컴퓨터 바둑에서 Group영역 획득과 String의
안정도 정의 및 구현

박 현 수*

<table>
<thead>
<tr>
<th>목</th>
<th>successive</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. 서 론</td>
<td>IV. String의 안정도</td>
</tr>
<tr>
<td>II. Link 정의</td>
<td>V. 안정도를 사용한 정적 형세 판단</td>
</tr>
<tr>
<td>III. Link에 의한 Group의 영역</td>
<td>VI. 결 론</td>
</tr>
</tbody>
</table>

요 약

본 논문은 1995년 세계 컴퓨터 바둑 대회에 출전한 큰돌의 일부분으로 Group의 내부영역과 String의 안정도 정의에 관한 것이다. 바둑에서 수를 둘 때 반면의 형세 판단을 근거로 다음 수 선택을 한다. 형세 판단은 반면의 집의 수와 세력과 돌들의 안정도가 매우 중요한 결정 요소로 사용되어진다. 본 논문에서는 String들간의 Link를 찾고 이를 기반으로 Group을 만들고, 그 Group의 내부 영역을 찾는다. 그리고 Group의 내부 영역 안에 있는 적 돌들의 안정도를 평가하고 수정된 내부 영역을 구하여 정적 형세 판단에 이들을 이용하는 것을 구현하였다.

I. 서 론

 컴퓨터 바둑은 흑과 백이 서로 변갈아 두어 영역을 많이 차지하는 쪽이 이기는 게임이다[1,2,3, 11]. 바둑판에는 모든 정보가 있고 그곳에서 새로운 정보를 만들어 내어 많은 후보 점을 생성시켜 게임을 진행한다[1].

후보 점 생성은 매우 어려운 문제 중의 하나인데 바둑판에서 얼마나 정확한 정보를 획득하는가에 따라 차이가 매우 크다[3]. Pattern들을 저장하였다가 그와 일치하는 것이 있으면 후보 점을 생성한다[7,12,13]. 그리고 상태와 영역의 크기 그리고 이런 요소를 고려한 형세판 단등으로 새로운 후보 점을 생성하게 된다. 그러므로 반면의 상태를 지속적으로 파악해서 전략을

*경동전문대학 전자계산학과 전임 강사

- 25 -
세워야 한다[3,4,5,6,8,9]. 본 논문은 1995년 서울에서 개최된 세계 컴퓨터바둑 대회에 출전한 큰들의 일부분으로 Link정보에 의한 Group영역과 String의 안정적 평가를 이용하여 정적 형제 판단을 구현하였다.

Link는 이웃점, 마름모, 한 칸, 날 일자 그리고 두칸 Link들이 있다[1]. Link는 Group을 형성하고 Group은 영역을 가진다. 그러므로 각 Group에 의한 영역이 구분이 되어진다. 그러나 Group은 서로 영역이 서로간에 영역을 간섭하고 침범하고 있다[8,9,10,14]. 그러므로 각 Group을 형성하고 있는 String들의 안정도를 사용한 정적 형제 판단을 해야한다.

본 논문은 먼저 Link정의를 하고 Link에 의한 Group의 영역을 구하는 알고리즘을 설명한다. 그리고 반면의 String의 안정도를 휴리스틱하게 정의(완전한 삽, 삽아 있는 상태, 모르는 상태, 위험한 상태, 죽음의 상태)하고 이러한 안정도를 사용하여 정적 형제 판단을 예로 보인다.

II. Link 정의

이웃 Link는 이웃점, 마름모, 한 칸, 날 일자 그리고 두칸 Link로 정의한다[1]. 그러므로 Group은 그림1과 같은 Link들에 의해서 형성되어진다. Group의 내부영역을 순환된 Group의 순환 내부 공백을 의미한다[1].

![Link Diagrams](image)

그림 1. (a)이웃점 Link (b)마름모 Link (c)한 칸 Link (d)날 일자 Link (e)두칸 Link.

III. Link에 의한 Group의 영역

정의된 Link를 사용하여 Group영역 형성 과정을 간략히 설명하면 아래와 같다.

배열 CG의 자료 구조

```
int CG[400]; /*정수형 값 초기치 -1 */
```

[알고리즘]

입력: 반면의 돌(stone)의 정보인 board[19][19]
출력: 각 Group의 내부 영역을 가진 String

- 26 -
int GroupCount : // GroupCount: 반면에 형성된 Group의 수
begin
for(k=0; k<GroupCount; k++)
{
end.

