获得的答案 **返回**

A list of final exams $F_1,...,F_k$ is to be scheduled for a list of students $S_1,...,S_l$, with each student taking a subset of the exams. The exams have to be scheduled into slots using only h slots and so that no student has two exams in the same slot. This problem can be **formulated as a language** and also can be proved as "**it is NP-complete**".

This problem can be formulated as the following language $SCHEDULE$:

$$SCHEDULE = \left\{ \begin{array}{l} \langle F, S, h \rangle \mid F \text{ is the list of finals, } S \text{ is the list of subsets indicating which} \\ \text{finals each student is taking, and } h \text{ is the maximum number of slots the} \\ \text{schedule can take such that no student has more than one exam in a slot} \end{array} \right\}$$

The verifier V for this language $SCHEDULE$ will take the schedule of time slots T for the final exams F as a certificate. It would then check each of the subsets of exams for the students, which are the members of S.

The **algorithm** for the verifier V is:

V = "On input $\langle F, S, h, T \rangle$:

1. For each sublist of S, test if more than one exam lies in the same slot of T.
2. If yes, *reject*.
3. Test if all elements of F have a slot in T.
4. If no, *reject*.
5. Test if the number of slots used by T is more than h.
6. If yes, *reject*. Else *accept*.

The steps 1, 3 and 5 of the verifier V all take polynomial time so the verifier takes polynomial time. Therefore**, the language $SCHEDULE$ is in NP**.

In the h Color Graph problem, the vertices of an undirected graph have to be colored in h colors such that for each and every edge (u, v), the vertices u and v are colored differently. Later it will be proven that the h Color Graph problem is NP-complete.

• Now, try to reduce all instances of the h Color Graph problem into instances of $SCHEDULE$ in polynomial time.

• An instance of the h Color Graph problem with k vertices can be mapped to an instance of $SCHEDULE$ with k final exams and a maximum of h slots taken by the schedule.

• The lists of exams taken by the students are a subset of the set of the k vertices. An edge between two vertices indicates a conflict, which is at least one student needs to take both exams.

This reduction can be performed in polynomial time. It is valid as:

• If an instance of the h Color Graph problem is true, then the instance of $SCHEDULE$ produced by the reduction will be true as exams (or vertices in the graph) colored identically can be scheduled in the same time slot and the length of the schedule will be h time slots as there are h different colors.

• From an instance of the $SCHEDULE$ problem, an instance of the h Color Graph problem can be created. The vertices are created from the list of exams. Edges are drawn between vertices if at least a single student has to give both the related exams.

• The vertices corresponding to exams in the same time slot are colored identically. This instance of the Graph Color problem is satisfied as only the vertices that do not lie in the same time slot can have edge between them and such vertices shall not have the same color.

Now it needs to check which class of problems the h Color Graph problem lies in. If it lies in NP-complete then the $SCHEDULE$ problem will also be in NP-complete as the NP-complete class of problems is reducible in polynomial time to it.

• The verifier V for the h Color Graph problem will take the $C = \{c_1,...,c_k\}$ colors assigned to the $N = \{n_1,...,n_k\}$ vertices as the certificate.

The **algorithm** for the verifier V will be:

V = "On input $\langle N, C \rangle$:

1. Test if all the colors C are valid.
2. If no, *reject*.
3. For each vertex n_i in N:

   Check if all adjacent vertices are colored differently.
4. If yes, *accept*; else *reject*.

The verifier takes polynomial time so the h Color Graph problem is in NP. Following which, NP-complete problems have to be shown to be polynomial time reducible to the h Color Graph problem**. The h SAT problem is reduced to the h Color Graph problem**.

An instance of the h SAT problem will have n clauses with each clause containing h variables $v_1, ..., v_h$. The variable gadget will be the vertices of the graph labeled with the variables and the clause gadget will be the vertices (and their edges) added corresponding to them.

• An instance of the h Color Graph problem is constructed by adding all the variables $v_1, ..., v_h$ and their complements $\overline{v_1}, ..., \overline{v_h}$ as vertices of the graph.

•

All the variable vertices are connected to their complements by an edge. Label the pair formed by a variable vertex and its complement as $true$ for the variable and $false$ for the complement.

Next, add a new set of vertices $x_1, ..., x_h$ to the graph in the form of a clique (by joining each new vertex with all the other new vertices). Color these vertices $x_1, ..., x_h$ with colors $c_1, ..., c_h$.

• Also connect each x_i with v_j and $\overline{v_j}$ except for the case when $i = j$. Color one of v_j or $\overline{v_j}$ with the same color as the corresponding x_j. This can be done with $h + 1$ colors.

Thus, the h Color Graph problem has been proven to be NP-complete. Consequently the $SCHEDULE$ problem is also in NP-complete.