Homework 11

Jiahao Liu April 14, 2024

1. Max Degree Spanning Tree Problem

Given an undirected graph G = (V, E), and positive integer k, the max-degree-spanning-tree problem asks whether G has a spanning tree whose degree is at most k. The degree of a spanning tree T is defined as the maximum number of neighbors a node has within the tree (i.e., a node may have many edges incident on it in G, but only some of them get included in T). Prove MDST problem is NP-complete.
Prove the MDST Problem is NP: Certificate: A spanning tree T of G with maximum degree at most k.
Certifier: Check if T is a spanning tree of G, i.e. all vertices are connected with |V|-1 edges, and the degree of each vertex is at most k. This can be done in polynomial time.
Hamiltonian Path $\leq_p$ MDST: Given any input of HP, a graph G, we input the exact graph into MDST problem and set K = 2. Now, G has a HP iff G has a spanning tree of max degree 2. Given a blackbox to solve the MDST with max degree k, for any instance of Hamiltonian Path problem, we can just call the blackbox with k = 2. If the blackbox returns true, then the original graph has a Hamiltonian Path. Otherwise, it does not.

Proof. **Suppose we have a Hamiltonian Path in G:** Then we can just construct a spanning tree with the same edges as the Hamiltonian Path, and the degree of each vertex is at most 2.
Suppose we have a spanning tree with maximum degree at most 2: Then the spanning tree is actually a Hamiltonian Path, because the degree of each vertex is at most 2, so there are at most two edges connected to each vertex, which is the definition of Hamiltonian Path. □

2. K-Cycle-Decomposition Problem

The k-cycle-decomposition problem for any k > 1 works as follows. The input consists of a connected graph G = (V, E) and k positive integers $a_1, a_2, \dots, a_k \leq |V|$. The goal is to determine whether there exist k disjoint cycles of sizes $a_1, a_2, \dots, a_k$ respectively, s.t. each node in V is contained in exactly one cycle. Show that this problem is NP-complete (for any k > 1).
Prove the k-cycle-decomposition Problem is NP
Certificate: A set of k disjoint cycles of sizes $a_1, a_2, \dots, a_k$ respectively, s.t. each node in V is contained in exactly one cycle.
Certifier: Check if the set of cycles is disjoint, each node is contained in exactly one cycle, and the size of each cycle is $a_1, a_2, \dots, a_k$ respectively. This can be done in polynomial time.
Hamiltonian Cycle $\leq_p$ k-Cycle-Decomposition: Given any instance of Hamiltonian Cycle Problem in G = (V, E), we can just copy G paste it twice in two layers in G' and add one more edge between two layers in G' to make it connected. Then there is a Hamiltonian Cycle in G iff there is a 2-cycle-decomposition in G' with size |V|,|V|.

Proof. **Suppose there is a Hamiltonian Cycle in G:** Then there should be two same Cycles in G' as in G, which is actually a 2-cycle-decomposition of G' with size |V| and |V|.
Suppose there is a 2-cycle-decomposition in G' with size |V| and |V|: Then there should be one cycle of size |V| in each layer, becuase the only one edge between layers can't be in any cycles. Otherwise, the only edge that connect two layers must be used at least twice. So we get a Hamiltonian Cycle of G in any one layer of G'. □

3. Half-Independent-Set Problem

Given a graph G = (V, E) with an even number of vertices as the input, the HALF-IS problem is to decide if G has an independent set of size |V| / 2. Prove that HALF-IS is in NP-Complete.
Prove the HALF-IS Problem is NP
Certificate: An independent set of size |V| / 2.
Certifier: Check if the set is an independent set, i.e. no two nodes are joined by an edge. Check the size of set is |V| / 2. This can be done in polynomial time.
Independent Set $\leq_p$ HALF-IS: Given any Independent Set problem in G = (V, E) with integer k, we need to construct a graph G' base on G depending on the parity of |V| and the relationship between k and |V| / 2.
1. If |V| is even and k < |V| / 2, then we can add m isolated nodes to G' to make k + m = (|V| + m) / 2, so m = |V| - 2k. Then there are an Independent set of size k + m = |V'| / 2 in G' if and only if there are an independent set of size at least k in G.

Proof. **Suppose there is an Independent Set of size k in G:** Then plus the added |V| - 2k isolated nodes, there should be an Independent Set of size (|V| - k) = |V'| / 2 in G'.
Suppose there is an Independent Set of size |V'| / 2 in G': Then in worst case, all |V| - 2k isolated nodes are

included in the Independent Set, then there should be an Independent set of size (|V'| / 2 - (|V| - 2k)) = (|V| - k - |V| + 2k) = k in G. □

2. If |V| is even and k > |V| / 2, then we can add m fully connected nodes to G' to make k + 1 = (|V| + m) / 2, so m = 2k + 2 - |V|. Then there are an Independent set of size k + 1 = |V'| / 2 in G' if and only if there are an independent set of size at least k in G.

