**AI Offline**

Flood It Solver

ID: 1105021

**Basic Approach:**

I converted the given 2D array into a Connected Component Graph (CCG). To do so, I have passed two consecutive (modified) DFS that explores every element as Flood-fill algorithm do. The first pass marks only the connected components, then second pass makes a connected graph from that components. To do so, I have used a class **Node**, and three functions such as **makeExtendedBoard(), findConnected(), makeGraph().** Apart from these, there are some helper functions like **copyExtendedBoard(), printGraph(), getNode(), getNodeColor(), getConnectedNodes()** etc. The overall conversion happens when a new **Board** variable is created.

**Complexity:**

As the conversation is a modified form of DFS, it takes **2 x O(V+E),** which is **O(V+E)**, in our case it is **O(N+N)** or **O(N)**

**Heuristic 1:**

With my **extendedBoard** and **graph** I have tried to calculate the heuristic of node X (Color CLRx) by counting the maximum shortest distance between the all disconnected CLRx (same color) starting from node X. To do so, I have used BFS algorithm and with that I have calculated the minimum distance between the initial node (X, CLRx) and all the other nodes with CLRx. After that, I took the maximum value from all those minimum distances. For tiebreaking, I have implemented the difference in number of tiles.  
**Admissibility:**

This algorithm, in my point of view should have been admissible. Say there is at least n edges between two same color nodes, then we will need at least n moves. Even if we explore the in-between nodes first, we will still need at least n moves.   
So, I think it is admissible. **Complexity:**The algorithm explores as a BFS algorithm. So it will need **O(V+E).** But, here V can be as large as **N2** and on every iteration of edge, **extendedNodes** (**N2)** is explored**.** So, it is **O(N2+EN2)** or **O(EN2).**

**Heuristic 2:**

I have tried so many approaches to find a good or at least reasonable heuristic but in vain. All those implements are still in my code as comments. So causes the system to run out of memory, while others produce incorrect results. The version that I am submitting is based on the height of the BFS tree. The heuristic of node X is the BFS tree height starting from the node X.   
**Admissible:**This algorithm is supposed to be admissible as the minimum height must be crossed and no two connected nodes are of same colors. But it is not, as it gives incorrect answer sometimes.  
**Complexity:**The algorithm explores as a BFS algorithm. So it will need **O(V+E).** But, here V can be as large as **N2** and so is **E.** So, it is **O(N2 + N2)** or **O(N2).**