Battleship: Optimal Play

Prepared By:
EYTAN CHONG
ONN QI HUAN
TAN EASON
ADELINE HAU YUN YI

Teacher Mentor:
MDM KHOO GEOK HWA

DUNMAN HIGH SCHOOL

Table of Contents

1 Introduction **4**

2 Research Question **6**

3 Terminology **7**

4 Notation **8**

4.1 Grid Notation 8

4.2 Move Notation 8

4.3 Game Notation 9

5 Optimal Strategy **11**

6 Other Strategies **14**

6.1 Random 14

6.2 Hunt Target 14

7 Testing the Optimal Strategy **16**

7.1 Formatted Data 16

7.2 Analysis 22

8 Optimal Layout **23**

9 Testing the Optimal Layout **25**

9.1 Formatted Data 25

9.2 Analysis 28

10 Conclusion **30**

11 Bibliography **31**

Appendix A **32**

Random Strategy 32

Hunt Target Strategy 32

Optimal Strategy 33

Optimal Layout 35

1 Introduction

Battleship is a classic board game that involves two players attempting to destroy all the other player's ships. The game is played on four grids, with two grids for each player. The grids are typically square – usually 10 by 10. Both players arrange their ships and record the shots by their opponent on their first grid. On their second grid, they record their own shots. Players are not permitted to search a square that has already been searched. Turns alternate between the two players. The game ends once a player's ships are all sunk.

Before play begins, each player secretly arranges his ships on their primary grid. Each ship occupies several consecutive squares on the grid, arranged either horizontally or vertically. The number of squares for each ship is determined by the type of ship. The ships cannot overlap (only one ship can occupy any given square in the grid). The classes and numbers of ships allowed are the same for each player. These may vary depending on the version the players are playing. The version of Battleship we will be using in this report will be the 1990 Milton Bradley version, where every ship has a breadth of 1 square. The class and length for each ship are as follows:

No.	Ship Class	Length (Number of Squares)
1	Carrier	5
2	Battleship	4
3	Destroyer	3
4	Submarine	3
5	Patrol Boat	2

Figure 1.1

A picture of the grids used in the Battleship board game

The 4 grids used in a game of Battleship. The bottom grids are where the players place their ships and record their opponent's shots. The top grids are where the players record their own shots.

2 Research Question

Our research question is:

What is the optimal play in Battleship?

Due to the nature of Battleship, there is only one way for a player to attack their opponent: searching their grid for ships using a strategy. For the sake of simplicity, we will assume that the strategy is consistent throughout the game. We have defined a strategy to be optimal if it destroys all of the opponent's ships in the least amount of moves possible.

Furthermore, the only way a player can defend their ships is through their arrangement at the start of the game. Hence, the best defence will be the layout that takes the most amount of moves for the opponent to possibly win. We shall call this arrangement the "optimal layout".

Therefore, to answer our question, we must find out both the optimal strategy and the optimal layout in Battleship.

3 Terminology

The following are the definitions of some terms we will be using frequently throughout this report.

- Search
    - The act of searching a square and changing its properties (an unsearched square will become a missed, hit or sunken square).
- Unsearched square
    - A square that has not been searched.
- Missed square
    - A square that has been searched and does not have a ship on it.
- Hit square
    - A square that has been searched and has an operational ship on it.
- Sunken square
    - A square that has been searched and has a sunken ship on it.
- Operational ship
    - A ship that has at least one square that is unsearched.
- Sunken ship
    - A ship whose squares have all been hit.
- Arrangement
    - The position of a ship on a grid.
- Layout
    - The positioning of the ships at the start of the game.
- Optimal strategy
    - The optimal strategy is the strategy that will destroy the opponent's ships in the least amount of moves possible.
- Optimal layout
    - The optimal layout is the layout that takes the most amount of moves for the opponent to possibly win.

