Midterm practice problems

Note: A *feasible algorithm* is an algorithm which runs in polynomial time, i.e., such that there exists a fixed positive integer k (thus independent of the input size n) such that $T(n) = O(n^k)$ where n is the length of the input instance of the problem which that algorithm solves. If the required run time of the algorithm which you are asked to design is not specified, it means that any feasible algorithm will do. Note that this excludes brute force solutions which run in exponential time $T(n) = \Theta(2^n)$ (or worse).

1. Assume you are given two arrays A and B, each containing n distinct integers and equation $x^8 - x^4y^4 = y^6 + x^2y^2 + 10$. Design an algorithm which runs in time $O(n \log n)$ which finds if A contains a value for x and B contains a value for y that satisfy the equation.

2. Assume that you are given an array A containing $2n$ numbers. The only operation that you can perform is make a query if element $A[i]$ is equal to element $A[j]$, $1 \leq i, j \leq 2n$. Your task is to determine if there is a number which appears in A at least n times using an algorithm which runs in linear time. (Warning: a tricky one. The reasoning resembles a little bit the reasoning used in the celebrity problem: try comparing them in pairs and first find one or at most two possible candidates and then count how many times they appear.)

3. Let M be an $n \times n$ matrix of distinct integers $M(i, j)$, $1 \leq i \leq n$, $1 \leq j \leq n$. Each row and each column of the matrix is sorted in the increasing order, so that for each row i, $1 \leq i \leq n$,

   $$M(i, 1) < M(i, 2) < \ldots < M(i, n)$$

   and for each column j, $1 \leq j \leq n$,

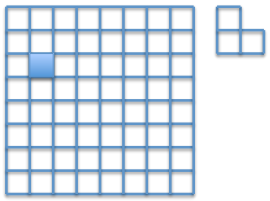
   $$M(1, j) < M(2, j) < \ldots < M(n, j)$$

   You need to determine whether M contains a given integer x in $O(n)$ time.

4. Assume you have an array of $2n$ distinct integers. Find the largest and the smallest number using $3n - 2$ comparisons only.

5. Assume that you have an array of 2^n distinct integers. Find the largest and the second largest number using only $2^n + n - 2$ comparisons.

6. You are given a $2^n \times 2^n$ board with one of its cells missing (i.e., the board has a hole); the position of the missing cell can be arbitrary. You are also given a supply of "dominoes" each containing 3 such squares; see the figure below. Your task is to design an algorithm for covering the entire board with such "dominoes", except for the hole which should remain uncovered.



7. Multiply the following pairs of polynomials using at most the prescribed number of multiplications where both numbers multiplied are large (large numbers are those which depend on the coefficients and thus can be arbitrarily large).

   (a) $P(x) = a_0 + a_2x^2 + a_4x^4 + a_6x^6$ and $Q(x) = b_0 + b_2x^2 + b_4x^4 + b_6x^6 + b_8x^8$ using at most 8 multiplications of large numbers;

   (b) $P(x) = a_0 + a_{100}x^{100}$ and $Q(x) = b_0 + b_{100}x^{100}$ with at most 3 multiplications of large numbers.

9. Describe all k which satisfy $i\,\omega_{64}^{13}\omega_{32}^{11} = \omega_{64}^{k}$ (i is the imaginary unit).

10. Consider the polynomial

$$P(x) = (x - \omega_{64}^{0})(x - \omega_{64}^{1})(x - \omega_{64}^{2}) \dots (x - \omega_{64}^{63})$$

    (a) What is the degree of $P(x)$? What is its coefficient of the highest degree of x present in $P(x)$?

    (b) What are the values of $P(x)$ at the roots of unity of order 64?

    (c) Can you represent $P(x)$ in the coefficient form without any computation? Think which polynomial shares the same roots and the same leading coefficient.

11. Describe how you would compute all elements of the sequence $F(0), F(1), F(2), \dots, F(2n)$ where

$$F(m) = \sum_{\substack{i+j=m \\ 0 \le i,j \le n}} \log(j+2)^{i+1}$$

    in time $O(n \log n)$.

12. In Elbonia coin denominations are 81c, 27c, 9c, 3c and 1c. Design an algorithm that, given the amount that is a multiple of 1c, pays it with a minimal number of coins. Argue that your algorithm is optimal.

13. Give an example of a set of denominations containing the single cent coin for which the greedy algorithm does not always produce an optimal solution.

14. Assume you are given n tasks each of which takes the same, unit amount of time to complete. Each task i has an integer deadline d_i and penalty p_i associated with it which you pay if you do not complete the task in time. Design an algorithm that schedules the tasks so that the total penalty you have to pay is minimized.

15. There is a line of 111 stalls, some of which need to be covered with boards. You can use up to 11 boards, each of which may cover any number of consecutive stalls. Cover all the necessary stalls, while covering as few total stalls as possible.

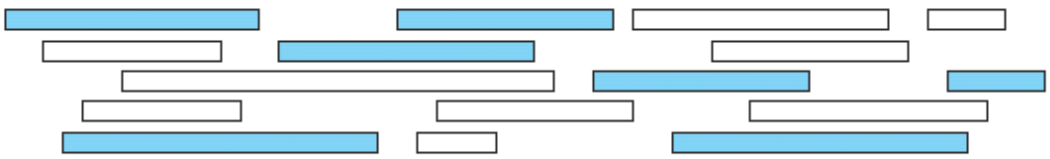
16. You are running a small manufacturing shop with plenty of workers but with a single milling machine. You have to produce n items; item i requires m_i machining time first and then p_i polishing time by hand. The machine can mill only one object at a time, but your workers can polish in parallel as many objects as you wish. You have to determine the order in which the objects should be machined so that the whole production is finished as quickly as possible. Prove that your solution is optimal.

