論文94-31B-8-20

작업장이 제한된 블럭세계에 대한 계획자

(A Planner for the Limited Table-Capacity Blocks World)

沈 東 熙*, 黃 鍾 善**

(Dong Hee Shim and Chong Sun Hwang)

要 約

Gupta는 블럭세계에서의 교착상태를 정의하였으며, 이로 인하여 블럭세계의 의사결정문제는 NP-Complete임을 밝혔다. 본 논문에서는 이러한 Gupta의 교착상태를 유형으로 분류하여, 교착상태의 성질을 분석하였다. 이 교착상태의 성질을 이용하고 또한 역위치 개념을 정의하여 아직까지 연구되지 못한 작업장이 제한된 블럭세계에 적용할 수 있는 계획자를 위한 휴리스틱 알고리듬을 제안하고 이를 구현평가하였다.

Abstract

Gupta defined deadlocks in the blocks world, and proved that decision making problem in the blocks world is NP-Complete. The types of deadlocks are classified, and the properties of deadlocks are analyzed in this paper. The heuristic planner algorithm for the limited table-capacity blocks world using both the inverse position of the block and properties of deadlocks is designed and its performance is also evaluated.

I. 서 론

인공지능의 계획분야에서 가장 많이 이용되는 문제영역은 블럭세계이다. 그리하여 STRIPS [1], ABSTRIPS [2], NOAH [3], SIPE [4], TWEAK [5], ABTWEAK [6] 등과 같은 많은 영역독립적 계획자(domain-independent planner)에서는 그 예로서 대부분 블럭세계를 이용하고 있다. 이러한 블럭세계는 기본적 블럭세계 (EBW:elementary blocks world)와 확장된 블럭세계 (XBW:extended blocks world), 작업장이 제한된 블럭세계 (LBW:limited table-capacity blocks world)로 나누어 볼 수 있다. EBW는 기존의 계획분야에서 사용된 바와 같은 문제이다. 즉 stack, unstack, pickup, putdown과 같은 4개의 원시연산자(primitive operator)를 사용하되 모든 블럭의 모양과 크기가 같고 블럭이름의 중복을 허용하지 않는 것이다. 한편 XBW에서는 블럭의 크기, 모양(shape), 색깔 등이

*正會員, 全州大學校 電子計算學科
(Dept. of Computer Science, Jeonjoo Univ.)
**正會員, 高麗大學校 電算科學科
(Dept. of Computer Science, Korea Univ.)
接受日字 : 1993年 5月 24日

다를 수 있고, 블럭이름의 중복도 허용하는 경우이다. LBW에서는 블럭을 놓을 수 있는 테이블의 용량에 제한이 가해지는 경우이다. 지금까지의 계획자들은 모두 EBW에 적용할 수 있으나, XBW와 LBW에 대해서는 적용이 곤란하다.

그러나 블럭세계에서 계획길이에 대한 의사결정문제는 블럭간의 교착상태 성질로 인하여 NP-Complete 임이 밝혀져 있다.[7,8] 따라서 위와 같은 영역독립적 계획자로 블럭세계를 다루는 데에는 그 한계가 있다. 즉 영역 고유의 휴리스틱 지식이 결부되어야 효율적인 계획의 수립이 이루어 질 수 있다. 그리하여 Gupta가 정의한 블럭세계에서의 교착상태 성질을 이용하여 XBW에 적용할 수 있는 휴리스틱 알고리듬에 근거한 계획자가 연구된 바 있다.[9] 또한 LBW에 적용할 수 있는 알고리듬은 Gupta에 의하여 제시된 바가 있지만[8] 계획자에 이용하기에 적절하지가 않다

그리하여 본 연구에서는 교착상태의 성질과 역위치의 개념을 이용하여 LBW에 적용할 수 있는 계획자 알고리듬을 제안하였으며, 이를 구현하여 그 성능을 Gupta가 제시한 알고리듬에 근거한 계획자와 비교평가하였다.

Ⅱ. 블럭세계를 위한 표현자

지금까지 STRIPS표현을 이용한 경우의 블럭세계에 대한 상태표현에서는 on(A,B), ontable(A), clear(A), holding(A), armempty 와 같은 5개의 서술자가 사용되었다. 이러한 서술자 이외에 다음과 같은 표현을 이용한다.

1) 블럭리스트 ${}_IBL_N = ({}_IB_1, {}_IB_2, \cdots, {}_IB_N)$ 는 $on({}_IB_1, {}_IB_2) \wedge on({}_IB_2, {}_IB_3) \wedge \cdots on({}_IB_{N-1}, {}_IB_N) \wedge ontable({}_IB_N) \wedge clear({}_IB_1)$ 라 하자. 여기서 N는 블럭리스트 ${}_IBL_N$ 의 길이에 해당한다. 블럭 ${}_IB_1$을 ${}_IBL_N$의 헤드라 하고, 마지막 블럭 ${}_IB_N$을 ${}_IBL_N$ 의 테일이라 하자. 이 정의에서 I는 블럭리스트의 번호를 의미한다.즉 ${}_IBL_N$은 N개의 블럭으로 구성된 I번째 블럭리스트를 의미한다.

위와 같은 정의를 이용하면 로봇을 제외한 블럭세계의 STRIPS 표현상태는 블럭리스트 ${}_IBL_N$ 을 원소로 하는 M 개의 블럭리스트 즉 $({}_1BL_{N1}, {}_2BL_{N2}, \cdots {}_MBL_{NM})$으로 표현된다.

