CS513 HW4

1. **Unit Ball Graph Dominating Set (3D):** Given a set of points in 3D, a unit ball graph is defined by connecting points of distance at most 1 away (a dominating set D is a subset of vertices such that every vertex in the graph is either in D or adjacent to a vertex in D). Design a polynomial time algorithm to approximate the minimum dominating set by a constant factor.

2. **Load Balancing (Greedy Makespan)**

   We are given m identical machines $M_1, \ldots, M_m$ and a set of n jobs. Each job j has a processing time $t_j > 0$. We need to assign each job to exactly one machine. Once assigned, a machine processes its jobs sequentially. Let L_i be the total load (sum of processing times) assigned to machine M_i. The objective is to minimize the **Makespan**, which is defined as the maximum load among all machines:

   $$\text{Makespan} = \max_{i=1\ldots m} L_i$$

   (Intuitively, we want to finish all jobs as early as possible, so we need to minimize the completion time of the busiest machine.)

   Now, consider the following greedy algorithm for this problem: Process the jobs in an arbitrary order $1, \ldots, n$. For each job j:

   1. Check the current load of every machine $M_1, \ldots, M_m$.
   2. Assign job j to the machine M_i that currently has the **minimum load**.
   3. Update the load of M_i: $L_i \leftarrow L_i + t_j$.

   Questions:

   (a) Show an example with $m = 2$ machines where this Greedy algorithm is not optimal.

   (b) What is the approximation ratio of this algorithm? Prove your answer.

3. **Maximum 3D Matching**

   Given disjoint sets X, Y, Z (each of size n) and a set $T \subseteq X \times Y \times Z$ of triplets. A subset $M \subseteq T$ is a *3D matching* if each element of X, Y, Z appears in at most one triplet in M. The goal is to find a 3D matching M of maximum size. Give a polynomial time algorithm to find a solution that is at least $1/3$ of the optimal solution.

4. **Facility Location (Supermarket Placement)**

   Given a set of n customers and a set of potential locations S for supermarkets, decide where to open the supermarkets to minimize a total cost function. The cost consists of two parts:

   - **Opening Cost:** A cost f_i if we open a market at location $s_i \in S$.
   - **Service Cost:** For each customer j, if they are served by a store at s_i, the cost is d_{ji}. Each customer connects to the closest open store.

   Design an $O(\log n)$ approximation algorithm for this problem.