4 Notation

Before we proceed with our research, it is crucial that we have an easy and intuitive notation system to use to describe a grid, move and game. We have hence devised such a system to communicate about Battleship more clearly.

4.1 Grid Notation

A grid can be represented as an array of strings, where each string represents a square on the grid. The strings are either a number or a blank space. If the string is a number, it represents a square that has a ship whose ID is that number. If the string is a blank space, it represents a square with no ship on it. Below is an example of grid notation for a 10 by 10 grid.

```
[ ][ ][ ][ ][ ][3][3][3][3][ ]
[ ][ ][ ][ ][0][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][0][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][2][2][2]
[ ][1][1][1][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][4][4][4][4][4][ ][ ]
```

4.2 Move Notation

A move comprises the player that made the move and the square that the player has searched. Hence, the move notation is simply

```
<Player's name>: <Coordinates of square searched>
```

Where the bottom-leftmost square is (1,1) and the top-rightmost square is (n,n).

An example of a move in move notation would be

```
Player 1: (3,7)
```

4.3 Game Notation

Putting grid and move notation together, we get game notation. Its format is

```
[<Player 1's name>]
<Player 1's grid in grid notation>

[<Player 2's name>]
<Player 2's grid in grid notation>

<All moves made in move notation>
```

An example of a game in game notation would be:

```
[Player 1]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][4]
[ ][2][ ][3][3][3][3][ ][ ][4]
[ ][2][ ][ ][ ][0][ ][ ][ ][4]
[ ][2][ ][ ][ ][0][ ][ ][ ][4]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][4]
[ ][ ][ ][ ][1][1][1][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]

[Player 2]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][4][ ][3][ ][ ][ ][ ][ ][ ]
[ ][4][ ][3][ ][ ][ ][ ][ ][ ]
[ ][4][ ][3][ ][2][ ][ ][1][ ]
[0][4][ ][3][ ][2][ ][ ][1][ ]
[0][4][ ][ ][ ][2][ ][ ][1][ ]
```

```
Player 1: (6, 6)
Player 2: (6, 6)
Player 1: (5, 5)
Player 2: (5, 5)
Player 1: (7, 7)
Player 2: (5, 7)
Player 1: (4, 4)
Player 2: (7, 5)
```

5 Optimal Strategy

In this section, we will come up with a strategy and prove that it is the best strategy in Battleship. Recall that the "best" strategy is defined as the strategy that destroys all of the opponent's ships in the least amount of moves possible.

In the standard version of Battleship, we are only given information about the squares that we have searched. We are also given information about the operationality of their ships. Hence, using this information, we can devise a strategy to efficiently hunt down enemy ships.

Our first step is to figure out which squares must have a ship. This can be done by deriving certain statements that immediately tell us if a square has a ship from the rules of Battleship. Firstly, if a ship has only one possible arrangement, that must be its arrangement. This is fairly obvious to see, since if that is not the ship's arrangement, there is no other place on the grid the ship can be. Secondly, if a hit square has only one unsearched adjacent square and has no hit adjacent squares, then said unsearched square must have a ship. This is because the other adjacent squares cannot have a ship on them, thus leaving the only possibility of having a ship to the unsearched square.

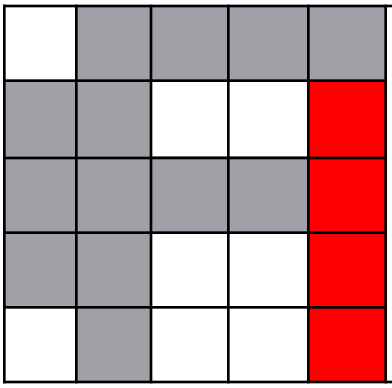

Figure 5.1
There is only one possible arrangement for the Battleship (4x1).

