기본적 블럭세계를 위한 선형계획자

심 동 희†

요 약

블럭세계는 인공지능연구에서 대표적인 문제영역인 바 지금까지 많은 계획자들이 제안되어 이용되어 왔다. Gupta는 블럭세계에서의 교착상태를 정의하였으며,이로 인하여 블럭세계의 의사결정문제는 NP-Complete임을 밝혔다. 본 연구에서는 블럭세계에서의 교착상태 성질을 분석하였으며, Gupta의 비결정적 알고리즘을 휴리스틱하게 접근하여 블럭세계에 대한 계획자에 이용가능하게 하였다.

Linear Planner for the Elementary Blocks World

Donghee Shim†

ABSTRACT

The blocks world is the primary problem domain in the artificial intelligence. Many planners have been developed in domain-independent. Gupta defined the deadlock in the blocks world, and he proved that decision making in the blocks world is NP-Complete problem. In this paper the properties of the deadlock are analyzed, and a heuristic algorithms which can handle the deadlock of the Gupta's nondeterministic algorithm is proposed.

1. 서 론

인공지능의 계획분야에서 가장 많이 이용되는 문제영역은 블럭세계이다. 그리하여 STRIPS[3], ABSTRIPS[7], NOAH[8], SIPE[9], TWEAK[1], ABTWEAK[10] 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)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 다를 수 있고, 블럭이름의 중복도 허용하는 경우이다.

Gupta[4, 5]는 이러한 블럭세계의 의사결정문제가 블럭간의 교착상태로 인하여 NP-Complete 문제임을 증명하였다. Gupta는 이 증명에서 블럭세계에서의 교착상태를 최초로 정의하였으며, 비결정적 다항식 계획자 알고리즘 (Nondeterministic Polynomial Planner Algorithm)을 제안하였다. 그리하여 블럭세계에서 최적계획을 수립하는 문제는 NP-Hard가 된다. 따라서 위와 같은 영역독립적 계획자로 블럭세계를 다루는 데에는 그 한계가 있다. 본 연구에서는 Gupta가 정의한 교착상태를 유형별로 분류하여 성질을 분석하였다. 또한 Gupta가 제시한 비결정적 알고리즘을 계획자에 이용할 수 있도록 교착상태의 성질을 이용하여 휴리스틱 알고리즘으로 제시하였으며, 이에 대한 성

※이 논문은 1996년도 전주대학교 학술연구조성비에 의하여 작성되었음.
† 종신회원: 전주대학교 컴퓨터공학과
논문접수: 1996년 5월 21일, 심사완료: 1996년 9월 17일

능평가를 하였다.

2. 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 ${}_IBL_N=({}_IB_1, {}_IB_2,..., {}_IB_N)$는 on$({}_IB_1, {}_IB_2)$ ^on$({}_IB_2, {}_IB_3)$ ^... on$({}_IB_{N-1}, {}_IB_N)$ ^ontable$({}_IB_N)$ ^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 M개의 블럭리스트 즉 $({}_1BL_{N1}, {}_2BL_{N2},..., {}_MBL_{NM})$으로 표현된다.

2) above(A, B)는 블럭A가 블럭B보다 위에 놓여있는 상태를 나타내되 on(A, B)는 제외한다. 즉 이는 on(A, X_1) ^on(X_1, X_2) ^... ^on(X_n, B) (단 n 는 1 이상, 이러한 X_i를 A와 B의 중간블럭이라함)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 ${}_IBL_N=({}_IB_1, {}_IB_2,..., {}_IB_C, X, {}_IB_{C+1},..., {}_IB_N)$에 대하여 목표상태에는 어떤 블럭리스트 ${}_JBL_K=({}_JB_1, {}_JB_2,..., {}_JB_A, X, {}_IB_{C+1}, {}_IB_{C+2},..., {}_IB_N)$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.

3. 블럭세계의 교착상태

3.1 Gupta의 교착상태 정의 및 계획자를 위한 비결정적 알고리즘

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

1) 상태 s에서 above(d_1, r_1) ^above(d_2, r_2) ^... ^above(d_p, r_p)

2) 목표상태 g에서 above(d_1, r_2) ^above(d_2, r_3) ^... ^above(d_p, r_1)

Gupta는 블럭세계의 복잡성을 분석하기 위하여 다음과 같은 비결정적 알고리즘을 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 알고리즘에서 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 스텝 4가 비결정적임을 이용하여 EBW에서의 계획길이문제는 NP-Complete 에 해당함을 증명하였다. 따라서 EBW에서 어떤 주어진 계획문제에 대한 최적계획 수립문제는 NP-Hard에 해당한다. 그러나 Gupta의 알고리즘은 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.

