

# Implementation of Decision Tree in AlphaGo Zero Deep Mind

Kevin Fernaldy 13516109

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia

kevin.fernaldy@students.itb.ac.id

**Abstract**—Decision Tree has been a vital part of Artificial Intelligence for years. It is a part for a robot/neural network to think and make a decision of their own. These robots are first taught by the engineers before they can do self-taught by themselves. However, a recent news about AlphaGo has made a breakthrough of Artificial Intelligence self taught, without any help of human. An AI that can pseudo-make his decision based on self made decision tree. How can a AI neural network make its decisions itself?

**Keywords**—AlphaGo Zero, Decision Tree, Go

## I. INTRODUCTION

We may have heard a project from Google's Deep Mind that has beaten professionals players of Go, such as Lee Sedol, 18-time world champion[1] and 9 dan rank[2] of Go. Lee Sedol is one of the best players in Go matches, however a computer (in this case, artificial intelligence) has beaten Lee Sedol in a 5 Go match, 4 out of 5. The artificial intelligence (AI) that beaten Lee Sedol is a project called AlphaGo. AlphaGo is one of the Google Deep Mind project that implements a complex decision making in a game of Go.

The success of AlphaGo has brought to us fascination and curiosity of how a computer can process and implement best moves out of all the other moves that it can choose. And now, Google has made a new project called AlphaGo Zero that is more smart and powerful than the AlphaGo Lee (the original AlphaGo that plays against Lee Sedol). The complexity of neural network and processing of the long binary tree has made this AlphaGo Zero the hardest computer to play against.

This paper will explore a simple breakdown of how AlphaGo Zero self-taught decision construct a complex and advanced decision tree, and also introduces a simple rules of Go, a traditional and very old Chinese board game.

## II. A SHORT SUMMARY OF GO PLAY RULES

Go is a traditional Chinese board game that has simple, but hard to implement set of rules. To get a grasp of the contents of this paper, one should understand the basics of Go. The simplified rules of Go[3] consist of :

### 1. The Go Board

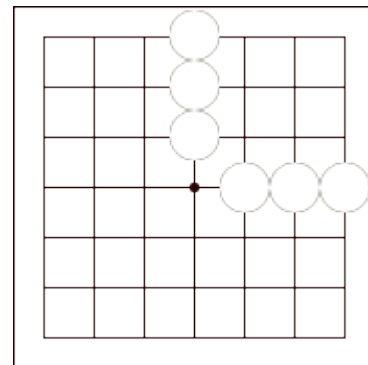
Go is played by two players, one black and one white.

The game is played on a square board with 19x19 lines. The smaller boards are 13x13 and 9x9 for

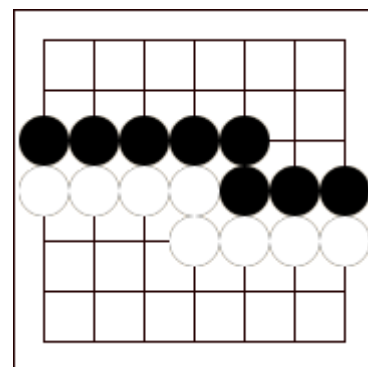
smaller games and teaching purposes

### 2. Territory

The goal of Go is to gain as many points as possible, by capturing territories or capturing opponent's stones. Territory is empty spaces that surrounded by stones of the same color, and the edge of the board acts as a natural border.



Picture 1 : Territory with natural border



Picture 2 : An example of finished Go match

Picture 1 shows that white has surrounded 9 spaces, which gives him 9 points. Picture 2 shows that white has surrounded 17 spaces, and black has surrounded 16, which is a win by 1 point by white.

