디지털산업정보학회 논문지 제17권 제3호-2021년 9월
http://dx.doi.org/10.17662/ksdim.2021.17.3.027

효율적인 계획생성을 위한 그래프 기반의 혼합 휴리스틱

박 병 준*·김 완 태**·김 현 식***

Graph-based Mixed Heuristics for Effective Planning

Park Byungjoon·Kim Wantae·Kim Hyunsik

〈Abstract〉

Highly informative heuristics in AI planning can help to a more efficient search a solutions. However, in general, to obtain informative heuristics from planning problem specifications requires a lot of computational effort. To address this problem, we propose a Partial Planning Graph(PPG) and Mixed Heuristics for solving planning problems more efficiently. The PPG is an improved graph to be applied to can find a partial heuristic value for each goal condition from the relaxed planning graph which is a means to get heuristics to solve planning problems. Mixed Heuristics using PPG requires size of each graph is relatively small and less computational effort as a partial plan generated for each goal condition compared to the existing planning graph. Mixed Heuristics using PPG can find partial interactions for each goal conditions in an effective way, then consider them in order to estimate the goal state heuristics. Therefore Mixed Heuristics can not only find interactions for each goal conditions more less computational effort, but also have high accuracy of heuristics than the existing max and additive heuristics. In this paper, we present the PPG and the algorithm for computing Mixed Heuristics, and then explain analysis to accuracy and the efficiency of the Mixed Heuristics.

Key Words : Classical Planning, Delete Relaxation, Domain-independent Heuristic, Relaxed Planning Graph, Partial Planning Graph

Ⅰ. 서론

인공지능 분야에서의 계획생성(AI planning)은 전통적인 인공지능 연구에서부터 현재까지 지속적으로 연구되고 있는 주요 연구 분야 가운데 하나로 일반적인 문제해결 방법을 구축하는 것을 기본 목표로 하고 있다. 일반적으로 전통적인 계획생성(Classical Planning)은 주어진 초기상태(initial state)에서 목표로 하고 있는 상태(goal state)까지 도달하기 위한 행위(action)를 결정하여 해결책(solution)을 찾는 과정으로서 주로 지능형 에이전트와 지능형 로봇, 무인 자동차의 자율 주행, 생산시스템 또는 소프트웨어 테스트 과정과 같은 다양한 분야에 적용하기 위해 이용된다[1-3].

* 서일대학교 소프트웨어공학과 교수(제1저자)
** 서일대학교 정보통신공학과 교수(참여저자)
*** 서일대학교 소프트웨어공학과 교수(교신저자)

이러한 계획생성을 위한 과정은 일반적으로 휴리스틱을 이용하여 상태 공간상의 탐색을 통해 수행되게 되는데, 이때 해결하고자 하는 계획문제(planning problem)가 서로 다른 지식 영역(domain)에서 자동으로 계획을 생성하기 위해서는 주어진 계획문제 명세로부터 얻어진 정보를 바탕으로 상태 공간상의 탐색을 수행할 수 있어야 한다. 이때, 이와 같이 주어진 서로 다른 계획문제가 특정한 지식 영역에 귀속되지 않고 탐색을 수행하기 위해서는 영역-독립적인(domain-independent) 탐색 휴리스틱이 필요한데 이를 위해 널리 사용되고 있는 방법이 삭제 완화(delete relaxation)[4, 5]를 이용하는 완화된 계획그래프(Relaxed Planning Graph)[6-9]이다. 여기서, 완화된 계획그래프에서 이용하는 삭제 완화는 계획문제에서 추출된 상태변화를 발생시키는 각 행위의 부정적 효과(negative effect)를 모두 제거하여 행위들 간의 부정적 상호작용을 배제하는 방법을 말한다. 이러한 완화된 계획 그래프는 삭제 완화를 통해서 부정조건을 완화하고 문제를 간략화 함으로써 주어진 계획문제 명세로부터 필요한 정보를 추출하여 휴리스틱을 얻을 수 있으며 최적의 해 계획을 구하기 위한 허용 가능한 휴리스틱(admissible heuristic)[10, 11]을 제공하는 것이 가능하다. 이러한 완화된 계획그래프를 기반으로 추출되는 대표적인 휴리스틱으로는 최대 휴리스틱(max heuristic)[6], 합산 휴리스틱(additive heuristic)[12] 그리고 겹침 휴리스틱(overlap heuristic)[5] 등이 대표적이며 이 가운데 최대 휴리스틱의 경우 최적의 해를 구하기 위한 허용 가능한 휴리스틱에 해당한다.

이러한 휴리스틱은 측정치가 목표 도달 거리에 가까울수록 계획생성을 위한 탐색 과정의 효율성이 높아지며 이러한 휴리스틱이 항상 실제 계획 경로보다 짧거나 같은 경우에는 허용 가능한 휴리스틱, 즉 최적의 해 계획을 구하는 것이 가능한 휴리스틱이 된다. 하지만, 실제 거리에 가까운 휴리스틱을 구하는 과정에서 이러한 허용 가능한 휴리스틱의 특징을 유지하는 것은 매우 복잡하고 그에 따른 계산량이 너무 크기 때문에 실시간 계획생성에서 시스템 또는 사용자가 요구하는 양질의 해 계획을 구하기 위한 휴리스틱의 계산 효율성과 정확성을 높이는 데는 많은 한계가 있다. 때문에 일반적으로 허용 가능한 휴리스틱이 반드시 필요한 경우에는 계산량은 적지만 정보력이 낮은 최대 휴리스틱과 같은 매우 단순한 형태의 휴리스틱을 주로 이용하고 있으며 최적의 해 계획 생성을 보장하지는 못하지만 휴리스틱의 정보력은 상대적으로 더 높은 합산 휴리스틱이나 계산량이 크고 복잡하지만 상대적으로 정보력이 높은 겹침 휴리스틱과 같은 휴리스틱을 주로 이용하고 있다. 때문에 효율적 탐색을 통한 양질의 해 계획 도출을 보다 효과적으로 진행하기 위해서는 계산량을 줄이면서도 높은 정보력을 가지는 휴리스틱에 대한 연구가 필요하다[9, 13, 14].

