바둑에서 휴리스틱 함수를 사용한 사활문제 풀이에 관한 연구

박 현 수*, 임 중 권**, 이 종 철***

Solving the Life-Death Problems Using Heuristic Function in Baduk

Hyun-Soo Park, Jung-Gweon Lim, and Chong-Cheol Lee

Abstract As an improvement in solving life-death problems of Baduk, we proposed a method of applying a new heuristic evaluation function to the ordered AND/OR search trees of baduk game. Our method consist of pattern matching for small problems and for larger problems, we consider cut vertex of an inner_space graph, special vertex, attack vertex, critical vertex, and eye_make vertex in due order.

I. 서 론

바둑은 흑과 백이 서로 번갈아 두어 자신의 영역을 차지하면서 적을 포획하여, 더 이상 둘 곳이 없는 상태가 되면 계가를 통하여 승/패를 판가름한다. 바둑을 두는 방법은 크게 감과 수 읽기로 구분 할 수 있다. 그러나 이 둘은 아주 밀접한 관계를 가지고있다.

간단히 말하자면 반면을 시각적 기능으로 보면서 큰 곳 급한 곳 등을 감으로 찾는데 이렇게 찾은 후보는 여러 개 일 것이다. 그 중에서 자신의 바둑진행에서 가장 중요한 수를 선택 하기 위해 후보들을 모두 조사해 보아야 한다.

그러므로 이들 후보 중 첫 수를 정하고 이 수로 수읽기를 하여 유리한 바둑인지를 검사해 본다. 그리고 나머지 모든 후보도 수 읽기로 검사 하여 가장 호수를 다음 수로 선택한다. 이러한 과정들을 세부적으로 보면 우선 자신의 모든 경험과 감각으로 다음에 착수할 여러 가지 중 제1감으로 떠오르는 수가 기초가된다. 그리고 이것으로 상대가 취할 수 있는 경우의 수를 모든 수 읽기를 하여 본다.

그리고 자신의 제일 착을 다른 지점으로 옮겨 제2착,제3착의 구도를 그려 나간다. 이렇게 몇 개의 구도가 완성되면 이번에는 그것을 비교, 검토하게된다. 그리고 구도 중 최선의 것을 선택하여 한가지만을 남기는데 이것이 다음에 착수 할 수로 정해 지는 것이라 할 수 있을 것이다. 이러한 감과 수 읽기를 기본으로 바둑문제를 해결하기 위한 여러 가지 접근인 정석,수순,사활을 비롯하여 공격과 방어,수습 더 나아가서는 마무리인 끝내기에 이르도록 많이 있지만 특히 감과 수 읽기의 문제를 단적으로 보여 주는 것이 사활문제이다. 또한, 사활문제는 바둑의 핵심이라할 수 있다. 왜냐하면 삶과 죽음의 판단은 다음 착수의 효율을 극대화 시킬 수 있는 기본 조건이기 때문이다.

즉 바둑의 기본이 적보다 많은 집을 확보하는 것이며, 이것은 필수적으로 적을 많이 포획하면서 자신은 잡히지 않는 전략을 기본으로 한다. 그러므로 적을 많이 포획하고 자신이 살기 위해서는 사활문제를 확실히 해결해야만 한다. 그러나 바둑문제의 광대함이 전체바둑 상에서 사활 판단 문제를 어렵게 하므로 기존의 문제풀이형 사활문제만을 국한하여 판단 하고자 한다.

1994년 6월 30일 접수
* 경북대학교 대학원 컴퓨터공학과 석사과정
** 삼보정보통신
*** 경북대학교 공과대학 컴퓨터공학과 부교수

그러나 문제풀이형 사활 문제도 단순히 착수 가능한 수를 다 해보는 방법은 무의미 하다. 왜냐하면 그림1처럼 (a,4),(g,1)의 내려 서기를 포함해서(사활에 관계되는 (a,4), (g,1)의 안쪽) 11! (대략 4000만)이 되고 여기에 따냄의 수를 생각한다면 더욱 큰 수가 되어진다.

