DFS & applications: contest analysis

Moscow International Workshop 2020

November 30, 2020

A. Party Planning

Interpret guests as boolean variables. The variable is true if the guest comes to the party. Let respectful guests be $x_1, \ldots, x_n$ and careless guests be $y_1, \ldots, y_m$.

First, let's check if at least any answer exists. Build a 2-SAT instance on the input. If no solutions exist then the overall answer is "NO", else it is "YES" or "MAYBE". The hard part is to distinguish between those cases.

The answer is "MAYBE" if we can find any assignment of y-s such that the 2-SAT has no solutions. In fact, this is true if and only if there are two vertices which correspond to some y-s such that one of them is reachable from the other. For example, y_1 should be reachable from y_2, or even y_1 from $\overline{y_1}$.

We denote "u is reachable from v" with $v \longrightarrow u$.

1. **Proof that it is sufficient** If $v \longrightarrow u$ and both vertices correspond to careless guests then they may plan their visits in a way that u is false and v is true, hence contradiction.

2. **Proof that it is necessary** Assume the opposite: no y-vertex is reachable from another y-vertex, but there is an assignment of y-s such that there is no 2-SAT solution. That means that there is a contradiction: both x and $\overline{x}$ are reachable from some truthy y-vertices (maybe distinct).

   - x and $\overline{x}$ are reachable from the same vertex v: $v \longrightarrow x$ and $v \longrightarrow \overline{x}$. Due to skew-symmectric structure $x \longrightarrow \overline{v}$, and by transitivity $v \longrightarrow \overline{v}$. Contradiction.
   - x and $\overline{x}$ are reachable from distinct vertices: $v \longrightarrow x$ and $u \longrightarrow \overline{x}$. Due to skew-symmectric structure $x \longrightarrow \overline{u}$, and by transitivity $v \longrightarrow \overline{u}$. Contradiction.

   This finishes the proof of nesessity.

3. **Algorithm** Now we have a graph with marked vertices and want to check if one marked vertex is reachable from another. Build a condensation of the graph. If there is more than one vertex in any SCC then we found two such vertices. Otherwise we get the same problem on an acyclic graph.

   For the acyclic graph, proceed its nodes in topological order. For each vertex maintain whether there is some marked vertex from which the current vertex is reachable. If current vertex is marked and some of its parents is reachable from some other marked vertex then we found the pair of such vertices. Otherwise no marked vertex is reachable from another one.

B. Flights

Binary search on the answer x, and check if the graph with edges weighted $\leqslant x$ is strongly connected.
We don't need to find SCCs, just run a DFS from any vertex using regular and reverse edges.

C. Recoloring

For a vertex v with color 1 make a variable x_v which is true if new color of v is 2 and false if it is 3. Make similar variables for vertices with other colors. Now each edge gives restrictions of kind "if color of u is t then color v is not t". All restrictions can be described with 2-SAT as there are only two options for each variable.

D. Bridges

Implement the algorithm from the lecture.

E. Cut Points

Implement the algorithm from the lecture.

F. Triplets

For each triplet (a, b, c), create a new triplet vertex v and connect it with all of a, b, c.
Suppose that we remove a triplet vertex v with all adjacent edges. The resulting graph is connected iff the original configuration with the respective triplet removed is connected.
It now suffices to find all triplet vertices which are articulation points.

G. Eulerian Path

Locate odd vertices (if any), and implement the algorithm from the lecture.

H. Even and Odd

Find vertex-biconnected components of the graph.
Consider edges of any vertex-biconnected component C. If the edges form a bipartite graph, then for any vertices v, u in C all paths between v and u have the same parity.
Otherwise, consider an odd cycle in C. We can use this cycle to find both an odd path and an even path between any two vertices v, u.
It follows that if a vu-path passes through a non-bipartite vertex-biconnected component, then either parity of a path between v and u is possible. In the other case, the parity is fixed (since it is fixed in each visited component).
We can now remove all edges of non-bipartite components from the graph. The remaining graph has several bipartite connected components, and vertices with fixed parity of a connecting path belong to the same component. Use DFS to 2-color each component, and count pairs with same/different colors.

I. Kingdom and Reforms

Consider connected components separately. Assume that the graph G is connected.
We can see that there is no solution if G has a vertex of degree 1, or if G is a cycle of odd length. If G is a cycle of even length, the solution is trivial.
In any other case, G has a vertex v_0 of degree $\geqslant 3$. Connect vertices of odd degress with additional edges to obtain a new graph G'. In G' all degrees are even, thus we can find an Euler cycle. Start looking for the cycle from v_0, and make sure that the first edge is an extra edge (if present).
Assign edge types alternately along the Euler cycle: `HTHT...`. Each vertex now has an adjacent edge of each type.
We argue that this is still true if we remove extra edges.
Vertices of even degrees have all adjacent edges intact.
v_0, and any vertex of odd degree will have a consecutive pair of edges that will stay intact.

In fact, a simpler solution is possible: add extra edges to the whole graph, and apply the solution to each component. If the resulting coloring after removing extra edges is invalid, then the answer is NO.