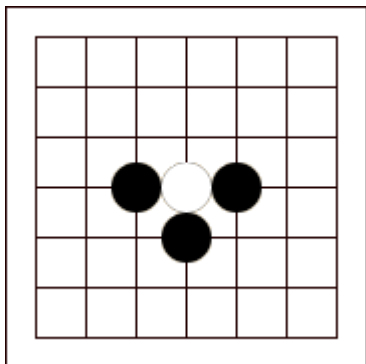
### 3. Movement

Go is placed by placing the stones on empty intersections of the lines. Black always moves first. The placement of stones are as fits as players like, with some exceptions of illegal moves that will be stated below. Once all territories are surrounded, a

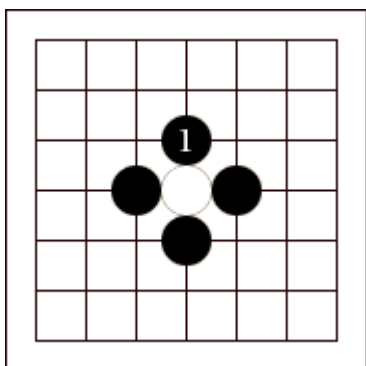
player may pass, and can be followed by the other player with another pass to end the game and begin the score counting. Note that placing a stone once all territories are surrounded will cause a penalty

4. Capturing

While the main goal of Go is to surround as many territories as possible, there are possibility of capturing stones, and the actions to prevent such captures. To capture a stone, a player must surround a group of other player's color stone, with no empty space, with an exception of diagonal placement.



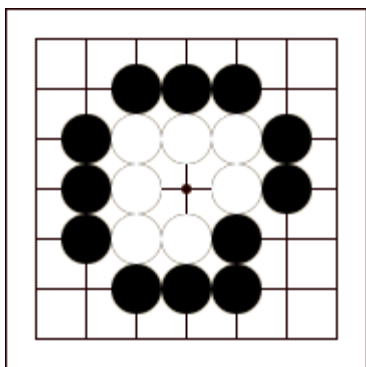
Picture 3 : Black has the white nearly surrounded



Picture 4 : Black has placed and surrounded the white

In Picture 3, it is shown that the black stones almost captures the white stone. This condition is called *atari* for the white player, in which white is prone to be captured in the next move by black. In Picture 4, the black player finally fills the last tile, and surrounding the white stone. In this situation, the owner of black stone has an option to capture the white stone.

5. Eyes



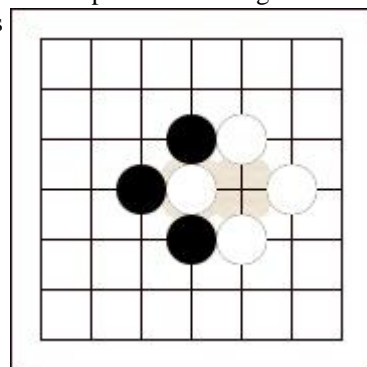
Picture 5 : An example of an eye

An eye is occurred when a single empty space exists inside a group of same color stones. One of the key themes in Go is to create eye spaces for your own groups and prevent the opponent from making ones.

In Picture 5, the black stone can kill the entire surrounded white group by placing a black stone inside the remaining empty space in the middle. This move is legal if and only if the white group is completely surrounded from the outside, as leaving one empty space outside the white group is a suicide move for the black stone, which counts as illegal move.

6. Ko

Ko is a situation where capturing a stone is vulnerable to being recaptured immediately, which would repeat the original situation, ending in an endless loop. A special rule prevents the game from going on endlessly.



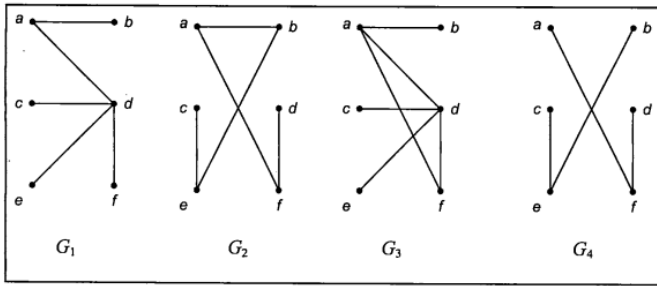
Picture 6 : A Ko situation in which both players can recapture

Picture 6 shows a Ko. If the owner of the black stone places its stone in the middle of white group, it will capture the white stone inside the black group. However, the owner of white stone can recapture the black stone. The special rules then interferes with a statement "the owner of the captured stone may not recapture to prevent endless loop". The white owner can however, make a threatening move on the other groups that must be prevented later by the black so the white stone owner can recapture the Ko.