Legend:
Grey - Sunk
Red - Position of Battleship

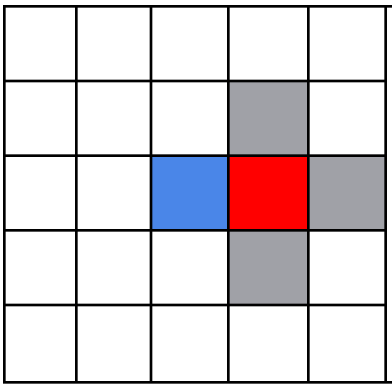

Figure 5.2
The blue square must contain an operational ship.

Legend:
Grey - Searched
Red - Hit
Blue - Our Shot

Our second step is to calculate the probability of each square having a ship and searching the square with the highest probability. This can be done using the following algorithm:

1. Iterate through all 30,093,975,536 unique layouts. This number of layouts is taken from research done by (Brown, n.d.).
2. Get the list of layouts that fit the current board position. This means that the board position can be superimposed upon the layout.
3. Get the square that appears the most in said list.
4. Search the square.

What this algorithm is essentially doing is maximising the probability of us hitting an enemy ship.

To increase the strategy's efficiency in step 2, parity could be implemented. This means that the number of times a square appears in the list described in step 2 of the optimal algorithm could be increased by a few percentage points. This is known as "boosting" a square. However, only squares that are of a certain parity would be eligible for such a boost. This parity boosting increases the chances of finding the smallest operational ship with the smallest length, hence increasing the efficiency of the strategy. This increases the probability of each shot hitting the smallest ship, as instead of all unsearched squares, only a handful are actually considered. In general, when the smallest operational ship is of length n, the distance between a boosted square and another boosted square would be $n - 1$ squares in all four directions.

For example, for n=2, the squares boosted would be like squares on a checkers board.

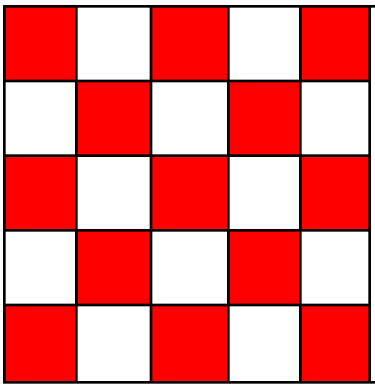

Figure 5.3
A representation of the squares boosted when the smallest ship has length 2.

Legend:
Red - A boosted square

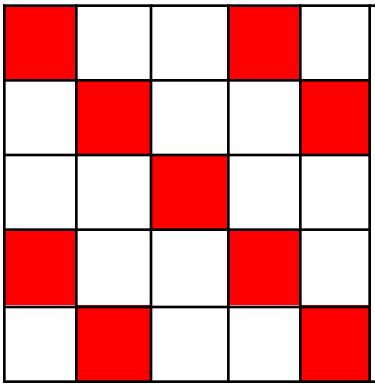

Figure 5.4
A representation of the squares boosted when the smallest ship has length 3.

Legend:
Red - A boosted square

By using this two-step approach to find squares that have the highest probability of having a ship on them, we have devised the optimal strategy for Battleship.

However, the algorithm would be infeasible in practice, as 31 billion initial layouts would most likely take a considerable amount of time to process. To mitigate this, a sub-optimal implementation of the algorithm would be as follows:

1. Iterate through all possible ship arrangements for all operational ships on the board.
2. Get the square that appears the most in step 1.
3. Search the square.

This reduces the problem into something much smaller while still being on par with the optimal algorithm.

6 Other Strategies

Apart from the optimal strategy, we will also be taking a closer look at two strategies, namely the Random and Hunt Target strategies. These two strategies were chosen as they are easy to understand and are frequently used by players.

6.1 Random

This strategy, as per its name, searches squares at random. No attempt will be made to deduce the whereabouts of the ship.