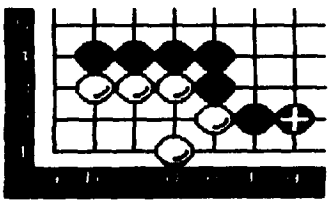

그림 1. 사활 문제 예
Fig. 1. Example of life-death problem.

그리고 착수 가능한 공배가 하나씩 늘어 남에 따라 전체 경우의 수는 기하급수적으로 늘어난다. 그러므로 불필요한 경우를 제외시키고 발생될 경우 중 빠른 시간에 해를 찾을 수 있도록 수 진행을 조작하는 것이 필수적이다. 이러한 사활문제를 해결 한 기존의 순서화된 AND/OR PRUNING 방법은 평가값으로 탐색 횟수를 감소시키고 정적분석을 통하여 TREE의 확장을 막아서 탐색TREE의 크기를 줄이고자 하였다[1]. 앞의 논문의 평가함수의 parameter로 쌍방간의 돌 수와 덩어리 수, 빈자리 수,단수된 수 등으로 구성 하였지만 이것은 인간의 사활문제 풀이 방식과는 큰 차이가 난다. 즉, 인간은 사활문제를 풀 때 빈자리의 모양과 빈자리의 수(넓이)에 대해 두 집이 나지 못하게 급소,맥,치중,그리고 젖힘 등으로 빠른 시간 내에 해를 구한다. 또한 죽는 모양을 기억하여 필요 없는 수 읽기를 하지 않는다. 이러한 인간적 사고 방식인 감과 수 읽기에 보다 근접한 새로운 평가함수를 제안하며 이와 함께 사/활 PATTERN을 만들어 사는 형태인지 죽는 형태 인지를 PATTERN(모양)으로 판단한다. 또한 방법 상에서 전체바둑에 접목도 유리하리라 판단된다.

II. 용 어 정 의

기존의 사활 문제에 대한 풀이 방식을 보면 순서화된AND/OR pruning 방법에서 순서화시키는 평가 함수식은 빈자리와 돌 수 와 덩어리 수 그리고 단수된 돌 수에 대해 흑/백의 차로 구했다[1].그리고 이 평가 함수값으로 순서화된 tree를 발생 시켜 수 읽기를 실행하게 된다. 그러나 인간은 사활 문제를 풀 때 빈자리의 모양과 빈자리의 수(넓이)에 대해 치중하는 방법과 젖히는 방법으로 두 집이 나지 못하게 하는 방법을 사용한다(결국은 포획한다). 인간의 능력은 모양에서의 급소나 치중 그리고 젖힘 등의 감이 아주 뛰어나(일반적으로 고수일 수록 더 뛰어남) 빈자리 모든 것에 대해 수진행을 하지는 않는다. 그러므로 본 논문은 감과 대응하는 순서화시키는 함수를 구하기 위해 파라미터로 치중과 젖히는 수를 가지고 계산하는 휴리스틱한 방법을 사용한다. 인간은 시각적 기능으로 반면에서 정보를 얻고 나서 감과 수 읽기를 통하여 사활문제를 해결한다. 여기서 치중은 급소라 할 수 있어 대부분 모양에서 비롯된다. 젖힘은 기본적으로 내부의 면적을 감소 시킴으로 적이 살 수 있는 집의 모양을 만들지 못하게 하여 포획을 하려는 방식이다.그리고 마지막으로 수 읽기는 모든 가능 수를 진행시켜 적의 죽음을 유도 할 수 있는 수를 찾는 방법이라 하겠다. 그래서 문제는 어떻게 이 수를 빠른 시간 내에 정확히 찾을 수 있는 가가 핵심이라 할 수 있다.

