Black Box
The Ultimate Game of Hide and Seek

Joey Shepard
Archie Monji
Helio Tejeda

August 17, 2013

Abstract

Black Box belongs to a class of mathematical and logical puzzle games, in which the objective is to strategically acquire information in order to deduce the game boards hidden solution. Much like the games Battleship and Mastermind, the tactics and strategies used to acquire information affects how accurately and quickly the solver is able to find the solution to Black Box. We have developed several artificial intelligence systems that undertake two main approaches in order to determine which strategies and properties of the board yield more accurate and optimal solutions. The first approach is a brute force method of gradually narrowing the range of candidate solutions until only the correct solution remains. The other is called the threshold approach; it involves scoring squares of the board as moves are played in order to indicate the likeliness that a hidden atom exists there. Our experimentation has revealed that there are several optimal first moves and that indistinguishable boards make some games unsolvable.

1 Black Box Game Information

1.1 Background

Black Box, also known as "The Ultimate Game of Hide and Seek", was invented in 1976 by Eric W. Solomon. Solomon, who has a Ph.D. in Mathematics from the University of Southampton, is a British game designer, software developer, and software programmer [2]. The game was first introduced to the public as a two-player board game. However, it quickly grew to become the one-player computer game we now play today. [2] Black Box was inspired from Godfrey N. Hounsfield's groundbreaking invention of the CAT scanner. This revolutionary invention which began in the late 1960's and was installed in 1971 would provide physicians valuable diagnostic information without the need for potentially dangerous exploratory surgery [3]. Hounsfield would later be awarded the 1979 Nobel Prize in Medicine [1].

1.2 How to Play

Black Box is a puzzle game in which atoms are hidden within the game board. The objective is to locate the position of all of the atoms with the lowest possible score by shooting rays, which interact with the atoms to give the user feedback on their positions, into the board. Shooting a ray into the game board is considered a game move. As mentioned, each ray shot into the game board will give the user feedback as to its interaction with the hidden atoms. Rays travel in straight lines and can either hit an atom directly in its path, or be detoured at a 90 degree angle by an atom in an adjacent row or column. These detours cause the atom to change direction then continue in a straight line (i.e. if the ray is going North and there is an atom to the left of it, the atom will be deflected 90 degrees to the right and will continue its path to the East). Points are calculated based on the feedback given to the user, and accumulated throughout gameplay until the user has submitted their atom guesses. Penalty points, worth five points, are given for every incorrect guess the user submits. More information regarding the different types of feedbacks will be described in the following sections. The following images represent the stages of game play, from the initial blank board, to the board filled with feedback from the rays shot, to a game board with guessed atoms based on the feedback received.

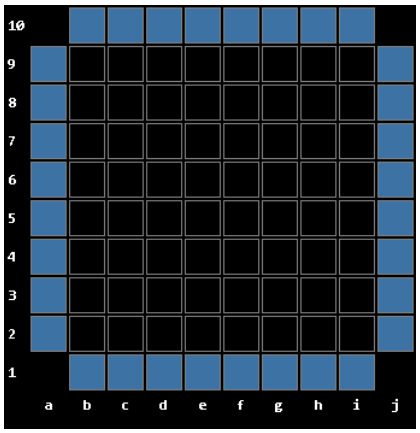

Figure 1: Initial Game Board

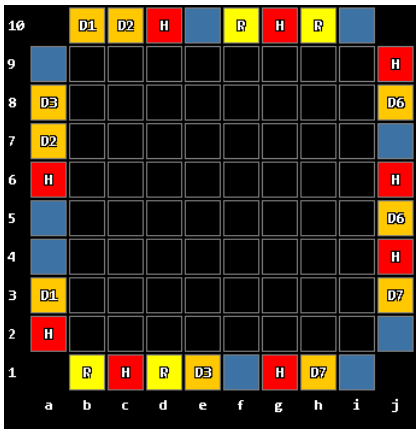

Figure 2: Board With Feedback

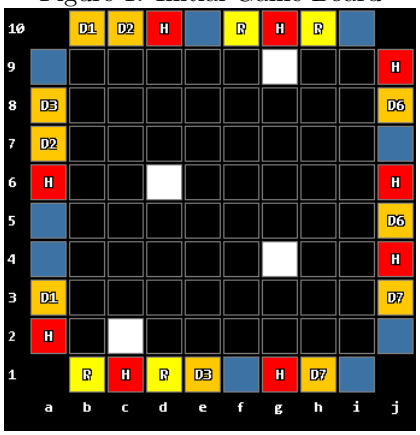

Figure 3: Guesses Based on Feedback

1.2.1 Scoring System

The objective of the game is to guess the location of the atoms within the game board with the lowest score possible. Each ray shot into the board results in feedback given to the user that will help locate the atoms positions, however, the feedback given also results in points being added to the players total score. A point value is given based on the feedback returned to the player. The point values are as follows:

Feedback	Point Value
Hit	1
Reflection	1
Detour	2

1.2.2 Hit

A move results in a hit when the ray shot into the board hits an atom directly in its path, meaning the ray never exited the game board. The following image depicts how a move can result in a hit. Remember however, that during game play, the only information received is the red highlighted square, the yellow arrows and the white atoms used are only used for visual effect.

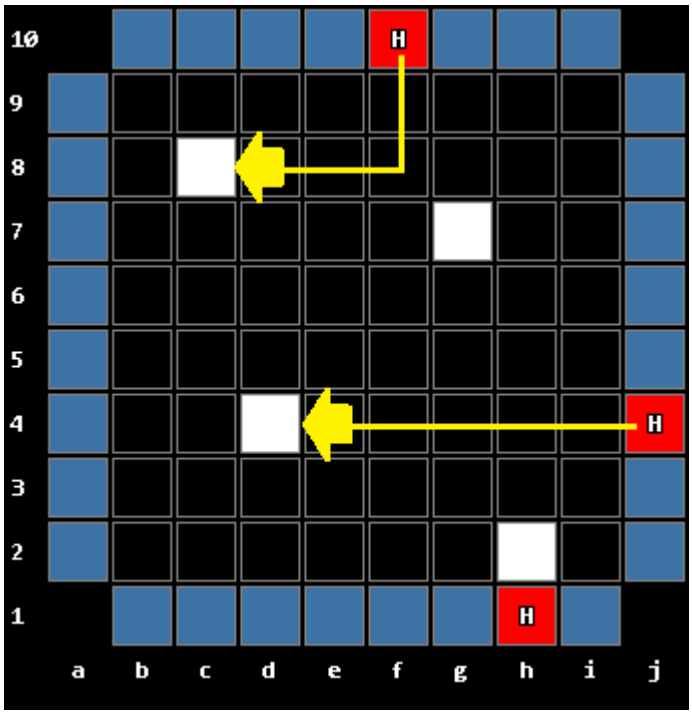

Figure 4: HITS

Demonstrated in Figure 4 a ray is being shot into the game board from port (f 10), traveling South and being deflected by an atom to its right, causing the ray to turn 90 degrees to the West. The ray then continues in a straight line West until finally hitting an atom at (c 8) resulting in feedback to the user as a Hit. The second ray shot from port (j 4) travels Westward in a straight path until it directly hits an atom at (d 4) also resulting in a hit. Finally, we see that the ray shot from port (h 1) also results in a hit, as there is an atom located directly in front of it, on the edge of the game board.

1.2.3 Reflection

A move results in a reflection when the ray that is shot into the game board exits through that same port (i.e. Ray shot from (a 7) exits through (a 7)).

