Journal of Korea Game Society 2014 Jun; 14(3): 47-54
http://dx.doi.org/10.7583/JKGS.2014.14.3.47

게임 프로그래밍

삼목 게임에 적용된 몬테카를로 트리탐색

이병두
세한대학교 체육학부 바둑학과
blee026@korea.com

Monte-Carlo Tree Search Applied to the Game of Tic-Tac-Toe

Byung-Doo Lee
Dept. of Baduk Studies, Division of Sports Science, Sehan University

요 약

바둑 게임은 가장 오래된 게임 중의 하나이며 적어도 2,500년 전에 기원되었다. 게임프로그래밍에서 대부분의 성공적인 접근법은 평가함수를 활용한 게임트리 탐색을 사용하는 것이다. 그러나 컴퓨터바둑에서 그럴싸한 평가함수를 구축한다는 것은 매우 어렵다. 몬테카를로 트리탐색(MCTS)은 9줄 바둑에서 프로기사를 제압한 MoGo와 CrazyStone과 같은 강력한 컴퓨터바둑 프로그램을 만들어 내었다. 몬테카를로 트리탐색은 몬테카를로 시뮬레이션에 의해 계산된 승률을 근간으로 한다. 몬테카를로 트리탐색을 컴퓨터바둑에 구현하기에 앞서 삼목에서 최상의 첫 수로 중앙, 귀, 변의 세 수에 대한 각각의 승률을 측정하려고 했다. 실험 결과로 최상의 첫 수는 중앙이 우선하고, 다음은 귀, 마지막으로는 변이라는 사실이 밝혀졌다.

ABSTRACT

The game of Go is one of the oldest games and originated at least more than 2,500 years ago. In game programming the most successful approach is to use game tree searches using evaluation functions. However it is really difficult to construct feasible evaluation function in computer Go. Monte-Carlo Tree Search(MCTS) has created strong computer Go programs such as MoGo and CrazyStone which defeated human Go professionals played on the 9×9 board. MCTS is based on the winning rate estimated by Monte-Carlo simulation. Prior to implementing MCTS into computer Go, we tried to measure each winning rate of three positions, center, corner and side, in Tic-Tac-Toe playing as the best first move. The experimental result revealed that the center is the best, a corner the next and a side the last as the best first move.

Keywords : computer Go(컴퓨터바둑), Monte-Carlo Tree Search(몬테카를로 트리탐색), Tic-Tac-Toe(삼목), MCTS(몬테카를로 트리탐색)

Received: May. 26, 2014 Accepted: Jun. 16, 2014
Corresponding Author: Byung-Doo Lee (Sehan University)
E-mail: blee026@korea.com

ISSN: 1598-4540 / eISSN: 2287-8211

1. 서 론

바둑은 대략 2,500년 전에 기원된 2인-제로섬-완전정보게임이다[1]. 1990년대부터 전 세계적으로 개발된 컴퓨터바둑에 있어, 종반전인 끝내기는 컴퓨터바둑의 수준이 프로기사와 대등하게 되었으며, 중반전은 상당한 아마 기력 수준까지 이르고 있으나, 초반전인 포석에 대한 연구는 미진한 편이다[2,3,4]. 그간 컴퓨터바둑의 기력 향상을 위해 많은 인공지능 연구자들은 데이터베이스를 기반으로 하는 패턴일치, 평가함수를 활용한 통제학습 등 여러 최신 인공지능 기법을 적용하였으나, 그 기력이 프로기사에 못 미친다는 사실을 알게 되어 새로운 방법을 고대하고 있었다.

몬테카를로 방법(Monte-Carlo method)은 난수를 이용하여 함수의 값을 확률적으로 계산하는 알고리즘이다[5]. 엔리코 페르미가 1930년 중성자의 특성을 연구하기 위해 이 방법을 사용한 것으로 유명하며, 주로 수학이나 물리학 등에 자주 사용된다[6]. 계산하려는 값이 수학적 형식으로 표현되지 않거나 복잡한 경우에 근사적인 값을 계산하기 위하여 사용되며, 현재는 금융, 물리, 게임뿐만 아니라 많은 분야에 적용되고 있다. 특히 인공지능 게임분야에 있어 포커, 브리지, 백개먼 등에서 성공적인 결과를 보였다. 현재는 컴퓨터바둑 기력 향상을 위해 몬테카를로 트리탐색(MCTS: Monte-Carlo Tree Search)을 적용하고 있다. 그 결과 MoGo와 CrazyStone과 같은 강력한 컴퓨터바둑 프로그램을 제작해 내어 9줄 바둑에서 프로기사들을 제압하였고, [Table 1]에서 보듯이 19줄 바둑에서도 선전하고 있다.