본 논문에서는 보다 효과적으로 계획생성 결과를 구하기 위해서 계산에 들어가는 노력을 최소화하면서도 휴리스틱의 정보력이 보다 높은 휴리스틱 계산을 위하여 기존의 완화된 계획그래프 기반의 새로운 자료구조인 부분 계획그래프(Partial planning Graph)와 이를 바탕으로 한 혼합 휴리스틱(Mixed Heuristics) 계산법을 소개한다. 부분 계획그래프는 기존의 완화된 계획그래프의 기능을 포함하고 있으며 목표 조건들 간의 부분적인 부정적 상호작용을 파악할 수 있는 정보가 포함된 자료구조이다. 이를 바탕으로 하는 혼합 휴리스틱의 경우에는 계획그래프 확장 과정에서 동작 간 긍정적 상호작용을 전제로 한 휴리스틱 계산과정과 목표조건 달성을 위한 동작 간의 상호작용을 파악하기 위한 동작 추적 과정이 혼합된 휴리스틱 계산 방법으로 기존 휴리스틱과 달리 휴리스틱 계산 이후에 해당 휴리스틱이 허용 가능한 휴리스틱인지 허용가능하지 않은 휴리스틱인지 파악도 가능하다. 본 논문에서는 이러한 부분 계획그래프와

이를 기반으로 한 혼합 휴리스틱에 대한 소개에 이어 혼합 휴리스틱의 정확도 및 탐색의 효율성을 분석, 확인하기 위한 비교 실험 결과에 대해여 설명한다.

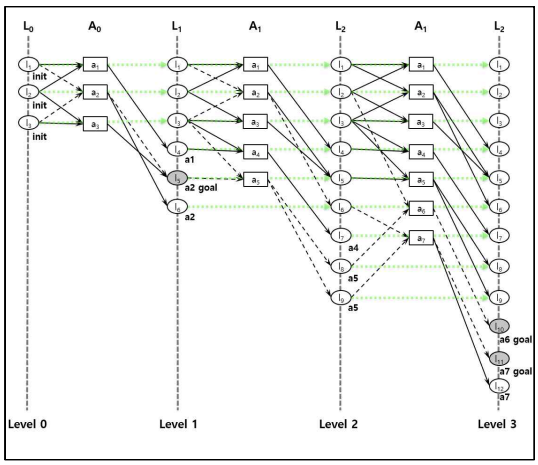
Ⅱ. 부분 계획그래프와 혼합 휴리스틱

부분 계획그래프의 기반이 되는 완화된 계획그래프는 STRIPS 기반의 Graphplan 계획기[15]에서 소개된 계획그래프에서 부정 효과를 무시하는 삭제 완화(delete relaxation)를 적용하여 초기상태에서 목표를 구성하는 각 목표조건들까지의 휴리스틱을 보다 쉽게 얻을 수 있도록 도와주는 대표적인 자료구조 가운데 하나로 전통적인 계획문제를 해결하기 위한 상태 공간상의 휴리스틱 계산과 계획 알고리즘에 널리 이용되고 있다[6-9, 15].

이러한 완화된 계획그래프는 <그림 1>에서 보는 것과 같이 초기상태 또는 동작으로 인해 변화된 현재 상태를 나타내는 상태 층(literal layer)과 이러한 상태 층에 적용 가능한 동작들로 이루어진 동작 층(action layer)이 반복적으로 펼쳐지는 형태로 구성되어 있으며, 초기상태에서 주어진 목표조건들이 모두 등장하여 만족할 때까지 상태 층과 동작 층이 교대로 반복적으로 펼쳐지며 목표를 찾는 하나의 계층화된 자료구조이다. 완화된 계획그래프는 이러한 그래프의 전개 과정에서 목표조건들을 모두 만족할 때까지 이전 상태조건과 동작을 유지하며 확장하고 앞에서 설명한 것과 같이 부정 효과가 제거된 동작들만 사용하며 동작들 간의 관계는 검사하지 않고 단지 목표조건들을 모두 만족했는지 만을 확인한다. 따라서 동작의 부정적 효과는 적용되지 않기 때문에 상태조건들로 구성된 임의의 한 상태에서 목표상태까지 도달하기 위한 최단 거리를 추정할 수 있는 정보 제공이 가능하다.