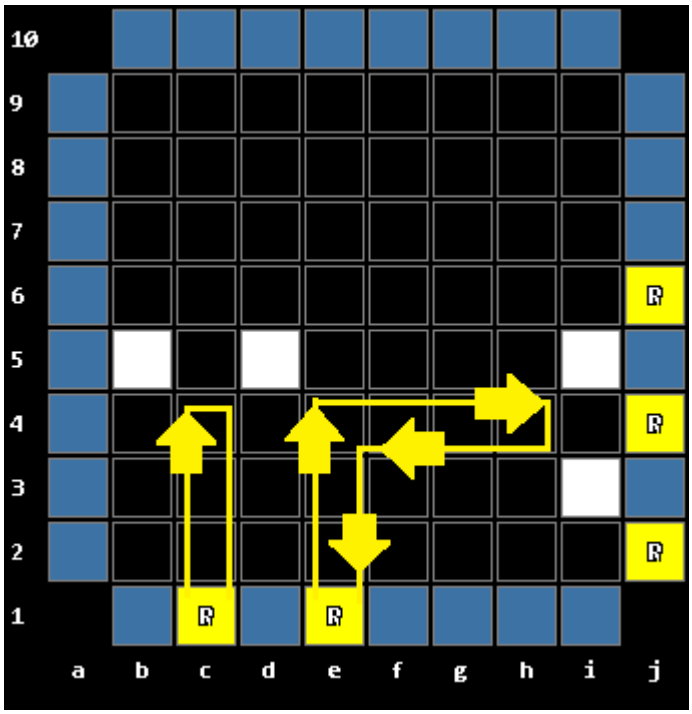

Figure 5: REFLECTIONS

As you can see in Figure 5 rays shot from ports (c 1) and (e 1) both enter the game board through their respective ports and are deflected by atoms in such a manner that they exit through those same ports. On the other hand, rays shot from ports (j 2), (j 4), and (j 6) are a special case for reflections. These are also considered reflections because the locations of the atoms are on the edge of the game board, with the rays being shot next to them.

1.2.4 Detour

The final type of feedback a user can receive is called a detour. A detour occurs when a ray shot into the game board is deflected in such a way by the atoms, causing the ray to exit through a different port from which it entered.

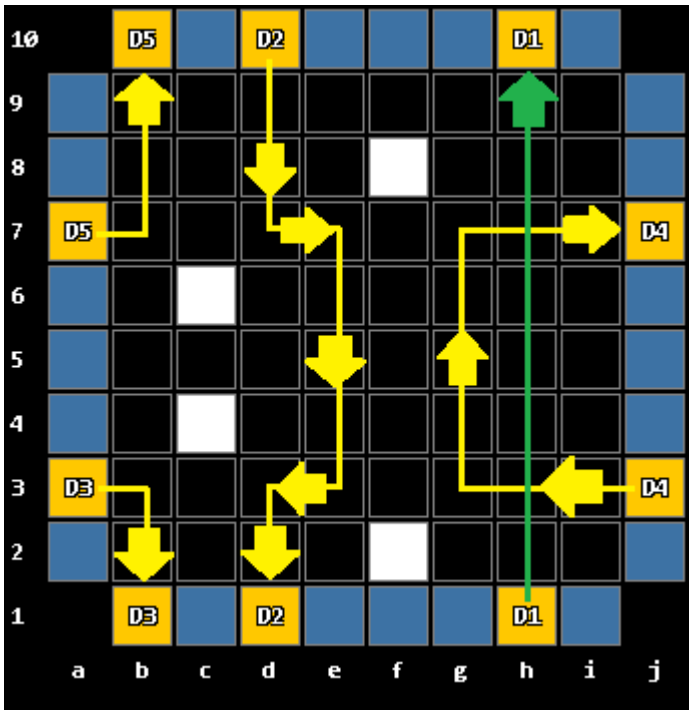

Figure 6: DETOURS

Figure 6 represents the different types of detours that can occur during game play. Remember, detours occur when a ray is shot through one port and exits through another. Therefore there are different types of detours that can occur. The first can be seen by detours (D5) and (D3) shot from ports (a 7) and (a 3) respectively and exiting through ports (b 10) and (b 1) respectively. This type of detour is known as a 90 degree detour. The following detour is called a miss, which occurs when a ray is shot through a port and travels directly across the game board exiting directly on the opposite side. This can be seen through detours (D1) and (D2) both of which travel directly across the game board, yet they both take different paths. The final type of detour is called the U-turn detour in which an atom enters through a port on one face of the game board, and exits through that same face but at a different port. The U-turn detour can be seen by detour (D4).

2 Combinatorics

2.1 Total Number of Board Configurations

The total number of board configurations is the number of combinations in which r atoms can be arranged on a board with n squares.

$$\binom{n}{r} \tag{1}$$

A typical black box game takes place on a board with 8 rows and 8 columns with 3 - 5 atoms. The number of board configurations for each case are:

Number of Atoms	Board Configurations
3	41,664
4	635,376
5	7,624,512

2.2 Minimum and Maximum Number of Guesses Needed to Fill All Ports

Considering, like above, a typical black box game takes place on a board with 8 rows and 8 columns with 3 to 5 atoms. The minimum and maximum number of fired rays needed to fill all ports are:

Number Of Atoms	Minimum	Maximum
3	20	24
4	20	25
5	24	27

3 Indistinguishable Boards

Every Black Box board has an associated set of feedback which is obtained by playing every move on the board. For most configurations of Black Box, boards and feedback sets are not one-to-one. Some boards, due to the nature of the game's rules, can provide the same exact same feedback as other boards although their configuration of atoms are different.