기존의 제안한 순서화된 AND/OR PRUNING 방법은 단순히 수진행의 방법을 개선 하였다고 볼 수 있다. 그러나 앞에서 말한 것처럼 바둑을 잘 두는 사람 일 수록 감과 수 읽기가 뛰어나 정확히 사활을 판단 할 수 있는 것이다. 즉, 제1감에서 수 읽기 후에 답을 찾는다면 아주 훌륭하다 하겠다. 그러나 아무리 프로라 할지라도 모든 문제를 제1감에서 찾지는 못하듯이 몇까지 후보 중에서 수 읽기 후 해를 찾을 수 있는 것이다. 그리고 이러한 수 읽기가 필요치 않은 모양이 있는데 여기에는 죽는 모양과 사는 모양이 정해져 있다. 이러한 문제는 가장 기본적인

사활문제이다. 이때는 수 읽기를 하는 것은 낭비이며 이미 알고 있는 사람은 한번 봄으로 해를 찾아낸다. 그러므로 사는 모양/죽는 모양을 많이 기억 하여 필요 없는 수 읽기를 줄인다. 예를 들어 그림 2처럼 기본적인 형태의 모양인 3집,정사궁, 5사궁,귀곡사,매화6궁,2선에서 7궁,번개궁 등은 단 한 수의 착수만 보면 사활이 판단되어진다.

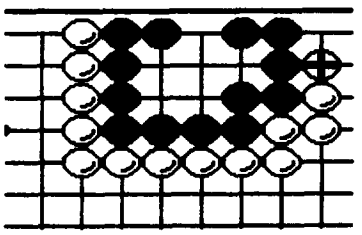

그림 2. 죽은 형태의 예
Fig. 2. Example of pattern of dead state.

이처럼 모양이 정형화된 PATTERN을 미리 정하여 비교하여 만약 이런 모양이 맞아지면 바로 사활이 판단 가능하다고 볼 수 있다. 그러므로 문제 해결을 위해 계속 수 진행을 하는 동시에 pattern matching을 하여 tree의 발생을 최소화 시켜 효율적으로 사활문제를 해결 할 수 있다.

1. 용어 정의

바둑판 B는 아래 조건을 만족하는 planar graph이다.

정의1.1] B=(V,E)

a. V = { (i, j) | 1≤ i,j ≤ 19},

b. E = {((i,j),(i'j')) | (i = i'∧|j-j'| =1) ∨ (|i-i'| = 1 ∧ j = j') },

c. f : V -> { black, white, empty}.

바둑판 B상에 돌(stone)는 함수 f에 의해 집합 {black,white}로 사상 되어진 집합 V.

정의1.2] S = { (i,j) | f((i,j)) = {black,white}, 1≤ i,j ≤19}

(i,j)와(s,k)가 완전히 연결 되었다.는 아래와 같이 정의한다.

정의1.3] (i,j),(s,k)∈V and f((i,j)) = f((s,k)),(i,j) con(s,k)={((i=s)∧(j=k))∨∃(x,z)∈V(B)(f((x,z))= f((i,j))∧(((i,j),(x,z))∈E(B)∧((x,z) con (s,k)))}.

마름모 연결(Marym mo)은 아래와 같이 정의한다.

정의1.4] (i,j),(s,k),(x,y),(w,z)∈V(B) 1≤i,j,s,k,x, y,w,z ≤ 19)

((i,j), (s,k)∈MM(B) iff f((i,j))=f((s,k))={black, white} and f((x,y)) = f((w,z)) = {empty} and

a. (s=i-1 and k=j-1) and (x=i-1 and y=j) and (w=i and z=j-1).

b. (s=i-1 and k=j+1) and (x=i-1 and y=j) and (w=i and z=j+1).

c. (s=i+1 and k=j-1) and (x=i+1 and y=j) and (w=i-1 and z=j-1).

d. (s=i+1 and k=j+1) and (x=i+1 and y=j) and (w=i and z=j-1).

한쪽 마름모 연결(one-side marym mo)은 아래와 같이 정의 한다.

정의1.5](i,j),(s,k),(x,y),(w,z) ∈ V(B) (1≤i,j,s, k,x,y,w,z ≤ 19),

((i,j),(s,k) ∈ OMM(B) iff f((i,j)) = f((s,k)) = {black,white} and [(f((i,j))≠f((x,y)) and f((w,z)) = {empty}) ∨(f((i,j))≠f((w,z)) and f((x,y)) ={empty})] and

a. (s=i -1 and k = j -1) and (x =i -1 and y =j) and (w=i and z=j-1).

b. (s = i -1 and k=j+1) and (x = i-1 and y = j) and (w=i and z =j+1).

c. (s = i + 1 and k = j-1) and (x = i + 1 and y = j) and (w = i - 1 and z = j -1).

d. (s = i + 1 and k = j + 1) and (x = i + 1 and y = j) and (w = i and z = j -1).