문제는 이러한 완화된 계획그래프는 상태 층과 동작 층에 상태조건들에 대한 정보와 여기에 적용 가능한 동작들에 대한 정보가 계속 유지되기는 하지만 이러한 내용을 제외한 다른 정보는 가지고 있지 않기 때문에 실제 상태조건들이 전개과정에서 어느 동작으로 인해 만족했는지 또는 이러한 상태조건이 또 다른 어떠한 관련 목표조건과 관계가 있는지 등에 대해서는 알기 어렵다. 때문에 주어진 상태에서 목표까지의 도달거리 예측치를 구하고자 할 때 동작들 간의 관계나 상호작용 등을 파악하는 데는 많은 한계를 가지고 있다.

본 논문에서는 이러한 문제를 해결하기 위해서 기존 완화된 계획그래프에 부여되는 정보와 동작 방식을 개선한 새로운 계획그래프인 부분 계획그래프와 이를 기반으로 한 혼합 휴리스틱을 제안한다. 부분 계획그래프와 이를 바탕으로 하는 혼합 휴리스틱은 계획문제 P= (s_0, G, O)가 주어졌을 때, 이를 풀기 위해 상태 공간상의 초기상태 s_0에 대해서 <그림 1>과 같이 계획그래프 생성과정을 진행한다. 이때, 목표 조건에 따라 그래프 생성과정은 <그림 2> <그림 3>에서와 같이 목표조건별로 그래프를 초기화하면서 목표 조건에 따른 그래프를 재생성하는 과정을 진행한다. 그리고 이러한 목표조건들에 따라 반복적으로 생성된 그래프를 바탕으로 휴리스틱을 계산하게 된다.



<그림 1> 완화된 계획그래프 / 동작-상태 정보

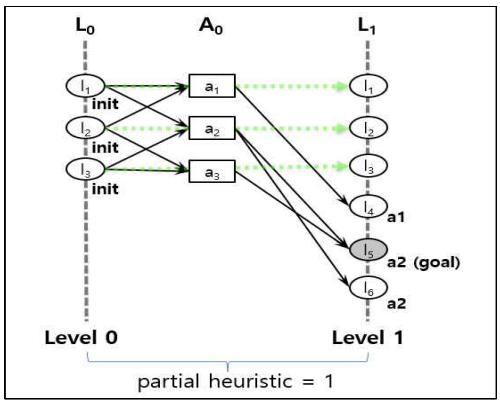
부분 계획그래프 생성 과정은 다음과 같다. 기존의 완화된 계획그래프와 달리 초기상태 s_0로부터 목표조건 가운데 하나 이상이 등장할 때까지만 계획그래프를 전개한다. 이때, 그래프 전개 과정에서는 새롭게 등장한 상태조건을 만족시키는 동작들을 확인하여 휴리스틱 계산을 위한 정보부여 과정을 진행한다. 정보부여 과정은 다음과 같다.

정보부여 과정 :

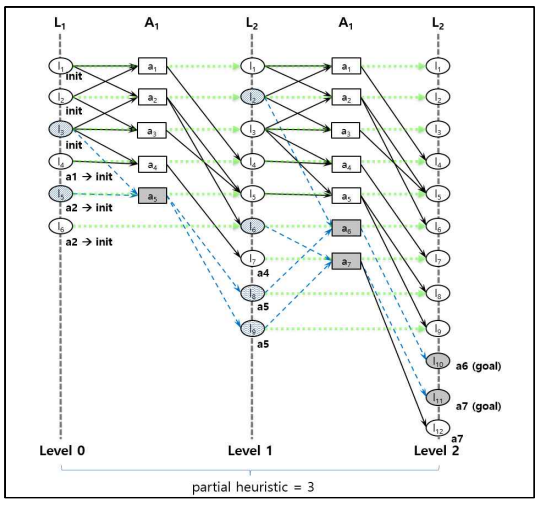
- 주어진 초기상태 층 L_0에 대해서 이를 구성하는 모든 상태조건들에 대해서 초기상태 조건임을 나타내기 위한 정보인 init 정보를 부여한다.
- <그림 2>에서와 같이 계획그래프 전개 과정에서 확장된 계층에 새롭게 등장하며 만족되는 상태조건들에 대해서는 (상태조건, 해당 동작)과 같이 해당 조건을 만족시키는 동작 정보를 부여한다. 이때, 유지 동작을 포함하여 동일한 동작 층에서 새롭게 상태조건을 만족시키는 동작이 있더라도 정보부여 과정을 통해 부여된 (상태조건, 해당 동작) 정보는 수정되지 않는다[9].

위와 같은 정보부여 과정을 포함하여 반복적으로 전개되는 계획그래프는 목표조건이 1개 이상 새롭게 만족되는 경우에 계획그래프 전개를 멈추고 확장된 상태를 초기상태로 갱신하면서 그래프가 초기화되고 다시 계획그래프 생성 과정을 진행하면서 모든 목표조건을 순서대로 만족하게 된다. 이때, 그래프 초기화 전에 해당 그래프를 바탕으로 개별 목표조건들에 대한 휴리스틱을 측정하게 되는데, 등장하는 목표조건의 수에 따라 측정 방법을 달리한다.

