For this project I have implemented three additional heuristic functions to supplement the existing ones named null_score, open_move_score, and improved_score. The inspiration for these new functions came from observations made while playing games with an instance of the HumanPlayer class against a CustomPlayer instance configured to use alpha-beta pruning with the improved_score scoring function.

The improved_score function works fairly well during all phases the game and against all the players in the tournament. Disappointingly, I discovered that none of the new heuristics alone produce very good results when compared with improved_score and so modified all of them to fall back onto or to combine with the results of that function in their final implementations.

The three general heuristics are named CSM, CCE, and CCM.

The CSM function checks to see whether the player and opponent share legal moves. More shared moves result in a lower score since I suspect that the improve_score function overstates the value of its calculation by not considering these common moves in its returned score.

Players can trap an opponent in a corner square by moving onto the last of the two unoccupied L-based moves leading out of the corner. Even if one player is not in a corner square, moving to these key squares on the board might effectively eliminate two squares with the single move. Thus the CCE function checks to see whether the player and opponent could move to any of the L-move board squares which give exit from the corner squares. The returned score is higher if the current player moves to one of these square and lower if the opponent could move to one.

The CCM function tests the hypothesis that positions near the center of the board have an advantage of leading to more open moves. The highest score is given if a player moves into the exact center of the board whereas the lowest one is when an opponent could do that. In addition the relatively small number of squares in the 7x7 game board used in the tournament do not allow for evaluating a score based upon how far a move is from the exact center. Instead, only player moves one square away are rewarded/penalized; for larger board sizes this should probably be changed to a percentage of the board area.

High variance in the results over many tournament runs made me unconfident about recommending any of the new heuristics instead of improved_score. The table of results included with this report reflects a single tournament run. For other runs, the results of even the ID_Improved player was observed to range from the low-60 to the mid-70 percentages.

The CCM and CCE heuristics did not appear to differ much in their scores in most tournament runs. In the run reported below the number of wins only ever differs by one between CCM and CCE. They did outperform the MM_Improved and AB_Improved opponents in most runs including this one. That isn’t surprising as, since it has already been mentioned, both functions rely upon special cases in addition to the base score from the improve_score function.

For most of the times when a new heuristic does beat that of improved_score it is the CSM function. I believe this may be due to three factors. First, recall that this function develops its initial score from improved_score. To refine that score further it make two additional checks. One check is to see if the player's last move will be blocked by the opponent (another special case). The other check is to adjust the score to remove common moves when it is not the last player's move in an effort to reward the player for making distinct moves from his opponent. Since the improved_score heuristic does not aid in implementing the "reflected" move strategy mentioned in the lectures (or anything similar) this means that both players tend to cover different paths through the square without colliding again (and taking move of each other's moves) in the future.

	ID_Improved	S_CSM	S_CCE	S_CCM
vs Random	17-3	19-1	18-2	17-3
vs MM_Null	16-4	14-6	14-6	14-6
vs MM_Open	13-7	14-6	14-6	13-7
vs MM_Improved	10-10	13-7	13-7	14-6
vs AB_Null	16-4	14-6	15-5	15-5
vs AB_Open	11-9	13-7	12-8	12-8
vs AB_Improved	11-9	15-5	12-8	11-9
Results	69.29%	72.86%	70.00%	68.57%