[Table 1] List of the first games won by computer programs against human Go professionals

Year	Handicap	Human level	Computer Program
2008	9	8 dan	MoGo
2008	8	4 dan	Crazy Stone
2008	7	4 dan	Crazy Stone
2009	7	9 dan	MoGo
2009	6	1 dan	MoGo

저자는 본격적인 MCTS의 컴퓨터바둑 적용에 앞서 이를 삼목(Tic-Tac-Toe)에 적용시켜 그 효용성을 살펴보았다.

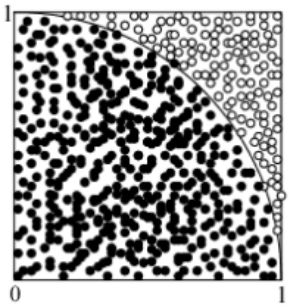
2. 본 론

2.1 몬테카를로 트리탐색

MCTS는 위치평가함수(position evaluation function)를 필요로 하지 않는 최대우선탐색(best-first search) 방법으로 탐색공간에 대한 무작위적 탐험(randomized exploration)에 근간을 두고 있다[6,7]. 이 알고리즘은 이전 탐험의 결과를 사용하여 메모리에 점진적으로 게임트리를 구축하며, 지속적인 시뮬레이션으로 가장 확신할 수 있는 착수를 정확하게 계산해낸다. MCTS를 적용하기 위해서는 다음과 같은 세 가지 조건이 만족되어야 한다[5].

(1) 득실값(payoff)의 한계가 있어야 한다. 즉 게임의 최대/최소 점수값이 있어야 한다.
(2) 게임 규칙이 정해져 있으며 완전정보(complete information) 게임이어야 한다.
(3) 게임의 길이가 제한되어 비교적 빨리 시뮬레이션이 끝나야 한다.

몬테카를로 방법은 임의추출(random sampling)을 이용하여 계산하는 알고리즘으로 MCTS를 구현하기 전에 알아야 할 방법이다.



[Fig. 1] Generated random points

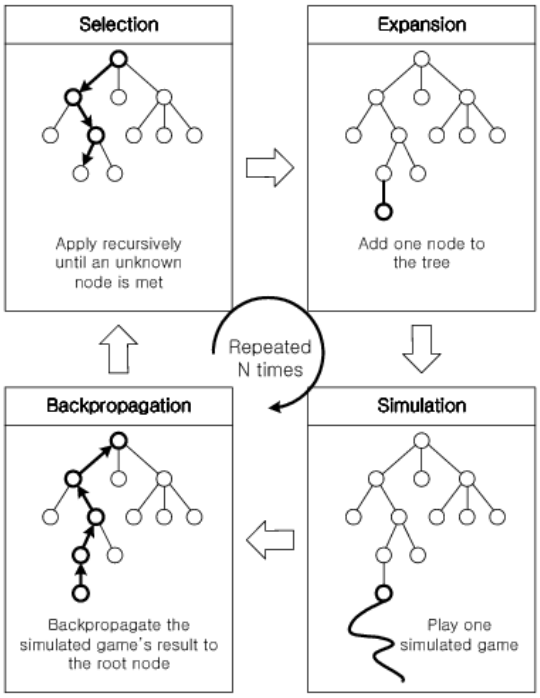
한 예로 [Fig. 1]은 몬테카를로 방법을 이용하여 원주율 $\pi = 3.14\cdots$를 계산하는 장면이 된다. 즉 [Fig. 1]에서 정사각형 내에 생성된 임의의 전체 (검은)점의 수를 N, 4분원 내에 생성된 임의의 전체 (흰)점의 수를 c라고 하면 (eq. 1)이 성립하며, 이를 이용하여 π값을 구할 수 있다.

$$P = \frac{c}{N} = \frac{\pi}{4} \qquad \text{(eq. 1)}$$

MCTS는 점진적으로 메모리에 이전의 정보를 기록하면서 승리의 우위를 가질 수 있는 착점을 중심으로 게임트리를 구성하며, 반면에 승리의 우위가 없는 착점은 과감하게 제거한다. MCTS를 수행하는데 필요한 두 가지 주요트리는 다음과 같다 [8].

(1) 탐색트리(search tree): 시뮬레이션을 위해 사용되는 정보를 담는 트리가 된다.
(2) 게임트리(game tree): 게임 진행 중에 생성된 착점들에 대한 정보를 담는 트리가 된다.

MCTS는 탐색트리를 초기화하면서 시작을 한다. 탐색트리가 초기화되면 [Fig. 2]와 같은 4단계의 작업이 주어진 작업 시간 범위 내에서 반복적으로 실시된다.



