

27. Give a polynomial time algorithm for the following problem. The input consists of two dimensional array R of non-negative integers, and an integer k . The value $R_{t,p}$ gives the number of of users requesting page p at time t (say from a www server. At each integer time, the server can broadcast either 0 or 1 pages. If the server broadcasts page p at time t , the the requests of all the users who requested page p strictly before time t are satisfied. The server can make at most k broadcasts. The goal is to pick the k times to broadcast, and the pages to broadcast at those k times, in order to minimize the total time (over all requests) that requests/users have to wait in order to have their requests satisfied. So problem 24 is a special case of this problem where there is only one page.

Tree Description:

All of the nodes in the tree are stored as [time t , broadcasts used b , page currently broadcasted p]. The nodes store the values as [total wait time W , wait time since last broadcast w]. Each of the children of any node represent the decision to either broadcast no page or broadcast one of the requested pages. The tree is rooted at $[0, 0, 0]$ representing the state in which no pages have been broadcast yet and no requests have come in yet.

Pruning Rules: Algorithm:

3. Show that if there is an $O(n^k)$, $k \geq 1$, time algorithm for squaring a degree n polynomial, then there is an $O(n^k)$ time algorithm for multiplying two degree n polynomials. Assume that the polynomials are given by their coefficients.

Polynomial Multiplication \leq Polynomial Squaring

Program Polynomial Multiplication:

```
read  $I, J$ 
 $P = I + J$ 
 $Q = \text{Polynomial Square}(P)$ 
 $Q = Q - I^2 - J^2$ 
output  $Q$ 
```

Constructing Q takes linear time since if $I = [i_1, i_2, \dots, i_n]$ $J = [j_1, j_2, \dots, j_n]$ $Q = [i_1 + j_1, i_2 + j_2, \dots, i_n + j_n]$.

4. Consider the following variant of the minimum Steiner tree problem. The input is n points in the plane. Each point is given by its Cartesian coordinates. The problem is build a collection of roads between these points so that you can reach any city from any other city and the total length of the roads is minimized. The collection of roads should be output as an adjacency list structure, that is, for each point p of intersection of roads there are a list of roads that meet that point. Roads must terminate when they meet one of the original input points, or when they meet another road. For example, if the points are $(1, 1)$, $(-1, 1)$, $(1,-1)$, $(-1,-1)$, $(2, 2)$ the output would be: From the point $(0, 0)$ there are roads to the points $(1, 1)$, $(-1, 1)$, $(1,-1)$, and $(-1,-1)$. From the point $(1, 1)$ there are roads to $(0, 0)$ and $(2, 2)$. From the point $(-1, 1)$ there is a road to $(0, 0)$. From the point $(1,-1)$ there is a road to $(0, 0)$. From the point $(-1,-1)$ there is a road to $(0, 0)$. Roads from $(1, 1)$ to $(-1,-1)$, and from $(1,-1)$ to $(-1, 1)$ would not be allowed because they cross. A road from $(-1,-1)$ to $(2, 2)$ would not be allowed since it would cross the point $(1, 1)$. Show by reduction that if you can solve this problem in linear time, then you can sort n numbers in linear time.

Extra Credit

22. The input to this problem is n points x_1, \dots, x_n on a line. A good path P has the property that one endpoint of P is the origin and every x_i is covered by P . Note that P need not be simple, that is, it can backtrack over territory that it has already covered. Assume a vehicle moves along this path from the origin at unit speed. The response time r_i for each x_i is the time until the vehicle first reaches x_i . The problem is to find the good path that minimizes $\sum_{i=1}^n \frac{r_i}{n}$, the average response time. For example, if the points are $x_1 = 1$, $x_2 = 8$ and $x_3 = -2$ and the path visited the points in the order x_1, x_3, x_2 , the average response time for this path would be $1/3 + (1 + 3)/3 + (1 + 3 + 10)/3$. Give a polynomial time algorithm for this problem.