3.2 교착상태의 성질

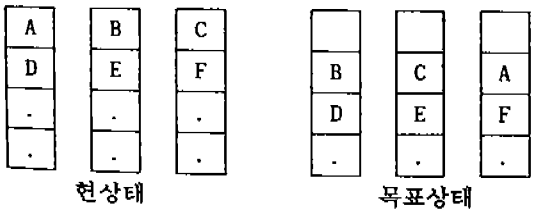
(1)교착상태의 유형분류

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

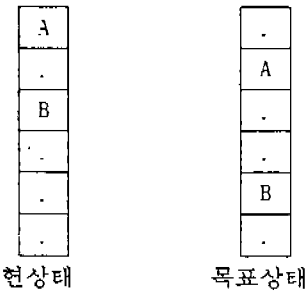
블럭리스트수 / 서술자		1개	2개 이상
상태 S	목표상태		
On	On	유형 1	유형 2
Above	Above	유형 3	유형 4
On	Above	유형 5	유형 6
Above	On		
On과 Above	On과 Above	해당없음	

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 세 개의 블럭리스트에 속하며, on의 관계에 있는 블록들 간의 교착상태이다. 단 이 그림에서 블럭 A, B, C가 반드시 헤드일 필요는 없다. 만약 이 배치에서 블록 간의 관계가 above가 되면 유형 4에 해당한다.



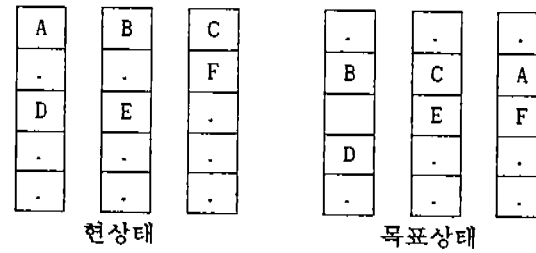
(그림 1) 유형2의 교착상태
(Fig. 1) deadlocks type 2

그림2에는 유형3을 나타냈다. 집합 D에 속한 블럭 A가 집합 R에 속한 블럭 B에 대하여 교착상태이다. 즉 한 개의 블럭리스트에 속하며, above관계에 있는 블록들 간의 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

그림3에는 유형6을 나타냈다. 집합 D에 속한 블럭 A, B, C가 집합 R에 속한 블럭 D, E, F에 대하여 교착상태이다. 즉 세 개의 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 알고리즘에 의한 계획을 수립하다가 스텝4에 도달하면 이는 상태공간

상의 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 여러 개가 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 Diagram[10]이라는 그래프에서는 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.

4. 계획자를 위한 알고리즘

4.1 교착상태 처리를 위한 휴리스틱알고리즘

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

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

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

③위의 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.

④조건3에 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.

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

이 휴리스틱 알고리즘을 Gupta의 비결정적 알고리

즘의 스텝4로 사용하면 기본적 블럭세계를 위한 계획자 알고리즘으로 이용할 수 있다.

4.2 알고리즘의 평가

〈표 2〉 알고리즘 각 단계의 복잡성
〈Table 2〉 Time Complexity of Algorithm

구분	세부 기능	시간 복잡성
주 알고리즘	초기화	O(n)
	목표상태 도달여부 조사	O(nlogn)
	최종위치로의 이동가능 블럭조사	$O(n^2)$
교착상태 처리를 위한 휴리스틱 알고리즘	교착상태 유형 1, 3, 5 파악처리	$O(n^3)$
	교착상태 유형2파악 및 처리	$O(n^d)$
	각 블럭리스트의 헤드 on^{-1} 블럭의 조건에 따라 처리	$O(n^2)$
	각 블럭리스트의 헤드 $above^{-1}$ 블럭의 조건에 따라 처리	$O(n^3)$
	가장 긴 블럭리스트 처리	O(n)

주알고리즘의 시간복잡성은 $O(n^2)$이며, 교착상태처리를 위한 휴리스틱 알고리즘의 시간복잡성은 $O(n^{\max(3,\ d)})$이다.

한편 이 알고리즘에서는 현상태와 목표상태에서의 블럭리스트를 유지하고, 계획에 선택된 연산자만 유지하면 되기 때문에 공간복잡성은 O(n)이다.

4.3 알고리즘의 구현

교착상태 처리를 위한 휴리스틱 알고리즘을 Golden Common LISP으로 IBM PC에서 구현하여 이를 임의로 처리한 경우와 비교하였다. 여기서 임의의 처리는 교착상태에 도달한 경우 블럭리스트의 길이가 가장 길거나 짧은 것의 헤드를 FS상태로 만드는 것이다.

교착상태를 포함하지 않는 경우에는 항상 최적계획을 수립하기 때문에 교착상태가 포함된 문제를 설정하였는 데, 블럭의 수가 10개, 20개, 30개, 40개인 경우에 대하여 본 연구에서 제안한 휴리스틱 알고리즘에 대하여 유리한 문제(교착상태 유형 1, 2, 3, 5를 포함)와 불리한 문제(교착상태 유형 4, 6을 포함)를 설정하여 계획을 수립하였다. 〈표3〉과 〈표4〉은 각 알고리즘별 교착상태의 휴리스틱 처리횟수와 계획의 총횟수를 나타내고 있다.

