Assignment 3 Domino Players Testing and Experimental Results

In this document I have first of all provided a list of all the tactics I have implemented along with a description of what each tactic does. Following this I have provided test results for all these tactics, note that these tactics use a number of other functions from my implementation but the testing from these highest order functions implies that the lower order functions work correctly. Then lastly in this document I have the experimental results of playing with different tactics showing that by adding more knowledge to a player results in that player winning more games.

|  |  |
| --- | --- |
| Tactic | Description |
| moveThatWillWin | If player has a domino in their hand that can win them the game then return the move of that winning domino |
| moveScoring59 | If player has a domino in their hand that can get them to a score of 59 then return the move of that domino, this is generally a good tactic because there are many ways of scoring 2 and hence finishing a game on the players successive turns. |
| checkOpponentWin | Return a list of doms in a player’s hand that the player can play that will prevent their opponent from winning the game. Takes into account what opponent can play based on what is believed to be in their hand (uses what is played on board, what is in players hand and when opponent may have been knocking). For each domino in this believed hand checks if a domino can win the game as a result of the player playing a playable domino. |
| score59MoveStopWin | This tactic is a combination of the movescoring59 tactic and checkopponentwin tactic. From the list returned that the player can play safely and won’t result in an opponent win checks if there is a move that will score 59 and if so returns it. |
| opponentDoesNotScoreMore | Plays the highest scoring domino in a hand that can be played and will not lead to opponent scoring more on their next turn based as a result of this move. Calculates opponents hand based on what is played on board, what is in players hand and when opponent may have been knocking |
| firstDrop | Plays optimal scoring (5,4) domino if player has first go, when playing this dom the score gained when placing is greater than any possible response the opponent can play. |
| playMajority | Returns move of highest scoring dom if player has a majority of one spot value from list of majority dominoes, if double is there then always plays that first. Takes into account dominoes of that spot value already played when calculating if a player has the majority of dominoes of one particular spot value. |
| highestKnockOff | Play highest scoring domino unless it is a dangerous domino (a double) and it can't be knocked off on a player’s successive moves. E.g. playing the (6,6) but player does not have any more doms with a spot value of a 6. |

Note in the testing where a ((7,7),L) is returned it means the conditions of that function have not been met and so the player will go through its next tactic, a player that has not met the conditions of a tactic will just play its highest scoring dom that will go.

**moveThatWillWin**

|  |  |  |
| --- | --- | --- |
| Test Condition | Call | Result |
| Test a valid winning move from a hand that wins when played right | moveThatWillWin [(3,2),(1,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,42) | ((4,6),R) |
| Test a valid winning move from a hand that wins when played left | moveThatWillWin [(3,2),(1,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (59,42) | ((3,2),L) |
| Test a hand that does not contain a winning move | moveThatWillWin [(3,2),(1,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (55,42) | ((7,7),L) |
| Test a hand that does not contain any doms that can be played | moveThatWillWin [(5,2),(1,0),(2,1),(5,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (55,42) | ((7,7),L) |
| Test with a hand that is empty | moveThatWillWin [] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (55,42) | ((7,7),L) |
| Test with an empty board and can’t win | moveThatWillWin [(5,2),(1,0),(2,1),(5,6)] (InitBoard) P1 (55,42) | ((7,7),L) |
| Test with an empty board but can win | moveThatWillWin [(5,4),(1,0),(2,1),(5,6)] (InitBoard) P1 (58,42) | ((5,4),L) |

**Move Scoring 59**

|  |  |  |
| --- | --- | --- |
| Test Condition | Call | Result |
| Test a valid 59 score move from a hand when played right | moveScoring59 [(5,2),(4,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,42) | ((4,0),R) |
| Test a valid 59 score move from a hand when played left | moveScoring59 [(3,2),(1,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,42) | ((3,2),L) |
| Test a hand that does not contain a 59 score move | moveScoring59 [(3,2),(1,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (51,42) | ((7,7),L) |
| Test a hand that does not contain any doms that can be played | moveScoring59 [(5,2),(1,0),(2,1),(5,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (55,42) | ((7,7),L) |
| Test with a hand that is empty | moveScoring59 [] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (55,42) | ((7,7),L) |
| Test with an empty board and can’t score 59 | moveScoring59 [(5,2),(1,0),(2,1),(5,6)] (InitBoard) P1 (55,42) | ((7,7),L) |
| Test with an empty board but can score 59 | moveScoring59 [(5,4),(1,0),(2,1),(5,6)] (InitBoard) P1 (58,42) | ((2,1),L) |