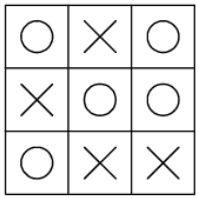
[Fig. 2] Four steps for MCTS algorithm in a search tree

(1) 선택(selection)단계: 선택함수 (selection function)를 반복적으로 적용하여 탐색트리의 단말노드를 만날 때까지 계속한다.
(2) 확장(expansion)단계: 선택단계에서 선택된 하나 이상의 노드의 자식이 탐색트리에 첨가된다. 일반적으로 하나의 노드가 탐색트리에 첨가된다.
(3) 시뮬레이션(simulation)단계: 탐색트리의 단말노드로부터 시뮬레이션을 시작하여 게임트리의 단말노드를 만날 때까지 시뮬레이션을 계속한다.
(4) 역전파(backpropagation)단계: 시뮬레이션 단계로부터 얻어진 보상값을 탐색트리의 뿌리노드로 역전파한다.

2.2 삼목에 적용된 MCTS

삼목은 두 대국자(○와 X)가 번갈아가며 3×3

판에 ○와 X를 써서 같은 모양이 가로, 세로 또는 대각선상에 연속하여 3개가 놓이면 이기는 게임이다. 참고로 [Fig. 3]은 첫 번째 대국자인 ○가 대각선 모양으로 승리한 장면이다.



[Fig. 3] A finished Tic-Tac-Toe game won by the first player, ○

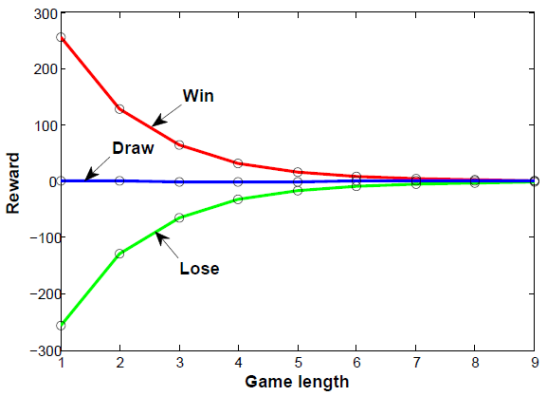
일반적으로 삼목에서 좀 더 많은 승리를 하기 위해서는 첫 수로 중앙에 두어야한다는 속설이 있으나[9,10,11], 어느 정도 승률을 보장하는지에 대한 자료는 없는 실정이다. 본 실험을 통해 1개의 중앙, 4개의 귀, 4개의 변이 있는 삼목에서 중앙, 귀, 변 중 어느 곳이 첫 수로 어느 정도의 승률을 보장하는지에 대해 살펴보았다.

삼목에서 게임결과는 일반적인 게임에서와 같이 승(win), 패(lose), 무승부(draw)로 구분할 수 있으며, 게임결과에 따른 보상값은 승리인 경우에 +1, 나머지인 경우에 0을 할당하는 방법[12,13]과 승리인 경우에 +1, 패한 경우에 -1, 무승부인 경우에 0을 부여하는 방법[5] 등이 있으나, 일반적으로 후자를 따른다.

본 실험에서는 게임결과에 따른 보상값을 계산하기 위해 게임길이(game length)를 고려한 (eq. 2)와 같은 새로운 보상함수를 제안하였다.

$$r(t)=\begin{cases} +2^t & \text{for } win \\ -2^t & \text{for } lose \\ \dfrac{\sin(t)}{t} & \text{for } draw \end{cases} \qquad \text{(eq. 2)}$$

여기서 l은 게임길이가 되며 $t=9-l$이 된다.



[Fig. 4] Reward functions with game length

즉 동일한 게임결과라도 [Fig. 4]와 같이 게임길이가 길면 작은 보상값을 부여하고, 반면에 게임길이가 짧으면 큰 보상값을 부여하는 보상함수를 적용하였다.

