My Original Heuristic Implementation:

My heuristic function takes the Manhattan cost as a base cost and adds other factors to formulate a total heuristic cost. It finds the location, in x and y coordinates, of the bottom left '1' square of the 2x2 piece. If the x-coordinate is 0, implying that the 2x2 piece is at the left edge of the board, I check if there are empty squares located at [3,2] and [4,2] respectively and I add a cost for every empty space that isn't there to the maximum of the two. This is because when the 2x2 piece slides to the desired destination at the finish, it requires BOTH empty squares to be at the aforementioned locations if the 2x2 is at the left edge. Similarly, I repeat this check if the 2x2 piece is at the right edge. I check if the x-coordinate of the bottom left '1' square is 2 and if the empty squares are located at [3,1] and [4,1] respectively, with my heuristic adding a cost of 1 per empty square that isn't at the checked location for the maximum of two. Again, this is because if the 2x2 piece is that the right side of the board, it requires both empty squares to be at those locations at its best case to solve the problem. If the empty pieces aren't there, we know we'll have to spend at least one move per empty square to get it to the desired location.

Now that we've looked at horizontal 2x2 piece shift possibilities, let's look at vertical possibilities. I continue to monitor the location of the bottom left '1' square; this time for it's y-coordinate. Knowing that the y-coordinate must be between 1 and 3, I use a for loop in range(y-coordinate, 4) and I check if the 2x2 piece can shift down vertically. It can only shift down vertically if the empty squares are besides each other and aligned horizontally adjacent to the 2x2 piece. Thus, in the for loop, I check for the horizontal coordinates directly below the 2x2 piece until the maximum of 3. I check directly under the 2x2 piece for empty spaces. If an empty space is NOT located at [y-coordinate+1, x-coordinate+1], I add one to the cost. If it isn't located at [y-coordinate+1, x-coordinate], I add one to the cost.

Thus, for the initial board configuration, my heuristic cost is **7**. This is because it adds the Manhattan cost, which is **3**, to any displaced horizontal empty spaces, which is 0 as the 2x2 piece is in the center and not the side, and any displaced vertical empty squares, which is **4** as the empty spaces are located at the very bottom center and the 2x2 piece has to cover the cost of going over two rows.

Where it's Located:

My heuristic function can be found under the "original_heuristic" function in my code. Here is a snippet of what I'm referring to; total_cost is initialized as the Manhattan heuristic already.

Why it's Admissible:

This is an admissible heuristic function because it depends on the movement of the 2x2 piece, which we know will always be stationary unless the empty spaces are besides each other and adjacent to the 2x2 piece. Thus, I look at the best scenario of the empty spaces being besides each other and the 2x2 piece going to the desired destination depending on the location of the 2x2 piece. Notice how when I check the vertical displacement, I don't assume that the 2x2 piece is at the top of the board to the left or top of the board to the right (which adds additional costs). I always assume that the 2x2 piece is at an optimal location and ready to move to the finish but check if the empty spaces will allow that finish and add one cost for every empty space that isn't where it needs to be for the 2x2 piece to finish.

Why it Dominates Manhattan:

We have already proved my heuristic is an admissible heuristic function. We also know that my heuristic function adds on to the already admissible Manhattan heuristic. Thus, we can deduce that my heuristic dominates the Manhattan heuristic because it's lowest value it can take is the Manhattan's highest value since Manhattan is its foundation. Any other costs that my heuristic incurs proves it's more dominant. Therefore, $h_{\text{original\_heuristic}} \geq h_{\text{Manhattan heuristic}}$ because Manhattan is a lower bound cost in original_heuristic.