한칸 벌림(hankan bulim)은 아래와 같이 정

의한다.

정의1.6] (i,j),(s,k),(x,y) ∈ V(B) (1 ≤ i,j,s,k, x,y ≤ 19)

((i,j),(s,k)) ∈ HB(B) iff f((i,j)) = f((s,k)) = {black,white} and f((x,y)) = {empty} and

a. (x=i-1 and y=j) and (s=i-2 and k=j).

b. (x=i and y=j+1) and (s=i and k=j+2).

c. (x=i+1 and y=j) and (s=i+2 and k=j).

d. (x=i and y=j-1) and (s=i and k=j-2).

날 일자 벌림은 아래와 같이 정의한다.

정의1.7] (a,b),(c,d),(e,f),(g,h),(i,j),(k,l) ∈ V(B) (1≤a,...,l≤ 19),

((a,b),(k,l))∈NB(B) iff f((a,b))=f((k,l))= {black, white} and f((c,d)) =f((e,f)) =f((g,h)) =f((i,j)) ={empty} and

a. (k=a+1 and l=b+2) and (c=a and d=b+1) and (e=a and f=b+2) and (g=a+1 and h=b) and (i=a+1 and j=b+1).

b. (k=a+1 and l=b-2) and (c=a and d=b-1) and (e=a and f=b-2) and (g=a+1 and h=b) and (i=a+1 and j=b-1).

c. (k=a-1 and l=b-2) and (c=a and d=b-1) and (e=a and f=b-2) and (g=a-1 and h=b) and (i=a-1 and j=b-1).

d. (k=a-1 and l=b+2) and (c=a and d=b+1) and (e=a and f=b+2) and (g=a-1 and h=b) and (i=a-1 and j=b+1).

정의1.8] (i,j) ∈ BV(B)(byun vertex) iff f((i,j)) = {empty} and

a. (i=1)∧(1≤j≤19).

b. (i=19)∧(1≤j≤19).

c. (j=1)∧(1≤i≤19).

d. (j=19)∧(1≤i≤19).

두 정점 (i,j)와(s,k)이 link를 가진다면 아래와 같이된다.

정의1.9](i,j) and (s,k) have a link iff

a. (i,j) con (s,k).

b. ((i,j),(s,k)) ∈ MM(B).

c. ((i,j),(s,k)) ∈ OMM(B).

d. ((i,j),(s,k)) ∈ HB(B).

e. ((i,j),(s,k)) ∈ NB(B).

f. (i,j)∈BV(B) and f((s,k)) = {black,white} and

ㄱ. if ((i=1)∧(1≤j≤19)) then (((s=2)∧(1≤k≤19))∨(s=3)∧(1≤k≤19)∧f((s-1,k))={empty})).

ㄴ. if ((i=19) ∧ (1≤j≤19)) then (((s=18) ∧ (1≤k≤19) ∨ ((s=17) ∧ (1≤k≤19) ∧ f((s+1, k)) = {empty})).

ㄷ. if ((j=1)∧(1≤i≤19)) then (((k=2)∧(1≤s≤19)∨(k=3)∧(1≤s≤19))∧f((s,k-1))={empty})).

ㄹ. if ((j=19) ∧ (1≤i≤19)) then (((k=18) ∧ (1≤s≤19) ∨ ((k=17) ∧ (1≤s≤19)) ∧ f((s, k+1)) = {empty})).

정의1.10] group은 link를 가진 돌(S)들의 집합이라 정의한다.

정의1.11] group(G)의 내부(INT(G))와 외부(EXT(G))를 다음과 같이 정의한다.