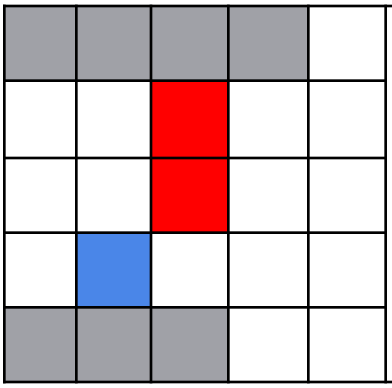

Figure 6.1
An example of the Random strategy

Legend:
Grey - Sunk
Red - Hit
Blue - Our shot

In this example with a 1x4 ship and two 1x3 ships on a 5x5 grid (where one 1x3 ship has not been sunk), the random strategy will fire a shot at a random unsearched square. Rolling a random number generator, we get (2, 2) as our shot.

6.2 Hunt Target

This strategy has a hunt mode and a target mode. The hunt mode is activated when there are no hit squares on the grid. Conversely, the target mode is activated when there is at least one hit square on the grid. During hunt mode, the algorithm behaves exactly like the Random strategy; it will randomly search an unsearched square. During target mode, a random square adjacent to a hit square will be searched.

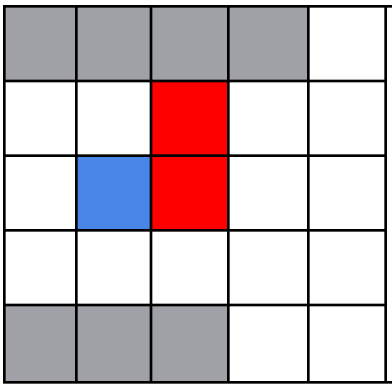

Figure 6.2
An example of the Hunt Target strategy

Legend:
Grey - Sunk
Red - Hit
Blue - Our shot

Using the same set-up as in Figure 6.1, we see that the domain of the squares searchable has shrunk, as there are hit squares. For example, (2,2), our guess for the Random strategy, is no longer an option as it is not adjacent to a hit square. Once again, rolling the random number generates yields (2,3) as our shot.

7 Testing the Optimal Strategy

7.1 Formatted Data

A comparison of how the optimal strategy fare against the Hunt Target and Random strategies was done, by simulating 10,000 games for each match for side lengths 6 - 15. This is to ensure that the algorithm works even when the board system is changed. In turn, this will give a generalisation. See Appendix A for the code used to implement these strategies. This section contains the formatted data generated by the simulation. The raw data can be found at https://github.com/asdia0/Battleship.Data.

Note that the win rate will be given as a value between 0 and 1. Also note that the number of moves have been scaled such that it refers to the number of moves divided by the maximum number of moves. For example, on a board of side length 6, if the median number of moves is 60, the scaled median number of moves is going to be $\frac{60}{2 \cdot 6^2}$, which would be $0.8\bar{3}$. All values are rounded to 3 decimal places.

Optimal Strategy Against the Other Strategies For Side Length 6

Side length	6
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.696
Number of moves (median, scaled)	0.708
Number of moves (mode, scaled)	0.708
Optimal-Hunt Target	
Win rate (optimal)	0.807
Number of moves (mean, scaled)	0.683
Number of moves (median, scaled)	0.681

Number of moves (mode, scaled)	0.681

Optimal Strategy Against the Other Strategies For Side Length 7

Side length	7
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.605
Number of moves (median, scaled)	0.602
Number of moves (mode, scaled)	0.602
Optimal-Hunt Target	
Win rate (optimal)	0.816
Number of moves (mean, scaled)	0.593
Number of moves (median, scaled)	0.592
Number of moves (mode, scaled)	0.602

Optimal Strategy Against the Other Strategies For Side Length 8

Side length	8
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.544
Number of moves (median, scaled)	0.539
Number of moves (mode, scaled)	0.539
Optimal-Hunt Target	
Win rate (optimal)	0.816

Number of moves (mean, scaled)	0.531
Number of moves (median, scaled)	0.523
Number of moves (mode, scaled)	0.539

