Second Level Algorithms

Prof. Palash Dey

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Week – 12

Lecture 59

Welcome to the 59th lecture of second level algorithm course. In the last lecture, we have proved that CNSAT and 3SAT are NP complete. In this lecture, we will see NP completeness proof of many other graph theoretic and important problems. So, let us begin. Our first problem is the clique problem.

So, what is the clique problem? input is a graph G and an integer k output is does there exist a clique of size k in G in particular or that is does there exist a subset of vertices such that size of s is k and for all vertices uv in s u not equal to v, there is an edge between u and v in G. In particular, pictorially I am given a graph G and an integer k and does there exist a subset S of size k such that between every pair of vertices there is an edge in this subset. So, this is the clique problem and we will show that click is NP complete proof the click problem belongs to NP because any clique of size k can be verified in polynomial time for YES instances to prove NP-hardness, we reduce from 3-set. Let I_1 is C_1 and C_2 and C_m be any instance of 3SAT. We construct an equivalent instance I_2 of clique as follows.

So, again, as usual, instead of writing this formally, let us understand the reduction with an example. So, suppose I_1 is, say, x_1 or $\bar{x}_2$ or x_3 and $\bar{x}_1$ or x_4 or $\bar{x}_5$ and x_3 or $\bar{x}_4$ or x_5 and so on. So, what we do for every clause is introduce 3 vertices. So, these are clauses; let us call it C_1, this is clause C_2, this is clause C_3, and so on.

So, C_1, each vertex is labeled by the corresponding variable. So, the clauses for C_1 are one vertex labeled with x_1, another vertex labeled with $\bar{x}_2$ and another vertex labeled with x_3, ok. So, these are the variables for the clause C_1; these are the 3 vertices corresponding to clause C_1. Similarly, for C_2, we introduce 3 vertices labeled by corresponding variables.

$\bar{x}_1$, x_4, $\bar{x}_5$. Similarly, for C_3, we introduce 3 vertices labeled by x_3,$\bar{x}_4$, x_5, and so on. So, the total number of vertices. So, if this graph is G, the cardinality of V(G) is 3m, OK? Now, we will introduce edges.

There will not be any edge between the 3 vertices corresponding to any clause, and across clauses, we will add edges if they are not contradicting each other. For example, there should not be an edge between $\bar{x}_1$ and x_1 because $\bar{x}_1$ and x_1 are contradicting each other; both cannot be true at the same time. We should add an edge between $\bar{x}_1$ and $\bar{x}_2$. There is an edge between $\bar{x}_1$, $\bar{x}_3$, and so on.

So, let me finish this drawing with all the edges. There will not be an edge between x_4 and $\bar{x}_4$, OK? So, this is our reduced incidence. This is the graph, and the clique that we are looking for, k, is m. So, this is I_2. Now, we claim that I_1 and I_2 are equivalent. OK.

So, if there is a satisfying assignment for I_1, at least one literal from every clause is set to true, by the satisfying assignment. So, we pick one literal from every clause which is set to true; the corresponding vertex we pick, one vertex from the three vertices, corresponding to every clause. Let S be the resulting set of vertices. The size of S is m, which is k, and because no two vertices from S can contradict each other—because all of them in S are simultaneously set to true—S forms a clique. Okay, hence I_2 is also an YES instance. Similarly, the other direction—let me give it as homework. Show that if I_2 is an YES instance, then I_1 is also an YES instance. So, this shows I_1 and I_2 are equivalent, and also the reduction takes a polynomial amount of time. Hence, this proves that the clique problem is NP-hard. Our next problem is the opposite of the clique problem, which you can call the independent set problem.

Here also. The input is a graph G and an integer k. The question that we are interested in is: does there exist an independent set of size k? That is, does there exist a subset S of vertices of size k such that for every pair of vertices uv from S, $u \neq v$, there should not be an edge between u and v in G. Pictorially, this is a graph G, and the question is: does there exist an independent set of size k?

What is an independent set? Between every pair of vertices, there is no edge between them. This is the theorem that the independent set problem is NP-complete proof. The independent set problem clearly belongs to the class NP, OK, because an independent set of size k can be verified in polynomial time for every YES instance.

To prove NP-hardness, we reduce an arbitrary instance of clique to an equivalent instance of independent set. So, let I_1 equal to (G, k). be an arbitrary instance of clique, then

We claim that I_2 equals G complement, k. What is G complement? Between every pair of vertices in G, if there is an edge, then that pair of vertices does not have an edge in G complement. On the other hand, if between a pair of vertices there is no edge in G, then there will be an edge in G complement. For example, if G is this graph, then G complement has the same set of vertices but whichever pair of vertices has an edge in G, there should not be an edge in G complement and vice versa. So, this will have edges AC and BD, OK. So, we claim that I_2 is an equivalent instance of I_1. This follows from the observation that a subset S of vertices is a clique in G if and only if S is an independent set in G complement. Hence, if I_1 is a yes instance of clique, that is, there exists a subset S of size k which is a clique in I_1. that same subset S of size k is an independent set in G bar, and hence I_2 is also a yes instance. This chain of implications follows in the reverse direction also. Hence, they are equivalent, hence independent set is also NP-hard.

We have already shown membership in NP; hence, it is NP-complete. Our next problem is vertex cover. The input is again a graph G and an integer k, and the question is: does there exist a vertex cover of G of size k? What is a vertex cover?

A set of vertices is called a vertex cover of G if, for every edge u v of G, either u belongs to S or v belongs to S, or both. So, it is a subset of vertices which contains at least one endpoint of every edge. In the next lecture, we will show that the vertex cover problem is NP-complete.

Okay, so let us stop here. Thank you.