- 목표조건이 1개인 경우, 동작들 간에 긍정적 상호작용이 존재하는 것으로 간주하고 최대 휴리스틱을 적용한다. 이를 위해서 <그림 2>에서 보는 것과 같이 그래프를 확장한 레벨의 수를 휴리스틱 계산 테이블에 (목표조건, 레벨) 형태로 기록한다.
- 목표조건이 2개 이상인 경우, 동작들 간의 부정적 상호작용이 존재하는 것으로 간주하고 확장과정에서 부여한 정보를 바탕으로 동작 추적 과정을 통한 동작 기반 휴리스틱 측정 과정을 진행한다. 동작 추적은 <그림 3>에서와 같이 초기상태 또는 이전 목표조건이 등장한 상태부터 새롭게 생성된 그래프를 통해 진행되기 때문에 이전에 등장한 목표조건이 포함된 레벨까지만 진행되며 이미 경로 탐색을 진행한 경로에 대해서는 탐색하지 않는다. 그리고 그 결과를 동작 테이블에 (목표조건, 해당 동작) 형태로 기록한다.



<그림 2> 단일 목표조건에 대한 부분 계획그래프



<그림 3> 다중 목표조건에 대한 부분 계획그래프

<표 1> 혼합 휴리스틱

```
1.  Mixed_Heuristic(s0, G, O)
2.  Begin
3.    L0 = s0; /* Graph : Initial Literal Layer */
4.    k = 1; /* Graph : Extend Level */
5.    h_value = 0;/* Mixed Heuristic */
6.    h_Table = null /* partial heuristic */
7.    a_Table = null /* Action-Literal Info */
8.    goal_Dependency_action = null;
9.    /* goal-related actions */
10.   If (G ⊄ L0) {
11.     For each literal li ∈ L0 do
12.         a_Table.put(li, init);
13.     End
14.   }
15.   while (G ⊄ Lk-1) {
16.     (Ak-1, Lk) = PG_Expand_Level(Lk-1, O);
17.     count = 0;
18.     count = Find_Action_Literal(Ak-1 , Lk , Lk-1);
19.     k = k + 1;
20.     If (count = = 1) {
21.         h_value = h_value + h_Table.getValue();
22.         L0 = Lk;
23.         k = 1;
24.         a_Table.actionUpdate( *, init);
25.         goalStateUpdate(h_Table.getLiteral());
26.     }
27.     If (count >= 2) {
28.       Find_Action_Dependency(G, Lk);
29.       L0 = Lk;
30.       k =  1;
31.       a_Table.actionUpdate( *, init);
32.     }
33.   }
34.    h_value = h_value +
35.              goal_Dependency_action.count();
36.    return h_value;
37. End
```

이러한 계획그래프 전개와 측정 과정은 최종적으로 모든 목표조건이 등장할 때까지 그래프를 반복적으로 재 생성하면서 진행된다. 생성되는 계획그래프는 하나의 거대한 계획그래프를 생성하거나 중복적으로 생성되는 것이 아니라 하나의 그래프를 초기화하면서 목표조건에 따른 부분 계획그래프의 재생성 과정을 거치는 것이기 때문에 보다 복잡한 계획문제를 해결하고자 할 때 발생하는 과도한 그래프 생성 과정 및 이로 인한 계산량의 증가 문제를 줄일 수 있을 것으로 기대된다. 또한 달성되는 목표조건들 간의 이용되는 동작들에 대한 긍정적 상호작용과 동작 간 상호작용을 부분적으로 적용하는 것 가능하며 모든 경로에 대한 동작 상호작용 관계를 검사하지 않기 때문에 계산량을 줄일 수 있다.

이러한 부분 확장 계획그래프를 통해서 계산되는 혼합 휴리스틱의 알고리즘은 <표 1>과 같다. 먼저 휴리스틱 계산을 위해 필요한 그래프와 부분적으로 계산된 휴리스틱을 관리할 수 있는 자료구조에 대한 초기화를 진행한다(표 1, line 3 ~ 8). 필요한 정보가 준비되면 주어진 초기상태와 목표 조건을 비교하여 그래프 확장 여부를 결정하는데, 초기상태를 구성하는 조건들이 목표상태를 구성하는 조건들을 만족하지 않으면 초기상태 조건들에 대한 정보부여 과정을 진행한다(표 1, line 10 ~ 14). 이때, 초기상태의 정보는 (상태조건, init)과 같이 부여된다. 이러한 초기상태에 대한 정보 부여는 알고리즘 내에서 목표조건과 관계된 동작 추적을 통한 동작의 관계성을 파악할 때 추적 종료 위치를 알아내기 위함이다. 초기상태를 구성하는 조건들에 대한 정보부여 과정이 완료되면 주어진 목표를 구성하는 목표조건들 가운데 1개 이상의 조건이 만족될 때까지 계획그래프를 한 단계씩 확장한다(표 1, line 15 ~ 33). 이때, 그래프를 확장하면서 새롭게 등장한 상태조건들에 대해 이를 만족시키는 동작들을 확인한다(표 1, line 18, 표 2).

<표 2>는 이러한 그래프 확장 과정에서 새롭게 등장하는 상태조건들을 만족시키는 동작들을 확인하는 과정을 나타내는데, 이전 상태 층을 구성하는 상태조건들을 제외하고 확장된 현 상태 층에서 새롭게 등장한 상태조건들에 대해서만 (상태조건, 해당 동작)의 정보부여 과정을 수행하며(표 2, line 3 ~ 13) 등장한

상태조건이 목표조건인지 확인하고 그 수를 계산한다(표 2, line 6 ~ 9). 이 때 측정된 목표조건의 수가 0개인 경우, 즉 목표조건이 새롭게 등장하지 않으면 휴리스틱 계산을 진행하지 않고 계획그래프를 계속 확장하게 되고, 1개인 경우에는 계획그래프의 확장 레벨의 수를 해당 목표조건에 대한 휴리스틱으로 이용하며(표 1, line 20 ~ 26), 등장한 목표조건의 수가 2개 이상인 경우에는(표 1, line 27 ~ 32) 그 때까지 확장된 계획그래프를 바탕으로 동작 추적과정(표 1, line 28, 표 3)을 진행한다.