Optimal Strategy Against the Other Strategies For Side Length 9

Side length	9
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.500
Number of moves (median, scaled)	0.5
Number of moves (mode, scaled)	0.5
Optimal-Hunt Target	
Win rate (optimal)	0.823
Number of moves (mean, scaled)	0.484
Number of moves (median, scaled)	0.475
Number of moves (mode, scaled)	0.451

Optimal Strategy Against the Other Strategies For Side Length 10

Side length	10
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.466
Number of moves (median, scaled)	0.465
Number of moves (mode, scaled)	0.455

Optimal-Hunt Target	
Win rate (optimal)	0.826
Number of moves (mean, scaled)	0.452
Number of moves (median, scaled)	0.445
Number of moves (mode, scaled)	0.425

Optimal Strategy Against the Other Strategies For Side Length 11

Side length	11
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.442
Number of moves (median, scaled)	0.434
Number of moves (mode, scaled)	0.393
Optimal-Hunt Target	
Win rate (optimal)	0.814
Number of moves (mean, scaled)	0.427
Number of moves (median, scaled)	0.417
Number of moves (mode, scaled)	0.376

Optimal Strategy Against the Other Strategies For Side Length 12

Side length	12
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.423

Number of moves (median, scaled)	0.413
Number of moves (mode, scaled)	0.406
Optimal-Hunt Target	
Win rate (optimal)	0.818
Number of moves (mean, scaled)	0.408
Number of moves (median, scaled)	0.403
Number of moves (mode, scaled)	0.378

Optimal Strategy Against the Other Strategies For Side Length 13

Side length	13
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.409
Number of moves (median, scaled)	0.399
Number of moves (mode, scaled)	0.363
Optimal-Hunt Target	
Win rate (optimal)	0.810
Number of moves (mean, scaled)	0.392
Number of moves (median, scaled)	0.388
Number of moves (mode, scaled)	0.393

Optimal Strategy Against the Other Strategies For Side Length 14

Side length	14
Optimal-Random	

Win rate (optimal)	1
Number of moves (mean, scaled)	0.398
Number of moves (median, scaled)	0.390
Number of moves (mode, scaled)	0.375
Optimal-Hunt Target	
Win rate (optimal)	0.816
Number of moves (mean, scaled)	0.381
Number of moves (median, scaled)	0.375
Number of moves (mode, scaled)	0.355

Optimal Strategy Against the Other Strategies For Side Length 15

Side length	15
Optimal-Random	
Win rate (optimal)	1
Number of moves (mean, scaled)	0.388
Number of moves (median, scaled)	0.38
Number of moves (mode, scaled)	0.349
Optimal-Hunt Target	
Win rate (optimal)	0.809
Number of moves (mean, scaled)	0.371
Number of moves (median, scaled)	0.367
Number of moves (mode, scaled)	0.358

