Performance Evaluation Report for Hexagon Game

The HexagonGame class is a Python implementation of a game that involves adding edges to a hexagonal grid and is played between a human player and a computer AI. The objective is to avoid forming a triangle of edges on the grid.

Overall, the performance of the HexagonGame class is reasonable. The game loop functions well and the moves of both the human player and the computer AI are implemented correctly. However, there are some areas where the performance could be improved.

One area where performance could be improved is in the triangleFormed() function. This function checks if a triangle of edges has been formed on the grid. It does this by looping through all of the vertices in the grid and checking if there are three edges that connect them. This is not an optimal operation because it gives O(n^3) runtime. As the size of the grid increases, this operation becomes increasingly expensive. A more efficient algorithm could be used to determine if a triangle has been formed.

Another area where performance could be improved is in the minimax() function. This function is used by the computer AI to determine the best move to make and it does this by recursively calling itself to explore all possible moves and their outcomes. This is an expensive operation, especially as the depth of the search increases. We decided to set the MAX_DEPTH parameter to 3 in the current implementation, but this could be increased or decreased depending on the desired level of difficulty for the game. Additionally, alpha-beta pruning could be used to improve the performance of the minimax algorithm.

Overall, the performance of the HexagonGame class is reasonable, but there is room for improvement. By optimizing the triangleFormed() and minimax() functions the game could be made to run more efficiently and provide a better user experience.