### III. TREE AND DECISION TREE

#### A. Tree and Rooted Tree

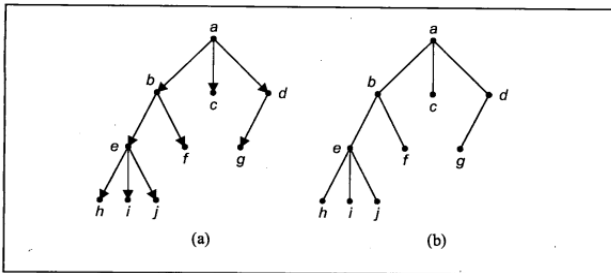
A tree is a connected, undirected graph that does not contains any circuits[4]. Because tree is referenced from graph theory, a tree can only have a vertex without any edges.



Picture 6 : G1 and G2 is a tree, while G3 and G4 is not

If  $G = (V,E)$  is a tree, then  $V$  cannot be an empty set, however  $E$  can be empty. This term is applicable to any form of tree. However, the definition of tree above is also called a free tree. For decision tree, we will be using a rooted tree.

A rooted tree is a tree that all of the vertex is enacted as a root, and all of the edges are directed away from the roots[5]. A root has an entry-degree of zero, while the others have one. Vertex that has zero exit-degree is called a leaf, whereas vertex that has non-zero exit-degree is called an inner-vertex or branch-vertex.



Picture 7 : Example of rooted-tree. The normal ones we use is the (b) one

### B. Decision Tree

A rooted tree can be used as a method to model problems, which a series of decisions can leads to a solution. This is called a decision tree. Decision tree is a rooted tree in which each internal vertex corresponds to a decision, with a subtree at these vertices for each possible outcome of the decision[6]. A decision tree is constructed with a directed graph,

$$G = (V, E), E \subset V^2$$

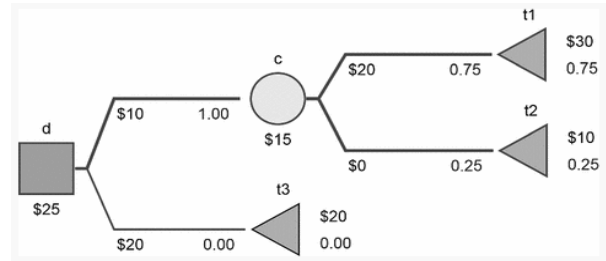
with set of nodes  $V$  split into three disjoint sets,

$$V = D \cup C \cup T \quad D = \text{Decision node}$$

$C = \text{Chance node}$

$T = \text{Terminal node}$

Reference [7] shows, in decision node, the decision maker selects an action from one of the edges that represents possible solutions to the problem to the leaves of the decision tree, chance nodes stems from the node is selected randomly, and terminal nodes represent the end of actions or a solution of a problem.



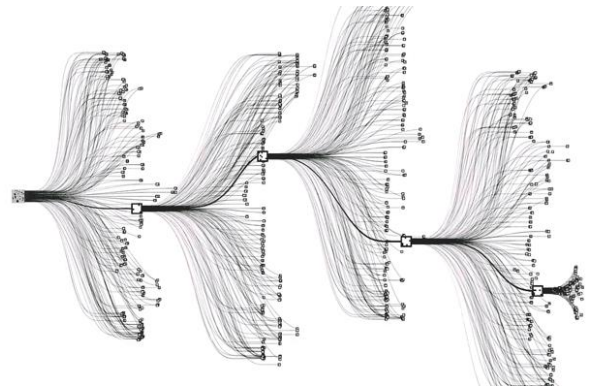
Picture 8 : Example of Decision Tree, decision node is square, chance node is circle, and terminal nodes are triangle

If we make a decision maker to select actions with the highest payoff from all, the decision tree will select the terminal node t1, with the highest probability and the most immediate action among the other terminal nodes.

