Assignment 7

Algorithms & Complexity (CIS 522-01)

Javier Arechalde

April 30, 2018

1. File transfer

Problem Model

In this problem we will have k network sources from source to destination, and n files that we want to transfer from source to destination. Each one of the files f_i has a length l_i. We want to find out how to assign the files to each one of the k routes so we can transfer the n files in the shortest time.

We will also assume in this problem that the network capacity is unlimited for each one of the networks.

Class

This algorithm belongs to the P class, as it can be implemented as an approximation greedy algorithm for load balancing.

Algorithm

In this algorithm, we will first sort the file lengths in decreasing order, and then we will start scheduling each one of the files to the less loaded network at each step. We will terminate this execution whenever we schedule all the files.

Pseudocode

Algorithm 1 File tranfers pseudocode

```
1: We first sort all the files f_i in decreasing order
2: while We didn't assign all the files do
3:     Assign file f_i to the least loaded network N_i
4: end while
5: return The list of the list assignments
```

Problem instance

An example of how to solve this problem can be found in the image attached to this assignment *Problem1.jpg*.

Time complexity

The time complexity of this implementation is $O(n \log m)$, as when we want to schedule each one of the jobs, we have to find which is the machine, or in this case the network, with the least load.

Approximation guarantee

The approximation guarantee for load balancing with list-scheduling is:

$$L \leq \frac{3}{2} L^*$$

2. Multiple Interval Scheduling

Problem Model

Class

3. Airport service

Problem Model

In this problem we have n airport sites, and m direct flights scheduled between these airports. Building a service facility that allow us to operate in some airport A_i has a cost C_i. We want to select a subset of airports in such a way that we can have all airports connected, at the minimum cost possible.

We will model this problem as a directed graph G, in which the airports will be the nodes, and the flights will be the edges. Also, each one of this nodes will have a cost associated C_i that will be the cost of building the service facility in that airport A_i.

Class

This problem belongs to class P, because it can be solved by an approximation method using a *greedy* approach.

Algorithm

To solve this problem, we will use a greedy vertex cover approach, or as it is usually named, the *pricing method*.

In this algorithm, we will first set the price paid for every edge to 0, and then , we will continuously increase the edges values, untill one of the nodes at one of the sides of the edge is 'tight', at that point we will add that node to the set of tight nodes S. We will then continue to look for edges that are not connected to tight nodes, to increase the edge value until one of the nodes that the edge is connected to becomes tight. We will continue to do this until we can't find an edge which is not connected to any tight node. Then we will return the set of tight nodes, which will be the solution to our problem. In our case the nodes returned would be the airports in which we should build the service facilities.

Pseudocode

Algorithm 2 Pricing method pseudocode

1: We first set $p_e = 0$ for all edges $e \in E$
2: **while** There is an edge $e = (i, j)$ such that neither i nor j is tight **do**
3: We selet an edge e
4: We increase p_e without violating fairness
5: **end while**
6: **return** S which is the set of all tight nodes

Problem instance

A small problem instance of this algorihtm can be found in *Figure 11.8* in Algorithm Design book page 622.

Time complexity

The time complexity of this algorithm will be $O(m)$, where m is the number of edges, or number of direct flights. As in the worst case scenario, we will have to go through all edges to find all the tight nodes.

Approximation guarantee

The approximation guarantee for this algorithm is:

$$w(S) \leq 2 \sum_{e \in E} p_e \leq 2w(S^*)$$

4. Telecommunication company

Problem Model

In this problem, we have to build base stations in various locations to provide mobile phone service. We have houses located in n different locations. We want to select k locations to build k base stations so that the maximum distance from any ouse to its closest station is minimized.

Class

This algorithm belongs to P class, as it can be implemented as an approximation algorithm with a greedy approach.

Algorithm

In this algorithm, we will start by setting one of the houses in the n locations to buold the base station, then we will build the next location on the house that is further from the set of centers or base stations. We will continue doing this until we can't place any more base stations.

Pseudocode

Algorithm 3 Center selection pseudocode

We assume that the number of centers is fewer that the number of points
Otherwise, our centers will be placed in top of our locations
We first select any site s and $C = s$
while $|C| < k$ **do**
    Select a site $s \in S$ that maximizes $dist(s, C)$
    Add site s to C
end while
return C as the selected set of sites

Problem instance

An example of how to solve this problem can be found in the image attached to this assignment *Problem4.jpg*.

Time complexity

The time complexity of this algorithm will be $O(k)$, where k is the number of points that we are looking to place.

Approximation guarantee

This greedy algorithm returns a set C of k points such that $r(C) \leq 2r(C*)$, where $C*$ is an optimal set of k points.

5. Job requests

Problem Model

Class

6. City roads monitoring

Problem Model

We will model this problem as a graph G, in which the nodes will be the crossroads, and the edges will be the roads.

Class

As k is small in this problem, we can solve this vertex cover problem in polynomial time. Then this problem belongs to P.

Algorithm

Our algorithm will check all the possible subsets of V of size k, and check if any of them is a vertex cover. This algorithm implementation will be based on brute force.

Pseudocode

Algorithm 4 Vertex cover pseudocode

```
1: while There is a subset S of V of size k that we didn't check yet do
2:     We check if subset S is a vertex cover in our graph
3:     if It's a vertex cover then
4:         return Vertex cover S
5:         Break the loop
6:     else if It's not a vertex cover then
7:         We continue with the next possible subset
8:     end if
9: end while
```

Problem instance

An example of how to solve this problem can be found in the image attached to this assignment *Problem6.jpg*.

Time complexity

There are $\binom{n}{k}$ possible subsets, and as it takes $O(kn)$ time to check wether a subset is a vertex cover or not, the time complexity of our algorithm will be $O(kn^{k+1})$