Positional Strategy and Minimax Algorithm in Reversi

Gaudensius Dimas Prasetyo Suprapto - 13514059
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13514059@std.stei.itb.ac.id

***Abstract*—Reversi, also known as Othello, is a tiled-board game that has to have exactly two players to play. Similar to Chess, Reversi has an 8×8 - tiled board, and requires in-depth strategy to aim for a win.**

***Keywords*— Reversi, board game, Minimax algorithm.**

I. Introduction

Games are instruments that human made to entertain themselves. The instruments used for a game varies, be it a ball to play basket ball, a racket to play tennis, or even some simple rules to play tag. There is an interesting type of game that used a certain instrument to play, namedly board game.

Board game also has its own classification, based on the age of the game. There are modern board games, which are usually made specifically for common people as the target players, for examples, Monopoly (©), Game of Life (©), etcetera. Modern board games usually has rules that are easy for common people to understand, and usually does not need in-depth strategy for a win, instead, in most modern board games, you need luck to win.

On the other hand, ancient board games have a not-really-that-hard-to-understand rules, probably much simpler than that of a modern board games, usually require exactly two players to play, yet interestingly, they require highly in-depth strategy for a win without relying on luck at all. Because of that, it is not uncommon for people to consider it as a “brain sport”, chess is one of the good examples.

This paper will tell you about how to improve your strategy in playing one of the ancient board, namedly Reversi, or also known as Othello, using a certain algorithm to make a decision tree which is based on positional strategy.

II. Basic Theory

Before we go to the interesting part of this paper, it is good for us to study first about graphs and trees, since this paper will mainly discuss about algorithm to make the decision tree to play reversi.

A. Graph Theory (defined in ref. [1])

Graph is a tuple set of (V, E), in which V stands for vertex and represents a set of vertices, and E stands for edge, and represents a set of edges that connect one vertex to the other. A graph should have at least one vertex in the set V, and such graph is known as a **trivial graph**.

Example of a graph notation:
G = (V, E) , in which:
V = {v_1, v_2, v_3, ... }
E = {e_1, e_2, e_3, ... }

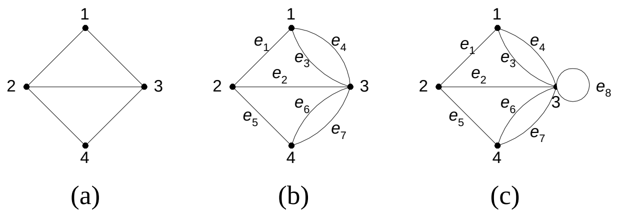


Diagram 1. Graph Examples

Take a closer look at the diagram 1.b. You will notice that there are edges that connects exactly the same two vertices, for example, e_6 and e_7 connect the same vertices, vertex 3 and vertex 4. Such edges are called as **parallel edges**. You can also find another type of edge at the diagram 1.c., the **loop edge**.

Based on the existence of such type of edges, graphs are classified into two types:

1. **Simple graph**,
   is a graph that does not contain neither the parallel edges nor the loop edge. Diagram 1.a is a good example.
2. **Unsimple graph**,
   is a graph that contain at least one of those two types of edges. Diagram 1.b and 1.c are the examples of it.

However, based on the edge direction of the graph, graph can be classified into two types:

1. **Undirected graph**,
   is a graph without any directions on all of its edges. All the graphs in diagram 1 are examples of undirected graphs.
2. **Directed graph**,
   is a graph with directions on all of its edges. All the graphs in diagram 2 are classified as directed graphs.

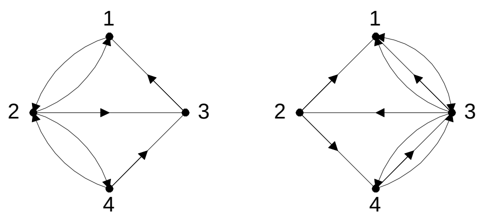


Diagram 2 Directed Graphs

In graph theories, we have several terminologies. Some important terminologies we need to know are:

1. **Adjacency**
   Two vertices are said to be adjacent to each other if they are connected an edge. For example, in G_1 in Diagram 3, vertex 1 is adjacent to vertex 2.