7.2 Analysis

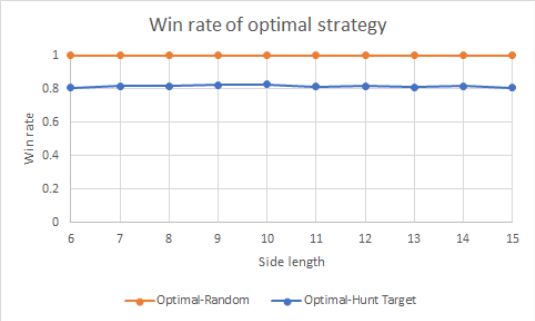

Figure 7.1
Win rate of optimal strategy against other strategies

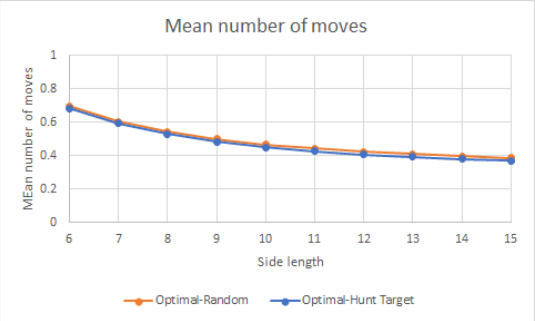

Figure 7.2
Mean number of moves for optimal strategy against other strategies

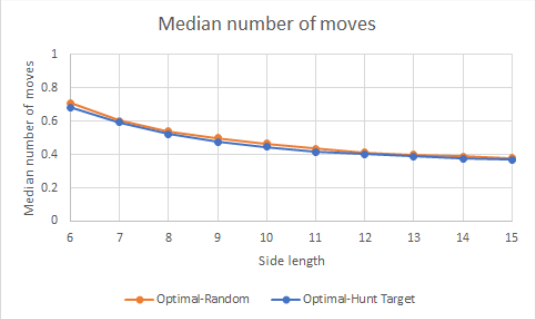

Figure 7.3
Median number of moves for optimal strategy against other strategies

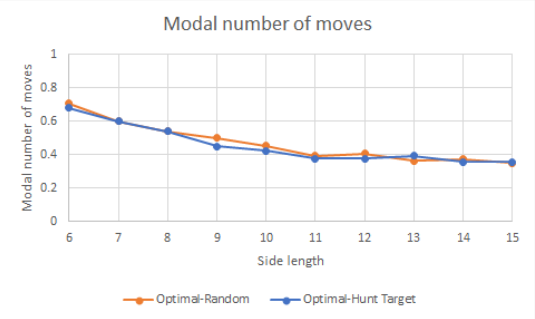

Figure 7.4
Modal number of moves for optimal strategy against other strategies

As seen in the graphs, the optimal strategy wins most of the time against the Hunt Target strategy, and completely crushes the Random strategy. Furthermore, the mean, median and modal number of moves for each match decreases as the side length increases. This means that as the side length increases, the optimal strategy is progressively getting better at finding and sinking enemy ships.

8 Optimal Layout

As the optimal strategy is the best strategy in Battleship, it is hence the best idea to develop a defence against it. If our opponent is using the optimal strategy, this layout will maximise the amount of moves we are safe from them. This is solely due to the fact that the optimal strategy is deterministic, that is, given the same inputs, in this case a grid, the strategy will always find the same square to search. This is because the strategy relies purely on logic and does not involve any randomness whatsoever. However, if our opponent is not using the optimal strategy, it is safe to say that the strategy we are using, in this case the optimal strategy, will most likely demolish their navy first.

Hence, our method to create the optimal layout is as follows:

1. Start off with an empty grid.
2. Get the best square to search according to the optimal strategy.
3. Check if the probability of the square found in step 2 as having a ship.
    a. If it is 100%, mark the square as a hit (or a sink, if necessary). This is due to the fact that if that particular square does not have an operational ship on it, the ship layout will be impossible.
    b. Else, mark the square as a miss. This ensures that this ship layout requires the most number of moves from the optimal strategy for all the ships to be sunk.

However, the optimal strategy is deterministic, which means that it is not random. If the opponent is also using the optimal layout, the two players will be doing the same moves each turn, causing the first player to always be one ply ahead of the second player, hence they will win first.

Using our algorithm as described above, we can now find the optimal layout for any board size. For example, when the board is 10 squares long, the optimal layout would be

[3][ ][0][ ][ ][ ][ ][ ][ ][ ]
[3][ ][0][ ][ ][ ][ ][ ][ ][ ]
[3][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[3][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[4][ ][ ][ ][ ][ ][ ][ ][ ][ ]
[4][ ][ ][ ][ ][ ][ ][ ][ ][2]
[4][ ][ ][ ][ ][ ][ ][ ][ ][2]
[4][ ][ ][ ][ ][ ][ ][ ][ ][2]
[4][ ][ ][ ][ ][ ][1][1][1][ ]

9 Testing the Optimal Layout

9.1 Formatted Data

To test the optimal layout, we simulated 10,000 games for side lengths 6 - 15, with one player using the optimal layout and the other using a random layout. This allows us to see if the optimal layout works for different board sizes. This section contains the formatted data for the simulation. Once again, the raw data can be found at https://github.com/asdia0/Battleship.Data. Appendix A contains the code used to implement these layouts. The values for the win rate are explained in Testing the Optimal Strategy.