〈표 3〉 휴리스틱 알고리즘에 대해 유리한 경우
〈Fig 3〉 Good Case for Heuristic Algorithm

블럭수		10		20		30		40	
알고리즘		DD	ST	DD	ST	DD	ST	DD	ST
임의의 처리	긴 블럭리스트 처리	4	12	8	25	10	39	19	52
	짧은 블럭리스트 처리	3	11	8	25	9	38	7	40
제안 알고리즘		3	11	4	21	5	34	7	40

(주) DD: 교착상태 발생횟수
ST: 압축계획의 총 스텝수

〈표 4〉 휴리스틱 알고리즘에 대해 불리한 경우
〈Fig 4〉 Worse Case for heuristic Algorithm

블럭수		10		20		30		40	
알고리즘		DD	ST	DD	ST	DD	ST	DD	ST
임의의 처리	긴 블럭리스트 처리	3	13	7	27	21	48	29	83
	짧은 블럭리스트 처리	4	14	7	27	21	48	28	82
제안 알고리즘		3	13	5	25	18	41	20	54

(주) DD: 교착상태 발생횟수
ST: 압축계획의 총 스텝수

위 표에서 보아 알 수 있듯이 모든 경우에 휴리스틱 알고리즘에 의해 수립된 계획이 임의의 처리에 의해 수립된 계획보다 교착상태의 발생횟수가 적음을 알 수 있다.

일반적으로 N개의 블럭으로 구성된 블럭문제는 2N개의 블럭이동을 포함하게 된다. 문제는 이러한 계획이 최적인지의 여부와 Polynomial Time내에 도출되느냐 하는 것이다. 여기서 수립된 계획은 최적에 근사한 것이다. 최적이 아닌 경우는 위의 교착상태 처리알고리즘의 휴리스틱 처리루틴에 기인한 부분이다. 즉 여기서 교착상태를 해결하기 위하여 선택된 블럭을 테이블에 내려놓아 FS상태로 만들어도 교착상태가 해결되지 않는 경우에는 최적계획이 아닐 가능성이 높아진다. 이때 작성되는 계획은 최적계획보다 이러한 경우의 횟수만큼 더 많은 블럭이동을 포함하게 된다.

탐색의 측면에서 이 알고리즘은 Steepest-Ascent Hill

Climbing에 해당한다. 다만 각 노드의 평가는 Final Position으로의 이동에 해당하는 블럭에 근거하기 때문에 Hill Climbing이 갖고 있는 Local Maximun, Plateau, Ridge 등의 현상이 발생하지 않게 된다.

5. 결 론

본 연구에서는 Gupta가 정의한 블럭세계의 교착상태를 각각의 유형으로 분류하고 이 성질을 분석하여 교착상태 처리를 위한 휴리스틱 알고리즘을 제안하였다. 이 휴리스틱 알고리즘은 기본적 블럭세계에 대한 계획을 최적해에 근사하게 수립해 준다. 또한 이 알고리즘은 Conflict-Free이며 완전순서화되고, Backtracking이 발생하지 않는다. 그러나 단점으로는 최적계획을 보장하지는 못한다는 점이다.

참 고 문 헌

[1] D. Chapman, "Planning for Conjunctive Goals", Artificial Intelligence Vol. 32, PP. 333-378, 1987.

[2] J. Christensen, "A Hierarchical Planner that Generates its Own Hierarchies", Proceedings of AAAI, 1990.

[3] R. E. Fikes,and N. J. Nilsson, "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving", Aritificial Intelligence, Vol. 2, pp. 189-208, 1971.

[4] Gupta N., and Nau D. S., "Complexity Results for Blocks-World Planning", Proceedings of AAAI ,1991.

[5] _____, "On the Complexity of the Blocks-World Planning", Artificial Intelligence, Vol. 56, pp. 223-254, 1992.

[6] C. A. Knoblock, J. D. Tenenberg, and Q. Yang, "Characterizing Abstraction Hierarchies for Planning", Proceeding of AAAI, 1991.

[7] E. D. Sacerdoti, "Planning in a Hierarchy of Abstraction Spaces", Artificial Intelligence, Vol. 5, pp. 115-135, 1974.

[8] _____, "The Nonlinear Nature of Plans", Proceedings of IJCAI 4, 1975.

[9] D. Wilkins, "Domain-independent Planning: Representation and Plan Generation", Artificial Intelligence, Vol. 22, 1984.

[10] Q. Yang, and J. D. Tenenberg, "ABTWEAK: Abstracting a Nonlinear, Least Commitment Planner", Proceeding of AAAI, 1990.

심 동 희

1980년 서울대학교 산업공학과 졸업(학사)

1982년 서울대학교 대학원 산업공학과 졸업(공학석사)

1994년 고려대학교 대학원 전산과학과 졸업(이학박사)

1982년~1985년 국토개발연구원

1985년~1990년 해운산업연구원

1990년~현재 전주대학교 컴퓨터공학과 부교수

관심분야: 기계학습, 전문가 시스템, 신경망 이론