Problem

The cat-and-mouse game is played by two players, “Cat” and “Mouse,” on an arbitrary undirected graph. At a given point, each player occupies a node of the graph. The players take turns moving to a node adjacent to the one that they currently occupy. A special node of the graph is called “Hole.” Cat wins if the two players ever occupy the same node. Mouse wins if it reaches the Hole before the preceding happens. The game is a draw if a situation repeats (i.e., the two players simultaneously occupy positions that they simultaneously occupied previously, and it is the same player’s turn to move).

$$\textit{HAPPY-CAT} = \{\langle G, c, m, h\rangle \mid G, c, m, h \text{ are respectively a graph, and positions of the Cat, Mouse, and Hole, such that Cat has a winning strategy if Cat moves first}\}.$$

Show that *HAPPY-CAT* is in P. (Hint: The solution is not complicated and doesn’t depend on subtle details in the way the game is defined. Consider the entire game tree. It is exponentially big, but you can search it in polynomial time.)

Step-by-step solution

Step 1 of 1

Firstly note that game can have only $2n^2$ configuration, defined by position of the cat, position of the mouse and if it is cat's turn. So can construct a directed graph consisting of $2n^2$ nodes where every node corresponds to a configuration of game and there is an edge from node u to the node v, if we can go from configuration corresponding to u to configuration corresponding to v in one move.

Now following algorithm solves HAPPY-CAT.

1. Mark all nodes (a,a,x) where a is a node in G, and $x \in \{true, false\}$.

2. If for a node $u = (a,b,true)$, there is a node $v = (c,b,false)$ which is marked and (u,v) is an edge then mark u.

3. If for a node $u = (a,b,false)$, all nodes $v = (a,c,true)$ are marked and (u,v) is an edge then mark u.

4. Repeat steps 2 and 3 until no new nodes are marked.

5. Accept if start node $s = (c,m,true)$ is marked.

Here algorithm takes $O(n^2)$ time to perform step 1, $O(n^2)$ time per iteration of the loop while loop is executed $O(n^2)$ times. Hence runs in polynomial time.

Comment