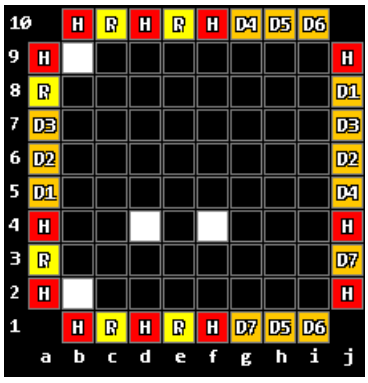

Figure 7: Indistinguishable Board 1

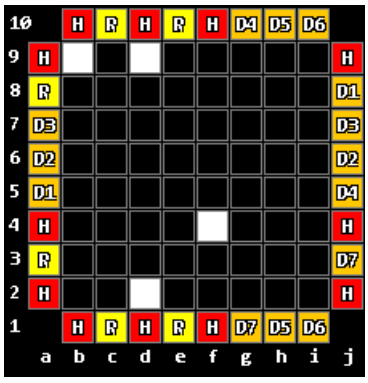

Figure 8: Indistinguishable Board 2

The boards shown in Figures 7 and 8 are indistinguishable from one another because, while they have different atom configurations, they provide identical feedback. Therefore, even if every possible move is tried and consequently all feedback is revealed, it is impossible for the player to determine which board configuration matches the hidden board. To address this quality of indistinguishable boards in our analysis, we consider solution boards that have identical feedback to that of the hidden board to be correct. The solution is valid on the basis that, while it is impossible to determine if the board is true solution, the solution's feedback matches that of the hidden board.

3.1 Percentage of Boards with Unique Feedback Sets

Boards that have a unique set of feedback share their feedback set with no other board. These boards are therefore distinguishable since their internal configurations can be determined from their feedback set. Figure 9 shows the percentage of boards which have a unique feedback set for several Black Box configurations:

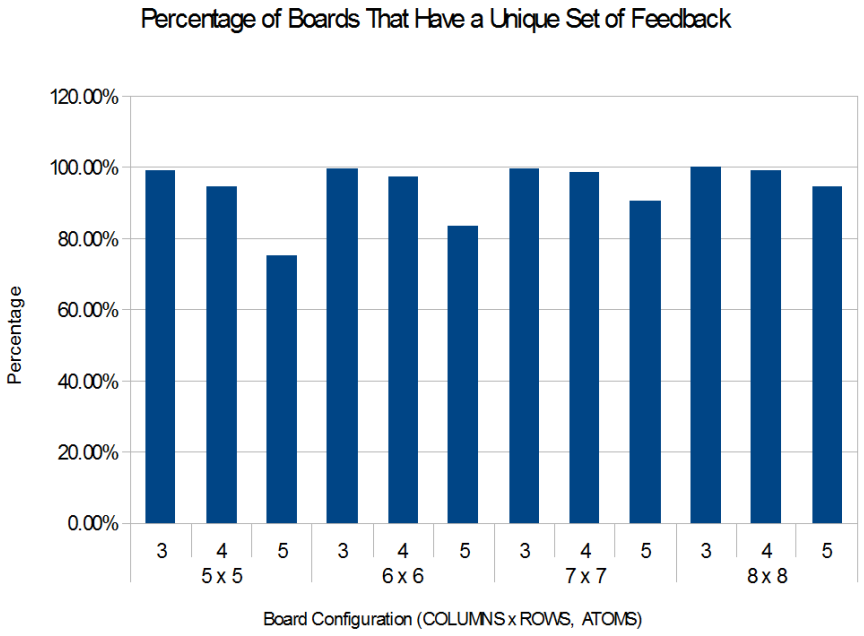

Figure 9: Percentage of Boards with Unique Feedback Sets

For most configurations, the majority of all possible boards have a unique feedback set. As the number of atoms placed on the board increases, the percentage of boards with unique feedback sets decreases. This makes sense since

certain atom configurations can guard squares from rays cast on the board. When this occurs, the position of an atom can no longer be determined.

