Assignment 6

Vaibhav Agarwal(B15139)

Q1.

First ,sorts the jobs in decreasing order of processing time and then proceeds as before. Then the algorithm makes one pass through the jobs; when it comes to job j, it assigns j to the machine i whose load is smallest so far. It takes n*logn to sort the jobs then for each job, we find the maximum yet,which takes O(n) time. There are total n jobs so time complexity of SortedBalance for solving load balancing problem is **$O(n^2)$**

<u>Psuedocode:</u>

SortedBalance:

Start with no jobs assigned
Set $T_i = 0$ and $A(i) = \varnothing$ for all machines Mi
Sort jobs in decreasing order of processing times t_j
Assume that $t_1 \geq t_2 \geq ... \geq t_n$
For $j = 1, \ldots, n$:
Let M_i be the machine that achieves the minimum $\min_k T_k$
Assign job j to machine Mi
Set $A(i) \leftarrow A(i) \cup \{j\}$
Set $T_i \leftarrow T_i + t_j$ EndFor

Q2. We repeatedly select one of the original sites s as the next center, making sure that it is at least 2r away from all previously selected sites. To achieve essentially the same effect without knowing r, we can simply select the site s that is farthest away from all previously selected centers: If there is any site at least 2r away from all previously chosen centers, then this farthest site s must be one of them. For each center we have to check every pair of points in centers and points to find the maximum center and there are total k centers. Its time complexity is **$O(k*n^2)$**.

<u>Psuedocode:</u>

CenterSelection(S,k):

Assume $k \leq |S|$ (else define $C = S$)
Select any site s and let $C = \{s\}$
While $|C| < k$:
Select a site $s \in S$ that maximizes dist(s, C)
Add site s to C
EndWhile
Return C as the selected set of sites

Q3.

Given a graph G(V,E) and associated with each vertex having a weight, find a minimum weight vertex cover using Integer Linear Programming.

Assign a decision variable x_i for each node $i \in V$ to indicate whether to include node $i \in$ in the vertex cover or not, $x_i = 1$ being that node i is in the vertex cover, and $x_i = 0$ being that it is not.

We use linear inequalities to encode the requirement that the selected nodes form a vertex cover and the objective function to encode the goal of minimizing the total weight. For each edge (i, j) E, it must have one end in the vertex cover, and we write this as ∈ the inequality $x_i + x_j \geq 1$, for all pairs of nodes (i,j). The minimization problem is then to minimize the value $w^T x$, where w denotes the set of node weights as an n-dimensional vector and x denotes the n-dimensional vector formed by assigning x_i values for all nodes to its ith component. The main computation for the problem is done to solve the augmented matrix. The augmented matrix is of size $O(E^2)$. Solving the augmented matrix requires O(E) computation and each computation takes $O(E^2)$.Therefore, the total time of the algorithm is **$O(E^3)$**.