**Check opponent Win**

Using the list returned if it’s not empty the player will play the highest scoring in the list of doms that can be played that prevent an opponent win based on what could be in opponent’s hand.

|  |  |  |
| --- | --- | --- |
| Test Condition | Call | Result |
| Test Where I have playable dominoes and opponent can’t win if I place them | checkOpponentWin [] [(5,2),(4,0),(2,1),(4,6)] [(5,2),(4,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,42) | [(4,0),(4,6)]  Opponents possible hand was calculated to contain any of : [(6,6),(6,5),(6,4),(6,2),(6,1),(6,0),(5,5),  (5,4),(5,3),(5,1),(5,0),(4,4),(4,2),(4,1),  (3,2),(3,0),(2,2),(2,0),(1,1),(1,0),(0,0)] |
| Test Where I have playable dominoes and opponent can win if I place them | checkOpponentWin [] [(5,2),(4,0),(2,1),(4,6)] [(5,2),(4,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,59) | []  Opponents possible hand was calculated to contain any of : [(6,6),(6,5),(6,4),(6,2),(6,1),(6,0),(5,5),  (5,4),(5,3),(5,1),(5,0),(4,4),(4,2),(4,1),  (3,2),(3,0),(2,2),(2,0),(1,1),(1,0),(0,0)] |
| Test Where I have an empty hand | checkOpponentWin [] [] [] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,59) | []  Opponents possible hand was calculated to contain any of : [(6,6),(6,5),(6,4),(6,2),(6,1),(6,0),(5,5),  (5,4),(5,3),(5,1),(5,0),(4,4),(4,2),(4,1),  (3,2),(3,0),(2,2),(2,0),(1,1),(1,0),(0,0)] |

**See later testing as to where I test calculating an opponent’s hand**

**Score 59 move that will stop win**

This tactic uses a combination of the checkopponentwin tactic and score 59 tactics and so testing required for this tactic can be seen above. This tactic uses the exact same function call as score 59 but when taking a hand, it takes the list of dominoes that can be played that will prevent an opponent from winning.

**First Drop**

|  |  |  |
| --- | --- | --- |
| Test Condition | Call | Result |
| Test Where I do not have the (5,4) and the domboard is not empty | firstDrop [(5,2),(4,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) | ((7,7),L) |
| Test Where I do have the (5,4) and the domboard is not empty | firstDrop [(5,4),(4,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) | ((7,7),L) |
| Test Where I do have the (5,4) and the domboard is empty | firstDrop [(5,4),(4,0),(2,1),(4,6)] (InitBoard) | ((5,4),L) |
| Test Where I do not have the (5,4) and the domboard is empty | firstDrop [(5,2),(4,0),(2,1),(4,6)] (InitBoard) | ((7,7),L) |

**Opponent does not score more**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Condition | Call | | Result |
| Test where I do not have a domino that can be played that will score higher than any of my opponents possible doms as a result of me playing that dom | opponentDoesNotScoreMore [(5,2),(4,0),(2,1),(4,6)] [(5,2),(4,0),(2,1),(4,6)] (Board (3,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (50,45) | ((7,7),L)  Opponents possible hand was calculated to contain any of : [(6,6),(6,5),(6,4),(6,2),(6,1),(6,0),(5,5),  (5,4),(5,3),(5,1),(5,0),(4,4),(4,2),(4,1),  (3,2),(3,0),(2,2),(2,0),(1,1),(1,0),(0,0)] | |
| Test where I do have a domino that can be played that will score higher than any of my opponents possible doms as a result of me playing that dom | opponentDoesNotScoreMore [(6,6),(4,0),(2,1),(4,6)] [(5,2),(4,0),(2,1),(4,6)] (Board (6,2) (3,2) [((6,2),P2,4),((2,4),P1,1),((4,3),P2,2),((3,2),P1,3)]) P1 (50,45) | ((4,6),L)  Opponents possible hand was calculated to contain any of :  [(6,5),(6,4),(6,3),(6,1),(6,0),(5,5),(5,4),(5,3),(5,2)  ,(5,1),(5,0),(4,4),(4,1),(3,3),(3,1),(3,0),(2,2),(2,0)  ,(1,1),(1,0),(0,0)] | |
| Test where I do not have a domino that can be played that will score higher than any of my opponents possible doms as a result of me playing that dom (empty board) | opponentDoesNotScoreMore [(5,2),(4,0),(2,1),(4,6)] [(5,2),(4,0),(2,1),(4,6)] (InitBoard) P1 (50,45) | ((7,7),L)  Opponents possible hand was calculated to contain any of : [(6,6),(6,5),(6,4),(6,3),(6,2),(6,1),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,4),(4,3),(4,2),(4,1),(3,3),(3,2),(3,1),(3,0),(2,2),(2,0),(1,1),(1,0),(0,0)] | |
| Test where I do have a domino that can be played that will score higher than any of my opponents possible doms as a result of me playing that dom (empty board) | opponentDoesNotScoreMore [(6,6),(5,4),(2,1),(4,6)] [(6,6),(5,4),(2,1),(4,6)] (InitBoard) P1 (50,45) | ((6,6),L)  Opponents possible hand was calculated to contain any of :  [(6,5),(6,4),(6,3),(6,2),(6,1),(6,0),(5,5),(5,3),(5,2),(5,1),(5,0),(4,4),(4,3),(4,2),(4,1),(4,0),(3,3),(3,2),(3,1),(3,0),(2,2),(2,0),(1,1),(1,0),(0,0)] | |