## II. ALPHAGO ZERO

### A. The Vast Probability Tree of Go

The game Go is a game with a huge probability tree. With 19x19 standard match board game and random placement of stones, an opening move of go has 361 probabilities. This stacks with the next remaining possible moves, ending with a total of  $2.08 \times 10^{170}$  moves[8]. This moves are much higher than the well-known and popular other board game, Chess.



Picture 9 : An image representing Go possible moves

As we can see from the picture that the possible moves of Go is really huge, with every moves creates tens and hundreds of new possible moves. Even a professional Go player have a limited capacity of processing the move possibility and the outcome of it afterwards. Google's AlphaGo Zero however, has an impressive neurological network that surpass human, computing every possible moves and choosing the best moves out of them all.

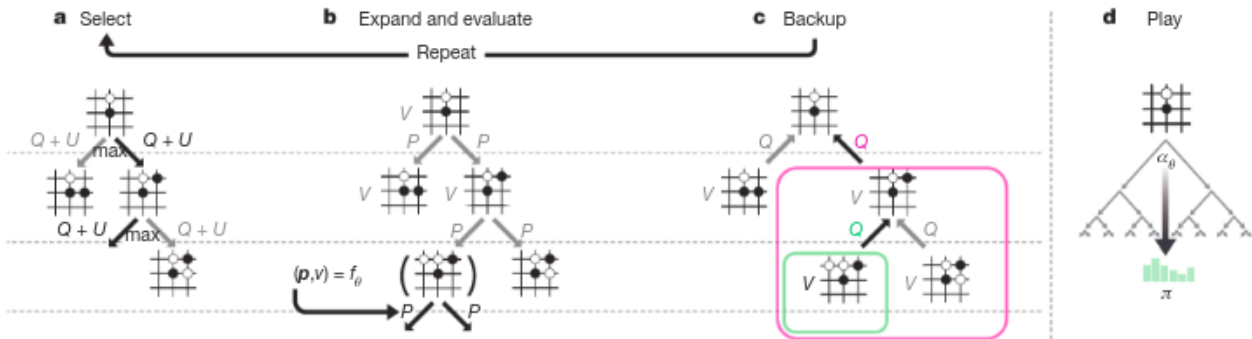
The vast probability of Go becomes a challenge for the engineers of AlphaGo to make a smart, independent machine that self-learns itself in Go and getting smarter and smarter each simulation of training. With the implementation of Artificial intelligence in Go, it can branches into various subjects of artificial intelligence and opens a new possibilities in the future.

## B. AlphaGo Zero Training

AlphaGo Zero uses a self-play reinforcement learning, that uses a deep neural network. The neural network takes a raw input of the board game position and history  $S$ , and gives 2 outputs consist of  $P$  and  $V$ , with  $P$  is a vector move probability of selecting each move including pass, and  $V$  estimating the probability of current player winning from position  $S$  [9]. The equation of the neural network input output system can be written as follows

$$(P, V) = f_{\theta}(S)$$

Upon learning, the AlphaGo Zero neural network uses MCTS or Monte Carlo Tree Search to search and execute the best moves out of the all probabilities ( $\Pi$ ). Reference [10] shows us a Picture 10 and a detailed explanation how the Decision Tree of AlphaGo Zero,



Picture 10 : AlphaGo Zero Learning Decision Tree

The first move is to select a section and traverse the tree by selecting the edge with maximum action value  $Q$ , plus an upper confidence  $U$  that depends on a stored probability  $P$  and visit count  $N$  for that edge. The  $Q$  minimizes the probability of opponent making a move. The  $U$  and  $N$  provides the probability of a next moves are played by the opponent and how to counter attack them.

