Journal of Korea Game Society 2018 Oct; 18(5): 77-82
http://dx.doi.org/10.7583/JKGS.2018.18.5.77

게임 프로그래밍/게임과 교육

몬테카를로 트리탐색을 활용한 초소형 바둑에서의 최상의 수순과 덤의 크기

이병두
세한대학교 체육학부 바둑학과
blee026@korea.com

The Best Sequence of Moves and the Size of Komi on a Very Small Go Board, using Monte-Carlo Tree Search

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

요 약

바둑은 최상의 착점을 찾기 위해 컴퓨터가 완전탐색을 하여 모든 가능한 착점들을 탐색할 수 없는 가장 복잡한 보드게임이다. AlphaGo 이전에 모든 강력한 컴퓨터바둑 프로그램들은 게임트리 내 매우 큰 분기수와 국면평가에서의 어려움을 극복하기 위해 몬테카를로 트리탐색(Monte-Carlo Tree Search)을 사용해 왔다. 본 논문에서는 MCTS를 활용하여 초소형 바둑에서의 최상의 수순과 덤의 크기를 알고자 했다. 2줄바둑에서의 게임결과는 빅이 되었으며 덤의 크기는 0집, 반면에 3줄바둑에서는 흑이 항상 승리하고 덤의 크기는 9집이 되어야 함을 알아냈다.

ABSTRACT

Go is the most complex board game in which the computer can not search all possible moves using an exhaustive search to find the best one. Prior to AlphaGo, all powerful computer Go programs have used the Monte-Carlo Tree Search (MCTS) to overcome the difficulty in positional evaluation and the very large branching factor in a game tree. In this paper, we tried to find the best sequence of moves using an MCTS on a very small Go board. We found that a 2×2 Go game would be ended in a tie and the size of Komi should be 0 point; Meanwhile, in a 3×3 Go Black can always win the game and the size of Komi should be 9 points.

Keywords : small Go(소형 바둑), MCTS(몬테카를로 트리탐색), sequence of moves(수순)

Received: Jul. 9. 2018 Revised: Oct. 5. 2018
Accepted: Oct. 7. 2018
Corresponding Author: Byung-Doo Lee(Korea Game Society)
E-mail: blee026@korea.com

ISSN: 1598-4540 / eISSN: 2287-8211

1. 서 론

바둑은 2,500여 년 전에 기원되었으며, 컴퓨터가 오랜 세월에 걸쳐 인간을 제압하지 못했던 게임이었다[1]. 그러나 2015년 말 Google DeepMind사의 AlphaGo-Fan이 최초로 19줄바둑에서 유럽 챔피언 Fan Hui를 5:0으로 승리를 하였고, 2016년 초 AlphaGo-Lee가 세계 최정상급 프로기사인 이세돌 9단에게 4:1로 완승을 거두어 전 세계인을 놀라게 하였다. 또한 DeepMind사는 2016년 말부터 2017년 초까지 AlphaGo-Master를 통해 전 세계 프로기사들에게 60:0이라는 경이적인 승리 기록을 남기었으며, 이후 AlphaGo-Master를 훨씬 뛰어넘는 기력을 가진 AlphaGo-Zero를 발표한 뒤 바둑계를 은퇴했다. 이와 같이 최정상급 프로기사들에게 컴퓨터바둑이 단기간에 위력적인 기력으로 이길 수 있었던 근원은 몬테카를로 트리탐색(MCTS: Monte-Carlo Tree Search)의 활용에 있었다.

몬테카를로 방법은 1940년대에 태동되었으며, B. Abramson은 1987년 그의 박사학위 논문에서 기존의 트리탐색이 아닌 새로운 탐색방식인 몬테카를로 방법을 제안했다[2]. 1992년 B. Brügmann은 Go-playing 프로그램을 통해 처음으로 몬테카를로 방법을 바둑에 구현하였으며[3], 2008년 R. Coulom은 몬테카를로 방법을 게임트리에 접목하여 MCTS를 창안해 냈다[4], 또한 MCTS를 활용한 컴퓨터바둑 Fuego는 2009년에 최초로 9줄바둑에서 최정상급의 프로기사를 제압하게 된다[5]. 세계 최강의 컴퓨터바둑인 AlphaGo(Zero 버전 제외) 역시 심층학습을 통한 MCTS를 적극 활용하였다[6,7].