2) above(A,B)는 블럭A가 블럭B보다 위에 놓여있는 상태를 나타내되 on(A,B)는 제외한다.

즉 이는 $on(A, X_1) \wedge on(X_1, X_2) \wedge \ldots \wedge on(X_N, B)$ (단 n는 1이상,이러한 X_i를 A와 B의 중간블럭이라 함)을 나타낸다.

3) obove(A,B) 는 on(A,B) 또는 above(A,B)를 나타낸다.

4) 최종위치(final position) FP(x)는 현재 블럭 x가 목표상태에서와 동일한 위치에 있는 것을 의미한다. 즉 현재상태에서 X가 속해있는 블럭리스트 ${}_IBL_N = ({}_IB_1, {}_IB_2, \ldots {}_IB_C, X, {}_IB_{C+1} \cdots, {}_IB_N)$에 대하여 목표상태에는 어떤 블럭리스트 ${}_JBL_K = ({}_JB_1, {}_JB_2, \ldots {}_JB_A, X, {}_IB_{C+1}, {}_IB_{C+2}, \cdots, {}_IB_N)$가 존재하여야 한다. (단 C=0, A=0일 수도 있음)

5) flat state FS(x)는 현재 ontable(x) ^ clear(x)의 상태를 나타낸다.

6) $on^{-1}(A,B)$는 on(A,B)의 역상태 즉 on(B,A)를 나타낸다.

7) $above^{-1}(A,B)$는 above(A,B)의 역상태 즉 above(B,A)를 나타낸다.

8) $obove^{-1}(A,B)$는 obove(A,B)의 역상태 즉 obove(B,A)를 나타낸다.

Ⅲ. 블럭세계의 교착상태

1. Gupta의 교착상태 정의 및 계획자를 위한 비결정적 알고리듬

Gupta는 블럭세계에 대한 교착상태를 다음과 같이 정의하였다.[7,8] 이 정의를 앞서 제시한 상태표현자를 이용하여 나타내면 다음과 같다. 상태s에서 블럭집합 $D=\{d_1,d_2,\cdots,d_p\}$는 만일 다음 조건을 만족하는 블럭집합 $R=\{r_1,r_2,\cdots,r_p\}$가 존재하면 교착상태이다.

1) 상태 s에서 $above(d_1,r_1) \wedge above(d_2,r_2) \wedge \ldots \wedge above(d_p,r_p)$

2) 목표상태 g에서 $above(d_1,r_2) \wedge above(d_2,r_3) \wedge \ldots \wedge above(d_p,r_1)$

Gupta는 블럭세계의 복잡성을 분석하기 위하여 다음과 같은 비결정적 알고리듬을 제시하였다.

1) 초기상태를 현재상태로 한다.

2) 현재상태가 목표상태와 같으면 완료한다.

3) 최종위치로 이동할 수 있는 블럭이 있으면 그 블럭을 옮기고 스텝 2로 간다.

4) 비결정적 루틴을 호출하여 교착상태를 해결하고 스텝 3으로 간다.

위 알고리듬에서 스텝 4가 비결정적이다. 이는 현재 교착상태의 블럭중 하나를 선택하여 FS상태(테이블에 내려 놓음)로 만들어 교착상태를 해결하는 것이다. Gupta는 위의 스텝1 - 스텝3은 $O(n^2)$이고 스텝4가 비결정적임을 이용하여 EBW에서의 계획길이문

제는 NP-Complete 에 해당함을 증명하였다. 따라서 EBW에서 어떤 주어진 계획문제에 대한 최적계획 수립문제는 NP-Hard에 해당한다. 그러나 Gupta의 알고리듬은 스텝4가 비결정적이기 때문에 계획자에 이용될 수 없다. 또한 스텝4에 도달한 경우 현상태를 구성하는 각 블럭리스트의 모든 헤드가 교착상태를 형성하는 블럭집합 D에 해당하지 않고, 헤드이거나 헤드가 아닌 일부의 블럭이 여기에 해당한다. 교착상태에 해당하지 않는 헤드는 이 교착상태인 블럭으로 인하여 연쇄적으로 최종위치로 이동이 될 수가 없다. 따라서 교착상태를 해결하려면 어떤 블럭이 교착상태인지를 파악해야 한다. 또한 스텝4를 결정적 루틴으로 만들기 위해서는 교착상태의 성질을 분석하여 이를 이용해야 한다.

2. 교착상태의 성질

1) 교착상태의 유형분류

교착상태는 항상 어떤 현재상태에서의 각 블럭리스트의 헤드에 해당하는 블럭의 교착상태가 관심사이다. 이는 계획에 나타나는 모든 연산자는 항상 블럭리스트의 헤드에 위치한 블럭에 가해지기 때문이다. 이러한 교착상태를 표1에 나타낸 바와 같이 6가지 유형으로 분류하였다. 여기서 블럭수는 집합 D의 원소수를 의미하는 데 이는 교착상태에 포함된 블럭리스트의 수와 같다. 한편 서술자는 현재상태와 목표상태에서 집합 D에 속한 블럭과 집합 R에 속한 블럭간의 관계를 의미한다.

표 1. 교착상태 유형

Table 1. deadlocks type.

