

Name _____ Student No. _____

No aids allowed. Answer all questions on test paper. Use backs of sheets for scratch work.

All four questions worth 25, for a total of 100.

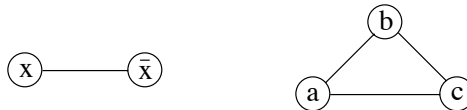
1. Define **NP** in terms of a verifier-certificate, and in terms of nondeterministic Turing machines. Show that the two definitions are equivalent.

Solution: One definition is in terms of a polytime verifier $V(w, c)$, which is polytime in $|w|$, and the language is defined as the set of strings w such that $\exists cV(w, c)$ (note that c , or its relevant prefix, is necessarily of polynomial length). The other definition is in terms of a polytime nondeterministic Turing machine. The equivalence holds because in one direction, the accepting branch of the computation provides a c , and in the other direction, the machine “tries-out” all the c ’s of polynomial length, and checks whether $V(w, c)$ holds.

2. In this question you will show that $\text{VERTEX-COVER} = \{ \langle G, k \rangle \mid G \text{ is an undirected graph that has a } k \text{ node vertex cover} \}$ is **NP-complete**. Recall that a *vertex cover* of a graph G is a subset S of the nodes of G such that each edge has at least one end-point in S .

(a) Show that VERTEX-COVER is in **NP**.

(b) Show that $3\text{SAT} \leq_p \text{VERTEX-COVER}$. Define $f(\langle \phi \rangle) = \langle G_\phi, k_\phi \rangle$, where ϕ is a 3CNF formula, and G_ϕ is defined as follows: for each variable x we have a “variable gadget” and for each clause we have a “clause gadget”:



where x is a variable and a, b, c stand for the literals in a given clause.

Finish the definition of G_ϕ by explaining what are the edges connecting the variable gadgets with the clause gadgets.

What is k_ϕ ?

Give a proof of correctness of the reduction (this is the bulk of the marks for this question).

(Use this page and the next for this question.)

Solution: To show VERTEX-COVER is in **NP**, let c be a subset of the vertices, and let V verify that each edge has at least one end-point in c .

In the reduction, connect each literal in the clause gadgets to the corresponding literal in the variable gadget (x to x and \bar{x} to \bar{x}).

Let m_ϕ be the number of variables in ϕ , and let l_ϕ be the number of clauses in ϕ . Let $k_\phi = m_\phi + 2l_\phi$.

If ϕ is satisfiable, let t be a satisfying truth assignment, and select those nodes in the variable gadgets which correspond to the literals satisfied by t . In the clause gadgets, find a node whose literal is satisfied by t , and select the other two nodes. The selected nodes form a vertex cover.

For the other direction of correctness, assume we have a vertex cover S , which necessarily contains one node from the variable gadgets, and two nodes from the clause gadgets. Let t make true the literals labeling the nodes (in S) in the variable gadgets. This t must satisfy at least one literal per clause: consider a clause gadget, note that one of the three nodes is not selected to be in S , so its other end-point must be in S , but its other end-point is labelled by the same literal, which is made true by t .

3. Show that the following two language are in **P**:

(a) CONNECTED = $\{\langle G \rangle \mid G \text{ is a connected undirected graph}\}$.

(b) TRIANGLE = $\{\langle G \rangle \mid G \text{ contains a 3-clique}\}$.

Solution: We give polytime algorithms for both: for CONNECTED, consider a node-marking algorithm, which starts at any given node, and marks nodes connected by an edge to it, and continues as long as new nodes are being marked. At the end, if there is an unmarked node, there is a disconnected component. Note that the algorithm is polytime. For TRIANGLE, consider the algorithm which looks at all triples of nodes, $\binom{n}{3} = O(n^3)$ many of them, and checks if any of them is a clique.

4. Show that if $\mathbf{P} = \mathbf{NP}$, then the language MIN-FORMULA is in \mathbf{P} . The language MIN-FORMULA = $\{\langle \phi \rangle \mid \phi \text{ is minimal, in the sense that if } \phi \equiv \phi', \text{ then } |\phi| \leq |\phi'|\}$.
- (a) First show that the language NOT-EQUIV consisting of $\langle \phi, \phi' \rangle$ such that ϕ is *not* equivalent to ϕ' (i.e., they differ on some truth assignment) is in \mathbf{NP} . Conclude that if $\mathbf{P} = \mathbf{NP}$, then EQUIV ($\langle \phi, \phi' \rangle$ such that $\phi \equiv \phi'$) is in \mathbf{P} .
- (b) Using (a) show that if $\mathbf{P} = \mathbf{NP}$ then NOT-MIN-FORMULA (the complement of MIN-FORMULA) is in \mathbf{P} , and hence MIN-FORMULA is in \mathbf{P} .

Solution: NOT-EQUIV is in \mathbf{NP} : let c be a truth assignment, and let V check that $t(\phi) \neq t(\phi')$. If \mathbf{P} and \mathbf{NP} are equal, NOT-EQUIV is in \mathbf{P} , which is closed under complementation, so EQUIV is in \mathbf{P} .

For NOT-MIN-FORMULA let c be a formula ϕ' , and let V check that two things are true: $|\phi'| < |\phi|$, and $\phi' \equiv \phi$. Note that the second condition is polytime verifiable by (a). Again, by equality of \mathbf{P} and \mathbf{NP} , and by the fact that \mathbf{P} is closed under complementation, we have the result.