또한 본 실험에서 삼목에 적용된 MCTS의 시뮬레이션에 해당되는 의사코드(pseudo-code)는 다음과 같다.

```
1  function MC_simulation(node, player)
2    for rounds
3      game_length = 0;
4      while game is over
5        generate random legal move
6        if random move is the first move
7          then first_move<- move
8        increase game_length
9        swap player
10     end while
11     evaluate first_move by game_length
12   end for
13   return the most winning first move
14 end function
```

의사코드를 살펴보면 1번 줄은 함수명으로 게임트리 내의 현재노드와 대국자를 매개변수로 받는다. 2~12번 줄을 시뮬레이션의 수만큼 반복을 한다. 3번 줄에서 게임길이를 초기화하고, 4번 줄은 게임이 종료되었는가를 확인하는 과정이 되며, 게

임이 종료가 되면 11번 줄에서 첫 번째 착수의 위치에 대해 보상함수를 이용하여 보상값을 계산한다. 5번 줄은 착수가 가능한 위치에 대한 임의의 착수를 생성하는 과정이 된다. 만약 생성된 착수가 첫 번째 착수인 경우에 6~7번 줄에서와 같이 이를 first_move로 보관한다. 8번 줄에서 게임길이를 증가시킨 후, 9번 줄에서 대국자를 변경시킨다. 이후 시뮬레이션이 종료가 되면 13번 줄에서 보듯이 first_move들에 대한 보상값을 비교하여 최대값을 갖는 first_move를 반환한다.

2.3 실험 결과

0	1	2
3	4	5
6	7	8

(a)

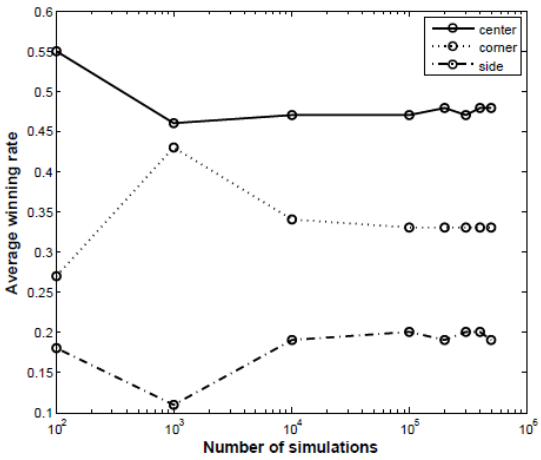
.376	.218	.360
.190	.506	.192
.361	.198	.341

(b)

[Fig. 5] (a) Positions and (b) its winning rates after 100,000 simulations

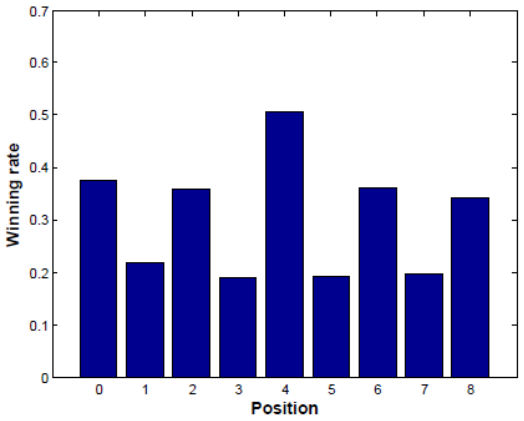
새로운 보상함수를 적용한 MCTS를 삼목에 시뮬레이션하기 위하여 100번~500,000번에 걸쳐 단계별로 시뮬레이션을 실시하였다. 또한 삼목내의 각 위치에 대한 평균 보상값을 계산하기 위하여 [Fig. 5](a)와 같이 각 위치에 대한 위치값을 부여하였다.

실험 결과 [Fig. 6]에서 보듯이 10,000번의 시뮬레이션부터 평균 승률값이 점차 수렴을 하기 시작했다.



[Fig. 6] Average winning rates by number of simulations

참고로 100,000번의 시뮬레이션에서의 각 위치에 따른 승률값은 [Fig. 5](b), [Fig. 7]과 같다.



[Fig. 7] Winning rates of each position after 100,000 simulations

[Table 2] Average winning rates by number of simulations

Position	$r=10^2$	$r=10^3$	$r=10^4$	$r=10^5$	$r=2\times10^5$	$r=3\times10^5$	$r=4\times10^5$	$r=5\times10^5$
center	.55	.46	.47	.47	.48	.47	.48	.48
corner	.27	.43	.34	.33	.33	.33	.33	.33
side	.18	.11	.19	.20	.19	.20	.20	.19

또한 100,000번 시뮬레이션에서의 중앙, 귀, 변에 대한 평균 승률값은 [Fig. 6], [Table 2]에서 보듯이 중앙이 .47, 귀가 .33, 변이 .20으로 나타나 삼목에서 최선의 첫 수로 '중앙이 우선, 귀가 다음, 변이 나중'이라는 사실을 밝혀냈다.

3. 결론 및 제언

예전의 컴퓨터바둑 대부분은 인공지능을 이용한 탐색, 패턴일치, 데이터베이스 등을 활용하여 기력을 향상시키고자 했으나 그 한계에 봉착하게 되었다. 이러한 문제를 해결하기 위하여 전 세계 연구자들은 그 대안책으로 몬테카를로 방법, 특히 MCTS을 제시했으며, 이를 통해 많은 진전을 이루어내었다.