블럭리스트 수 / 서술자		1 개	2개이상
상태 S	목표상태		
On	On	유형 1	유형 2
Above	Above	유형 3	유형 4
On	Above	유형 5	유형 6
Above	On		
On과 Above	On과 Above	해당없음	

위 유형에 대하여 몇 가지 예를 보면 다음과 같다. 그림1에는 유형2를 나타냈다. 집합 D에 속한 블럭 A, B, C가 집합 R에 속한 블럭 D, E, F에 대하여 교착상태이다. 즉 세개의 블럭리스트에 속하며, on의 관계에 있는 블럭들간의 교착상태이다. 단 이 그림에서 블럭 A, B, C가 반드시 헤드일 필요는 없다. 만약 이 배치에서 블럭간의 관계가 above가 되면 유형4에 해당한다.

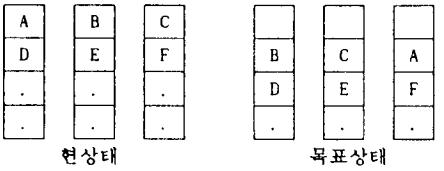

그림 1. 유형2의 교착상태

Fig. 1. deadlocks type 2.

그림 2에는 유형3을 나타냈다. 집합 D에 속한 블럭 A가 집합 R에 속한 블럭 B에 대하여 교착상태이다. 즉 한개의 블럭리스트에 속하며, above관계에 있는 블럭들간의 교착상태이다. 만약 현상태와 목표상태에서 A, B의 관계가 모두 on이면 유형 1에 해당하며 하나의 상태에서만 관계가 on이면 유형5에 해당한다. 여기서도 A가 반드시 헤드일 필요는 없다.

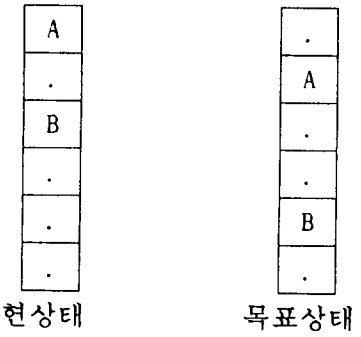

그림 2. 유형3의 교착상태

Fig. 2. deadlocks type 3.

그림 3에는 유형6을 나타냈다. 집합 D에 속한 블럭 A, B, C가 집합 R에 속한 블럭 D, E, F에 대하여 교착상태이다. 즉 세개의 블럭리스트에 속하며, 현상태나 목표상태에서 모두 블럭간의 관계가 on, above가 혼합되어 있는 경우다. 여기서도 A, B, C가 반드시 헤드일 필요는 없다.

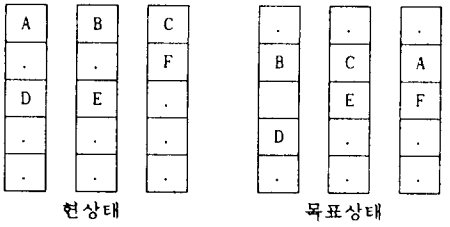

그림 3. 유형6의 교착상태

Fig. 3. deadlocks type 6.

2) 교착상태의 성질

앞에 소개한 Gupta의 비결정적 알고리듬에 의한 계획을 수립하다가 스텝4에 도달하면 이는 상태공간상의 탐색에서 막다른 골목(death-ends)에 도달한 것에 해당한다. 물론 계획의 수립시에 교착상태에 다다르면 교착상태에 해당하는 블럭을 FS상태로 만들면 교착상태는 해소된다. 그러나 교착상태에 해당하지 않는 블럭을 FS상태로 만들면 여전히 최종위치로 이동될 수 있는 블럭이 여전히 없어 교착상태가 계속 지속된다. 또한 이 교착상태를 형성하는 블럭들은 여러 집합이 있을 수 있다. 각 유형별로 여러개가 있을 수 있고, 같은 유형이 여러 개 있을 수도 있다. 교착상태에 해당하는 블럭을 FS상태로 만들어도 이 블럭은 그후에 최종위치로 한번 더 이동이 이루어져야 하기 때문에 최적계획보다 블럭 이동수를 증가시킬 가능성이 많게 된다. 따라서 교착상태 해결방법이 최적계획에 많은 영향을 미치게 된다. 따라서 최적계획에 근사하게 교착상태를 해결하는 방법이 필요하다.

한편 교착상태의 유형은 블럭세계를 그래프로 표현하여 파악할 수 있다. Hasse Diagram이라는 그래프[10]에서는 블럭세계의 블럭을 노드로 간주하고 블럭간의 관계 on을 에지로 간주하였다. 이 그래프를 유향그래프(directed graph)로 다음과 같이 변형시키자. 즉 현상태에서의 on(A,B)를 노드 A에서 노드 B로의 유향에지(directed edge)로 정의하고, 목표상태에서의 on(A, B)는 노드 B에서 노드 A로의 유향에지(directed edge)로 정의하자. 이렇게 정의하면 교착상태를 형성하는 블럭집합 D와 R에 속하는 블럭들은 현상태의 유향그래프와 목표상태에서의 그래프를 유니온시킨 유향그래프에서 사이클(cycle 또는 circuit)에 포함된다.

한편 다음의 몇 개 유형에 대해서는 최적계획의 수립이 보장된다.

가) 교착상태 유형1,3,5

이 유형에서는 교착상태에 해당하는 블럭을 FS상태로 만드는 것이 최적계획에 배치되지 않는다. 또 어떤 임의의 블럭이 유형 1에 해당하는 지의 여부는 O(n)에 결정될 수 있으며, 유형 3에 해당하는 지의 여부는 $O(n^3)$에 결정될 수 있으며, 유형 5에 해당하는 지의 여부는 $O(n^2)$에 결정될 수 있다.

