

Problem 12.1. (6 points) Problem 7.16 of Sipser.

Solution 12.1

1. *SPATH* is in P. Here is a polynomial time algorithm to decide *SPATH*:

- (a) Run breadth first search algorithm on G starting from a .
- (b) If the distance from a to b is less than or equal to k , then accept.
- (c) Otherwise, reject.

This works because BFS will find the shortest path from a to b in polynomial time. If the shortest path has length less than or equal to k , then the instance (G, k) is in *SPATH*. If G contains a simple path of length at most k from a to b , then the shortest path from a to b will also be less than or equal to k .

2. First, note that *LPATH* is in NP. The certificate is simply the path of length at least k from a to b . To show *LPATH* is NP hard, we reduce from *UHAMPATH* to *LPATH*:

- (a) On input $\langle G = (V, E), a, b \rangle$ return $\langle G, a, b, |V| - 1 \rangle$.

This reduction clearly runs in polynomial time. If G has an Hamiltonian path from a to b , then that path is a simple path from a to b of length $|V| - 1$ and is therefore in *LPATH*. Conversely, if there is a simple path of length at least $|V| - 1$ from a to b , then it must pass through every other node in the graph, and is a Hamiltonian path.

Problem 12.2. (8 points)

a. Problem 7.10 of Sipser.

b. Problem 7.21 of Sipser.

Solution 12.2

a. Here is an easy n^3 algorithm for **triangle**, where n is the number of vertices of the graph. Simply iterate over all three tuples (a, b, c) of states and check if all the three edges (a, b) , (b, c) , and (c, a) are present.

b. **HALF-CLIQUE** is in NP. The certificate is simply the list of nodes in the clique. It is also NP-hard. We will give a polynomial time reduction from *CLIQUE* to **HALF-CLIQUE**:

1. On input $\langle G = (V, E), k \rangle$ do the following:
2. Construct a new graph $G' = (V', E')$ that contains all the vertices and edges in G .
3. If $|V| < 2k$, then add $2k - |V|$ additional nodes to G' so that $|V'| = 2k$. Leave the additional nodes unconnected to the rest of the graph.
4. Else if $|V| > 2k$, then add $|V| - 2k$ additional nodes to G' so that $|V'| = 2|V| - 2k$. Connect each of these new nodes to all other vertices in G' .

5. Return $\langle G' \rangle$.

This runs in polynomial time since we are adding $O(|V|)$ vertices to the graph. This reduction works by constructing a new graph G' such that G has a clique of size at least k nodes if and only if G' has a half-clique. First, consider the case where $|V| < 2k$. If G has a clique of size k , it is a half-clique in G' since $k = |V|/2$. Conversely, if G' has a half-clique, then it must be within the nodes of V since the extra nodes added to G' are not connected. Since $|V|/2 = k$, then this is a clique in G of size at least k . Second, consider the case where $|V| > 2k$. If G has a clique of size k , then the same clique in G' will also be connected to the $|V| - 2k$ new nodes, forming a clique of size $|V| - k = |V|/2$. This is a half-clique in G' . Conversely, if G' contains a half-clique, the largest half-clique in G' will contain all the new nodes in G' because they are connected to all other nodes. Removing the new nodes leaves at least $|V| - k - (|V| - 2k) = k$ nodes in the clique. This means that G has a clique of size k .

Problem 12.3. (8 points)

- a. Problem 7.37 of Sipser.
- b. Problem 7.19 of Sipser.

Solution 12.3

- a. For brevity we will refer *2-cnf satisfiability* as 2SAT. Let ψ be a instance of 2SAT, that is, the set of clauses with two literals each. We can define a graph $G(\psi)$ as follows: the vertices of $G(\psi)$ are the literals (variables and their negation) of ψ ; and there is an edge (α, β) if and only if there is a clause of the form $(\neg\alpha \vee \beta)$ (or $(\beta \vee \neg\alpha)$). Intuitively, these edges capture the logical implication (\Rightarrow) of ψ . As a result the graph $G(\psi)$ has a nice symmetry property: If (α, β) is an edge then so is $(\neg\beta, \neg\alpha)$ (see Fig 1 for example). Paths in $G(\psi)$ are also valid implications (by transitivity of \Rightarrow).

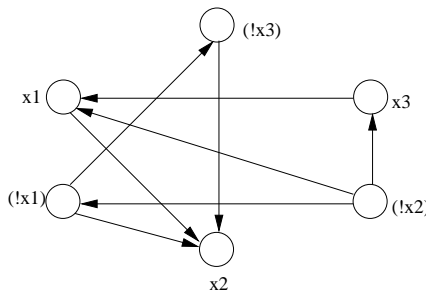


Figure 1: The graph for the formula $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_2) \wedge (x_2 \vee x_3)$

Now we prove the following:

ψ is unsatisfiable if and only iff there is a variable x such that there is a path from x to $\neg x$ in $G(\psi)$.