<표 2> Find_Action_Literal 함수

```
1.  Find_Action_Literal(Ak-1 , Lk , Lk-1)
2.  Begin
3.    For each new literal li ∈ Lk — Lk-1 do
4.          Select an action o from
5.                    Ak-1 s.t. li ∈ effect(o);
6.          If (li ∈ G) {
7.              count = count + 1;
8.              h_Table.put(li, k);
9.          }
10.         If (¬a_Table.CONTAINS(li)) {
11.             a_Table.put(li, o);
12.         }
13.     End
14.   return count;
15. End
```

<표 3>은 이러한 목표조건들을 달성시키는 관련된 동작들과 이와 관련된 상태조건들에 대한 검사를 수행하는 과정을 나타내고 있다[9]. 살펴보면 계획그래프 확장 시 정보부여 과정을 통해 부여된 정보를 바탕으로 목표조건을 만족시키는 동작들을 검출하고 해당 동작을 계획생성 과정에 이용되는 목표 관련된 동작들(goal-dependent actions)에 포함한다. 이때, 포함된 동작은 동작 검사를 통해서 관련 동작들 간에 중복 저장이 발생하지 않도록 하며 해당 동작에 대해서는 추적 과정이 발생하지 않는다. 검출된 동작이 목표와 관련된 동작에 포함되게 되면 동작이 수행되기 위해 필요한 해당 동작의 전제조건들(preconditions)을 경로 추적을 위한 검사 항목에 포함시키며 검사 항목에 포함된 상태조건들을 바탕으로 동작 확인 과정과 검사 항목 포함 과정을 반복하며 경로를 추적한다(표 3, line 16 ~ 29). 주어진 개별 검사 항목에 대한 추적 과정은 초기상태 또는 이미 경로 탐색을 진행한 상태에 도달하게 되면 마치게 되며 검사해야 할 항목이 더 이상 없을 때까지 진행된다.

<표 3> Find_Action_Dependency 함수

```
1.  Find_Action_Dependency(G, Lk)
2.  Begin
3.    Stack new_literal = null;
4.    boolean query_result = true;
5.    For each g ∈ Lk do
6.      Select an action o from
7.                  Ak-1 s.t. li ∈ effect(o);
8.      If each g ∈ effect(o) s.t. o is different {
9.          action ak = actionTable.query(g);
10.     }
11.     If ¬goal_Dependency_action.CONTAINS(ak) {
12.       goal_Dependency_action.put(ak);
13.       new_literal.PUSH(precondtion(ak));
14.     }
15.   End
16.   while (¬new_literal = = null) {
17.     li = new_literal.POP();
18.     action ak = a_Table.query(li);
19.     If (ak = = init ) {
20.     /* ak is initial state condition */
21.     } Else {
22.       query_result =
23.         goal_Dependency_action.CONTAINTS(ak);
24.     }
25.     If (query_result == false) {
26.       goal_Dependency_action.put(ak);
27.       new_literal.PUSH(precondtions(ak));
28.     }
29.   }
30.   return null;
31. End
```

이러한 과정을 통해서 얻어진 목표와 관련된 동작들은 해당 그래프에서 등장한 목표조건들을 달성하기 위한 최소한의 관련된 동작들이 되고 목표조건에 따른 반복적인 그래프 확장과 동작 추적을 통해 누적된 동작들을 이용하여 휴리스틱 계산을 진행하게 된다. 이러한 계획그래프는 등장한 목표조건의 수가 1개 이상인 경우, 휴리스틱 계산이 수행된 계획그래프는 현재까지 확장된 상태를 초기상태로 지정하면서 그래프는 초기화되고 목표상태를 구성하는 모든 목표조건이 등장할 때까지 그래프를 다시 확장하게 된다. 계획그래프에서 모든 목표조건을 만족하게 되면 그동안 목표조건의 상태에 따라 개별적으로 계산된 휴리스틱 계산 결과를 더하여 최종 휴리스틱 계산 결과를 도출하게 된다(표 1, line 34 ~ 35).