나) 교착상태 유형2의 경우는 집합R에 속한 블럭중 최종위치에 해당하거나 최종위치로의 이동이 가능한 블럭이 존재하는 경우 이 블럭과 on의 관계에 있는 블럭을 FS상태로 만드는 것이 최적계획에 배치되지 않는다. 또 임의의 어떤 블럭이 이 유형에 해당하는 지의 여부는 $O(n^d)$(단 d는 이 유형에 해당하는 집합 D에 속한 블럭수)에 이루어질 수 있다.

다) 유형 4, 6

이 유형은 최적계획을 보장해주는 해결방법이 없으며 이 유형의 탐지 복잡성은 앞서 정의한 유향그래프에서 해밀턴 회로의 탐지에 상응하게 되어 NP에 해당한다.

라) 교착상태를 해결해주는 휴리스틱 방법

최종위치가 아닌 헤드의 $above^{-1}$에 위치한 블럭중 최종위치에 해당하거나 최종위치로의 이동이 가능한 블럭이 있으면 이 헤드를 FS상태로 만든다. 예를 들어 ${}_iB_3$이 최종위치로의 이동이 가능하다고 가정하면 ${}_iB_1$, ${}_iB_2$를 FS상태로 만들면 ${}_iB_3$가 최종위치로 이동될 수 있으므로 이 방법을 이용한다.

3) 교착상태 처리를 위한 휴리스틱알고리듬

위에 설명한 교착상태의 성질을 이용하여 교착상태 처리를 위한 휴리스틱 알고리듬을 다음과 같이 설계하였다. 다음의 5단계를 순서적으로 처리하되 블럭이동이 발생하면 Gupta의 알고리듬 스텝3으로 간다.

① 교착상태 유형1,3,5에 해당하는 유형이 있으면 이 유형에 해당하는 블럭을 FS상태로 만든다.

② 교착상태 유형2에 해당하되 집합R에 속한 블럭이 최종위치에 있거나, 최종위치로의 이동이 가능하면 이러한 블럭의 on 위치에 있는 블럭을 FS상태로 만든다.

③ 위의 2가지 경우에 해당하는 처리가 없는 경우에는 각 블럭리스트의 헤드의 on^{-1}에 위치한 블럭중 최종위치에 해당하거나 최종위치로의 이동이 가능한 블럭이 있으면 이 블럭리스트의 헤드를 FS로 만든다.

④ 조건3에 의한 처리도 없는 경우에는 각 블럭리스트 헤드의 $above^{-1}$에 위치한 블럭중 최종위치에 있거나 최종위치로의 이동이 가능한 블럭이 있는지 조사하여 이러한 블럭의 블럭리스트의 헤드를 FS상태로 만든다.

⑤ 조건4에 의한 처리가 여전히 없는 경우에는 길이가 가장 긴 블럭리스트의 헤드를 FS상태로 만든다.

이 휴리스틱 알고리듬을 Gupta의 비결정적 알고리듬의 스텝4로 사용하면 기본적 블럭세계를 위한 계획자 알고리듬으로 이용할 수 있다.

Ⅳ. LBW의 정의 및 특성

작업장이 제한된 블럭세계는 Limited Table-Capacity Blocks World(LBW) 라고 불리는 문제이다.[8] 즉 다른 모든 조건은 기본적 블럭세계와 같

은 데 작업장이 되는 테이블의 능력에 제한이 있는 경우이다. 따라서 어떤 블럭을 FS 상태로 만드는 데 제한이 있기 때문에 블럭리스트의 갯수에 상한값이 존재한다. 한편 이런 문제는 만일 블럭 갯수만큼의 작업장이 주어지는 경우는 기본적 블럭세계의 문제와 같게 된다.

1. 문제의 표현과 해결성

기본적 블럭세계에서와 같은 상태표현자를 이 모델에서도 그대로 적용할 수 있으며, 다만 작업장 즉 블럭리스트의 최대허용 갯수 C를 추가해 문제를 표현해야 한다. 즉 초기상태 I, 목표상태 G, 최대허용 블럭리스트 수 C (블럭갯수로 표현되는 테이블의 밑면적에 해당)에 의하여 문제를 표현하도록 한다. LBW 문제 (I,G,C)에 대하여 초기상태와 목표상태에 나타난 블럭 이름이 모두 일치하고 각각의 블럭리스트의 갯수가 C 이하일 때 이를 일관적(consistent)이라고 하자. 또한 LBW 문제 (I,G,C)가 일관적이고 이에 대한 계획이 존재하는 경우 이를 해결적(solvable 또는 feasible) 이라고 하자. 만약 LBW 문제 (I,G,C)에서 C가 무한대이면 이 문제는 EBW가 되기 때문에 3장에 제시한 알고리듬으로 계획을 수립할 수 있다.

[예 1] 그림4와 같은 Sussman의 Anomaly 문제의 경우 C=2로 하면 이 문제는 해결적이지 않다. 그러나 C 〉 2 이면 이 문제는 해결적이다.

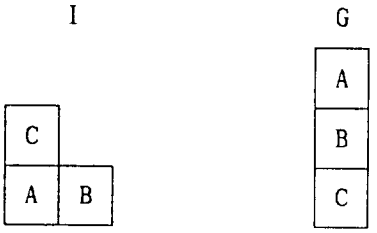

그림 4. Sussman의 Anomaly
Fig. 4. Sussman 's Anomaly.