17. You are given a set S of n overlapping arcs of the unit circle. The arcs can be of different lengths. Find a largest subset P of these arcs such that no two arcs in P overlap (largest in terms of total number of elements, not in terms of total length of these arcs). Prove that your solution is optimal.

18. You are given a set S of n overlapping arcs of the unit circle. The arcs can be of different lengths. You have to stab these arcs with minimal number of needles so that every arc is stabbed at least once. In other words, you have to find a set of as few points on the unit circle as possible so that every arc contains at least one point. Prove that your solution is optimal.

19. Let X be a set of n intervals on the real line. A subset of intervals $Y \subseteq X$ is called a tiling path if the intervals in Y cover the intervals in X, that is, any real value that is contained in some interval in X is also contained in some interval in Y. The size of a tiling cover is just the number of intervals. Design and estimate the time complexity of an algorithm to compute the smallest tiling path of X as quickly as possible. Assume that your input consists of two arrays $X_L[1..n]$ and $X_R[1..n]$, representing the left and right endpoints of the intervals in X.

20. A photocopying service with a single large photocopying machine faces the following scheduling problem. Each morning they get a set of jobs from customers. They want to do the jobs on their single machine in an order that keeps their customers happiest. Customer i's job will take t_i time to complete. Given a schedule (i.e., an ordering of the jobs), let C_i denote the finishing time of job i. For example, if job j is the first to be done we would have $C_i = t_j$, and if job j is done right after job i, we would have $C_j = C_i + t_j$. Each customer i also has a given weight w_i which represents his



A set of intervals. The seven shaded intervals form a tiling path.

or her importance to the business. The happiness of customer i is expected to be dependent on the finishing time of is job. So the company decides that they want to order the jobs to minimize the weighted sum of the completion times, $\sum_{i=0}^{n} w_i C_i$. Design an efficient algorithm to solve this problem. That is, you are given a set of n jobs with a processing time t_i and a weight w_i for job i. You want to order the jobs so as to minimize the weighted sum of the completion times, $\sum_{i=0}^{n} w_i C_i$.

21. You are given n points x_i $(1 \leq i \leq n)$ on the real line and n intervals $I_j = [l_j, r_j]$, $(1 \leq j \leq n)$. Design an algorithm which runs in time $O(n^2)$ and determines if each point x_i can be assigned to a distinct interval I_j so that $x_i \in I_j$.

22. You are given a connected graph with weighted edges. Fins a spanning tree such that the largest weight of all of its edges is as small as possible.

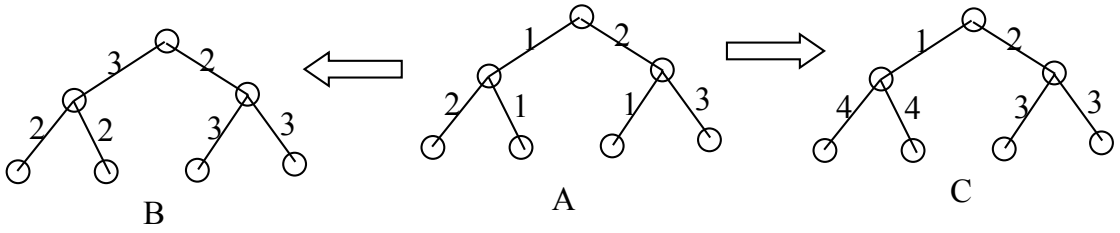
23. You need to write a very long paper "The meaning of life". You compiled a sequence of books in the order you will need them, some of them multiple times. Such a sequence might look something like this:

$$B_1, B_2, B_1, B_3, B_4, B_5, B_2, B_6, B_4, B_1, B_7, \ldots$$

Unfortunately, the library lets you keep at most 10 books at home at any moment, so every now and then you have to make a trip to the library to exchange books. On each trip you can exchange any number of books (of course, between 1 and all of 10 books you can keep at home). Design an algorithm which decides which books to exchange on each library trip so that the total number of trips which you will have to make to the library is as small as possible.

24. Timing Problem in VLSI chips. Consider a complete balanced binary tree with $n = 2^k$ leaves. Each edge has an associate positive number that we call the length of this edge (see picture below). The distance from the root to a leaf is the sum of the lengths of all edges from the root to this leaf. The root sends a clock signal and the signal propagates along the edges and reaches the leaf in time proportional to the distance from the root to this leaf. Design an algorithm which increases the lengths of some of the edges in the tree in a way that ensures that the signal reaches

all the leafs in the same time while the sum of the lengths of all edges is minimal. (For example, on the picture below if the tree A is transformed into trees B and C all leaves of B and C are on the distance 5 from the root and thus receive the clock signal in the same time, but the sum of lengths of edges in C is 17 while sum of lengths in B is only 15.)



25. Alice wants to throw a party and is deciding whom to call. She has n people to choose from, and she has made up a list of which pairs of these people know each other. She wants to pick as many people as possible, subject to two constraints: at the party, each person should have at least five other people whom they know and at least five other people whom they do not know. Give an efficient algorithm that takes as input the list of n people and the list of all pairs who know each other and outputs a subset of these n people which satisfies the constraints and which has the largest number of invitees. Ague that your algorithm indeed produces a subset with the largest possible number of invitees.