**HEURISTICS 1**

**Description:**

Heuristics 1 returns a score based on the number of moves of a node two levels deep in a search tree from the current board state.

**def heuristics\_1**(game, player):

score = 0.0  
 **for** move **in** game.get\_legal\_moves(player):  
 score = max(score,len(game.forecast\_move(move).get\_legal\_moves(player)))  
 **return** score

**Result:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Evaluating: ID\_Improved

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Playing Matches:

----------

Match 1: ID\_Improved vs Random Result: 78 to 22

Match 2: ID\_Improved vs MM\_Null Result: 68 to 32

Match 3: ID\_Improved vs MM\_Open Result: 39 to 61

Match 4: ID\_Improved vs MM\_Improved Result: 43 to 57

Match 5: ID\_Improved vs AB\_Null Result: 51 to 49

Match 6: ID\_Improved vs AB\_Open Result: 35 to 65

Match 7: ID\_Improved vs AB\_Improved Result: 41 to 59

Results:

----------

ID\_Improved 50.71%

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Evaluating: Student

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Playing Matches:

----------

Match 1: Student vs Random Result: 81 to 19

Match 2: Student vs MM\_Null Result: 63 to 37

Match 3: Student vs MM\_Open Result: 62 to 38

Match 4: Student vs MM\_Improved Result: 59 to 41

Match 5: Student vs AB\_Null Result: 65 to 35

Match 6: Student vs AB\_Open Result: 43 to 57

Match 7: Student vs AB\_Improved Result: 42 to 58

Results:

----------

Student 59.29%

**Analysis:**

The game agent outperforms ID\_Improved with a win rate of 59.29% to 50.71% in a 100-match game each against various opponents.

**HEURISTICS 2**

**Description:**

Heuristics 2 returns a score based on common legal moves between game agent and opponent. If such a move does not exist, it returns the difference between legal moves available to the game agent and opponent

**def heuristics\_2**(game, player):

own\_moves = len(game.get\_legal\_moves(player))  
 opp\_moves = len(game.get\_legal\_moves(game.get\_opponent(player)))  
 move = [own\_move **for** own\_move **in** game.get\_legal\_moves(player) **for** opp\_move **in** game.get\_legal\_moves(game.get\_opponent(player)) **if** own\_move == opp\_move]  
  
 **if** move:  
 game\_copy = game.forecast\_move(move[0])  
 result = len(game\_copy.get\_legal\_moves(player))  
 **if** result > 2:  
 **return** float('inf')  
 **else**:  
 own\_moves = len(game\_copy.get\_legal\_moves(player))  
 **return** float((2 \* own\_moves) - opp\_moves)  
 **return** float((2 \* own\_moves) - opp\_moves)

**Result:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Evaluating: ID\_Improved

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Playing Matches:

----------

Match 1: ID\_Improved vs Random Result: 53 to 47

Match 2: ID\_Improved vs MM\_Null Result: 57 to 43

Match 3: ID\_Improved vs MM\_Open Result: 54 to 46

Match 4: ID\_Improved vs MM\_Improved Result: 41 to 59

Match 5: ID\_Improved vs AB\_Null Result: 52 to 48

Match 6: ID\_Improved vs AB\_Open Result: 51 to 49

Match 7: ID\_Improved vs AB\_Improved Result: 54 to 46

Results:

----------

ID\_Improved 51.71%

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Evaluating: Student

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Playing Matches:

----------

Match 1: Student vs Random Result: 84 to 16

Match 2: Student vs MM\_Null Result: 71 to 29

Match 3: Student vs MM\_Open Result: 57 to 43

Match 4: Student vs MM\_Improved Result: 50 to 50

Match 5: Student vs AB\_Null Result: 47 to 53

Match 6: Student vs AB\_Open Result: 41 to 59

Match 7: Student vs AB\_Improved Result: 60 to 40

Results:

----------

Student 58.57%

**Analysis:**

The game agent outperforms ID\_Improved with a win rate of 58.57% to 50.71% in a 100-match game each against various opponents.

**HEURISTICS 3**

**Description:**

Heuristics 3 is a modification of heuristics 2.Scores are based on the number of available spaces left in the game.

**def heuristics\_3**(game, player):

own\_moves = len(game.get\_legal\_moves(player))  
 opp\_moves = len(game.get\_legal\_moves(game.get\_opponent(player)))  
  
 **if** len(game.get\_blank\_spaces()) > (game.width \* game.height) \* 0.7:  
 move = [own\_move **for** own\_move **in** game.get\_legal\_moves(player) **for** opp\_move **in** game.get\_legal\_moves(game.get\_opponent(player)) **if** own\_move == opp\_move]  
  
 **if** move:  
 game\_copy = game.forecast\_move(move[0])  
 result = len(game\_copy.get\_legal\_moves(player))  
 **if** result > 2:  
 **return** float('inf')  
 **else**:  
 own\_moves = len(game\_copy.get\_legal\_moves(player))  
 **return** float((2 \* own\_moves) - opp\_moves)  
 **return** float((2 \* own\_moves) - opp\_moves)  
 **else**:  
 **return** float((2 \* own\_moves) - opp\_moves)

**Result:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Evaluating: ID\_Improved

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Playing Matches:

----------

Match 1: ID\_Improved vs Random Result: 61 to 39

Match 2: ID\_Improved vs MM\_Null Result: 56 to 44

Match 3: ID\_Improved vs MM\_Open Result: 42 to 58

Match 4: ID\_Improved vs MM\_Improved Result: 48 to 52

Match 5: ID\_Improved vs AB\_Null Result: 47 to 53

Match 6: ID\_Improved vs AB\_Open Result: 52 to 48

Match 7: ID\_Improved vs AB\_Improved Result: 54 to 46

Results:

----------

ID\_Improved 51.43%

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Evaluating: Student

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Playing Matches:

----------

Match 1: Student vs Random Result: 78 to 22

Match 2: Student vs MM\_Null Result: 75 to 25

Match 3: Student vs MM\_Open Result: 64 to 36

Match 4: Student vs MM\_Improved Result: 52 to 48

Match 5: Student vs AB\_Null Result: 47 to 53

Match 6: Student vs AB\_Open Result: 48 to 52

Match 7: Student vs AB\_Improved Result: 52 to 48

Results:

----------

Student 59.43%

**Analysis:**

The game agent outperforms ID\_Improved with a win rate of 59.43% to 51.43% in a 100-match game each against various opponents.

**Recommendation:**

Heuristics 3 is recommended for the game agent, as it is the best performing heuristics with the highest win rate of 59.43%. It generalizes better than the other two heuristics when playing against various opponents. Though winning marginally against MM\_Improved and AB\_Improved, it performs better than heuristics 1 and 2.