LBW에서 C 〉 3 인 경우 초기상태와 목표상태가 일관적인 모든 문제는 해결적이다. 즉 이는 Gupta가 제시한 알고리듬에 의하여 최적계획이라는 보장은 없지만 polynomial time에 타당한(초기상태로부터 목표상태로 변화시켜줌) 계획을 수립할 수 있다. 그러나 C=2의 경우 일관적인 문제라 하더라도 교착상태를 포함하고 있으면 이 문제는 해결적이지 않다.

2. Gupta의 알고리듬

Gupta [8] 는 일관적인 LBW문제에 대한 알고리듬을 다음과 같이 제시하였다.

1) 모든 블럭을 임의의 2개 블럭리스트로 이동시킨다.
2) 목표상태에 존재하는 블럭리스트 1,2,…h-2를 완성한다. 이 이동에서 블럭리스트 h와 h-1을 이용한다.
3) h와 h-1 리스트에 남아있는 블럭을 블럭리스트 1과 2로 이동시킨다.
4) 블럭리스트 1과 2 위에 있는 블럭을 목표상태의 블럭리스트 h와 h-1에 일치하게 이동시킨다.

Gupta의 위 알고리듬은 계획의 최적성 여부보다는 polynomial 스텝에 문제의 해결성을 강조하는 성격을 갖고 있다.

Ⅴ. LBW에 대한 계획자 알고리듬

일관적인 (I,G,C)의 문제에서도 기본적 블럭세계에서 정의된 교착상태는 존재한다. 특히 교착상태 정의에서 집합 D, R에 속하는 블럭이 모두 같은 블럭리스트에 속하는 경우(유형1, 3, 5) 기본적 블럭세계에서는 교착상태인 블럭을 FS 상태로 만들면 교착상태가 해결될 뿐만 아니라 최적계획에 해당하지만 작업장이 제한된 블럭세계에서는 FS상태로 만드는 데 제약이 있기 때문에 다른 방법으로 교착상태를 해결해야 한다. 그림4에서 블럭 C에 대한 최종위치가 존재하지만 작업장의 제약(C=2인 경우) 때문에 이동이 될 수가 없으므로 문제가 해결적이지 않게 되었다. 따라서 LBW에서는 다른 개념의 도입을 통하여 교착상태의 해결을 시도해야 한다.

1. 역위치와 완벽블럭리스트

현상태에서 자신의 obove^{-1}에 위치한 블럭중 목표상태에서도 자신의 obove^{1}에 존재하는 블럭이 전혀 없을 때 이 블럭의 위치를 역위치(inverse position)라고 하자. 따라서 어떤 블럭이 교착상태 유형 1, 3, 5에 해당하지 않으면 이는 역위치가 된다. 이러한 역위치중에서도 자신의 obove^{-1}에 위치한 블럭중 적어도 하나의 블럭이 목표상태에서 자신의 obove에 있는 블럭을 강한 역위치(strong inverse position)이라고 하자. 다만 목표상태에서 헤드에 해당하는 블럭이 현상태에서 테일이면 이 블럭은 강한 역위치로 간주한다.

임의의 블럭리스트 ${}_iBL_p$ (${}_iB_1$, ${}_iB_2$, ${}_iB_3$,…,${}_iB_p$)에 대하여 목표상태에도 동일한 블럭리스트 (${}_iB_1$, ${}_iB_2$, ${}_iB_3$,…,${}_iB_p$)가 존재하는 경우 블럭리스트 ${}_iBL_p$를 완벽블럭리스트 (compact block list : ${}_iCBL_p$)라 한다. 완벽블럭리스트의 헤드위에 어떤 다른 블럭을 놓아도

이는 역위치가 된다.

2. Gupta 알고리듬의 개선안

4장에 제시한 Gupta의 LBW 문제를 위한 알고리듬을 계획자에 보다 효율적으로 적용할 수 있도록 다음과 같이 개선할 수 있다.

1) 모든 블럭을 임의의 블럭리스트 2개 위로 이동시킨다. 단 가급적 역위치에 해당하는 곳으로 이동시킨다.

2) 목표상태가 h개의 블럭리스트로 구성되어 있으면 블럭리스트 1,2,…h-2를 완성한다. 이는 블럭리스트 h와 h-1을 이용한다. 이 스텝에서는 2개 블럭리스트 헤드중 최종위치로의 이동이 가능한 블럭을 먼저 이동시킨다. 만일 이런 블럭이 없으면 헤드의 obove^{-1}에 위치한 블럭중 먼저 최종위치로의 이동이 가능한 블럭(헤드에서 가깝게 밑에 있는 블럭)을 찾는다.

3) h와 h-1 리스트에 남아있는 블럭을 1번부터 h-2번의 완벽블럭리스트로 이동시킨다. 이때 역시 역위치에 해당하는 위치로 이동시킨다.

4) 1번부터 h-2번까지의 완벽블럭리스트 위에 있으면서 최종위치가 아닌 블럭을 목표상태의 블럭리스트 h와 h-1에 일치하게 이동시킨다. 이 스텝에서도 h-2개의 블럭리스트 헤드중 최종위치로의 이동이 가능한 블럭을 먼저 이동시킨다. 만일 이런 블럭이 없으면 헤드의 obove^{-1}에 위치한 블럭중 가장 먼저 최종위치로의 이동이 가능한 블럭을 찾는다.