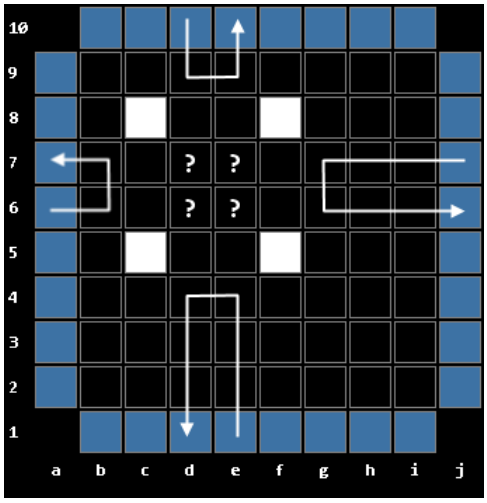

Figure 10: Atoms can hide other atoms

In Figure 10, four atoms are positioned such that it is impossible to determine the position of any atom within the area indicated. If another atom were to be placed on the board, the 5th atom can be in any one of the 4 guarded squares without affecting what feedback is returned. In this case, at least 4 boards out of the many possible board configurations will be indistinguishable from one another. Such situations as well as mere coincidence lend to the existence of indistinguishable boards.

4 Brute Force Approach

4.1 Iterative Filtering

In analyzing Black Box, one approach that we tried was a brute force method of iteratively reducing a list of candidates solutions until the solution was found. The algorithm is provided a board to solve and a list of candidate solutions. These candidate solutions can simply be a generated list of all possible board configurations to ensure the solution is inclusive in the list. A move is played on the board to solve and the feedback received is recorded. The same move is then tried on all candidate boards. If a board returns different feedback, that board is discarded since it cannot be a solution. After all candidates are processed, another move is then played. This process continues, iteratively reducing the set of candidates, until one board remains, i.e., the solution. After a filtering cycle, if all remaining candidates are indistinguishable from each other, then the program has reached a situation where making further moves will not eliminate

any more candidates. At this point, any candidate is considered to be correct since there is no way to determine the precise correct board. If the entire range of possible board configurations are input as candidate boards, this approach is guaranteed to find a satisfactory solution.

4.2 Move Selection Using Iterative Filtering

Using this same approach of filtering out invalid boards, we are able to find the move which eliminates the most boards in hopes of obtaining a solution in fewer moves. Since it is difficult to predict the feedback that will be returned by a move, we filter out candidates as though some feedback were to have occurred and remember the number of remaining candidates. We do this for all feedback cases and remember the greatest value, representing the worst case for that move. We then select the move which has the minimum of these values. This provides us the move which eliminates the most boards given the worst case scenario. Applying this method, we are able to calculate the first move for the game that will eliminate the most boards given the worst case scenario. Figure 11 shows the best first move for various boards of size $n \times n$ with 3 to 5 atoms allocated on the board. In general, for these boards, the 2nd port from all the edges will always remove the most candidates in the worst case. An exception to this trend is shown in Figure 12. In the case where 4 atoms are allocated on an 8x8 board, the preferred moves are the edge ports.

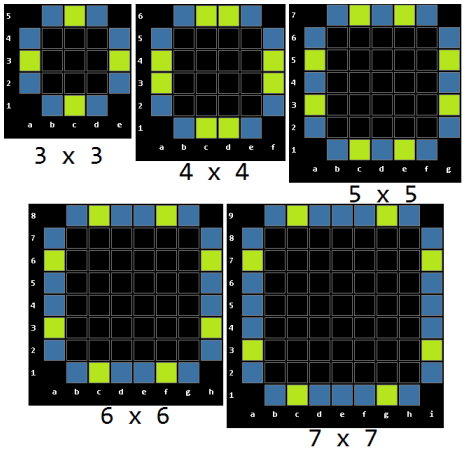

Figure 11: Best First Moves for 3x3 - 7x7 (denoted by lighter colored ports)

