Approximation Algorithms Assignment 2

1. Consider the following scheduling problem: there are n jobs to be scheduled on a single machine, where each job j has a processing time p_j, a weight w_j, and a due date $d_j, j = 1, \ldots, n$. The objective is to schedule the jobs so as to maximise the total weight of jobs that complete by their due date. First prove that there always exist an optimal schedule in which all on-time jobs complete before all late jobs, and the on-time jobs complete in the earliest due date order. Use this structural result to show how to solve this problem using Dynamic Programming in $O(nW)$ time, where $W = \sum_j w_j$. Now use this result to derive a fully polynomial-time approximation scheme.

2. Consider the following scheduling problem: there are n jobs to be scheduled on a constant number of machines m, where each job j has a processing time p_j, and a weight $w_j, j = 1, \ldots, n$. Once started, each job must be processed to completion on that machine without interruption. For a given schedule, let C_j denote the completion time of job $j, j = 1, \ldots, n$ and the objective is to minimise $\sum_j w_j C_j$ over all possible schedules. First show that there exists an optimal schedule where, for each machine, the jobs are scheduled in non-decreasing p_j/w_j order. Then use this property to derive a dynamic programming algorithm that can be used to obtain a fully polynomial-time approximation scheme.

3. In the *directed Steiner tree problem*, we are given as input a directed graph $G = (V, E)$, non-negative costs $c_{ij} \geq 0$ for arcs $(i, j) \in E$, a root vertex $r \in V$, and a set of terminals $T \subseteq V$. The goal is to find a minimum-cost tree such that for each $i \in T$ there is a directed path from r to i.

   It is NP-hard to approximate set-cover to a factor better than $c \log n$, for some constant c, where n is the number of elements. Use this fact to argue that for some constant d there can be no $d \log |T|$-approximation algorithm for the directed Steiner tree problem, unless P=NP.

4. The *k-suppliers problem* is similar to the k-center problem discussed in class. The input to the problem is a positive integer k, and a metric on a set of points $V, |V| = n$. However, now the points are partitioned into *suppliers* $F \subseteq V$ and customers $C = V \setminus F$. The goal is to pick k suppliers such that the maximum distance between a customer and its nearest picked supplier is minimized. In other words, we wish to find $S \subseteq F, |S| \leq k$ that minimizes $\max_{j \in C} d(j, S)$ where $d(j, S)$ is the distance between j and the nearest point in S. Give a 3-approximation algorithm for the k-suppliers problem.