Edin Guso
23435

CS407 HW8

1) i) We first need to show that SI $\in$ NP.

Verification of a solution can be done in polynomial time because the numbers of edges of a graph is bounded polynomially. Therefore, checking if all the edges of one graph exist in another is polynomial time. Thus, SI $\in$ NP.

ii) We need to reduce an NP-complete problem to SI.

Take Clique problem. Say that parameters of the clique problem are k and G'. Now, we will map the clique problem to SI by setting G and H as following:

G = complete graph on k vertices (construction takes polynomial time)

$H = G'$ (assignment takes constant time)

Clique problem on G' and k has a solution iff SI on G and H has a solution.

SI $\in$ NP-Complete.

2) i) We first need to show that IP $\in$ NP.

Verification of a solution can be done in polynomial time because given the assignment to n variables, it needs to check m equalities and verify that they hold. Thus, IP $\in$ NP.

ii) We need to reduce an NP-Complete problem to IP.

Take SAT problem. For every variable x_i in the SAT problem, create two variables x_i and x_i^c in IP. Additionally, for each variable x_i add the following:

$$x_i + x_i^c = 1$$

This way exactly one of the variables will be 1 (true) and the other will be 0 (false).

For every clause $C_i = (x_j, \dots, x_k)$ in the SAT problem, add the following:

$$x_j + \dots + x_k \neq 0$$ [checking for inequalities is the same difficulty as checking for equalities]

This way, all the clauses will have at least one true variable.

SAT problem has a solution iff IP problem has a solution.

IP $\in$ NP-Complete.

Edin Guso
23435

3) i) We first need to show that 3C∈NP

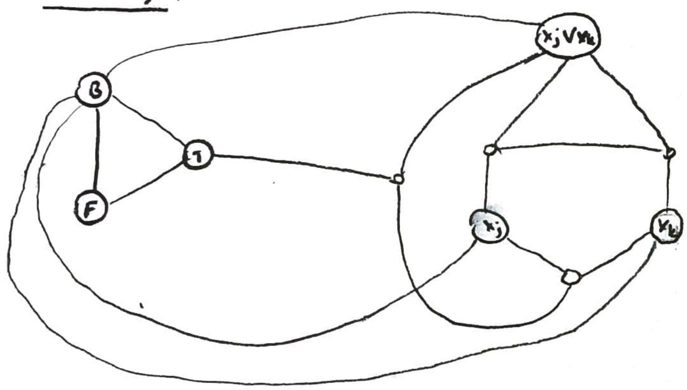
Verification of a solution can be done in polynomial time because verification is as simple as checking the two ends of each edge and the number of edges is polynomially bounded. Thus, 3C ∈ NP.

ii) We need to reduce an NP-Complete problem to 3C.

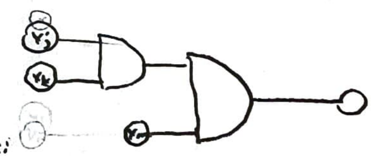
Take 3SAT Problem. For every variable x_i in 3SAT, create two vertices x_i and x_i'. Additionally, create 3 vertices called T, F and B (base). Additional vertices will be added later through OR-gadgets.

First, connect T, F and B into a triangle and let their colors be color T, F and B respectively. Connect each x_i and x_i' to each other and to B so that they each get one of T or F colors.

OR-Gadget:



This or-gadget simulates a real or-gate. Or gates will be used to simulate clauses. Since this is 3-SAT we will need two of these or-gadgets for each clause. Representation of or gates:



Representation for two or-gates used to simulate a clause:

Finally, we will need to connect output of every 3-or-gadget to F and B to force its coloring to T.

When we generate our graph like this in polynomial time every x_i and x_i' get opposite T and F colors, output of every clause is computed correctly and forced to be T.

Therefore, 3SAT problem has a solution iff 3C has a solution.

3C ∈ NP-Complete.