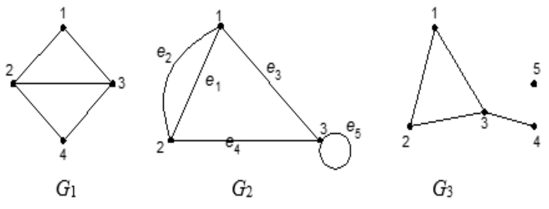


Diagram 3 Another Graph Examples

2. **Incidency**
   If an edge connect vertex 1 and vertex 2, it can be said that the edge is incident with vertex 1, and at the same time, is incident with vertex 2. For example, in G_2, edge e_1 is incident to vertex 1, and at the same time, is incident to vertex 2.
3. **Isolated Vertex**
   is a vertex that does not has any edge attached to itself. Vertex 5 in G_3 in diagram 3 is a good example of it.
4. **Null/Empty Graph**
   is a graph without any edges. It can be inferred that a null graph consists of only isolated vertices. An example of a null graph is Diagram 4.

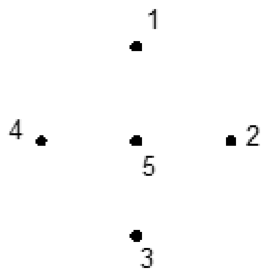


Diagram 4 Null Graph

5. **Degree**
   A vertex' degree is defined as the number of the edges attached to the vertex. For example, in G_1 in Diagram 4, vertex 2 has a degree of 3, and vertex 1 has a degree of 2.
6. **Path**
   Path is like an exploration of edges from a certain vertex to another certain vertex as the destination. The length of a path is defined by the number of the edges the path has passed to reach its destination.
7. **Cycles / Circuits**
   is a path with its starting vertex as its destination, as if the path was a big loop involving 2 or more vertices.

B. Tree Theory (defined in ref. [1])

Trees are defined as undirected graphs that does not contain any circuits in them. A group of trees that do not connect to each other is called as a forest.

If given a certain graph with circuits, we can get spanning trees from the graph by cutting off the circuits. From any graph, we can at least get one spanning tree from it.

From a weighted graph, we can get a minimum spanning tree by applying the **Prim's Algorithm**. The steps of using the Prim's Algorithm are:

1. Choose an edge with minimum weight from all of the edges in the graph and add it into the spanning tree *T*
2. Choose an edge incident with the vertex in *T*, but does not form a circuit inside *T*. Add it into *T*.
3. Repeat step 2 until you get all the vertex without getting any circuit.

Like the graph theories, tree theories also have terminologies we need to understand. Those terminologies are:

1. **Child(s) and Parent**
   In Diagram 5, *b, c,* and *d* are the children of *a*, and otherwise, *a* is the parent of *b, c,* and *d.*

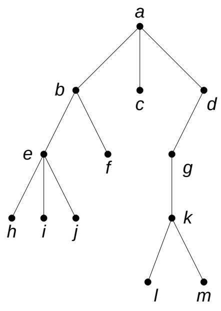


Diagram 5 Tree

2. **Path**
   Path in tree theory is similar to that of a graph theory. For example, the path from *a* to *f* is *a, b, f.* Therefore, the length of the path is 2.
3. **Siblings**
   In Diagram 5, *h, i,* and *j* are siblings to each others, however, they are not siblings with *k* as they have different parents.
4. **Subtrees**
   are the trees inside a tree. For example, in Diagram 5, if you cut the edge that connects *a* and *b*, you will get a forest out of a tree, with tree with *b, e, f, h, i,* and *j* as the subtree of the former tree.
5. **Degrees**
   of a vertex in a tree is defined by number of the children the vertex have. For example, in Diagram 5, *k* has a degree of 2, since *k* has two children, *l* and *m*.
6. **Root**
   is the starting vertex, for example, *a* is the root of the tree in Diagram 5.
7. **Leaf**
   is a vertex in a tree that does not have any children. For example, *h, i, j, l,* and *m* are the leaves of the tree in Diagram 5.
8. **Internal nodes**
   is the vertex that has at least a child, except the root itself.
9. **Level**
   The level of a vertex is defined by the length of path from the root of the tree to the vertex. The level of a root is 0.
10. **Depth/Height**
    is the maximum level of a tree. The level of the tree in Diagram 5 is 4.

