Xiaoyu Pen CMPSC464 Collaborator: None

1. a) $(x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$

The formula is not satisfiable, since both values for x and y will result in False.

x	y	$x \vee y$	$x \vee \bar{y}$	$\bar{x} \vee y$	$\bar{x} \vee \bar{y}$	formula
F	F	F	T	T	T	F
F	T	T	F	T	T	F
T	F	T	T	F	T	F
T	T	T	T	T	F	F

b) Assume there are 2 languages where $L_1 \in P$ and $L_2 \in P$
so there are polynomial-time TM M_1 decides L_1 and M_2 decides L_2.
for $L_1 \circ L_2$. we construct a TM M with following. reference: web.njit.edu/~marvin/cs341/hw/hwsoln11.pdf page 2.

M= on input w with length n
1) For each i between 1 and n
2) partition w into 2 parts where $w_1 = a_1 a_2 \cdots a_i$, $w_2 = a_{i+1} \cdots a_n$
3) Run M_1 on w_1, M_2 on w_2
4) if both M_1 and M_2 accept, accept.
5) if for loop ends and not accept, reject.

the for loop will loop at most n+1 times, n is length of input. the partition takes $O(1)$
and running M_1, M_2 on w_1, w_2 will give us runnig time $O(n^{k_1})$ and $O(n^{k_2})$ where k_1 and k_2 are constants
thus, the overall running time is $O(n+1)(O(n^{k_1}) + O(n^{k_2}))O(1) = O(n^{1+MAX(k_1,k_2)})$ which $\in P$
Therefore, P is closed under concatenation

2. ISO = {<G,H> | G and H are isomorphic graph}.

Construct a non-deterministic TM N that decides ISO in polynomial time

N = on input <G,H>, G and H are undirected graphs

1) check whether graph G and H have same number of nodes n, if not, reject

2) select permutation f of n nodes nondeterministically

3) for each pair of nodes, x and y nodes

4) check (x,y) is edge in G IFF (f(x),f(y)) is edge in H.

5) if all loops agree, accept. else, reject.

Since the procedure can be done in polynomial time, with some part non-deterministic, therefore ISO ∈ NP

reference: web.cse.ohio-state.edu/~rademacher.10/Sp16_6321/ps3sol.pdf

3. Assume P=NP

let $A \in P$ such that $A \neq \emptyset$ and $A \neq \Sigma^*$, thus strings x and y exists where $x \in A$, $y \in \bar{A}$

let $L \in NP$, an arbitrary language from NP=P, thus there is a M decides L in polynomial time

Show $L \leq_p A$

we can give a polynomial-time reduction from L to A as follows

f: on input w

1) Run M on w first

2) if M accepts then output x, if M rejects the output y.

obvious this is a polynomial time reduction from L to A, therefore, A is NP-complete

reference: people.cs.aau.dk/~srba/courses/tutorial-CC-10/t12-sol.pdf

cs.rpi.edu/~goldberg/14-CC/HW/14-test-2-sol.pdf

4.

① PATH is not NP-complete

assume PATH is NP-complete

then for all $A \in NP$, L is reducible to PATH in polynomial time

but, it implies that for all $L \in NP$, L is in P.

So it results in: PATH is NP-complete IFF P=NP, which we belief is not true.

② Assume we can prove PATH is not NP-complete

thus it means $L \in NP$, L is not reducible to PATH in polynomial time

However, we know PATH is in P

therefore, with contradiction, it implies $P \neq NP$

reference: stackoverflow.com "Proving PATH problem is not a NP-complete problem"
github.com "ryanbaugherty/Introduction-to-the-Theory-of-Computation-Solution/chapter7.tex"

5. On input ϕ, a non-deterministic polynomial-time TM can guess 2 assigns and if both assignments satisfy ϕ, accept.

thus, we say DOUBLE-SAT is in NP

to show $SAT \leq_p$ DOUBLE-SAT

we construct a TM M computes the polynomial-time reduction f

F= on input $\langle\phi\rangle$, ϕ is a boolean formula with var $x_1, x_2, \ldots, x_n$

1) let ϕ' be $\phi \wedge (x \vee \bar{x})$, where x is a new variable

2) output $\langle\phi'\rangle$

if $\phi \in$ SAT, then ϕ' will have at least 2 satisfying assign that obtained from original by changing x.

if $\phi' \in$ DOUBLE-SAT, then ϕ is also satisfiable, since x is not in ϕ.

therefore, we can say that $\phi \in$ SAT IFF $f(\phi) \in$ DOUBLE-SAT

hence, DOUBLE-SAT is NP-Complete.

reference: cobweb.cs.uga.edu/~potter/theory/7-time-complexity_II.pdf