**See later testing as to where I test calculating an opponent’s hand**

**Play Majority**

Will get the highest scoring majority domino if player has a majority of one domino, if double there play that first.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Condition | Call | Result | |
| Test where I do not have a majority of one domino based on what is in my hand and what is played (board empty) | playMajority [(6,3),(4,0),(2,1),(4,6)]  (InitBoard) | | ((7,7),L) |
| Test where I do have a majority of one domino based on what is in my hand and what is played (board empty) | playMajority [(6,3),(6,0),(6,1),(4,6),(6,6)]  (InitBoard) | | ((6,6),L) |
| Test where I do not have a majority of one domino based on what is in my hand and what is played (board not empty) | playMajority [(1,3),(4,0),(1,1),(4,6)]  (Board (6,2) (3,2) [((6,2),P2,4),((2,4),P1,1),((4,3),P2,2),((3,2),P1,3)]) | | ((7,7),L) |
| Test where I do have a majority of one domino based on what is in my hand and what is played (board not empty) | playMajority [(4,3),(4,0),(1,1),(4,6)]  (Board (6,2) (3,2) [((6,2),P2,4),((2,4),P1,1),((4,3),P2,2),((3,2),P1,3)]) | | ((4,6),L) |

**Highest Knock off**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Condition | Call | Result | |
| Test where I have a domino that is a double, it is my highest scoring dom and I can knock it off | highestKnockOff [(4,5),(1,0),(4,4),(4,6)] (Board (2,3) (3,4) [((3,3),P2,4),((3,6),P1,1),((1,3),P2,2),((3,4),P1,3)]) P1 (57,42) | | ((4,4),R) |
| Test where I have a domino that is a double, it is my highest scoring dom but I cannot knock it off (will get next highest scoring dom if there is one that can be played) | highestKnockOff [(2,1),(1,0),(4,4),(1,6)] (Board (2,3) (2,4) [((2,3),P2,4),((3,6),P1,1),((6,2),P2,2),((2,4),P1,3)]) P1 (57,42) | | ((2,1),L) |
| Test where I have a domino that is a double, it is my highest scoring dom but I cannot knock it off and I do not have another dom that can be played | highestKnockOff [(5,1),(1,0),(4,4),(1,6)] (Board (2,3) (2,4) [((2,3),P2,4),((3,6),P1,1),((6,2),P2,2),((2,4),P1,3)]) P1 (57,42) | | ((7,7),L) |

**opponentsPossHand**

Gets doms that could be in opponent’s hand based on doms played (allow for flips), doms in my hand and dom values where player was knocking if they ever were.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Condition | Call | Result | |
| Test where I have a full hand but board is empty | opponentsPossHand [(5,4),(1,0),(4,4),(6,4),(6,6),(5,5),(3,2),(0,0),(3,1)] (InitBoard) | | [(6,5),(6,3),(6,2),(6,1),(6,0),(5,3),(5,2),(5,1),(5,0),(4,3),(4,2),(4,1),(4,0),(3,3),(3,0),(2,2),(2,1),(2,0),(1,1)] |
| Test where I have a full hand and board contains a few doms | opponentsPossHand [(5,4),(1,0),(4,4),(6,4),(6,6),(5,5),(3,2),(0,0),(3,1)] (Board (0,3) (2,4) [((0,3),P2,4),((3,6),P1,1),((6,2),P2,2),((2,4),P1,3)]) | | [(6,5),(6,1),(6,0),(5,3),(5,2),(5,1),(5,0),(4,3),(4,1),(4,0),(3,3),(2,2),(2,1),(2,0),(1,1)]  Note: opponent not knocking in history here so this implicitly tests getting end values where player not knocking in history |
| Test where I have a full hand and domBoard contains a state where opponent was knocking in the history | opponentsPossHand [(5,4),(1,0),(4,4),(6,4),(6,6),(5,5),(3,2),(0,0),(3,1)] (Board (2,3) (2,4) [((0,3),P1,4),((3,6),P1,1),((6,2),P2,2),((2,4),P1,3)]) | | [(6,5),(6,1),(6,0),(5,2),(5,1),(5,0),(2,2),(2,1),(2,0),(1,1)]  Note: player 1 played on move 3 and 4, so opponent was knocking after move 3, the end values after move 3 were (4,3) so any doms containing the values (4,3) can not possible be in opponent’s hand |