3. LBW 계획자를 위한 휴리스틱알고리듬

C 〉 2이고 일관적인 LBW문제에 적용할 수 있는, 계획자를 위한 휴리스틱알고리듬을 제안한다. 이 알고리듬의 각 단계에서 블럭 이동이 발생하면 이 이동(이동블럭이름, 이동장소)을 계획에 추가해 나가며 현상태의 블럭리스트도 갱신한다.

1) 주알고리듬

① 초기상태를 현상태로 한다.

② 현재상태가 목표상태와 같으면 완료한다.

③ 블럭리스트의 헤드이며 최종위치에 있지 않은 블럭중 C의 제약조건을 만족하면서 최종위치로 이동할 수 있는 블럭이 있으면 그 블럭을 옮기고 b)로 간다.

④ 이 스텝에 도달하면 다음의 휴리스틱알고리듬을 이용하여 교착상태를 해결하고 c)로 간다.

이 주알고리듬은 기본적 블럭세계에 대한 Gupta의 알고리듬과 동일하다. 다만 3번째 스텝에서 C의 제약조건을 고려하는 점이 다르다. 이 3번째 스텝에서 최종위치로 이동이 가능한 헤드가 더 이상 없으면 4번째 스텝으로 진행된다. 이 4번째 스텝은 교착상태를 해결하기 루틴으로 작업장이 제한됨점을 감안하여 다음과 같이 설계하였다.

2) 교착상태 처리를 위한 휴리스틱알고리듬

이 알고리듬은 최종위치에 있지 않은 헤드를 역위치로 옮겨서 교착상태의 해결을 시도한다. 이 이유는 최종위치에 있는 헤드를 역위치로 옮기는 것은 무의미하고, 또한 현재 역위치에 있는 것을 다른 역위치로 옮기면 사이클 현상이 발생할 수 있기 때문이다.

① 각 블럭리스트의 헤드중 목표상태에서도 헤드인 블럭이 있으면 이 블럭을 다음 순서에 따라 처리하고 주알고리듬 c)로 간다.

보충설명: 현상태에서 헤드에 있는 블럭중 목표상태에서도 헤드인 블럭은 이 블럭이 최종위치로 이동될 때 가장 늦게 최종위치로 옮겨져야 하기 때문에 먼저 역위치가 있으면 이동시키는 것이다. 역위치의 선정도 다음과 같은 순위로 처리한다.

a) 만약 현재 블럭리스트의 갯수가 C보다 적다면 위 블럭을 FS상태로 만든다.

b) 완벽블럭리스트가 존재하면 이 리스트의 헤드위로 옮긴다.

c) 최종위치가 아닌 블럭중 역위치조건이 성립하는 블럭위로 옮긴다.

d) 최종위치에 해당하는 블럭수가 가장 적은 블럭리스트 위로 옮긴다.

e) 위 3조건에 해당하지 않으면 역위치로의 이동도 불가능하므로 길이가 가장 작은 블럭리스트의 헤드를 길이가 가장 긴 블럭리스트의 헤드로 옮긴다.

② 각 블럭리스트의 헤드에 대하여 강한 역위치로의 이동이 가능한 블럭이 있으면 이 블럭을 강한 역위치로 옮기고 주알고리듬 c)로 간다.

- 최종위치가 아닌 블럭중 강한 역위치조건이 성립하는 블럭위로 옮긴다.

③ 교착상태 유형 1,2,3,5에 해당하는 헤드를 다음 순서에 따라 처리하고 주알고리듬 c)로 간다.

a) 완벽블럭리스트가 존재하면 위 블럭을 이 완벽블럭리스트 위로 이동시킨다.

b) 완벽블럭리스트가 없는 경우 C가 현상태의 블럭리스트의 갯수보다 크면 위 블럭을 FS상태로 만든다.

c) 최종위치가 아닌 블럭중 역위치조건이 성립하는 블럭위로 옮긴다.

d) 위의 2 조건이 만족되지 않으면 위 블럭을 최종위치에 있는 블럭만 포함하고 있는 블럭리스트

중 길이가 가장 적은 블럭리스트에 이동시킨다.

e) 위 3조건에 해당하지 않으면 역위치로의 이동도 불가능하므로 길이가 가장 작은 블럭리스트의 헤드를 길이가 가장 긴 블럭리스트의 헤드로 옮긴다.

④ 최종위치가 아닌 헤드중 헤드의 on[1]에 위치한 블럭이 최종위치이면 스텝 c)에서의 순위에 따라 처리한다. 단 이헤드가 역위치로의 이동(a와 c스텝의 4순위에 의한 이동)에 의하여 이동되었던 블럭이 아니어야 한다.

⑤ C에 여유가 있는 경우에는 헤드가 역위치도 아니고 최종위치도 아닌 것중 가장 길이가 긴 블럭리스트의 헤드를 이동시킨다. C에 여유가 없는 경우에는 헤드가 역위치도 아니고 최종위치도 아닌 것중 가장 길이가 작은 블럭리스트의 헤드를 이동시킨다.

- 이때의 이동 장소는 스텝 c)에서의 순위에 따른다.

4. 알고리듬의 복잡성

n을 블럭문제에서의 블럭갯수라 하면 위 알고리듬의 공간 복잡성은 O(n)이다. 왜냐하면 현상태와 목표상태에서의 블럭리스트만 유지하고, 각 블럭의 위치 특성(최종위치 또는 역위치 여부), 계획에 선택된 연산자들만 유지하면 되기 때문이다.