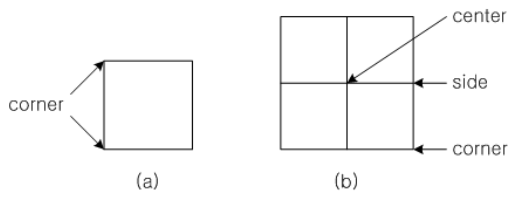
본 논문에서는 아무런 전략을 구사하지 않는 순수 MCTS를 활용하여 초소형 바둑인 2줄바둑과 3줄바둑에서의 최상의 수순과 덤의 크기를 찾고자 했다. 이를 위해 Window 8.1 64bit 운영체제하에서 Intel Core i5-3337U 프로세서를 이용하여 MS C++ 프로그래밍 언어로 1년여에 걸쳐 자체 제작하였다.

2. 본 론

2.1 초소형 바둑

전 세계적으로 공식 대국용으로 사용하는 바둑은 19줄바둑이지만, 바둑 입문자나 어린이 학습을 위해 9줄바둑이 사용하고 있다[8].

본 실험에서는 바둑을 통해 MCTS의 성능을 알아보기 위해 문제의 영역을 최소화한 2줄바둑과 3줄바둑에 MCTS를 적용하였다. [Fig. 1](a)에서 보듯이 2줄바둑은 4개의 귀로만 구성되어 있는 가장 작은 바둑판이 되며, 반면에 3줄바둑은 [Fig. 1](b)에서 보듯이 중앙, 귀, 변으로 된 가장 작은 바둑판이 된다.



[Fig. 1] (a) 2×2 and (b) 3×3 Go boards

2.2 바둑규칙

바둑은 두 대국자인 흑과 백이 19줄 바둑판 위에 있는 361개의 교차점에 흑돌과 백돌을 교대로 착수하여 상대방보다 더 많은 영역을 갖는 쪽이 이기는 경기이다[9]. 게임의 종료는 두 대국자가 연속하여 순서넘김(pass)을 한 경우와 게임 중 똑같은 형태가 반복되는 동형반복(super ko)의 경우가 된다. 순서넘김으로 게임이 종료된 경우에는 계가(scoring)를 하며, 동형반복인 경우에는 계가를 하지 않고 무승부 처리가 된다.

바둑규칙에는 크게 한국식(또는 일본식) 규칙과 중국식 규칙이 있다. 두 규칙에는 다소 차이가 있으나 그 중에 대표적인 것이 덤과 계가가 된다.

2.2.1 덤

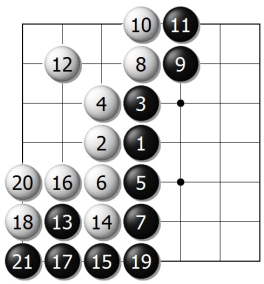
덤은 먼저 두는 흑 대국자가 유리하기 때문에 나중에 두는 백 대국자에게 그 불리함을 보상하는 규칙이 되며, 한국(또는 일본)에서는 6집 반을, 중국에서는 7집 반을 계가할 때에 백에게 이를 공제를 하고 있다[10].

2.2.2 계가

계가는 경기가 끝난 후 승패를 판가름하기 위한 것으로, 한국은 집을 세는 계가(territory scoring) 방식을, 중국은 집과 돌을 세는 계가(area scoring)방식을 사용한다[11]. 즉 다음과 같이 계가를 한다.

- □ 한국식 계가: 자신의 집 안의 빈 점의 개수 + 따낸 상대의 돌의 개수 + 덤
- □ 중국식 계가: 자신의 집 안의 빈 점의 개수 + 살아있는 자신의 돌의 개수 + 덤

한 예로 7줄바둑에서의 게임이 종료된 [Fig. 2]인 상황에서 한국식 계가를 적용하면 흑 19점, 백 9점이 되어 덤이 없는 경우에 흑이 10집(= 19점 - 9점) 승이 된다. 반면에 중국식 계가를 적용하면 흑 30점(= 19점 + 11점), 백 19점(= 9점 + 10점)이 되어 덤이 없는 경우에 흑이 11점 앞서게 된다.



[Fig. 2] Final position

2.2.3 빅

빅(seki)은 먼저 단수(單手)를 치는 쪽이 손해를 보게 되기 때문에 서로 단수를 칠 수 없는 무승부 상태를 말한다[12]. 한국식 규칙에 따르면 빅을 구성하고 있는 돌들은 모두 살아있는 것으로 처리하고, 계가를 할 때 빅을 구성하고 있는 돌들을 제외한다. 반면에 중국식 규칙에 따르면 빅을 구성하고 있는 돌들은 살아있는 자신의 돌로 간주하여 계가를 한다[11].

2.3 MCTS

MCTS는 국면평가함수[1]를 사용하지 않는 최대우선탐색이 되며, 탐색공간에 대한 무작위적인 탐험을 실시하여 최상의 근사값을 구하면서 게임트리의 크기를 획기적으로 줄여준다[13-15].