C. Reversi (defined in ref. [2])

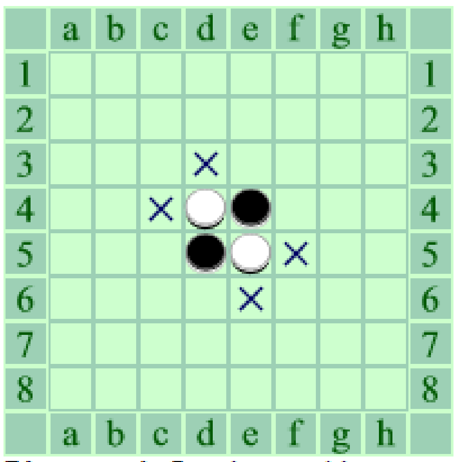


Diagram 6 Starting Position of Reversi [2]

Reversi requires two players, opposing each others by identifiable color similar to chess, black and white. The difference is, black get the first turn. At each turn a player must place a disc with their color face up on one of the empty squares of the board near to the opposing disc such that one or more straight lines (horizontal, vertical or diagonal) are formed from the newly placed disc through one or more of the opponent's discs from the other discs of their own color already on the board. All the opposing discs in between are flipped into the color of newly laid disc afterwards. The symbol X in Diagram 6 shows the example of the legal move black can make. A player cannot pass his turn unless he has no legal move.

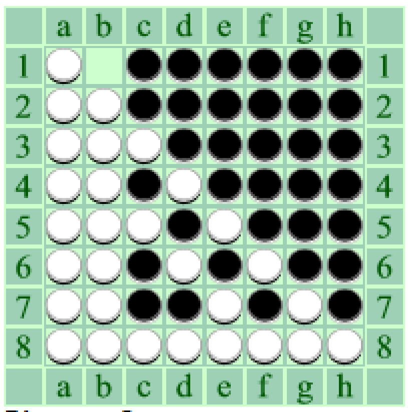


Diagram 7 The game ends even though not all tiles are filled by the discs [2]

The game will continue on and on until none of the player can make a valid move, commonly when all 64 tiles on the board is filled by the disc. The victor is decided by the most discs that turned into their colors at the end of the game.

There are many strategies on how to win reversi. The one this paper will explain about is the positional strategy.

III. Positional Strategy and Decision-tree-Making Algorithm in Playing Reversi

A. Positional Strategy [3]

In playing Reversi, certain tiles on the board hold different values to each other. There are tiles which can lead you to a victory, but there are also tiles that you will want to avoid taking in order to win.

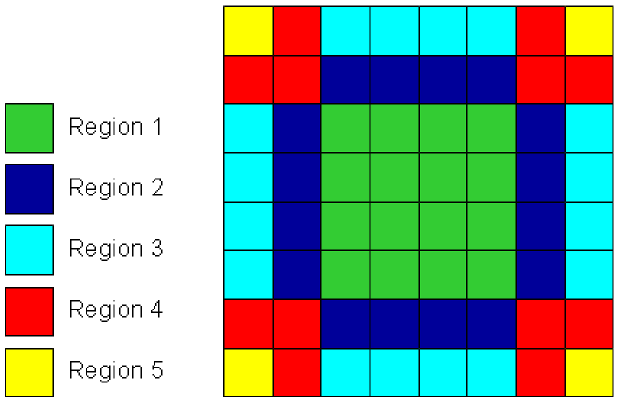
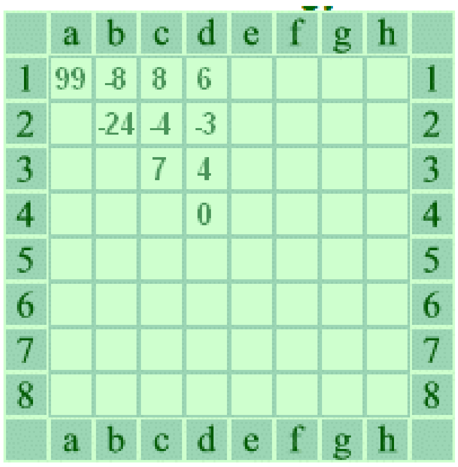