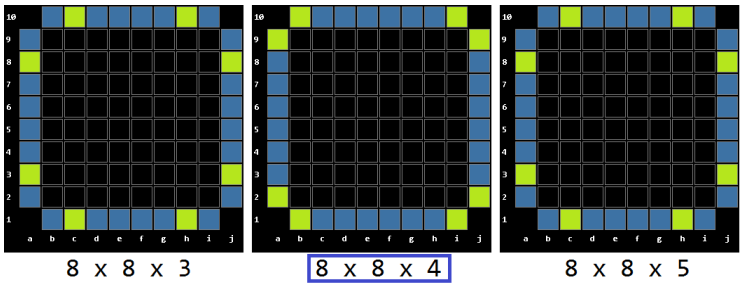

Figure 12: Exception in the 8x8 case

This method of selecting moves is relatively expensive to compute. For all available moves, all possible feedback, which includes all of the various possible detours, must be tested. This means that the set of candidates must be filtered for all of these combinations of moves and feedback. An upper bound for the number of boards which to be filtered can be expressed as $O(m^2 * n)$ where m represents the number of available moves and n represents the number of candidate boards. If there are m available moves, then the number of possible feedback possibilities is $O(m)$ since we count $O(m)$ detour combinations (some of these combinations represent impossible situations, hence $O(m)$) as well as a hit and a reflection. $O(m^2)$ thus represents the number of all feedback cases to be tried. Ultimately, this method scales poorly with regard to board size and the number of candidate states to be considered.

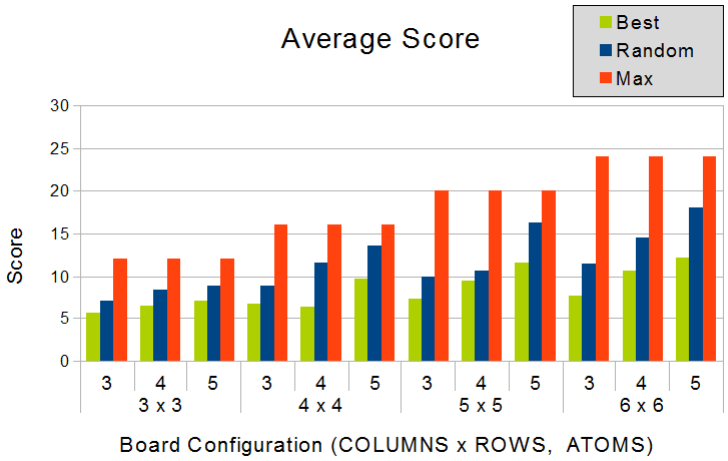

Figure 13: Average Score

Figure 13 shows the average scores for various Black Box setups using the aforementioned move selection method. Trials using random move selection as well as the maximum possible score are also included for comparison. The move

selection strategy, as expected, provides better results than randomly selecting the move to make. With the move selection strategy, the iterative filtering solver is able to narrow down the number of candidates to one board and provide a solution with less moves. It is interesting that, even with random move selection, on average, the solver was able to converge upon a solution without needing to make every move on the board. This suggests that a good strategy for picking moves should produce better scores. In addition, for all board dimensions, as the number of atoms increases, and consequently the number of possible boards configurations, we observe that the average score needed to solve a board increases.

5 Threshold Approach

Another approach we took to solving Black Box was based on setting an arbitrary threshold that must be passed by any given square to be considered in the final guess. Each square on the grid was initially given the value 0. Upon receiving feedback from each casted ray, we augmented the values in the squares we deemed affected by each ray. When any squares value surpasses or recedes below the threshold, the square is added to or removed from the list of likely atom locations, respectively. Once the number of squares that have surpassed the threshold equals the number of hidden atoms, a guess is made consisting of those very squares.