MCTS는 [Fig. 3]과 같이 선택단계, 확장단계, 시뮬레이션단계, 역전파단계라는 네 단계를 수행한다[16].

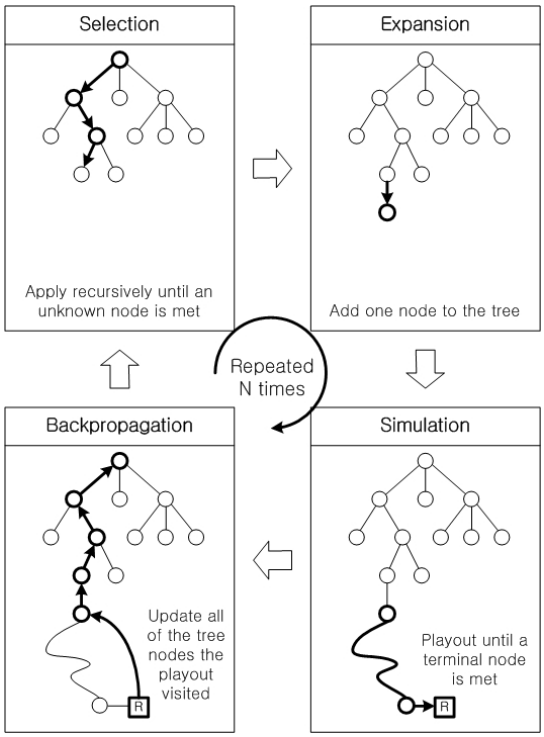
□ 1단계(선택단계): 트리정책(tree policy)을 반복적으로 적용하여 트리 내 단말노드를 만날 때까지 계속 수행을 한다.

□ 2단계(확장단계): 선택단계에서 선발된 하나 이상의 자식노드가 트리에 첨가된다.

□ 3단계(시뮬레이션단계): 확장단계에서 선발된 자식노드로부터 디폴트정책(default policy)에 준하여 시뮬레이션을 시작하여 단말노드를 만날 때까지 시뮬레이션을 지속한다. 참고로 시뮬레이션을 플레이아웃(play-out) 또는 롤아웃(roll-out)이라고도 한다.

□ 4단계(역전파단계): 시뮬레이션 단계로부터 얻어진 결과값을 선택단계에서 선발된 트리 내 자식노드로부터 뿌리노드로 역전파하며 결과값을 갱신해 나간다.

1) 국면(局面): 바둑 진행의 단면(單面) 또는 형세(形勢)



[Fig. 3] Four steps for MCTS

2.4 실험 시 고려된 사항

한 번의 플레이아웃 후 결과값을 계산하기 위해 흑이 이기는 경우에는 1, 백이 이기는 경우에는 0, 그리고 무승부인 경우에는 0.5의 승률값을 부여하였다. 또한 프로그래밍 작성 시 고려된 사항은 다음과 같다.

□ 대국 종료는 두 대국자가 연속으로 순서넘김을 한 경우로 하였다.

□ (덤을 고려하게 되면 착수 시 전략이 바뀔 수 있으며, 이를 고려하는 것은 매우 방대한 작업이 되는 이유로 인해) 덤은 고려하지 않았다.

□ (한국식 계가를 적용하는 경우에는 프로그래밍 시 고려해야 될 사항이 많은 이유로) 중국식 계가를 적용하였다.

□ 마지막으로 (사활, 장문, 촉촉수 등과 같은 하위 바둑지식을 고려하지 않은) 무전략 순수 MCTS 기법을 적용하였다.

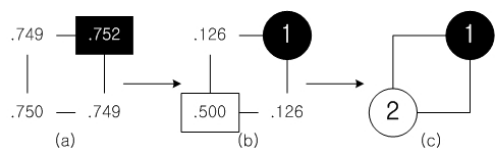
2.5 실험 결과

2.5.1 2줄바둑에서의 최상의 수순과 덤

2줄바둑에서 최상의 첫 번째 수를 찾아내기 위해 MCTS를 사용하여 착수가 가능한 네 귀에 대해 각각 250,000번의 플레이아웃을 실시하였다. [Table 1], [Fig. 4](a)에서 보듯이 첫 번째 수로 착수가 가능한 4곳의 평균승률값이 75.1±0.1%가 됨을 알 수 있었다.