한편 각 스텝의 시간 복잡성은 다음 표2와 같다.

표 2. 알고리듬의 시간 복잡성

Table 2. time complexity of algorithm.

스텝구분	세부 기능	시간 복잡성
주 알고리듬	a 초기화	O(n)
	b 목표달성여부	O(nlogn)
	c 최종위치이동가능	$O(n^2)$
교착상태 처리 알고리듬	a 조건	$O(n^2)$
	작업장 조건	O(n)
	완벽블럭리스트	$O(n^2)$
	역위치	$O(n^4)$
	최종위치 적은 곳	O(nlogn)
	짧은 블럭리스트	O(nlogn)
	b 조건	$O(n^5)$
	c 조건	
	교착상태 유형 1	$O(n^2)$
	교착상태 유형 2	$O(n^{d+1})$
	교착상태 유형 3	$O(n^4)$
	교착상태 유형 5	$O(n^3)$
	이동위치 a와 동일	
	d 조건 이동위치 a와 동일	$O(n^2)$
	e 조건 이동위치 a와 동일	O(nlogn)

위의 표에서 보면 주알고리듬의 시간복잡성은 $O(n^2)$이 되며, 교착상태처리루틴은 $O(n^{\max[5,\ d+1]})$가 된다. 한편 n개의 블럭으로 구성된 문제에 대하여 이 알고리듬은 각 블럭에 대한 최대이동횟수를 고려하면 초기상태의 위치에서 역위치로 이동, 그 후 역위치에서 최종위치로의 이동이 될 수 있다. 따라서 이 알고리듬은 n개로 구성된 문제에 대하여 계획을 수립해 주는 경우 2n 횟수 이하의 블럭이동을 나타내는 계획을 생성한다. 즉 이 알고리듬이 계획을 수립하는 데에는 최대 2n회만큼 각 스텝을 반복하여 수행해야 한다. 따라서 이 알고리듬의 시간복잡성은 $O(n^{\max[6,\ d+2]})$이 된다.

5. 구현

위에 설계한 알고리듬을 Golden Common LISP으로 MS-DOS에서 구현하였다. 이 구현에서는 다음과 같은 자료의 구분이 필요하다.

1) 현상태의 블럭리스트는 3종류의 블럭리스트로 구분한다.
   - FPL (final position list) : 최종위치에 해당하는 블럭리스트의 집합
   - DAL (deadlock list) : 헤드가 최종위치에 있지 않고, 최종위치로 이동될 수 없는 블럭리스트의 집합
   -CHKL (check list) : 최종위치로의 이동가능 여부가 필요한 블럭리스트의 집합
   -위와 같이 구분하면 FPL, DAL, CHKL의 공통부분은 공집합이며, 합집합은 현상태가 된다.
2) 블럭이 현재 최종위치에 있는 지, 역위치에 있는 지 구분하기 위하여 다음과 같이 블럭의 특성을 부여한다.
   - 최종위치에 있는 경우 특성값을 'F' (final position의 의미)
   - 역위치에 있는 경우 특성값을 'I' (inverse position의 의미)
   - 초기상태에는 'A' (arbitrary position의 의미)

본 연구에서 제안하는 LBW를 위한 계획자 알고리듬의 구현결과를 비교하기 위하여 Gupta의 LBW를 위한 알고리듬, 이 알고리듬의 개선안, 앞서 3장에 제시한 EBW를 위한 알고리듬을 모두 MS-DOS에서 Golden Common LISP으로 구현하였다.

6. 성능평가

블럭수가 10개, 20개 , 30개, 40개로 구성된 문제를 랜덤하게 작성하여 작업장의 수(C)를 변화시켜가면서 계획을 수립하였다. 표 3은 이 결과 계획에 포

합된 블럭의 총 이동횟수를 나타내고 있다. 본 논문에서 제안하는 알고리듬의 성능비교를 위하여 Gupta 알고리듬에 의하여 수립된 계획의 블럭이동횟수, V장 2절에 제시한 개선된 Gupta알고리듬에 의하여 수립된 계획의 블럭이동수도 같이 표기하였다. 앞에서 설명한 바와 같이 LBW문제에서 C의 제약을 없애면(C= ∞) EBW 문제가 되는 데, 이 문제에 대해서는 앞서 3장에 제시한 EBW를 위한 알고리듬에 의해 수립된 계획의 블럭이동수를 함께 표기하였다.

블럭수가 10개인 문제에서는 목표상태 블럭리스트 갯수는 3개였다. 이때 작업장 갯수 C가 4이하인 경우에는 본 알고리듬에 의해서는 계획이 수립되지 않았으며, C가 5일 때는 블럭의 총이동횟수가 18회이였다. C가 6이상인 경우에는 모두 블럭의 이동횟수가 14회이였다. Gupta의 알고리듬에 의한 경우는 블럭의 총이동횟수가 37회였으며, 개선된 Gupta의 알고리듬은 이 문제에 적용이 안된다(C=3이므로). C에 제약이 없는 경우 블럭의 이동횟수는 13회였다.

만일 어떤 EBW 문제가 블럭 n개로 구성되어 있다면 2n회 이하의 블럭이동에 의하여 초기상태로부터 목표상태에 도달할 수 있다. 왜냐하면 EBW에서는 작업장의 제약이 없으므로 모든 블럭을 FS상태로 만들 수 있다(블럭 n개 이동). 모든 블럭이 FS상태에 있으면 교착상태가 전혀 없으므로 모든 블럭이 1회 이내의 이동에 의하여 최종위치로 옮겨질 수 있기 때문이다.