Optimal Layout Against Random Layout For Side Length 6

Side length	6
Win rate (optimal)	0.991
Number of moves (mean, scaled)	0.695
Number of moves (median, scaled)	0.708
Number of moves (mode, scaled)	0.708

Optimal Layout Against Random Layout For Side Length 7

Side length	7
Win rate (optimal)	0.997
Number of moves (mean, scaled)	0.604
Number of moves (median, scaled)	0.602
Number of moves (mode, scaled)	0.602

Optimal Layout Against Random Layout For Side Length 8

Side length	8

Win rate (optimal)	0.992
Number of moves (mean, scaled)	0.542
Number of moves (median, scaled)	0.539
Number of moves (mode, scaled)	0.523

Optimal Layout Against Random Layout For Side Length 9

Side length	9
Win rate (optimal)	0.965
Number of moves (mean, scaled)	0.498
Number of moves (median, scaled)	0.5
Number of moves (mode, scaled)	0.475

Optimal Layout Against Random Layout For Side Length 10

Side length	10
Win rate (optimal)	0.99
Number of moves (mean, scaled)	0.467
Number of moves (median, scaled)	0.465
Number of moves (mode, scaled)	0.445

Optimal Layout Against Random Layout For Side Length 11

Side length	11
Win rate (optimal)	0.979
Number of moves (mean, scaled)	0.443
Number of moves (median, scaled)	0.442

Number of moves (mode, scaled)	0.409

Optimal Layout Against Random Layout For Side Length 12

Side length	12
Win rate (optimal)	0.999
Number of moves (mean, scaled)	0.424
Number of moves (median, scaled)	0.420
Number of moves (mode, scaled)	0.392

Optimal Layout Against Random Layout For Side Length 13

Side length	13
Win rate (optimal)	0.983
Number of moves (mean, scaled)	0.408
Number of moves (median, scaled)	0.399
Number of moves (mode, scaled)	0.411

Optimal Layout Against Random Layout For Side Length 14

Side length	14
Win rate (optimal)	0.815
Number of moves (mean, scaled)	0.388
Number of moves (median, scaled)	0.390
Number of moves (mode, scaled)	0.490

Optimal Layout Against Random Layout For Side Length 15

Side length	15
Win rate (optimal)	0.995
Number of moves (mean, scaled)	0.388
Number of moves (median, scaled)	0.38
Number of moves (mode, scaled)	0.38

9.2 Analysis

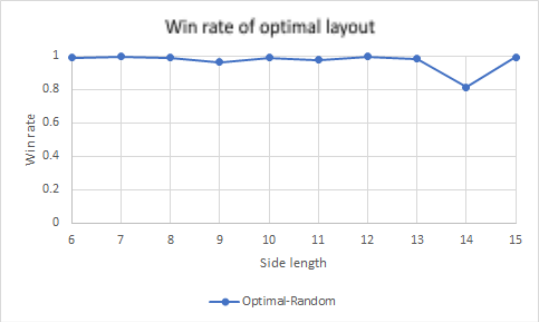

Figure 9.1
Win rate of optimal layout against random layout

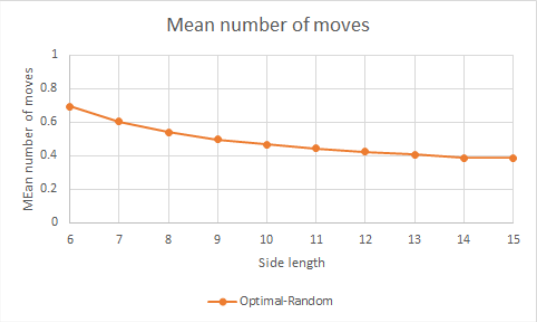

Figure 9.2
Mean number of moves to sink all ships in optimal layout against random layout