**Experimental Results**

**Testing single tactics**

First of all, I will test each tactic solely on its own to see how it affects a hsd player, if the tactics conditions are not met then the player will simply result in playing its highest scoring domino (hsdPlayer). In all these tests the players will face a hsd player using the same seeds and same number of games (1000). In all these tests the tactical player will be player 2 on the console input.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Player1 tactics** | **Player 2 tactics** | **Seed** | **Results** | **Additional Games Won** | **Analysis** |
| hsdPlayer | hsdPlayer | 42 | (480,520) | N/A | N/A |
| 6452 | (524,476) |
| 463781185 | (507,493) |
| hsdPlayer | 1. **firstDrop** 2. hsdPlayer | 42 | (474,526) | 16 | Provides slight improvement over standard hsdPlayer |
| 6452 | (522,478) |
| 463781185 | (499,501) |
| hsdPlayer | 1. **moveThatWillWin** 2. hsdPlayer | 42 | (390,610) | 265 | Provides Huge improvement over standard hsdPlayer |
| 6452 | (441,559) |
| 463781185 | (430,570) |
| hsdPlayer | 1. **moveScoring59** 2. hsdPlayer | 42 | (595,405) | -349 | Provides massive decline over standard hsdPlayer when used on its own |
| 6452 | (638,362) |
| 463781185 | (630,370) |
| hsdPlayer | 1. **checkOpponentWin** 2. hsdPlayer | 42 | (432,568) | 144 | Provides moderate improvement over standard hsdPlayer when used |
| 6452 | (470,530) |
| 463781185 | (465,535) |
| hsdPlayer | 1. **score59MoveStopWin** 2. hsdPlayer | 42 | (506,494) | -97 | Provides moderate decline over standard hsdPlayer when used on its own |
| 6452 | (558,442) |
| 463781185 | (546,454) |
|  |  |
| hsdPlayer | 1. **opponentDoesNotScoreMore** 2. hsdPlayer | 42 | (481,519) | 0 | No improvement when used on its own |
| 6452 | (526,474) |
| 463781185 | (504,496) |
| hsdPlayer | 1. **playMajority** 2. hsdPlayer | 42 | (528,472) | -87 | Provides moderate decline over standard hsdPlayer when used on its own |
| 6452 | (540,460) |
| 463781185 | (530,470) |
| hsdPlayer | 1. **highestKnockOff** 2. hsdPlayer | 42 | (950,50) | -1316 | Seems like a terrible tactic |
| 6452 | (948,52) |
| 463781185 | (929,71) |

Based on these results we can see clearly that some tactics are performing better than others, however the tactics that performed worse than a standard hsdplayer should not be ignored because when used in logical combinations with other tactics they will improve performance as you will see later on. It is however safe to say that the tactic ‘**highestKnockOff’** seems to be a disaster of a tactic.