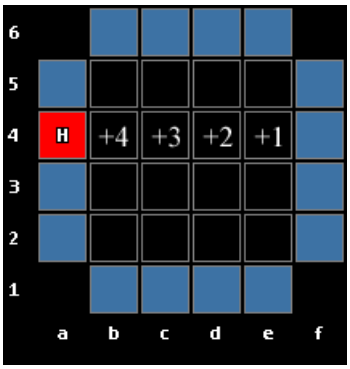

Figure 14: Hit Evaluations

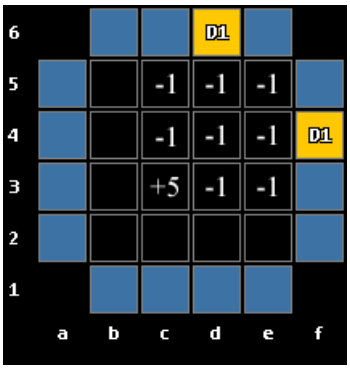

Figure 15: Detour Evaluations

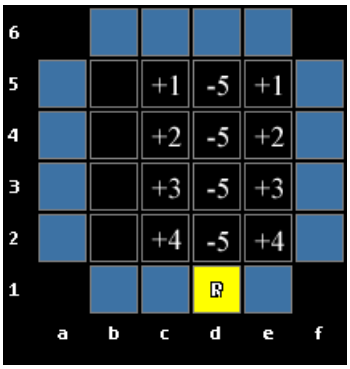

Figure 16: Reflection Evaluations

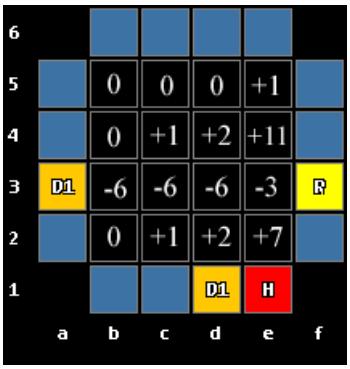

Figure 17: All Evaluations

In the rare case that all ports are filled and the number of squares whose values have passed the threshold is less than the number of hidden atoms, the next highest evaluation is repeatedly added as a location in the final guess until that condition is satisfied.

6 Restrictions

This section explores some of the ideas we implemented in the hope of optimizing the solvers average score. The first concept is that of board trimming, in which the computer tries to reduce the size of the board by quarantining the atoms in the least-area bounding box, which then becomes the new starting board. This comes a cost however, as each guess made to try and trim the board adds to the score. Secondly, we addressed the question of whether or not the equality of revealed ports on each of the four sides significantly influenced the average score. Rather than guess randomly, we placed a constraint on the solver to guess any port on the side with the least amount of revealed ports.

6.1 Board Trimming

In order to successfully reduce the board size, we try to cast a ray along each outermost border. If the ray is travelling up or down and ends in the same column it began, or is travelling left to right and ends in the same row it began, we are confident that there are no atoms in that column/row or the adjacent one, thus reducing the size of the respective dimension by two. In this case the maximum score one can receive through ports only has been reduced by 4, at the cost of the ray, +2. The net change is -2 to the maximum score which proportionally translates to an average score decrease of -1, when compared to the original board.

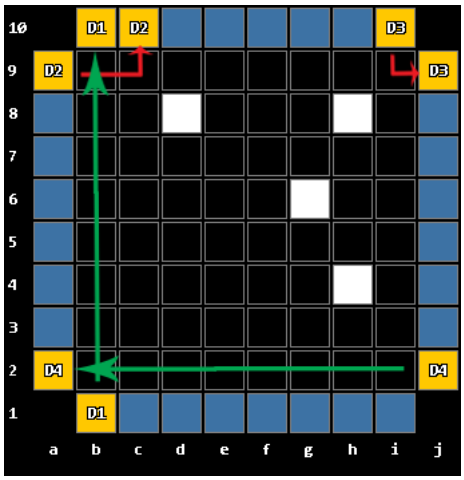

Figure 18: An illustration of the trimming algorithm

If any of the four sides is successfully trimmed, the process is repeated along those sides again until the ray cast along the border does not pass straight through the board, ensuring the maximum decrease in board size. Unfortunately, if this strategy fails and little or no sides are trimmed (in some cases the trimmed board stays the same dimension as the original board), the solver has set itself back quite a bit by guessing the uninformative corner ports and increasing the score drastically, considering how little information was obtained. The density of a board can be determined by the following equation:

$$D_{board} = a/(c * r) \tag{2}$$

D_{board}, the density of a given board
a, the number of hidden atoms in a given board
c, the number of columns in a given board
r, the number of rows in a given board

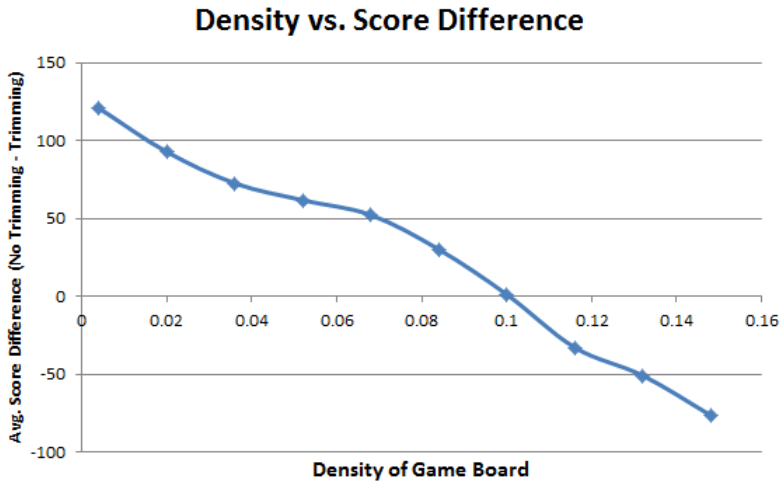

Figure 19: Model of Density and Average Score Difference

The density of a board is important to determine the utility of the trimming operation. Trimming is more efficient in lower density boards (less than ~0.1), as there is a lesser chance for collision along the edges. In denser boards, enabling the trimming algorithm actually punishes the solver in the sense that the score overcompensates for the information gained. In other words, the unsuccessful edge guesses were suboptimal.