Byun_Byun()에서는 변 점들을 중 액 끝점을 찾는다. 액 끝점이라는 것은 내부 영역을 포괄적으로 포함하는 점을 찾기 위해서이다. 두 점이 있을 수 있는 형태는 같은 변에 있든지 아니면 이웃 변 혹은 대응변에 있을 수 있다. 모든 변 점들을 중 같은 변, 이웃 변, 대응변으로 구분하여 계산하여 가장 큰 값을 선택한다. 두 점들을 A(Xa,Ya), B(Xb,Yb)이라 두자. 그림2와 같은 방법으로 양 끝점을 구할 수 있다.

모든 점들을 계산하여 가장 큰 값을 양측 점으로 선택한다. Search%Intern()은 배열 CG에 포함된 Link관계의 empty 점과 변 점들을 사용하여 X축을 기준으로 같은 Y값을 가지는 점들을 선택하고 그 두 점들의 Y값 중 최소 값과 최대 값을 정한다. 그리고 최소 값에서 최대 값으로 증가시키면서 거쳐가는 점들을 Xtemp에 저장한다.

**Xtemp의 자료 구조**

*int Xtemp[400]:*

![Diagram](image)

그림 2. (a) 같은 변일 경우: $|X_a - X_b| + |Y_a - Y_b|$
(b) 이웃 변일 경우: $|X_a - X_b| + |Y_a - Y_b|$
(c) 대응변일 경우: $X_a + Y_a + X_b + Y_b - 2$

다음으로 다시 배열 CG에 포함된 모든 점들을 사용하여 Y축을 기준으로 같은 Y값을 가지는 점들을 선택하고 그 두 점들의 X값 중 최소 값과 최대 값을 정한 후 최소 값에서 최대 값으로 증가시키면서 거쳐가는 점들을 Xtemp에 존재하는지 조사하여 존재하면 내부 영역이라는 Inter 배열에 첨가한다.

**Inter의 자료 구조**

*int Inter[400]*

결과적으로 Inter 배열에 남은 점들이 내부 영역이 되어진다.
IV. String의 안정도

String의 안정도는 휴리스틱하게 정의한다. 안정도는 String이 완전한 상태, 살아 있는 상태, 모르는 상태, 위험한 상태 혹은 죽은 상태를 수치로 정의한다. 안정도를 구하기 위해 필요한 여러 요소를 정의하고 그 요소를 사용하여 안정도를 구하고자 한다. 안정도를 구하기 위한 요소로서 그 String의 눈의 수, 유사 눈, 특수 눈, 확장점, 내부 영역의 크기, 공배수 그리고 이웃한 야군 String과의 연결 정도를 이용한다. String의 자료구조는 아래와 같다.

```c
struct stone_list {
    int position;
    struct stone[list huge *Link;
};

struct String {
    int stone_num; /* String구성 돌 수 */
    struct stone_list huge *member;
    int connect_num; /* 연결점의 수 */
    int extend_num; /* 확장점의 수 */
    int empty_num; /* 공배의 수 */
    int eye_num; /* 눈의 수 */
    int eye_like_num; /*유사눈+특수눈 수 */
    int influence_size; /* 영향력의 크기 */
    int InternalAreaSize; /*내부영역의 크기 */
    int stability; /* 안정도 */
    int groupid; /* Group의 번호 */
} String[STRING_MAX];
```

그림 3. 한 점에서의 8방향.

위와 같은 안정도 요소를 사용하여 실제로 안정도를 구할 수 있는데 그 상태로는 완전한 삶, 살아있는 상태, 모르는 상태, 위험한 상태 그리고 죽은 상태로 구분 지을 수 있다.
### Table 1. 안정도 구성 요소.

<table>
<thead>
<tr>
<th>안정도 구성 요소</th>
<th>설명</th>
</tr>
</thead>
<tbody>
<tr>
<td>눈</td>
<td>임의의 공배에 인접한 4방향 string이 같은 string일 때 이 공배를 눈이라 한다. 한 String의 특수눈은 그림3에서 a-g의 각들고 아닌 g-g 중 눈이면 count를 1</td>
</tr>
<tr>
<td>특수눈</td>
<td>증가시키고 적이 못 놓는 자리 일 경우 count를 1증가시키고 같은 String반호일 때 count를 1증가시킨다. 만약 count가 6이상이면 이 공배를 특수한 눈이라 한다.</td>
</tr>
<tr>
<td>유사눈</td>
<td>한 String의 유사눈은 그림3에서a-g점 중에서 a,c,e,g중 적군 돌이 1개 이하이고 아군 돌이 6개 이상이고 a-g중 공백인 것끼리는 연결하지 않을 때 유사 눈이라 한다.</td>
</tr>
<tr>
<td>확장점</td>
<td>3선 이상의 위치에 있는 string이 두칸이나 눈목차 위치의 공배 정점(8방향이 모두 공배)을 가지고 link를 가질 때 확장점을 가진다.</td>
</tr>
<tr>
<td>내부영역의 크기</td>
<td>link에 의한 영역 분할 후 그 group의 영역이 내부영역 크기이다.</td>
</tr>
<tr>
<td>공배수</td>
<td>String에 이웃한 empty point의 수가 공배수이다.</td>
</tr>
<tr>
<td>연결도</td>
<td>String의 돌에서 두 칸,한 칸, 날 일자 혹은 마름모 위치에 돌 때 다른 아군 String과 Group으로 연결되면 각각 연결도가 1이다. 연결도가 높으면 눈물수록 더욱 안정적임</td>
</tr>
</tbody>
</table>