**Interesting logical combinations of tactics**

In the below tests I have combined a series of tactics in combinations that would logically improve performance. If the tactics conditions are not met then the player will simply result in playing its highest scoring domino (hsdPlayer). In all these tests the players will face a hsd player using the same seeds and same number of games (1000). In all these tests the tactical player will be player 2 on the console input.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Player1 tactics** | **Player 2 tactics** | **Seed** | **Results** | **Additional Games Won** | **Analysis** |
| hsdPlayer | hsdPlayer | 42 | (480,520) | N/A | N/A |
| 6452 | (524,476) |
| 463781185 | (507,493) |
| hsdPlayer | 1. **opponentDoesNotScoreMore** 2. **firstDrop** 3. hsdPlayer   (PLAYER 1) | 42 | (471,529) | 22 | Provides slight improvement over standard hsdPlayer |
| 6452 | (515,485) |
| 463781185 | (503,497) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. hsdPlayer   (PLAYER 2) | 42 | (383,617) | 273 | Provides Huge improvement over standard hsdPlayer |
| 6452 | (434,566) |
| 463781185 | (421,579) |
| hsdPlayer | 1. **checkOpponentWin** 2. **opponentDoesNotScoreMore** 3. **firstDrop** 4. hsdPlayer   (PLAYER 3) | 42 | (426,574) | 149 | Provides moderate improvement over standard hsdPlayer |
| 6452 | (470,530) |
| 463781185 | (461,539) |
| hsdPlayer | 1. **moveThatWillWin** 2. **opponentDoesNotScoreMore** 3. **firstDrop** 4. **playMajority** 5. hsdPlayer   (PLAYER 4) | 42 | (423,577) | 190 | Provides Huge improvement over standard hsdPlayer when used |
| 6452 | (440,560) |
| 463781185 | (458,542) |
| hsdPlayer | 1. **moveThatWillWin** 2. **moveScoring59** 3. hsdPlayer   (PLAYER 5) | 42 | (381,619) | 267 | Provides Huge improvement over standard hsdPlayer |
| 6452 | (435,565) |
| 463781185 | (428,572) |
| hsdPlayer | 1. **checkOpponentWin** 2. **moveScoring59** 3. **opponentDoesNotScoreMore** 4. **firstDrop** 5. **playMajority** 6. hsdPlayer   (PLAYER 6) | 42 | (444,556) | 77 | Provides moderate improvement over standard hsdPlayer |
| 6452 | (501,499) |
| 463781185 | (489,511) |

The results from these combinations of tactics are interesting since they all show improvement over a standard hsdPlayer. However, the most interesting thing here is that some tactics that previously scored badly such as moveScoring59 when used on its own improved performance when combined with other tactics. Therefore, a good player should consist of a logical order of going through a series of tactics in order to get the best possible move every time from its hand and based on the situation of the game.

**Building the best Player**

Here I demonstrate that by adding more knowledge/tactics to a player it will improve their performance based on the amount of knowledge on the game it captures. In all these tests the players will face a hsdPlayer using the same seeds and same number of games (1000). In all these tests the tactical player will be player 2 on the console input.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Player1 tactics** | **Player 2 tactics** | **Seed** | **Results** | **Additional Games Won** | **Analysis** |
| hsdPlayer | hsdPlayer | 42 | (480,520) | N/A | N/A |
| 6452 | (524,476) |
| 463781185 | (507,493) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. hsdPlayer   (PLAYER 1) | 42 | (383,617) | 273 | Provides Huge improvement over standard hsdPlayer |
| 6452 | (434,566) |
| 463781185 | (421,579) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. **moveScoring59** 4. hsdPlayer   (PLAYER 2) | 42 | (377,623) | 278 | Provides Huge improvement over standard hsdPlayer |
| 6452 | (429,571) |
| 463781185 | (427,573) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. **moveScoring59** 4. **opponentDoesNotScoreMore** 5. hsdPlayer   (PLAYER 3) | 42 | (381,619) | 288 | Provides Huge improvement over standard hsdPlayer |
| 6452 | (414,586) |
| 463781185 | (428,572) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. **moveScoring59** 4. **opponentDoesNotScoreMore** 5. **firstDrop** 6. hsdPlayer   (PLAYER 4) | 42 | (377,623) | 301 | Provides Huge improvement over standard hsdPlayer when used |
| 6452 | (409,591) |
| 463781185 | (424,576) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. **checkOpponentWin** 4. **moveScoring59** 5. **opponentDoesNotScoreMore** 6. **firstDrop** 7. hsdPlayer   (PLAYER 5) | 42 | (335,665) | 401 | Provides Huge improvement over standard hsdPlayer when used |
| 6452 | (387,613) |
| 463781185 | (384,616) |
| hsdPlayer | 1. **moveThatWillWin** 2. **score59MoveStopWin** 3. **checkOpponentWin** 4. **moveScoring59** 5. **opponentDoesNotScoreMore** 6. **firstDrop** 7. **playMajority** 8. hsdPlayer   (PLAYER 6) | 42 | (333,667) | 401 | Provides Huge improvement over standard hsdPlayer when used |
| 6452 | (389,611) |
| 463781185 | (384,616) |

**Conclusion**

As demonstrated by slowly adding more tactics to a player, the performance of the player is improved, if we look at seed 42 the tactical player wins 2/3 of all games which is a good result. Interestingly both player 5 and 6 scored the same however, they got different scores based on the seed value. With further testing it seems that playMajority added on top does improve performance when more than 1000 games are played but only by a handful of games. However, this still enforces my statement.