score = 3 \* (my\_pieces - their\_pieces) + 2 \* (my\_mills - their\_mills) + (my\_mobility - their\_mobility)

In this function, my\_pieces and their\_pieces represent the number of pieces on the board belonging to the current player and their opponent, respectively. my\_mills and their\_mills represent the number of open mills that each player has, and my\_mobility and their\_mobility represent the number of legal moves that each player has.

The weights in this function (3, 2, and 1) are chosen somewhat arbitrarily and could be adjusted to emphasize different aspects of the game state. For example, if you wanted to create a more defensive AI, you might give greater weight to the number of pieces each player has on the board, while a more aggressive AI might prioritize open mills or mobility.

Ultimately, the "best" static estimation function for 9 Men's Morris will depend on the specific needs of your application and the goals of your AI system.

The function I suggested and the function you provided have different approaches to evaluating the game state of 9 Men's Morris. The function I suggested uses a weighted evaluation function that takes into account multiple factors, such as the number of pieces on the board, the number of open mills, and the mobility of the pieces. On the other hand, the function you provided uses a simpler approach that prioritizes specific conditions, such as having fewer than two pieces on the board or having no legal moves left.

The function you provided may be effective in some cases, such as when the game is close to ending and there are only a few pieces left on the board. In these cases, the function's focus on the number of pieces left and the number of legal moves remaining may be a good indicator of the game's state.

However, the function I suggested is more comprehensive and takes into account more factors that can affect the game state, such as the presence of open mills, which can lead to capturing opponent's pieces. It also includes weights that prioritize certain factors, such as mobility, which can be important in the early game when players are trying to position their pieces for future attacks.

Therefore, the effectiveness of these two functions will depend on the specific game state and strategy of the player. In general, a more comprehensive static evaluation function that considers multiple factors is likely to be more effective than a simpler function that focuses on specific conditions, especially in the early game when there are many possible moves and strategies to consider.

score = 5 \* (my\_pieces - their\_pieces) + 3 \* (my\_mills - their\_mills) + 2 \* (my\_blockable\_pieces - their\_blockable\_pieces)  
  
In this function, my\_pieces and their\_pieces represent the number of pieces on the board belonging to the current player and their opponent, respectively. my\_mills and their\_mills represent the number of open mills that each player has. my\_blockable\_pieces and their\_blockable\_pieces represent the number of pieces that can still be moved to form mills or block opponent's mills.

score = 10 \* (my\_pieces - their\_pieces) + 5 \* (my\_mills - their\_mills) + 2 \* (my\_blockable\_pieces - their\_blockable\_pieces) + 3 \* (my\_double\_mills - their\_double\_mills)

In this function, my\_pieces and their\_pieces represent the number of pieces on the board belonging to the current player and their opponent, respectively. my\_mills and their\_mills represent the number of open mills that each player has. my\_blockable\_pieces and their\_blockable\_pieces represent the number of pieces that can still be moved to form mills or block opponent's mills. my\_double\_mills and their\_double\_mills represent the number of mills that the player has where both adjacent spaces are empty.