[Table 1] Average win rates of the first positions

Go size	center	corner	side
2×2	N/A	75.1±0.1%	N/A
3×3	82.7%	41.6±0.2%	61.4±0.9%



[Fig. 4] Average win rates of each position

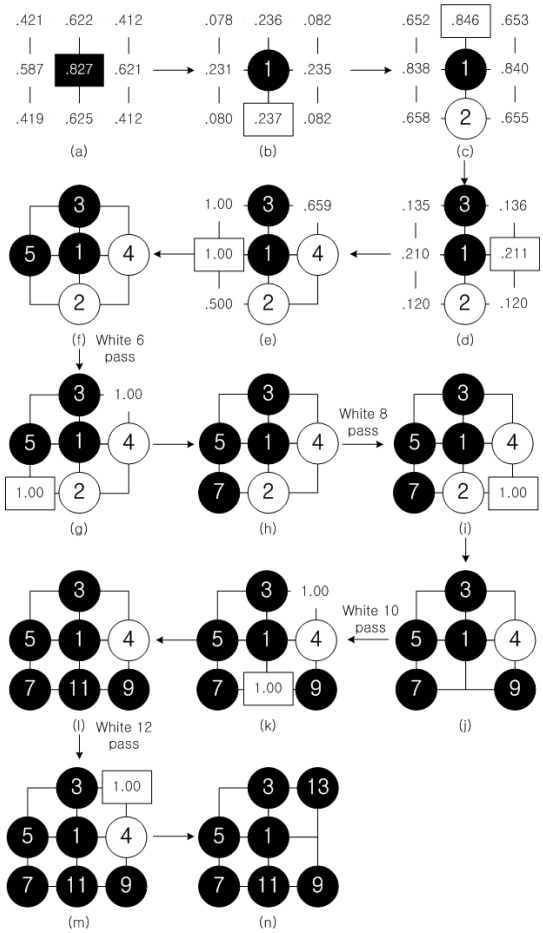
결국 [Fig. 4](a)에서 보듯이 흑은 75.0% 정도의 평균승률을 보여 첫 수로 아무 곳에 두어도 쉽게 지지 않는다는 것을 알 수 있다. 그러나 두 번째 대국자인 백은 흑1의 대각선 방향의 곳에 대한 평균승률값이 [Fig. 4](b)에서 보듯이 50.0%로 가장 크기 때문에, [Fig. 4](c)와 같이 백2는 흑1의 대각선 방향으로 두게 된다. 이렇게 함으로서 쌍방은 빅의 상태가 되어 무승부가 되는 것을 알 수 있다.

결국 2줄바둑에서의 최상의 수순은 흑1에 대해 백은 흑돌을 마주보는 대각선 위치인 백2가 되며, 게임 결과는 빅이 된다. 그러한 이유로 덤의 크기는 0집이 된다.

2.5.2 3줄바둑에서의 최상의 수순과 덤

3줄바둑 역시 아주 작은 크기의 바둑이 되며, 2줄바둑과 달리 중앙, 귀, 변으로 구성되어 있다. 즉 중앙 1곳, 귀 4곳, 변 4곳으로 구성되어 있다.

3줄바둑에서의 최상의 첫 번째 수를 구하기 위해 2줄바둑에서와 같이 MCTS를 활용하여 착수 가능한 9곳에 대해 각각 250,000번의 플레이아웃을 실시하였다. 실시 결과 [Table 1], [Fig. 5](a)에서 보듯이 착수 가능한 9곳의 평균승률값을 살펴보면 중앙 82.7%, 귀 41.6±0.2%, 변 61.4±0.9%가 되어 중앙이 가장 높다는 것을 알 수 있다.



[Fig. 5] The most promising sequence of moves

흑1에 이어 두 번째 대국자인 백은 [Fig. 5](b)에서 보듯이 하변의 평균승률값이 23.7%로 가장 크기 때문에, [Fig. 5](c)와 같이 하변에 백2를 두게 된다. 비슷한 방법으로 다음 대국자인 흑은 [Fig. 5](c)에서 보듯이 상변의 평균승률값이 84.6%로 가장 크기 때문에, [Fig. 5](d)와 같이 상변에 흑3을 두게 된다. 이와 같은 방법으로 진행해 보면 [Fig. 5](n)과 같은 상황으로 게임은 종료가 되며, 이후 중국식 계가를 하게 되면 흑이 9점을 이긴 것을 알 수 있다.

결국 3줄바둑에서의 최상의 흑의 첫 번째 수는 중앙이 되며, 이후 게임을 진행하여 [Fig. 5](f)와 같이 흑이 1, 3, 5 형태를 유지하게 되면 집이 되어, 게임의 결과는 [Fig. 5](n)에서 보듯이 항상 흑승으로 끝나는 것을 알 수 있다. 그러한 이유로 중국식 계가를 적용한 3줄바둑에서의 덤의 크기는 9집이 되어야 함을 알 수 있다.