임의의 group을 구성하는 돌들이 cycle을 이루거나 그 돌들이 변과의 link를 2개 이상 가질 때 link를 가진 변의 한 정점에서 또 다른 link를 가진 같은 변이나 다른 변의 한 정점사이에 연결 그래프(connected graph)가 존재한다. 이러한 사이클이나 연결 그래프에 의해 이루어진 다각형에서 빈 정점이 적은 곳을 그 group의 내부 영역(INT(G))이라하며 빈 정점이 많은 곳을 그 group의 (EXT(G))이라 정의한다.

정의1.12] 바둑에서 "눈"(eye)은 아래와 같이 정의한다.

a. 중앙의, 임의의 빈 정점에서 이웃한 네 방향의 자리에 돌들의 색이 같은 색이면서 각각은 단수로 몰리지 않는 경우 빈 정점을 눈이라 한다.

b. 변에서, 임의의 빈 정점에서 이웃한 세 방향의 자리에 돌들의 색이 같은 색이면서 각각은 단수로 몰리지 않는 경우 빈 정점을 눈이라 한다.

c. 귀의 네 정점((1,1),(1,19),(19,1),(19,19))중 빈 정점에 이웃한 두 방향의 자리에 돌들의 색이 같은 색이면서 각각은 단수로 몰리지 않는 경우 빈 정점을 눈이라 한다.

정의1.13] 바둑에서 "눈 형성 가능 정점"(make-eye-vertex)을 아래와 같이 정의한다.

a. 중앙에서, 이웃한 네 방향 중 이미 두(세)방향에 같은 색의 돌이 놓여있고 나머지 이웃한 두(한)정점이 빈 정점일 때 이 빈 두(한) 정점을 눈 형성 가능 정점이라 정의한다.

b. 변에서, 이웃한 세 방향 중 이미 두 방향에 같은 색의 돌이 놓여있고 나머지 이웃한 한 정점이 빈 정점일 때 이 빈 한 정점을 눈 형성 가능 정점이라 정의한다.

c. 귀에서, 이웃한 두 방향 중 이미 한방향에 돌이 놓여있고 나머지 이웃한 한 정점이 빈 정점일 때 이 빈 한 정점을 눈 형성 가능 정점이라 정의한다.

정의1.14] CUT_VERTEX를 아래와 같이 정의한다. GRAPH에서의 CUT_VERTEX와 동일하게 CONNECTED GRAPH G에서 G - v는 G보다 더 많은 COMPONENT를 가지는 G'이 되게 하는 정점들을 cut-vertex라 정의한다.

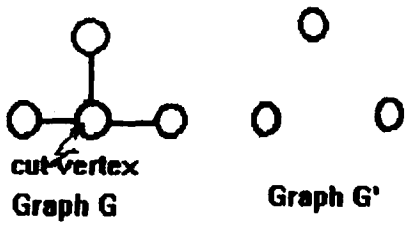

그림 3. CUT-VERTEX의 보기
Fig. 3. Example of CUT-VERTEX.

정의1.15] 외부와 연결점

group A가 group B에 의해 폐쇄형의 형태로 둘러싸여 있을 때 INT(A)에 있는 돌(i,j)((s,k)∈B, f((i,j))=f((s,k)))가 group B와 연결 되게 하는 수로 group A의 일부분의 돌을 포획하는 정점이나 1선에서 INT(A)로 젖히는 정점.

2. 문제 정의

바둑사활문제(이하 문제)의 대부분이 귀와 변에 위치하고있는 특성을 가진다. 이러한 위치/형태적인 특성으로 분류가 가능하다.

(1) 첫째, 폐쇄정도에 따라 분류해보면 아래와 같다.

1) 폐쇄형 : 사활 문제에서 group A가 다른 group B((i,j)∈A, (s,k)∈B,f((i,j)) ≠ f((s,k))))에 의해 둘러싸여 있을 때 EXT(B)로 완전히 연결되어 나갈 수 없고 group B를 포획할 수 없는 형태

2) 미 폐쇄형 : 사활 문제에서 폐쇄형이 아닌 모든 형태

(2) 두번째, 문제 유형 별로 분류해 보면 아래와 같다.