Proof: Suppose that such paths exist and still ψ can be satisfiable by a truth assignment T . Suppose $T(x)$ is true (symmetric argument works when $T(x)$ is false). Since there is a path from x to $\neg x$, and $T(x)$ is true and $T(\neg x)$ is false, there must be an edge (α, β) such that $T(\alpha)$ is true and $T(\beta)$ is false. However, since (α, β) is an edge in $G(\psi)$, it follows that $(\neg\alpha \vee \beta)$ is a clause in ψ which is not satisfied and hence we arrive at a contradiction.

Suppose there is no such path in $G(\psi)$ then we are going to construct a satisfying truth assignment, that is, a truth assignment such that there is no edge true to false. We repeat the following step: Pick a node α whose truth value is not decided. We consider all reachable nodes of α and assign them true. We also assign false to the negations of these nodes (the negations correspond to all these nodes from which $\neg\alpha$ is reachable). This step is well defined because, if there were paths α to both β and $\neg\beta$ then there would be paths to $\neg\alpha$ from both these (by symmetry of $G(\psi)$), and therefore a path from α to $\neg\alpha$, a contradiction to our hypothesis. Furthermore, if there were a path from α to a node already assigned false in a previous step, then α being a predecessor of that node was also assigned false at that step.

We repeat the above step until every node is assigned a truth value assignment. Since we assumed there is no path from variable x to $\neg x$ and back, all nodes will have a truth value. It is easy to prove that the truth assignment satisfy ψ . (why?)

- b. DOUBLE-SAT is in NP . The certificate is any two satisfying assignments of ϕ . We can reduce SAT to DOUBLE-SAT with the following polynomial time reduction: On input $\langle\phi\rangle$ return $\langle\phi \wedge (x_1 \vee x_2)\rangle$, where x_1 and x_2 are variables that do not appear in ϕ . Note that $x_1 \vee x_2$ has exactly three satisfying assignments. The reduction obviously runs in polynomial time. Also, $\langle\phi\rangle$ is in SAT iff $\langle\phi \wedge (x_1 \vee x_2)\rangle$ is in DOUBLE-SAT. If we have a satisfying assignment for ϕ , then we can construct two satisfying assignments for $\phi \wedge (x_1 \vee x_2)$ by using the same assignment with $x_1 = 1, x_2 = 0$ and with $x_1 = 0, x_2 = 1$. If we have any satisfying assignments of $\langle\phi \wedge (x_1 \vee x_2)\rangle$, there will be at least two satisfying assignments. In fact there will be at least three: one with $x_1 = 1, x_2 = 0$, one with $x_1 = 0, x_2 = 1$, and one with $x_1 = 1, x_2 = 1$. The satisfying assignment for any one of these cases must also satisfy ϕ , simply by the rules of the boolean AND.

Problem 12.4. (8 points)

- a. Problem 7.11 of Sipser.
b. Prove that the following problem is NP-complete:

SUBGRAPHISOMORPHISM: given two undirected graphs $G = (V, E)$ and H , is there a subset $E' \subseteq E$ of the edges of G such that the graphs $G' = (V_{E'}, E')$ and H are isomorphic? Here, $V_{E'} = \{v \in V : (\exists w \in V)(\{v, w\} \in E')\}$ is the set of nodes incident to edges in E' .

Hint: reduce CLIQUE to SUBGRAPHISOMORPHISM. Does your reduction work for GRAPHISOMORPHISM (problem ISO of part a)?

Solution 12.4

- a. Graph isomorphism is in NP. Let $\langle G_1, G_2 \rangle$ be an instance of the graph isomorphism problem. A nondeterministic Turing machine can guess the isomorphism function f between the vertex sets, i.e., for each vertex v of G_1 , the corresponding vertex $f(v)$ of G_2 . This is clearly polynomial in the size of the graphs. Then it needs to check that f is one-to-one and onto (to make sure it is an isomorphism), and also for every vertex v of G_1 , the set of neighbors of v is isomorphic to the set of neighbors of $f(v)$ under the function f . All this can be done in polynomial time.

b. SUBGRAPHISOMORPHISM is in NP. The certificate is the subgraph (V', E') isomorphic to H and the isomorphism. It is also NP-complete. Here is a polynomial time reduction from CLIQUE:

1. On input $\langle G = (V, E), k \rangle$, do the following:
2. Construct a fully connected graph H with k nodes.
3. Return $\langle G, H \rangle$

This clearly runs in time polynomial to the input. The graph G has a clique of size k if and only if it is subgraph isomorphic with the fully connected graph with k nodes. If G has a clique of size k , then the clique is isomorphic with H . Conversely, if G has a subgraph that is isomorphic with H , then that subgraph is a clique of size k . This reduction does not work for the GRAPHISOMORPHISM problem since G may not even contain the same number of nodes as H .