본 논문에서 저자는 향후 본격적인 MCTS의 컴퓨터바둑 적용에 앞서 이를 삼목에 적용시켰다. 즉 MCTS를 이용하여 삼목에서 어느 위치의 첫 수가 승리를 하기 위한 최선인가를 알아보았다.

3.1 연구 결과

삼목에서 승리를 위해 첫 수로 중앙, 귀, 변 중 어느 부분을 먼저 두어야 하는지에 대한 정확한 이론 및 계산값이 없는 실정이다. 일반적으로 "중앙이 가장 좋은 첫 수일 것"이라는 속설이 있다.

실험을 위해 저자는 MCTS와 보상함수를 이용하여 여러 번의 시뮬레이션을 하였다. 실험 결과 평균 승률값은 100,000번의 시뮬레이션에서부터 수렴을 하기 시작했으며, 그 평균 승률값은 중앙(.48) > 귀(.33) > 변(.19)으로 나타나 첫 수로 "중앙이 우선, 귀가 다음, 변이 나중"이라는 사실을 알아냈다. 또한 저자는 기존의 보상함수(승: +1, 패: -1, 무승부: 0) 대신에 게임길이 l을 고려한 변수 $t = 9 - l$에 대한 새로운 보상함수(승: $+2^t$, 패: -2^t, 무승부: $\sin(t)/t$)를 제시하였다.

3.2 제언

MCTS를 19×19 바둑판에 적용하는 것은 매우 방대한 작업이 된다. 대신에 문제영역이 다소 적은 사활문제나 바둑판의 규모가 작은 9×9 바둑판 등에 적용하는 것은 단순 실험을 위해 현실성이 있다. 또한 이러한 결과와 기존의 인공지능 방법인 인공신경망, 단순탐색의 결과와 비교하는 것은 향후 매우 흥미로운 실험이 될 것이다.

REFERENCES

[1] B.D Lee and J.W. Park, "Applying Principal Component Analysis to Go Openings", Journal of Korea Game Society, Vol. 13, No. 2, pp. 59-70, 2013.

[2] B.D. Lee, "Applying Neuro-fuzzy Reasoning to Go Opening Games", Journal of Korea Game Society, Vol. 9, No. 6, pp. 117-126, 2009

[3] B.D. Lee, "Korean Pro Go Player's Opening Recognition Using PCA", Journal of Korean Society for Computer Game, Vol. 26, No. 2, pp. 228-223, 2013

[4] B.D. Lee, "Comparison of LDA and PCA for Korean Pro Go Player's Opening Recognition", Journal of Korea Game Society, Vol. 13, No. 4, pp. 15-23, 2013

[5] G. Chaslot, "Monte-Carlo Tree Search", Research Report, Netherlands Organisation for Scientific Research, p. 110, 2010.

[6] Wikipedia, "Monte-Carlo Method", from http://Kowikipedia.org/wiki, 2014.

[7] M.H.M. Winands and Y. Brőrnsson, "Evaluation Function Based Monte-Carlo LOA", from http://www.ru.is/~yngvi/pdf/WinandsB09.pdf, 2014.

[8] A.A.J. van der Kleij, "Monte Carlo Tree Search and Opponent Modeling through Player Clustering in no-limit Texas Hold'em Poker", Master thesis, University of Groningen, 2010.

[9] Yahoo, "To win a tic tac toe game, what is the best first move?", from https://answers.yahoo.com/question/index?qid=

20071126092114AAIyE3Y, 2014.
[10] Autodesk Inc., “Winning tic-tac-toe Strategies”, from http://www.instructables.com/id/Winning-tic-tac-toe-strategies/, 2014
[11] R. Aycock, “How to Win at Tic-Tac-Toe”, from http://www.cs.jhu.edu/~jorgev/cs106/ttt.pdf, 2002.
[12] S. Gelly and D. Silver, “Achieving Master Level Play in 9×9 Computer Go”, Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, 2008.
[13] K. Hughart, “Monte Carlo Tree Search in Borad Game AI”, from http://ebookbrowsee.net/kyle-hughart-monte-carlo-tree-search-in-board-game-ai-docx-d422628740, 2014.

이 병 두 (Lee, Byung Doo)

1982 한양대 원자력공학 학사
1991 서강대 정보처리학 석사
2005 Auckland University 컴퓨터공학 박사
2012-현재 세한대 체육학부 바둑학과 조교수

관심분야 : 컴퓨터공학, 인공지능, 컴퓨터바둑