흑선 흑생, 흑선 백사, 백선 흑사, 백선 백사

(3) 세번째, 폐쇄 상황에서의 내부에 적의 유무에 대한 분류

1) 적이 있다.

2) 적이 없다.

본 논문에서는 폐쇄정도에서는 문제의 대부분을 차지하는 폐쇄형으로 문제 범위를 정하였다. 다음으로는 문제 유형은 흑/백과 사/활의 조합이므로 흑선 백사를 해결하는 범위로 정하였다.(다른 유형은 비슷한 알고리즘임을 알 수 있다.) 마지막으로 내부에 적의 존재 유무에 따라 적이 있는 상태는 적이 없는 상태에서 수진행의 중간 상태라 정의 한다. 그러므로 본 방법에서는 폐쇄형의 문제를 흑선 백사의 해를 찾는 것이다.

3. Empty planar Graph(EPG)

EPG는 일반적인 Graph G에 속한다. 그러나 바둑에서 empty vertex만을 가진 EPG를 일반

적인 planar graph에 대해 아래의 조건을 만족하는 것으로 정의한다.

표 1. 공 평면그래프와 일반그래프의 비교
Table 1. Comparison of Empty Planar Graph and G.

										EPG	G
	E	0	1	2	3	4	5	6	7	총수	총수
V											
1		1								1	1
2		1	1							2	2
3		1	1	1						3	4
4		1	2	2	1					7	11
5		1	2	3	4	1				12	34
6		1	1	2	4	7	7	5	1	28	156

f : V -> { EMPTY }

a. 임의의 GRAPH G가 만약 EDGE가 3이상인 홀 수로 구성된 CYCLE을 포함한다면 G는 EPG에서 제외된다.

b. 임의의 GRAPH G가 만약 DEGREE >= 5 인 VERTEX를 포함한다면 G는 EPG에서 제외된다.

그러므로 EPG와 GRAPH G의 VERTEX수와 EDGE수에 대한 표1을 만들 수 있다.

표1처럼 EPG는 G보다 매우 적은 수를 가진다.

III. PATTERN 정의 및 저장법/산출법/탐색법

1. PATTERN 정의

EMPTY VERTEX로 구성된 EPG에서 두 가지 형태로 분류가능하다.

활형 : VERTEX는 2개 이상 존재하고 COMPONENT가 2이상인 형

사형 : VERTEX는 1개 이상 존재하고 COMPONENT가 1개인 형 중 치중/수 진행에 의해 활형이 될 수 없는 형

활형은 실제 살아있는 형으로 집이 둘 이상이 났다는 의미이므로 내부에 있는 적을 포획하기 위하여 저장할 필요가 없다. 그러나 사형은 치중에 의하여 적을 죽게 만듦으로 매우 중요하게 사용될 수 있다. 그러므로 사형을 확장하여 확장 패턴을 만들 수 있다. 확장할 때 VERTEX는 1개씩 확장되는 반면 EDGE는 1개 혹은 2개가 확장된다. 또한 새로운 VERTEX가 첨가되어 연결되는 기존의 VERTEX를 확장 VERTEX라 하고 이러한 확장 VERTEX는 다시 또 다른 급소점이 되어진다.만약 2개의 EDGE가 첨가되어 CYCLE을 이루는 GRAPH라면 DEGREE가 가장 큰 VERTEX가 급소점이 되어진다.단 새로운 VERTEX가 확장 될 때는 이전의 PATTERN과 동형이 될 수 없다. 그리고 급소점이 또다시 새로운 EPG가 되어 순환적으로 이런 PATTERN을 적용 할 수 있다.

2. PATTERN의 저장법/산출법/탐색법

(1) 저장법

EPG로 이루어진 PATTERN들은 다음 특징으로 분류가능하다.

첫째는 VERTEX수

둘째는 EDGE수

셋째는 각 VERTEX의 DEGREE 순

이러한 순서로 PATTERN을 저장한다.단 동형의 EPG는 S/W적으로 판단하여 PATTERN의 수를 줄인다. 즉 아래 그림4와 같은 형은 ISOMORPHIC하다.

