Design & Analysis of Algorithm
Homework 5

Due Date: 12/06/2020 11:45pm (Eastern)

1 Question

The Subset Sum problem is as follows: Given a set of integers S and an integer t, is there a subset $S^{'}$ of S such that the sum of all elements in $S^{'}$ is equal to t? The Equal Partition problem is as follows: Given a set of integers K, is it possible to split K into two sets such that two subsets have the same sum? Show that Equal Partition reduces to Subset Sum.

2 Question

Vertex Cover Problem: A vertex cover of an undirected graph is a subset of its vertices such that for every edge (u, v) of the graph, either u or v is in vertex cover. Given an undirected graph, the vertex cover problem is to find minimum size vertex cover.

Set Cover Problem: Given a set of elements $\cup$ (called the universe) and a collection S of m sets whose union equals the universe, the set cover problem is to identify the smallest subcollection of S union equals the universe.

For example, consider the universe $\cup = (1, 2, 3, 4, 5)$ and the collection of sets $S = (1, 2, 3), (2, 4), (3, 4), (4, 5)$. Clearly the union of S is $\cup$. However, we can cover all of the lements with the following smaller number of sets:(1,2,3),(4,5)

Prove the Vertex Cover problem reduces to the Set Cover Problem

3 Question

We discussed the Hamiltonian Cycle problem for undirected graphs. Define an equivalent problem, called D-Hamiltonian Cycle, for directed graphs (i.e., given a directed graph, does it contain a cycle that vists every node exactly once?). Show that D-Hamiltonian Cycle is NP-Complete.

4 Question

The Subset Sum problem is as follows: Given a set of numbers $S = (s_1, s_2, ..., s_n)$ and a target value t, is there some subset $S^{'}$ of such that the sum of numbers in $S^{'}$ is equal to t? We saw earlier that this problem is NP-Complete. Design a reasonable backtracking algorithm to solve the Subset Sum problem. As always, make sure to include enough detail for us to implement your algorithm.

5 Question

Consider the N Queens problem, where we had to place N queens on an N-by-N chess board such that no two queens were in the same row, column, or diagonal (i.e., no two queens are threatening each other). We designed a simple local search algorithm for this problem as follows:

First, because we know that each column can contain only one queen, assign each queen to her own column. Place each queen at random location within her column. Calculate the number of pairs of queens that are threatening each other (e.g., if Q is threatened by both Q_2 and Q_3, this counts as 2 threats). For each queen Q, consider moving queen Q, to a different location in the same column, and calculate the new number of conflicts that would exist if you performed that move. Once you have performed these calculations, perform the move that results in the greatest reduction in the current number of conflicts (subject to the constraint that each queen must remain in her assigned column). If there is no move that reduces the current number of conflicts, then terminate. Repeat until termination.

Give an example showing that this algorithm may fail to find a correct solution. (Hint: try $N = 4$. Show that (a) there is a solution in which all queens are safe, and (b) there is some starting position and some sequence of moves following the above description that fails to find such a solution.)