이와 같이 부분 계획그래프를 기반으로 한 혼합 휴리스틱은 계획그래프를 확장하는 과정에서 생성되는 정보를 동작과 목표조건 간의 관계 파악에 이용하기 때문에 그래프 확장 과정 이외의 별도의 추가적인 노력을 필요로 하지 않으며 계획그래프를 확장하며 생성된 여러 동작 층에 포함된 동작들을 모두 검사하지 않으면서도 관련 동작, 또는 동작 수행 횟수를 보다 쉽게 추출할 수 있다. 또한 목표조건을 달성하기 위한 관련된 동작 검사 과정에서 중복 검사로 인한 낭비와 동일한 상태조건에 대한 복수의 관련된 동작 선택 문제 등을 방지할 수 있으며 그래프에서 모든 목표조건들이 등장할 때까지 확장하는 것이 아니라 목표상태를 구성하는 개별적인 목표조건을 위한 부분 그래프를 생성하기 때문에 그래프 크기를 줄여서 휴리스틱 계산을 위한 상태 크기를 줄일 수 있다. 따라서 부분 계획그래프를 바탕으로 하는 혼합 휴리스틱은 목표조건에 따른 부분 계획그래프를 이용하여 휴리스틱을 계산하기 때문에 과도한 그래프 생성과 계산 복잡도의 증가 문제를 줄일 수 있다. 또한 동작 간의 상호의존성을 고려하기 때문에 최대, 합산 휴리스틱 보다 높은 정보력을 가지며 주어진 계획문제에 따라서는 겹침 휴리스틱보다 더 적은 계산 노력을 통해서 같은 수준의 휴리스틱 평가치 계산이 가능하며 사용자 혹은 시스템에 측정된 휴리스틱 평가치가 허용가능한 휴리스틱인지 허용가능하지 않은 휴리스틱인지 판별하여 제공하는 것 또한 가능하다.

Ⅲ. 평가

본 논문에서 제안하고 있는 부분 계획그래프와 이를 기반으로 하고 있는 혼합 휴리스틱의 정확성과 탐색 효율성을 분석하고자 하였다. 일반적으로 측정된 휴리스틱이 실제 값에 가까울수록 탐색을 보다 정확하게 수행할 수 있고 효율적으로 계획 생성을 위한 탐색을 유도한다는 것을 알 수 있다. 따라서 실제 최소 도달거리와 휴리스틱 측정치의 비교를 통해 휴리스틱의 정확성을 확인하기 위한 평가 항목으로 이용하였다. 그리고 이러한 휴리스틱 측정치를 구하는 과정에서 필요한 동작 검사 횟수를 계산 효율성에 대한 평가 항목으로 사용하였다. 평가의 대상은 동일한 자료구조를 사용하는 휴리스틱이 없기 때문에 제안하고 있는 부분 계획그래프의 기반이 되는 완화된 계획그래프 기반의 대표적인 휴리스틱인 최대 휴리스틱과 모든 목표조건들 사이에 독립적 관계가 존재한다고 가정하는 합산 휴리스틱, 그리고 긍정적 상화작용과 독립적 관계가 공존한다고 가정하는 겹침 휴리스틱을 이용하였으며 몇 가지 서로 다른 계획 문제 도메인을 이용하여 비교 실험을 진행하였다. 평가의 대상이 된 휴리스틱 가운데, 최대 휴리스틱의 경우에는 허용 가능한 휴리스틱(admissible heuristic)이고 합산 휴리스틱과 겹침 휴리스틱의 경우에는 허용 가능하지 않은 휴리스틱(non-admissible heuristic)이지만 여기서는 휴리스틱의 정확성을 평가하기 위해 모두 포함하였다. 탐색 효율성에 대한 분석에서는 최대 휴리스틱과 합산 휴리스틱의 경우, 상호의존성을 검사하

지 않기 때문에 상호의존성을 검사하는 겹침 휴리스틱만을 대상으로 하였다. 이러한 비교 평가를 통해서 혼합 휴리스틱의 정확성과 계산 과정의 효율성을 확인하고자 하였다.

실험에서 필요한 탐색 알고리즘은 전통적 계획문제에서 가장 일반적으로 사용되는 A* 알고리즘을 이용하였으며 실험을 위해 3가지 서로 다른 도메인(생활 서비스 도메인, 블록 이동 도메인, 트럭 도메인)에 임의로 생성한 각 5개의 계획문제, 총 15개의 계획문제를 이용하였다. 비교 실험에 이용된 도메인 내용을 간단히 살펴보면 다음과 같다. 생활 서비스 도메인은 생활환경에서 사용자에게 필요한 일정 생성을 위한 서비스와 가정 내 로봇이 제공하는 서비스가 결합된 도메인으로서 사용자 일정과 로봇에게 주어지는 업무가 서로 별도의 계획이 아닌 로봇이 사용자에게 물을 가져다주는 동작과 같이 서로 연관된 업무 등으로 구성되어 사용자 일정 생성과 로봇이 수행해야 하는 업무에 대한 계획이 함께 진행된다. 때문에 사용자 일정과 로봇이 제공하는 서비스 간의 동작 간 상호작용이 발생할 수 있어 이를 확인하기 위하여 실험 도메인으로 포함하였다. 이를 위해 생활 서비스 도메인은 물을 전달하는 give_the_water_to_user, 로봇 이동과 관련한 move, open_door, 사용자 지시 사항을 수행하기 위한 carry, putdown 등의 동작과 user_move, have_dinner, visit, lesson과 같은 사용자 일정 생성을 위한 동작을 함께 포함하고 있다. 블록 이동 도메인은 블록을 집거나 내려놓는 pickup, pickup_table, putdown, putdown_table과 같은 동작을 포함하며 트럭 도메인은 화물을 싣고 내려놓는 load, unload, 위치 간 이동을 위한 move 등의 동작을 가지고 있다.