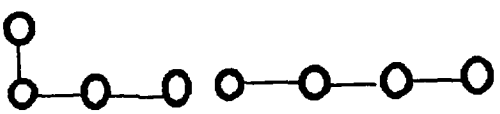

그림 4. 동형의 패턴
Fig. 4. Patterns of isomorphism.

(2) 산출법

첫번째, 백의 불완전 LINK 를 완전 LINK화 한다[1].

두번째, 내부에 흑이 없는 경우 : 내부 영역

의 빈 정점을 사용 각 좌표를 EPG로 바꾸어 기억한다.

내부에 흑이 있는 경우 :

1) 흑이 내부에 있는 자리를 EMPTY로 본다.

2) 흑의 위치를 기억하여 둔다.

(3) 탐색법

문제에서 산출되는 정보로 VERTEX수와 EDGE수 그리고 DEGREE와 연결형태를 얻을 수 있다.그러므로 탐색 키로 첫번 째가 VERTEX수이다. 그리고 두번째 키로 EDGE수가 된다. 마지막으로 DEGREE와 연결형태를 비교하여 탐색하게된다. 아래와 같은 단순한 문제가 주어졌다면 어떻게 문제해결이 이루어지는지를 살펴보자.

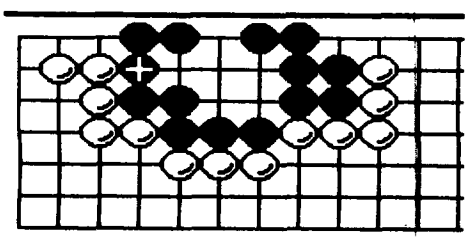

그림 5. 사활 문제 예
Fig. 5. Example of life-death problem.

이렇게 그림5와 같이 문제가 주어지면 EMPTY VERTEX를 가지고 EPG그림을 아래처럼 산출한다.

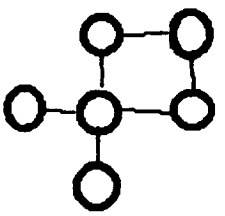

그림 6. 그림 5의 공 평면그래프
Fig. 6. Empty planar graph of Fig5.

위 그림처럼 EPG가 나오면 이것으로 탐색에 필요한 정보를 추출 할 수 있다. 먼저, VERTEX수는 6개가 나오며 EDGE의 수도 6개가 나온다 그리고 연결형태는 위 그림처럼 좌표로 표시 될 수 있다. 그래서 VERTEX의 수가 6이고 EDGE의수도 6인 정보로 탐색을 시작한다. 그러면 앞의 PATTERN 정의에서 VERTEX수가 6이고 EDGE수가 6인 것 중 패턴 매칭을 시작한다. PLANAR GRAPH에서 동형 판단방법의 알고리즘은 linear time에 수행된다. 그러므로 일반 그래프에서 동형 판단보다 빠른 시간에 수행 되어진다. 이렇게 찾아진 PATTERN은 사활의 정보를 가지고있다.

IV. 새로운 평가함수 제안 및 타당성

사활문제 해결을 위해 기존의 평가함수 보다 많은 parameter가 필요하다. 그러므로 본 논문에서는 인간적 사고에 기초로 한 소수의 몇 후보를 산출하여 기존의 순서화방법보다 효율을 높이고자한다. 아래에 평가함수의 parameter 타당성을 열거하였다.

1. CUT_VERTEX가 되는 모든 VERTEX들

앞에서 정의한 것으로서 타당한 이유는 CUT_VERTEX들은 EPG를 2개 이상의 COMPONENTS로 분리시키므로 이것은 사활의 맥점과 같다.

2. 특수 점 (SPECIAL VERTEX)

좌표상 (1,2),(2,1)에 위치하는 4곳의 VERTEX로 사활문제가 귀에서 발생되어 질 때 귀의 특수성으로 인하여 해가 될 수 있는 확률이 매우 높다.

3. 외부와 연결점

ㄱ. 제 1선에서 젖히는 점.

사활 문제에서 내부의 면적을 줄이면서 수를 찾는 것임에 수 있다는 기본적 해결방법에서 그 근본을 둔다.

