HW #6

Michael Smith

Problem #1:

A) "If Y is NP-Complete then so is X" = $\boxed{\text{False}}$

* For NP-Complete, X has to be the subset of NP-complete

B) "If X is NP-complete then so is Y" = $\boxed{\text{False}}$

* Y cannot be NP-Complete since X is reducible to Y, not Y is reducible to X

C) "If Y is NP-complete & X is in NP then X is NP-complete"

$\boxed{\text{False}}$

* X is reducible to Y, so X can be effiriently solved by Y, but not vice-versa given the relation of X & Y

D) "If X is NP-Complete & Y is in NP then Y is NP-complete"

$\boxed{\text{True}}$

* The problem X reduces to problem Y so that if there is a black box to solve Y efficiently, then that can be used to solve problem X efficiently.

* X is easier than Y

E) "If X is in P, then Y is in P" = $\boxed{\text{False}}$

* Problem X is reducible to Y, therefore, if Y is P then X is P

F) "If Y is in P, then X is in P" = $\boxed{\text{True}}$

* Problem X reduces to problem Y, if there is a black box to solve Y efficiently, then that can be used to solve X efficiently

G) "X & Y can't both be NP-Complete" = $\boxed{\text{False}}$

* X can be NP-complete if and only if Y is NP-complete

Problem #2:

A) "3-SAT $\leq_p$ TSP" = $\boxed{\text{True}}$

* If one NP-complete problem can be solved in polynomial time, then all NP problems can be solved in polynomial time.

* Here, 3-SAT is NP-complete & TSP is also an NP Problem. So, it can be solvable in polynomial time

B) "If P $\neq$ NP, then 3-SAT $\leq_p$ 2-SAT" = $\boxed{\text{False}}$

* If P $\neq$ NP & there is a polynomial-time reduction from 3-Sat to 2-Sat; then 2-Sat is NP-complete, but 2-Sat is also solvable in polynomial time. So, for all decision problems $A \in NP$ you can decide $x \in A$ reducing x to the corresponding 2-Sat instance in polynomial time and solve it in polynomial time. So $A \in P$ and hence $NP \subseteq P$, but we also have $P \subseteq NP$, so $P = NP$ is a contradiction.

C) "TSP $\leq_p$ 2-SAT, then P=NP" = $\boxed{\text{False}}$

* If one NP-complete problem can be solved in polynomial time, then all NP problems can be solved in polynomial time. If that is the case, then NP & P set become same which contradicts the given statement.

Problem #3:

* HAM-PATH $\in$ NP
* Certificate = A sequence of vertices y
* Verification = check whether y is Hamiltonian path from u to v in G
* Agorithm $A(\langle G, u, v\rangle, \langle y\rangle)$ steps:
  1. Check whether y has n vertices, if not return 0
  2. Check whether $y = (y_1, y_2 \dots y_n)$ has repeated vertices, if not return 0
  3. Check whether $\{y_i, y_{i+1}\} \in E$ for $i = 1, \dots n-1$ and whether $\{y_n, y_1\} \in E$, if some of tests fail return 0
  4. Check whether $y_1 = u$ and $y_n = v$, if not return 0, otherwise return 1

* Algorithm A runs in polynomial time since:
  - Step 1 runs in $O(n)$ steps
  - Step 2 runs in $O(n)$ Steps
  - Step 3 runs in $O(n)$ steps
  - Step 4 runs in $O(1)$ steps
* So A runs in $O(n)$ steps, showing how algorithm A is a correct verification for HAM-PATH.

Problem #4:

* Steps are to prove that 4-Color is in NP and prove that 4-Color is NP-hard, therfore proving 4-Color is NP-complete.

* Proving 4-Color is in NP:

  - The coloring denotes the certificate that represents the nodes list and colors, We can state a verifier V for 4-Color:

    V = via input (G, C)

    1. We will examine that C comprises ≤ 4-Colors

    2. We will Color every node of G as stated by C

    3. In respect of every node, we will examine that it has a distinct color from it's neighbors.

    4. If all examinations achieve success the Accept otherwise Reject

* Proving 4-Color is NP-Hard:

  - Put a polynomial-time reduction from 3-Color to 4-color. This will map a graph G into a new graph G′ like G ∈ 3-Color if G′ 4-Color. Do so by establishing G′ to G and then add new node z, linking z to every node in G′.

  - If G is 3-Colorable then G′ can be 4-colored exactly like G with z is the only node colored with the extra color. Accordingly, if G′ is 4-Colorable then we are aware that nod z must be the sole node of its color because it is linked to each different node in G′. We are aware that G must be 3-Colorable. This will execute in linear time to add a single nod and G edges.

* b/c 4-Color is in NP & NP-hard, it is NP-complete