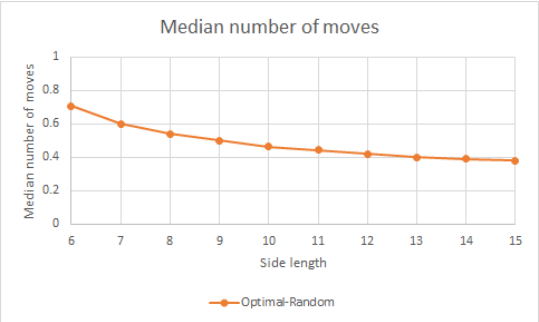

Figure 9.3
Median number of moves to sink all ships in optimal layout against random layout

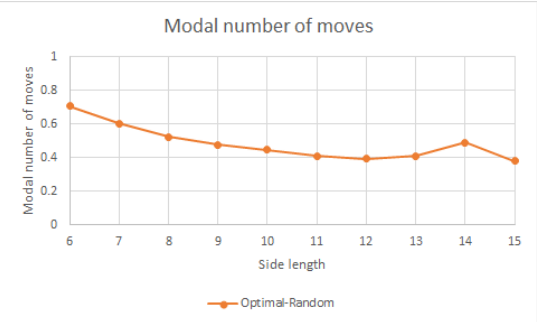

Figure 9.4
Modal number of moves to sink all ships in optimal layout against random layout

According to our simulations, the player with the optimal layout is more likely to win against players with random layout. Furthermore, as seen and explained in Testing the Optimal Strategy, the mean, median and modal number of moves decreases as the side length increases. While there are anomalous results when the side length is 14, it is not of much concern as a win rate of 80% is far higher than 50%, which is approximately the win rate of the first player when using a random layout against a random layout.

10 Conclusion

The strategy described in Optimal Strategy is proven to be the best strategy in Battleship. The Optimal Layout is also argued to guarantee players the best chances of survival from enemy fire. Two other well-known strategies were introduced and compared against the optimal strategy. Apart from that, we have also pitted the optimal strategy against itself, with one player using a random layout while the other using the optimal layout. In the first simulation, we saw that the optimal strategy vastly outperformed the other two strategies, while in the second simulation, a huge majority of games were won by the player employing the optimal layout.

We strongly believe further research can be made in this topic. Firstly, a rectangular grid of dimensions m and n could be considered. Secondly, more strategies can be tested and analysed. Thirdly, ships of various shapes and sizes could also be explored (e.g. having tetromino-shaped ships instead of rectangular ships). Fourthly, more variants of Battleship could be investigated. For example, an interesting variant would be one where players need not announce that their ships have sunk. Overall, Battleship is a relatively unexplored topic, and we hope that more research could be put into it.

11 Bibliography

- Brown, C. L. (n.d.). *C. Liam Brown — Battleship Probability Calculator — Methodology.* C. Liam Brown. Retrieved April 29, 2021, from https://cliambrown.com/battleship/methodology.php
- Berry, N. (2011, December 3). *Battleship.* DataGenetics. Retrieved April 29, 2021, from https://www.datagenetics.com/blog/december32011/
- P. (2016, March 12). *Analysis of (hexagonal) Battleship.* Possibly Wrong. Retrieved April 29, 2021, from https://possiblywrong.wordpress.com/2016/03/12/analysis-of-hexagonal-battleship/
- Wikipedia contributors. (2022). *Battleship (game).* Wikipedia. Retrieved April 29, 2021, from https://en.wikipedia.org/wiki/Battleship_(game)