Diagram 8 Positional Strategy

As you all may know, region 5, alias the corner, is often considered as the key to victory, and it is often to be true, since no disc will be able to reclaim the corner once taken, enabling the corner-taker to expand the unreclaimable discs. Because of that, normal players would want to avoid region 4, since it is highly risky because the opponent may use your disc in that region as a stepping stone to get the corner. Region 3 is also favorable to take, since it is hard for opponent to reclaim the edge even though it is still possible, and normal player would tend to avoid region 2 with the same reason as they avoid taking the region 4.

To calculate the value of the tile, we can give the value of each tile certain points.



	a	b	c	d	e	f	g	h	
1	99	-8	8	6					1
2		-24	-4	-3					2
3			7	4					3
4				0					4
5									5
6									6
7									7
8									8
	a	b	c	d	e	f	g	h	

Diagram 9 Value Point in Each Tiles [2]

B. Minimax Algorithm [3]

Now that we have learned about the positional strategy, we can move on to the algorithm to make the decision tree to improve the strategy in playing reversi, namedly the Minimax Algorithm. However, the Minimax Algorithm explained in this paper is not applied to the scores progress, but it will be combined with the positional strategy.

Minimax Algorithm, found by John von Neumann in 1928, is an algorithm used to evaluate the score progress. This algorithm requires us to explore all of the possible moves of a player while calculating the maximum or the minimum score the player can make from all of those possible moves.

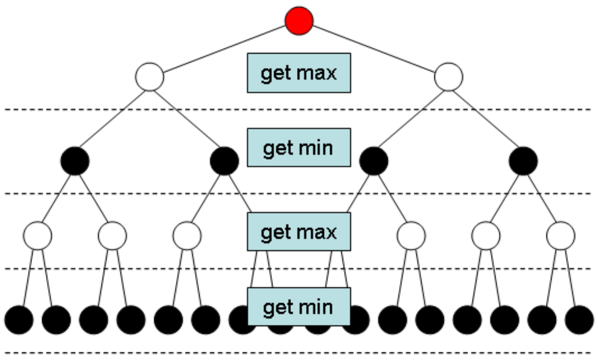


Diagram 10 Minimax Algorithm Sketch [3]

With this algorithm, we can discover which move that will yield maximum score in our turn and which move that will yield minimum score in the opposing's turn, improving the strategy to play in reversi.

Moreover, such algorithm maybe used to evaluate your positional strategy. By combining the minimax algorithm with the positional strategy, we will be able to evaluate our positions, as well as the opposing's positions.

However, such algorithm also has drawbacks as well. We cannot use this algorithm to play reversi that has a time limit because this algorithm require us to calculate all possible moves and evaluate them, which obviously takes too much time unless you are already highly-experienced about that.

That said, although unable to be used in most competitions this algorithm allows you evaluate about your strategy in playing reversi.

IV. Appendix

1)	Algorithm	: a process or set of rules to be followed in calculations or other problem - solving operations
2)	Terminologies	: the body of terms used with a particular technical application in a subject of theory
3)	Vertex	: each angular point of a polygon, polyhedron, or other figure (in this context, a graph).

V. Acknowledgment

All my deepest gratitude to DRa. Harlili, M.Sc and Dr. Ir. Rinaldi Munir, M.T. as the lecturers of IF 2120 – “Matematika Diskrit” for giving such assignment to enrich my knowledge about the application of Discrete Mathematics.

Also my special thanks for all of my family, my friends, and other people who have supported me so I can finish this paper.

References

[1] Munir, Rinaldi. *MATEMATIKA DISKRIT* Revisi Kelima. Bandung: Penerbit Informatika. 2012.

[2] http://www.samsoft.org.uk/reversi/strategy.htm, 21:15, 9 December 2015.

[3] http://mnemstudio.org/game-reversi-example-2.htm, 22:05, 9 December 2015

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, 8 Desember 2015

Dimas

Gaudensius Dimas Prasetyo Suprapto - 13514059