Task: TOU
Tour (author: Arkadiusz Czarkowski)

BOI 2025, Day 1: Analysis.

We are given a directed graph $G = (V, E)$, each edge has a colour. We seek a (not necessarily simple) cycle such that the colours of every two consecutive edges are different. Let us call such a cycle *alternating*.

The first idea is to define (or construct) a new larger graph G'. The vertices of G' are the edges of G. We add an edge from the vertex of G' corresponding to an edge (x, y) of G to the vertex of G' corresponding to an edge (y, z) of G when the colours of (xy) and (y, z) are different. Then, alternating cycles in G correspond to cycles in G'. We can thus build G' in $O(m^2)$ and use, say, the DFS algorithm to check if there is a cycle there in the same complexity.

The second idea is to observe that if colours c and c' are different then they must differ on some bit. This is a simple but powerful idea that is often used to reduce problems to their bichromatic versions. The previous solution can be seen as traversing the graph while keeping track of the last edge. Alternatively, we could only keep track of its colour, but this does not decrease the number of states in the worst case. However, we can try to only keep some partial information about the colour of the last edge. More precisely, we keep a bit position i and the bit b at that position. This is only $O(\log m)$ possibilities. Then, if we are currently at node x and consider if we can follow edge (x, y) then we check if the i-th bit of its colour is different than b. If so, we move to y, and decide which of the bits of the colour of the edge (x, y) to keep. This can be rephrased as running a DFS in a graph of size $O(m \log^2 m)$. Formally, we define a graph G' in which each node is a triple (x, i, b). Then, for every edge $(x, y) \in E$ such that the i-th bit of its colour is different than b and for each $i \in \{0, 1, \ldots, \log m\}$, we add an edge from (x, i, b) to (y, i', b'), where b' is i'-th bit of the colour of (x, y). The number of nodes is $O(n \log m)$, and the number of edges $O(m \log^2 m)$, and alternating cycles in G correspond to cycles in G', so we obtain an $O(m \log^2 m)$ time algorithm.

To further improve the complexity, we can separate checking if the i-th bit of the colour of (x, y) is different than b from guess which bit of its colour should be now stored. We add an edge from (x, i, b) to an intermediate node (x, y), and then from the intermediate node (x, y) to (y, i', b'). This possibly increases the number of nodes to $(m + n \log m)$, but guarantees that the number of edges is $O(m \log m)$, while alternating cycles in G still correspond to cycles in G', so we obtain an $O(m \log m)$ time algorithm.

However, the model solution works in $O(m)$ time by running the first $O(m^2)$ DFS solution on G' without explicitly constructing the whole G' and avoiding visiting the same node (other than the root node r connected to every node u with an edge of colour 0) of G more than three times. This can be implemented in $O(m)$ time.

In more detail, we prune the visited part of G' as follows. Whenever we visit a node u, we reach it by following an edge with some colour c. We denote this state by (u, c). Now, whenever the current state is (u, c) and the recursion stack already contains the state (u, c), we have found an alternating cycle. Otherwise, if (u, c) has already been fully processed by the recursion, there is no need to process it again. The remaining case is that (u, c) is neither already fully or partially processed.

Now consider the case when we visit u for the third time, and let the states corresponding to all three visits be (u, c_1), (u, c_2), and (u, c_3). We know that c_1, c_2, c_3 are pairwise distinct. But then, if we are currently at node u after following an edge with colour c_3, and subsequent edge (u, v) with colour $c' \neq c_3$ could have been as well traversed either when we were at node u after following an edge with colour c_1 or c_2. Thus the only reason for visiting u for the third time is to close a cycle, but then we can immediately terminate.