ㄴ. 적의 내부로 밀고 들어가는 점

이것 역시 그과 같은 이유로 내부의 궁도를 좁히는 효과를 가진다.
ㄷ. 단수하는 수와 적을 포획하여 연결하는 점.
ㄹ. 제 1선으로 넘는 수.

4. 정적 PATTERN MATCHING으로 급소점

PATTERN MATCHING을 통해 현 문제의 EPG를 죽는 EPG로 만드는 급소점을 찾는다.

5. 기존의 논문에서 완전LINK를 불완전LINK화 하는 점

기존의 논문에서 CG를 정의한 것의 주요 FACTOR인 완전LINK를 불완전LINK화 하여 흑에게 유리한 점[1].

6. 눈 형성점

앞에서 정의한 눈 형성점에 흑이 두어 눈의 형성을 막아 전체 2집이 나는 것을 막고자한다.

V. 실험 및 고찰

문제 그림처럼 사활 문제가 주어지면 기존의 순서화된 AND/OR PRUNING방법과 위 parameter 1)에서 6)을 만족하는 EPG상의 VERTEX들을 모두 산출하여 1)에서 6)까지중 가장 많이 나온 순으로 정렬하여 TREE의 노드를 순서화하는 방법을 비교하여 보았다.

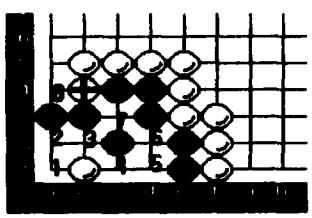

그림 7. 사활 문제 예
Fig. 7. Example of life-death problem.

기존의 방법으로 평가하여 나온 순서로는 2,3,4,5,8,6,1이 된다. 새로운 평가 함수를 사용하면 먼저 아래처럼 EPG를 먼저 산출하고 1)에서 6)까지의 parameter를 구한다.

각 parameter에 해당하는 후보지 산출을 하면 표2와 같이 되었다.

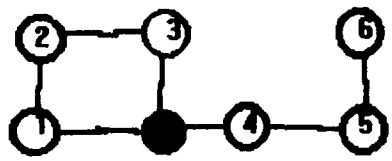

그림 8. 그림 7의 공 평면그래프
Fig. 8. Empty planar graph of Fig. 7.

표 2. 후보지 산출
Table 2. Produce candidate.

함 수	후 보 지
6.1	4,5
6.2	2
6.3	5,6,6
6.6	5

그러므로 새로운 평가 함수를 사용한 결과 후보지의 순서는 5,6,4,2와 나머지 후보지는 1,3,8 올림순으로 순서화한다. 실제의 해는 6으로 기존의 함수를 사용할 때의 순서화 보다 더욱 효율적인 결과를 내었다

V. 결 론

본 논문은 PATTERN MATCHING시 기존의 다른 바둑논문에서 시간이 $O(n^2)$ 걸리는 것을 바둑판 정의를 planar graph(EPG)로 정의하여 선형시간에 실행되도록 하였다[3].

그리고 바둑의 사활 문제풀이 방법에 대한 연구로 기존 논문의 평가함수에 비해 더욱 인간적 사고 방식에 맞는 parameter인 CUT VERTEX, SPECIAL VERTEX, 젖히는 점, 단수하여 연결하는 점, 넘는 점 , 패턴 매칭의 급소점, 그리고 눈 형성 점 등으로 이루어진 평가함수를 제안 하였으며 평가 결과치 역시 기존의

평가 결과치 보다 더욱 효율적 이었다[1].

참 고 문 헌

1. 최영숙, "순서화된 AND/OR Pruning을 이용한 바둑 사활의 효율 개선," 경북대학교 컴퓨터공학과, 석사학위논문, 1992.
2. 임중권, 김영상, 이종철, "연결 정도에 따른 반면의 안정도 평가 기법," 한국정보과학회 학술발표회, vol. 20, no. 1. 1993.
3. 임중권, 김경아, 김진환, 박현수, 이두환, 이종철, "패턴의 계층적 분류에 의한 다음 수 선택 기법," 한국정보과학회 학술발표회, vol. 20, no. 2. 1993.