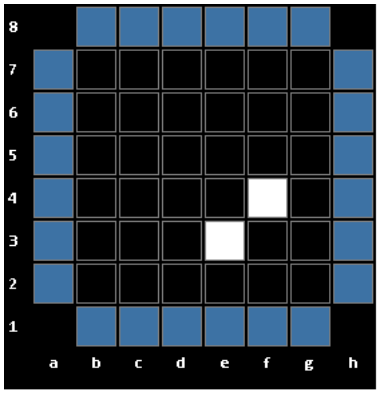

Figure 20: Board Density = 0.06

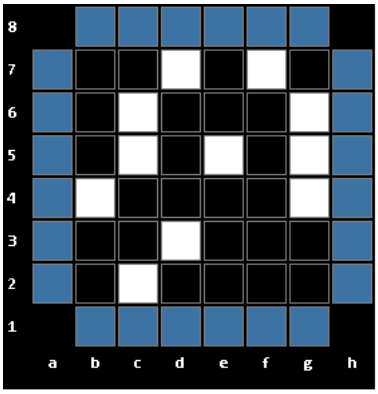

Figure 21: Board Density = 0.31

6.2 Even Guessing

We were naturally curious as to whether the distribution of revealed ports among the four sides of the grid affected the efficiency of the solver as well as the overall score. To accomplish this, we instructed our solver to intelligently choose a to shoot a ray from the side with the fewest revealed ports and we compared the average score to a solver that guessed at random among all sides. It is important to note that this algorithm can fail to produce the result intended under certain circumstances. For example if the first ray shot enters and leaves the same side, and the following ray starts on a new side but comes out of the first, the distribution of revealed ports remains skewed despite that it complies with the even-guessing algorithm.

7 Results

		Guess Stability	
		Enabled	Disabled
Trimming	Enabled	29.7	29.5
	Disabled	28.6	28.7

Table 1: Average score per game (8x8 grid, 4 atoms) over 100,000 trials

The results seen in the above table contradict our initial hypothesis that trimming the board and altering the guess logic would improve the overall score of the solver. When applied to the standard 8x8 size board (4 atoms), the partitioning algorithm increases the average score. This result is somewhat expected due to the moderate density of the standard board. We expect that a much lower density would yield results that support our hypothesis. Lastly, applying an even guessing filter to the solver had a negligible effect on the overall score, thus we believe that the manner in which information is obtained and used to make the final guess is far more important than the order in which particular ports are chosen.

8 Conclusions

Through iterative filtering, we find that, on average, every move does not need to be played to narrow down the list of candidates to the solution. With the same process of iteratively filtering boards, we are able to find a move that is good in terms of reducing possible candidates. However, since iterative filtering and the move selection method require that all candidate boards be filtered through, their performance is affected directly by the number of possible candidate states. There are scaling issues as the state space of board configurations increases. However, for smaller game sizes, both methods provide interesting results in terms of average score. As for the threshold approach, we have determined that

the choice of port is not as important as the way feedback is handled. There are two ways which this feedback is interpreted. First, we score the grid based on the type of response, then we add or remove certain squares from our final guess, based on their value. Our scoring system is not optimal, there are special cases which we have left generalized in the algorithm, thus ensuring functionality over efficiency. Still, the threshold solver shows that placing a restriction on our final guess (by requiring enough evidence to meet or exceed the threshold) means that otherwise erroneous assumptions can be avoided.

9 Acknowledgements

Dr. Sarah Eichhorn
UCI iCAMP Summer Research Program
National Science Foundation - Grant# DMS-0928427

References

[1] Nobel Media AB. The nobel prize in physiology or medicine 1979, 2013.

[2] Erik Arneson. Eric solomon, 2010.

[3] National Inventors Hall of Fame. Hall of fame/inventor profile, 2007.