<표 4>는 혼합 휴리스틱의 정확성을 알아보기 위해서 주어진 계획문제의 초기상태에서 목표상태까지의 실제 최소 도달거리와 이에 대한 각 휴리스틱의 평가치를 비교한 결과이다. 실제 최소 도달거리와 휴리스틱 간의 차이를 비교해보면, 도메인 모두에서 제안하고 있는 혼합 휴리스틱이 비교 대상인 최대 휴리스틱과 합산 휴리스틱에 비해 휴리스틱 평가치가 더 정확하다는 것을 알 수 있다. 이러한 결과는 혼합 휴리스틱이 동작의 상호의존성을 검사하기 때문으로 판단된다. 또한 동작 간의 상호의존성을 모두 검사하는 겹침 휴리스틱에 비해서도 비록 일부 정확도가 낮은 결과가 나왔으나 그 정확도의 차이는 크지 않은 것을 알 수 있다. 이는 제안하는 혼합 휴리스틱의 경우, 계산량의 줄이기 위해서 목표조건이 만족되는 상황에 따라 부분적으로 동작 간의 검사를 하지 않기 때문에 그 상황에서 발생하는 동작 간의 상호작용을 검출하지 못하기 때문으로 판단된다.

<표 4> 초기상태에서 목표상태까지의 최소 도달거리와 휴리스틱 평가치 비교

도메인 / 휴리스틱	로봇 서비스 도메인				
	p1	p2	p3	p4	p5
최소 도달거리	5	8	9	13	15
최대 휴리스틱	3	7	6	10	12
합산 휴리스틱	7	16	12	22	26
겹침 휴리스틱	5	9	8	12	14
혼합 휴리스틱	5	8	7	11	13
도메인 / 휴리스틱	블록 이동 도메인				
	p1	p2	p3	p4	p5
최소 도달거리	5	7	11	14	19
최대 휴리스틱	3	4	4	4	6
합산 휴리스틱	8	17	22	34	47
겹침 휴리스틱	5	8	11	16	22
혼합 휴리스틱	5	8	11	16	22
도메인 / 휴리스틱	트럭 도메인				
	p1	p2	p3	p4	p5
최소 도달거리	4	8	10	11	12
최대 휴리스틱	3	5	7	7	8
합산 휴리스틱	6	14	21	24	27
겹침 휴리스틱	4	8	11	12	13
혼합 휴리스틱	4	7	9	10	10

<표 5> 목표 관련 동작 검사 수

도메인 / 휴리스틱	로봇 서비스 도메인				
	p1	p2	p3	p4	p5
겹침 휴리스틱	5	9	9	14	16
혼합 휴리스틱	4	7	8	11	13
도메인 / 휴리스틱	블록 이동 도메인				
	p1	p2	p3	p4	p5
겹침 휴리스틱	11	14	24	27	32
혼합 휴리스틱	6	9	11	15	19
도메인 / 휴리스틱	트럭 도메인				
	p1	p2	p3	p4	p5
겹침 휴리스틱	4	9	12	15	17
혼합 휴리스틱	3	6	9	11	12

<표 5>는 합산 휴리스틱의 계산 효율성을 알아보기 위한 실험 결과이다. 휴리스틱 계산 과정에서 이용되는 목표 관련 동작을 확인하기 위해 검사한 동작들의 수를 비교한 결과로써 이를 통해서 계산 과정에서의 계산 효율을 확인해 보고자 하였다. 최대 휴리스틱과 합산 휴리스틱의 경우에는 목표 관련 동작의 상호의존성을 검사하지 않기 때문에 비교 대상에서 제외하였으며 상호의존성을 검사하는 겹침 휴리스틱만을 대상으로 하였다. 본 논문에서 제시하고 있는 혼합 휴리스틱과 겹침 휴리스틱은 목표조건과 관련한 동작들을 찾고 이를 휴리스틱 계산에 이용하기 때문에 관련 동작을 검사한 횟수를 통해서 계산 효율성을 일부 분석할 수 있다. <표 4>, <표 5>의 결과를 함께 살펴보면 혼합 휴리스틱이 겹침 휴리스틱에 비하여 측정된 평가치에 비해 검사하는 동작의 횟수가 더 적은 것을 알 수 있다. 이러한 결과는 겹침 휴리스틱의 경우에는 확장이 완료된 전체 계획그래프 상에서 목표와 관련한 전체 경로상의 동작을 검사하기 때문에 검사 횟수가 많은 것으로 판단되며 제안하는 혼합 휴리스틱의 경우에는 목표조건 별로 부분 그래프를 만들며 계산하기 때문에 전체 경로를 검사하지 않아서 더 적은 것으로 판단된다. 이를 통해서 혼합 휴리스틱이 실험 도메인에서 최대 휴리스틱과 합산 휴리스틱에 비해 더 높은 정확성을 가진다는 것과 겹침 휴리스틱보다 정확성에서는 일부 낮은 결과를 보이기는 했으나 최대 휴리스틱과 합산 휴리스틱보다 높은 정확성을 적은 계산 노력으로 얻을 수 있다는 것을 확인하였다.

