Based on the configuration of the current board and the current utility function, I added 3 more aspects with **Node ordering** and **State Caching** when considering the heuristic value, apart from only considering the number of pawns and number of Kings. In order to win faster, more weight has been added to aggressive strategies and less weight has been added to playing-safe strategies.

1. **Number of safe pawns and Kings - playing safe**

In order to win a game, you want to avoid being captured by the opponent. To be a safe pawn or a safe King, its best position will be on the edge. In this case, it will be either position at x==0, x==7, y==0 or y==7. So, for a red turn, we want to keep more red pawns and Kings safe on the edge, and fewer black pawns and Kings on the edge.

2. **Number of moveable pawns and Kings**

In order to win a game, if you cannot capture the opponent's pieces, another strategy is to move to a place that will be not blocked by the opponent (ie. Do not move to a place where there is no legal move left for yourself). In this case, for a red turn, considering more moveable red pawns and Kings and fewer movable black pawns and Kings is important.

Implementation: If the current red piece position is [x,y], we will check if two steps away are empty (ie. [x-2, y-2], [x-2, y+2] for red pawn and [x+2, y-2], [x+2, y+2] more for red King). The reason why we check two steps away is considering the following cases:

- if there is a black piece in between, then if it is a movable space aside, red can do a capture.
- if there is a red piece in between, then if it is a movable space aside, at least the red piece in between is able to move.
- if there is an empty space in between, then it is guaranteed that this red piece is movable.

3. **To promote a pawn - playing aggressive**

In order to win the game, you want to make as many kings as possible so that you can not only go forward but also go backward. So, the following two fields are important to be considered.

- **Aggregated distance of the pawns to the promotion line**

More weight will be added to the pawns that are on the opponent's side of the board so that pawns will continuously be pushed to the promotion line to be promoted to King.

- **Number of unoccupied fields on the promotion line for pawn**

The number of unoccupied fields on the opponent side/promotion line is also important. We want to make sure there is space on the edge for promotion, rather than just going forward and not considering anything. Because if there is no space on the promotion line, all your pawns will end up having no space to move forward before becoming Kings, and you lose due to there is no legal move left.

After adding these to the heuristic function, the **Node ordering** has also been implemented. For alpha pruning, we want the successor state with a higher heuristic value to run first so that it can prune those lower-value states. For beta pruning, we want the successor state with a lower heuristic value to run first so that it can prune those higher-value states.

Caching states also have been implemented. The caching creates a dictionary that maps board states to their minimax values. It also checks the depth to make sure that we are not underestimating the values with different depths.