3. 결론 및 제언

2016년 초 AlphaGo-Lee와 이세돌과의 세기적인 바둑대국의 결과로 전 세계인은 인공지능에 대한 관심이 고조됐으며, 아울러 경계심마저 불러 일으켰다. 저자는 MCTS의 성능과 결과만을 보여주는 기존의 논문과는 달리 MCTS를 활용한 초소형 바둑에서의 가장 이상적인 돌의 수순과 바둑판의 규모에 따른 덤의 값을 측정하고자 했다. 실험결과에 따르면

□ 2줄바둑에서의 최상의 수순은 먼저 놓은 흑돌에 대해 백은 흑돌의 대각선 방향으로 놓아야 하며, 게임 결과는 빅이 된다는 사실과 덤의 크기는 0집이 되어야 함을 알 수 있었다.

□ 한편 3줄바둑에서의 흑은 첫 번째 수로 중앙에 두어야 하며, 최상의 수순으로 진행된 경우의 게임 결과는 항상 흑의 승리가 되며, 덤의 크기는 9집이 되어야 함도 알 수 있었다.

향후 컴퓨터바둑의 성능을 개선시키기 위해서는

본 논문에서 시행한 무전략 순수 MCTS가 아닌 전략과 전술을 구사할 수 있는 MCTS를 구현할 필요가 있다. 즉 MCTS에서의 제3단계인 시뮬레이션단계에서 디폴트정책인 아닌 트리 내 단말노드의 상태를 제대로 짧은 시간 내에 정확히 파악해 낼 수 있는 새로운 정책 개발이 요구된다.

ACKNOWLEDGEMENTS

This paper was supported by the Sehan University Research Fund in 2018.

REFERENCES

[1] B.D. Lee, “The first move in the game of 9×9 Go, using non-strategic Monte-Carlo Tree Search”, Journal of Korea Game Society, Vol. 17, No. 3, pp. 63-70, 2017.
[2] B. Abramson, “The Expected-Outcome Model of Two-Player Games”, PhD thesis, Columbia University, 1987.
[3] B. Brügmann, “Monte Carlo Go”, Technical report, Syracuse University, 1993.
[4] R. Coulom, “The Monte-Carlo Revolution in Go”, Japanese-French Frontiers of Science Symposium, 2008.
[5] M. Enzenberger and M. Müller, “Fuego - An Open-Source Framework for Board Games and Go Engine Based on Monte Carlo Tree Search”, Technical report, University of Alberta, 2008.
[6] D. Silver, et al., “Mastering the game of Go with deep neural networks and tree search”, Journal of Nature, Vol. 529, Issue 7587, pp. 484-489, 2016.
[7] T. Song, “Monte Carlo Tree Search and Its Application in AlphaGo”, from https://stlong0521.github.io/20160409%20-%20MCTS.html, 2017.
[8] Sense's Library, “Small board Go”, from https://senseis.xmp.net/?SmallBoardGo, 2018.
[9] S.H. Jung, et al., “Modern Baduk theory”, Dacom Press, 2016.
[10] Wikipedia, “Komidashi”, from https://en.wikipedia.org/wiki/Komidashi, 2018.
[11] Wikipedia, “Scoring”, from https://ko.wikipedia.org/wiki/%EA%B3%84%EA%B0%80, 2018.
[12] Wikipedia, “Seki”, from https://ko.wikipedia.org/wiki/%EB%B9%85_(%EB%B0%94%EB%91%91), 2018.
[13] H. Baier and M.H.M. Winands, “Monte-carlo Tree Search and Minimax Hybrids”, Computer Games, Vol. 504, pp. 45-63, 2014.
[14] M.H.M. Winands and Y. Brörnsson, “Evaluation Function Based Monte-Carlo LOA”, from http://www.ru.is/~yngvi/pdf/WinandsB09.pdf, 2017.
[15] S. Gelly, et al., “The Grand Challenge of Computer Go: Monte Carlo Tree Search and Extensions”, Communications of the ACM, Vol. 55, No. 3, pp. 106-113, 2012.
[16] E. Powley, et al., “Heuristic move pruning in monte carlo tree search for the strategic card game lords of war”, In Computational Intelligence and Games (CIG) of IEEE, pp. 1-7, 2014.

이 병 두 (Lee, Byung-Doo)

1982 한양대학교 원자력공학 학사
1991 서강대학교 정보처리학 석사
2005 Auckland University 컴퓨터공학 박사
2010～현재 세한대 체육학부 바둑학과 조교수

관심분야 : 컴퓨터공학, 인공지능, 컴퓨터바둑