Proof. **Suppose there is an Independent Set of size k in G:** Then plus any one from the added 2k + 2 - |V| fully connected nodes, there should be an Independent Set of size (k + 1) = |V'| / 2 in G'.
Suppose there is an Independent Set of size |V'| / 2 in G': Then in worst case, there is one node from the added fully connected nodes, then there should be an Independent set of size (|V'| / 2 - 1) = (2k + 2) / 2 - 1 = (k + 1 - 1) = k in G. □

3. If |V| is even and k = |V| / 2, then we can just copy G to G'. There is an Independent set of size at least |V| / 2 in G if and only if there are an independent set of size |V'| / 2 in G'.

Proof. **Suppose there is an Independent Set of size |V| / 2 in G:** Then there should be an Independent Set of size |V'| / 2 in G'. **Suppose there is an Independent Set of size |V'| / 2 in G':** Then there should be an Independent Set of size |V| / 2 in G. □

4. If |V| is odd and k < (|V| + 1) / 2, the same as the first case.
5. If |V| is odd and k >= (|V| + 1) / 2, the same as the second case.

4. Set Cover Problem

In a certain town, there are many clubs, and every adult belongs to at least one club. The town's people would like to simplify their social life by closing off as many clubs as possible, but they want to make sure that everyone will still belong to at least one club afterwards. Formally the Redundant Clubs problem has the following input and output.
INPUT: List of people; list of clubs; list of members of each club; number K.
OUTPUT: Yes if there exists a set of K clubs such that, after removing all these K clubs, each person still belongs to at least one club. No otherwise.
Prove that the Redundant Clubs problem is NP-Complete
Prove the Redundant Clubs Problem is NP
Certificate: A set of K clubs such that, after removing all these K clubs, each person still belongs to at least one club.
Certifier: Check if the set is a set of K clubs, and after removing all these K clubs, each person still belongs to at least one club. This can be done in polynomial time.
Set Cover $\leq_p$ Redundant Clubs: Given any Set Cover problem with universe $U = \{1, \ldots, n\}$, a collection of subsets $\{S_1, \ldots, S_m\}$, and an integer k, we can construct a Redundant Clubs problem with people = U, clubs = $\{S_1, \ldots, S_m\}$, members of each club is the elements in each subset, and number $K = m - k$. Then there is a collection of at most k subsets that covers U if and only if there is a set of K clubs such that after removing all these K clubs, each person still belongs to at least one club.

Proof. **Suppose there is a collection of at most k subsets that covers U:** Then we can just remove all the left m - k subsets to get a set of K clubs such that after removing all these K clubs, each person still belongs to at least one club. **Suppose there is a set of K clubs such that after removing all these K clubs, each person still belongs to at least one club:** Then we can just add all the left m - k subsets to get a collection of eaxctly k subsets that covers U. □

5. Zero-Weight-Cycle Problem

Given a directed graph G = (V, E) with weights on its edges $e \in E$. The weights can be negative or positive. The Zero-Weight-Cycle Problem is to decide if there is a simple cycle in G so that the sum of the edge weights on this cycle is exactly 0. Prove that this problem is NP-complete by using the Subset-Sum Problem.
Prove the Zero-Weight-Cycle Problem is NP
Certificate: A simple cycle in G so that the sum of the edge weights on this cycle is exactly 0.
Certifier: Check if the cycle is simple, and the sum of the edge weights on this cycle is exactly 0. This can be done in polynomial time.
Subset-Sum $\leq_p$ Zero-Weight-Cycle: Given any Subset-Sum problem with a set of integers $a_1, \ldots, a_n$ and determine whether there is a subset of them that sums to 0. We can construct a weighted directed graph G with 2n vertices such that every integer a_i corresponds to two vertices v_i, u_i with an edge from v_i to u_i with weight a_i and an edge from u_i to all other v_i with weight 0. Then there is a subset of integers that sums to 0 if and only if there is a simple cycle in G so that the sum of the edge weights on this cycle is exactly 0.

Solution2: Given any Subset-Sum problem with a set of integers $a_1, \dots, a_n$ and determine whether there is a subset of them that sums to W. Construct G with nodes $0, 1, 2, \dots, n, n+1$ and an edge (i, j) with weight w_j for all pairs $i < j < n+1$, with weight 0 for all pairs $i < j = n+1$. Add a final edge from n+1 to 0 with weight -W.
Proof: If there is such a subset S, then we define a cycle that starts at node 0, **goes through the nodes in increasing order of indices correspond to elements in S**, then jumps from the last to node n+1 with weight 0 and then returns to node 0 with wegiht -W. The weight -W on the edge (n+1, 0) should precisely cancel the sum of the other edge weights since the elements in S summed to W.
Conversely, all cycles in G must use the edge (n+1, 0), and so if there is a zero-weight-cycle, then the other edges must exactly cancel -W, i.e. their indices must form a set that adds up to exactly W.

Proof. **Suppose there is a subset of integers that sums to 0:** Then we construct the cycle by picking all edges (v_i, u_i) corresponding to the integers in the subset. And connecting these edges by other 0 weighted edges to form a simple cycle.
Suppose there is a simple cycle in G so that the sum of the edge weights on this cycle is exactly 0: Then we can just pick the integers corresponding to the edges (v_i, u_i) on the cycle to form a subset that sums to 0, because the left edges are all 0 weighted and the sum of the cycle is 0. □