom</parameter>
        <parameter>kitchen</parameter>
      </step>
    </plan>
  </result>
```

그림 10 계획 응답 메시지: 계획 성공의 경우

그러면 작업 계획기는 지식베이스로부터 현재 작업환경의 상태정보를 받아온 다음, 여기에 동작모델을 적용하여 작업목표 달성을 위한 작업 계획을 생성한다.

```
- <result>
    <status>failure</status>
  - <failure-description type="component_absence">
    - <precondition>
        <pred>door_closed</pred>
        <pred>connect</pred>
        <pred>robot_in_room</pred>
        <pred>robot_at_door</pred>
      </precondition>
      <postcondition />
    </failure-description>
  </result>
```

그림 11 계획 응답 메시지: 계획 실패의 경우

그리고 계획 생성에 성공하였을 경우는 [그림 10]과 같이

작업 계획을 담은 XML 메시지를, 실패하였을 경우는 [그림 11]과 같이 실패의 원인과 새로운 기능 컴포넌트들에 대한 조건식을 담은 XML 메시지를 응답으로 보내준다. 따라서 필요한 경우, 작업 관리기는 이러한 조건식을 기초로 컴포넌트 서비스기에 조건에 맞는 새로운 컴포넌트의 검색과 설치를 요청하고, 새로운 컴포넌트 설치가 성공적으로 이루어지면 해당 컴포넌트의 기능명세를 작업 계획기에 전해준다. 그러면 작업 계획기는 새로 설치된 컴포넌트 명세로부터 새로운 동작모델을 추출하여 이것을 포함한 새로운 작업 계획을 생성한다.

4.4 평가

데모시스템의 구현과 실행을 통해 작업 계획기 자체의 효율성뿐만 아니라 블랙보드에 기초한 다른 지식원들과의 연계 가능성을 확인할 수 있었다. 하지만 아직 기능 컴포넌트의 명세와 검색 등 컴포넌트 서비스기와의 협력 연구가 미흡하다는 사실과, 가변적 작업환경에 효과적으로 대처하기 위해 조건부 계획 기능(conditional planning)에 대한 연구가 필요하다는 것을 확인할 수 있었다.

5. 결론

본 논문에서는 블랙보드 중심의 분산 제어 구조에서 하나의 독립적인 지식원으로 동작하는 작업 계획기의 설계에 대해 자세히 설명하고, 데모시스템을 소개하였다. 현재 프로토타입 수준의 작업 계획기를 더욱 안정화하고 조건부 계획 기능을 추가 개발하는 방향으로 향후 연구가 진행되어야 할 것으로 판단한다.

참고 문헌

[1] Blum, A. and Furst, M., "Fast Planning through Planning Graph Analysis," *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI95)*, pp.1636-1642, 1995.

[2] Bonet, B. and Geffner, H., "Planning as Heuristic Search," *Artificial Intelligence*, Vol.129, No.1-2, pp.5-33, 1995.

[3] Bryce, D., Kambhampati S. and Smith D.E., " Planning Graph Heuristics for Belief Space Search," *Journal of Artificial Intelligence Research*, Vol.26, pp.35-99, 2006.

[4] Fox, M. and Long, D., "PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains," 2003.

[5] Hoffmann, J., " FF: The Fast-Forward Planning System," *The AI Magazine*, 2001.