The second move is to expand the leaf node from the selection. The current position  $S$  is evaluated by the neural network equation  $(P(S, \cdot), V(S)) = f_{\theta}(S)$ . The output of the measurement  $P$  then stored in the outgoing edges from  $S$ .

The action value  $Q$  is now updated to track the mean of all evaluations of in the subtree below the current selection  $S$ . Action value  $Q$  are updated every time a cycle of selection and expanding occurs for every possible positions are there in a decision tree. The result measurement of the sections are summarized into search probability  $\Pi$ , and the AlphaGo Zero then chooses the best moves out of all the probability it measured.

The possible solutions of the problems are shown in the paths to the leaves of the rooted trees.

## V. CONCLUSION

In conclusion, the AlphaGo Zero is very impressive in calculating the best solution in every move from a Decision Tree. Decision Tree has been and always been a vital part of Artificial Intelligence decision making.

## VII. ACKNOWLEDGMENT

I would like to thank you for Mr. Rinaldi Munir, our teacher of IF2120 Matematika Diskrit for sharing to us his knowledge and wisdoms, my family giving emotional supports, and my close friends who also gives me emotional supports making

this paper.

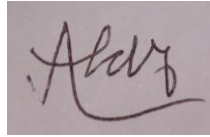
## REFERENCES

- [1] XINHUA, "Lee Sedol expects 'not easy' game with AlphaGo in 3<sup>rd</sup> Go match", [https://www.shine.cn/archive/article/article\\_xinhua.aspx?id=322918](https://www.shine.cn/archive/article/article_xinhua.aspx?id=322918), accessed on December 3<sup>rd</sup>, 2017.
- [2] Lee Sedol Go Biography, <http://gobase.org/information/players/?pp=Lee+SeDol>, accessed on December 3<sup>rd</sup>, 2017.
- [3] [https://www.pandanet.co.jp/English/learning\\_go/learning\\_go\\_2.html](https://www.pandanet.co.jp/English/learning_go/learning_go_2.html), accessed on December 3<sup>rd</sup>, 2017.
- [4] Rinaldi Munir, "Matematika Diskrit Revisi Keempat", Bandung, Indonesia: Informatika Bandung, 2010, pp. 444.
- [5] Rinaldi Munir, "Matematika Diskrit Revisi Keempat", Bandung, Indonesia: Informatika Bandung, 2010, pp. 457.
- [6] Bogumił Kamiński, et. al., "A framework for sensitivity analysis of decision trees", Online Paper, <https://link.springer.com/article/10.1007%2Fs10100-017-0479-6>, Central European Journal of Operations Research, accessed on December 3<sup>rd</sup>, 2017.
- [7] *loc. cit.*
- [8] Sensei's Library, <https://senseis.xmp.net/?NumberOfPossibleGoGames>, accessed on December 3<sup>rd</sup>, 2017.
- [9] David Silver, et al., "Mastering the game Go without human knowledge", Online Article, [https://www.nature.com/articles/nature24270.epdf?author\\_access\\_token=VJXbVjaSHxFoctQQ4p2k4tReN0jAjWel9jnR3ZoTv0PVW4gB86EEpGqTRDtplz-2rmo8-KG06gqVobU5NSCFeHILHeVFUeMsbvwS-lxjqQGg98faovwjxeTUgZAUMnRQ](https://www.nature.com/articles/nature24270.epdf?author_access_token=VJXbVjaSHxFoctQQ4p2k4tReN0jAjWel9jnR3ZoTv0PVW4gB86EEpGqTRDtplz-2rmo8-KG06gqVobU5NSCFeHILHeVFUeMsbvwS-lxjqQGg98faovwjxeTUgZAUMnRQ), pp. 355, accessed on December 3<sup>rd</sup>, 2017.
- [10] *loc. cit.*

## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 3 Desember 2017

A square image containing a handwritten signature in dark ink. The signature is cursive and appears to read 'Kevin Fernaldy'.

Kevin Fernaldy  
13516109