이 표에서 "-"로 표기된 경우는 C의 제약으로 인하여 이 본 논문에서 설계한 휴리스틱 알고리듬으로는 계획의 수립이 곤란한 경우이다. 즉 교착상태 처리에서 블럭을 이동시킬 장소가 부족하여 발생하는 현상이다. 이런 경우는 2절에 제시한 개선된 Gupta의 알고리듬을 이용해야 한다. 또한 C의 값이 증가하여도 어느 이상으로는 계획에 포함된 블럭의 이동횟수가 감소하지 않았는 데, 이는 역위치로의 이동을 수행할 때 FS상태를 만들어 블럭리스트를 새로 생성하는 하기 때문에 발생하는 현상이다. 한편 이 표에서 블럭수가 20개인 경우 C가 10일 때의 스텝수는 29인데도 불구하고 C가 11일 때 스텝수가 34로 증가한 것은 작업장의 여유가 있을 때 교착상태 해결시 FS상태로 만드는 것을 우선적으로 하는 데 기인한다. 그러나 전체적으로 적절한 C 값의 범위 내에서는 작업장의 능력이 커짐에 따라 전체적으로 블럭의 이동횟수는 줄어드는 것을 알 수 있다.

Ⅵ. 결 론

본 연구에서는 Gupta가 정의한 블럭세계의 교착상태를 유형으로 분류하고, 이 성질을 분석하여 교착상태처리 방법을 제시하였다. 한편 작업장이 제한된 블럭세계에서는 목표상태에서 블럭리스트의 테일에 위치한 블럭이라 할지라도 작업장의 여유가 있을 때에만 최종위치로 이동이 이루어질 수 있다. 따라서 작업장의 제약이 있는 경우의 블럭세계의 문제에서는 작업장을 여유있게 만들어 주는 것이 필요하다. 그리하여 본 연구에서의 계획자는 역위치의 개념을 이용하고, 블럭세계에 내재하는 교착상태의 성질을 이용한 알고리듬에 근거하고 있다. 그러나 작업장 크기제약조건의 모든 범위에 대하여 계획을 수립해 주지는 못하고 있다. 즉 어느 한계값 이하에 대해서는 계획을 생성하지 못하며, 어느 한계값 이상에 대해서는 모두 동일한 계획을 생성하고 있다. 그러나 이러한 값의 사이값에 대해서는 최적에 근사한 계획을 수립하고 있다.

표 3. 성능평가 결과
Table 3. performance evaluation result.

총블럭수 / 목표상태 스택수		10 / 3	20 / 4	30 / 5	40 / 6
알	5	18	-	-	-
	6	14	-	-	-
	7	14	-	-	-
	8	14	33	-	-
	9	14	34	51	-
제 고	10	14	29	46	-
	11	14	34	46	66
안 리	12	14	32	46	65
	13	14	29	45	64
듬	14	14	29	45	64
	15	14	29	45	63
Gupta알고리듬		37	115	196	360
개선된 Gupta 알고리듬		-	86	120	197
C의 제약이 없을 경우		13	25	43	57

參 考 文 獻

[1] R.E.Fikes, and N.J.Nilsson, "STRIPS : A New Approach to the Application of Theorem Proving to Problem Solving", Aritificial Intelligence, Vol. 2, 1971 ,

pp. 189-208.
[2] E. D. Sacerdoti, "Planning in a Hierarchy of Abstraction Spaces", Artificial Intelligence, Vol. 5 No. 2 , 1974 ,pp.115-135.
[3] E. D. Sacerdoti, "The Nonlinear Nature of Plans", Proceedings of IJCAI 4, 1975, pp. 206-214.
[4] D. Wilkins, "Domain-independent Planning: Representation and Plan Generation", Artificial Intelligence, Vol. 22, No. 3, 1984, pp. 269-301.
[5] D. Chapman, "Planning for Conjunctive Goals", Artificial Intelligence 32, No. 3, 1987, pp. 333-378.
[6] Q. Yang, and J.D.Tenenberg, "ABTWEAK: Abstracting a Nonlinear, Least Commitment Planner", Proceeding of AAAI, 1990 , pp. 204-209.
[7] N.Gupta, and D.S.Nau, "Complexity Results for Blocks-World Planning", Proceedings of AAAI ,1991, pp. 629-633.
[8] _____, " On the Complexity of the Blocks-World Planning", Artificial Intelligence, Vol. 56, 1992 , pp. 223-254.
[9] 심동희, 황종선, "확장된 블럭세계를 위한 계획자에 대한 연구", 한국정보과학회 학술발표논문집, Vol. 19, No.2 ,1992.
[10] F. P. Preparata, and R.T.Yeh, Introduction to Discrete Structures, Addison-Wesley, 1973.

著 者 紹 介

沈 東 熙(正會員)
1980年 2月 서울대학교 산업공학과(공학사). 1982年 2月 서울대학교 대학원(공학석사). 1994年 2月 고려대학교 대학원 전산과학과(이학박사). 1990年 ~ 현재 전주대학교 전자계산학과 조교수. 주관심 분야는 Reasoning, Machine Learning, Expert System 등임.

黃 鍾 善(正會員) 第 31卷 B編 第 5號 參照
현재 고려대학교 전산과학과 교수