Ⅳ. 결론

본 논문에서는 휴리스틱을 보다 효율적인 방법을 통해 양질의 결과를 자동으로 계산해 내기 위한 새로운 자료구조인 부분 계획그래프와 이를 기반으로 한 혼합 휴리스틱을 제안하였다. 그리고 비교 실험을 통해서 기존 최대 휴리스틱과 합산 휴리스틱에 비해 더 높은 정확성을 보인다는 것을 확인하였으며 보다 적은 계산 노력으로 동작의 상호의존성을 포함한 휴리스틱 계산이 가능함을 확인하였다. 비록 동작의 상호의존성을 전체 검사하지 않기 때문에 전체 경로에 대한 동작을 검사하는 동작 기반 휴리스틱에 비해서는 정확도가 일부 낮아지는 결과가 있기는 하였으나 그 차이는 크지 않았으며 휴리스틱 계산에 이용된 혼합 알고리즘의 함수를 검사하면 제공하는 휴리스틱 결과의 허용성 여부를 파악할 수 있다는 장점이 있다. 추후에는 제안하는 휴리스틱 알고리즘의 정확성과 효용성을 보다 명확히 파악하기 위하여 보다 다양한 도메인과 계획문제를 이용한 실험을 진행할 예정이며 동작 간 상호의존성 검사에서 부정 효과를 포함하여 휴리스틱을 구할 수 있도록 확장할 예정이다. 또한 지속적으로 연구되고 있는 다른 휴리스틱과의 비교와 불확실성을 포함한 실세계 계획문제에 적용하기 위한 확률을 포함한 조건부 계획문제(contingent planning)로 확대 적용을 위한 연구가 필요할 것으로 판단된다.

참고문헌

[1] 이석준·김인철, “행위 온톨로지로부터 로봇 작업 계획 생성,” 제어로봇시스템학회, 제어로봇시스템학회논문지, 제23권, 제8호, 2017, pp.650-660.

[2] 최병관, “인공지능 객체인식에 관한 파라미터 측정 연구,” 디지털산업정보학회, 디지털산업정보학회논문지, 제15권, 제3호, 2019, pp.15-28.

[3] 송호정·김현기, “어댑티드 회로 배치 유전자 알고리즘의 설계와 구현,” 디지털산업정보학회, 디지털산업정보학회논문지, 제17권, 제2호, 2021, pp.13-20.

[4] B. Bonet, and H. Geffner, “Planning as Heuristic Search,”Artifical Intelligence, 2001, pp.5-33.

[5] J. Hoffmann and B. Nebel, “The FF Planning System: Fast Plan Generation through Heuristic Search:, Journal of AI Research, vol14, 2001, pp.253-302.

[6] D. Bryce and S. Kambhampati, “A Tutorial on Planning Graph-Based Reachability Heuristics”, AI Magazine, vol.28, no.1, 2006, pp.47-83.

[7] D. Cai, M. Yin and J. Wang, “Improving Relaxed Plan-Based Heuristics via Simulated Execution of Relaxed-Plans”, ICAPS, 2009.

[8] 김현식·김인철, “최적 계획수립을 위한 확장된 그래프 기반의 휴리스틱,” 한국정보과학회, 정보과학회논문지, 제17권, 제12호, 2011, pp.667-671.

[9] 김현식, “효율적인 계획 수립을 위한 동작-기반의 휴리스틱,” 한국산학기술학회, 한국산학기술학회논문지, 제16권, 제9호, 2015, pp.6290-6296.

[10] P. Haslum and H. Geffner, “Admissible Heuristics for Optimal Planning”, AIPS2000, 2000, pp.140-149.

[11] S. Russell and P. Norving, Artificial Intelligence: A Modern Approach, 3nd Edition, PeasonEducation, 2010.

[12] E. Keyder and H. Geffner, “Set Additive and TSP Heuristics for Planning with Action Costs and Soft Goals”, ICAPS, 2007.

[13] 김완태·김현식, “최적 계획생성을 위한 동작비용 기반의 휴리스틱,” 디지털산업정보학회, 디지털산업정보학회논문지, 제13권, 제2호, 2017, pp.27-34.

[14] O. Marom and B. Rosman, “Utilising Uncertainty for Efficient Learning of Likely-Admissible Heuristics,” ICAPS2020, 2020, pp.560-568.

[15] A. Blum and M. Furst, “Fast Planning through Planning through Planning Graph Analysis”, Artificial Intelligence, vol.90, 1997, pp.281-300.

■ 저자소개 ■

박 병 준
(Park, Byungjoon)

2013년 3월~현재 서일대학교 소프트웨어공학과 조교수
2010년 2월 국민대학교 전산정보공학과 (이학박사)
2002년 8월 고려대학교 의료정보기기학과 (공학석사)
2000년 2월 고려대학교 전자정보공학과 (공학사)

관심분야 : 영상처리, 패턴인식, 객체추적, IoT, 인공지능
E-mail : 20130029@seoil.ac.kr

김 완 태
(Kim, Wantae)

2011년 3월~현재 서일대학교 정보통신공학과 조교수
2011년 2월 한국항공대학교 정보통신과 (공학박사)
2004년 2월 한국항공대학교 정보통신과 (공학석사)

관심분야 : 통신시스템 설계, IoT 시스템, 모바일 응용 S/W
E-mail : wtkim@seoil.ac.kr

김 현 식
(Kim, Hyunsik)

2011년 3월~현재 서일대학교
소프트웨어공학과 조교수
2010년 8월 경기대학교 일반대학원
전자계산학과(이학박사)
2004년 2월 경기대학교 일반대학원
전자계산학과(이학석사)
2001년 2월 경기대학교 전자계산학부
전자계싼전공(이학사)

관심분야 : 자동계획, 시맨틱웹, 지능형 시스템, 데이터마이닝, 모바일 응용 S/W
E-mail : hskim@seoil.ac.kr

논문접수일 : 2021년 8월 20일
수 정 일 : 2021년 9월 9일
게재확정일 : 2021년 9월 13일