아래에 Table 2에서 안정도의 각 상태를 정의한다.

### Table 2. 안정도.

<table>
<thead>
<tr>
<th>상태</th>
<th>안정도</th>
<th>구성 요소</th>
</tr>
</thead>
<tbody>
<tr>
<td>완전 삐(C)</td>
<td>590 눈이 2개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>580 눈이 1개이고 확장점이 4개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>570 눈이 1개이고 확장점이 3개이고 유사 눈이 1개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>560 눈이 1개이고 확장점이 1개 이상이고 2개 이하이고 유사 눈이 2개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>550 유사 눈 3개 이상</td>
<td></td>
</tr>
<tr>
<td>삐(A)</td>
<td>490 눈이 1개이고 확장점이 3개</td>
<td></td>
</tr>
<tr>
<td></td>
<td>480 눈이 1개이고 확장점이 1개이고 유사 눈이 1개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>470 확장점 4개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>460 확장점 3개이고 내부영역크기가 5이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>450 유사눈 2개 이상</td>
<td></td>
</tr>
<tr>
<td></td>
<td>440 내부영역크기가 7이상.</td>
<td></td>
</tr>
</tbody>
</table>
V. 안정도를 사용한 정적 형세 판단

영역 내부에 적의 String이 존재하지 않는 경우는 Link에 의한 영역분할만으로 내부영역을 구할 수 있다. 그러나 영역 내부에 적의 String이 존재하는 경우는 Link에 의한 영역분할과 서로 영향을 주는 String의 안정도를 사용하여 내부영역을 구한다. 즉, 적의 상태를 고려하여 내부영역으로 고려해야할 것인지 아니면 내부의 적의 영역을 제외한 나머지 부분을 내부영역으로 추출 할 것인지에 판단하여야한다. 만약, 적이 안정도 평가에서 즉음(D)의 상태나 위험(K)의 상태이면 그 영역은 내부영역에 포함시킨다. 그림4에서 백이 위험(D) 상태이므로 혹은 색(A)의 상태인 Group의 내부영역에 포함되어진다.

그림 4. 위험한 상태의 적을 포함한 경우.  그림 5. 색의 상태를 포함한 경우.

그러나, 내부에 적이 완전한 색(C)의 상태, 색의 상태(A) 혹은 모름(U)의 상태등 세 가지 상태에 속한다면 내부영역을 구하기 전에 먼저 적의 내부영역을 먼저 구한 후-앞에서 제시한 방법을 사용-내부영역을 구한다. 이때에 발생되는 문제는 변 정점을 수정해야한다. 원래의
변 경점에서 적의 내부영역에 속하는 변 경점을 바탕한 후 양쪽 점으로 선택될 변 경점 사이에 마크된 경점이 하나라도 존재하면 양쪽 점으로 선택될 수 없다. 그래서 내부에 있는 적의 영역을 제외한 나머지 부분을 추출해 낼 수 있다.
즉, 그림 5에서 백이 살아 상태에 있고 흑의 내부에 있을 때는 흑을 변 경점을 수정하여 내부의 백 영역을 제외한 나머지 부분을 영역으로 추출한다.

VI. 결 론

컴퓨터 바둑에서 형세판단은 매우 중요한 결정요소로 사용되어진다. 본 논문은 정적형세판단을 위한 Link에 의한 Group영역구분과 각 String의 안정보를 정하였다.

안정보의 구성요소는 눈, 공백수, 내부 영역 크기, 확장점 그리고 연결정보로 정의하였다. 이를 기반으로 String의 안정보 상태를 흐리 섬(K), 섬(A), 모름(U), 위험(D) 그리고 죽음(K) 등으로 세분화하고 각각의 상태에 따라 안정보를 정했다. Group영역과 안정보를 사용 정적 형세 판단을 구현하였다. 앞으로의 과제는 정밀한 동적 형세 판단이 과제이다.

참 고 문 헌