Algorithms

Exercises

1 Introduction

1. Prove that the solution presented in class for finding the maximal sum of consecutive elements of an array is correct.

2. An array of length n of positive numbers is given. Suggest an algorithm with $O(n)$ runtime to find the maximal possible product of (any number of) consecutive entries of the array.

2 The Greedy Algorithm

3. Consider the knight tour problem, discussed in class, on any rectangular $m \times n$ chessboard.

(a) Find an infinite set of pairs (m, n), for each of which there exist starting points from which the tour cannot be completed.

(b) Conclude that there exists an infinite set of pairs (m, n), for each of which the problem does not admit any solution in which the knight can move from the final square to the initial one.

4. An $m \times n$ chessboard is given, from which two opposite corners have been cut out. For which pairs (m, n) is it possible to cover the board by domino tiles? (Each tile covers two adjacent squares of the board, and each square should be covered by a single tile.)

5. Consider a system consisting of n particles, distributed between K sites. If n_i of the particles are at site i, $1 \leq i \leq K$ (where

$\sum_{i=1}^{K} n_i = n$), then the energy of the system is $\sum_{i=1}^{K} w_i n_i^2$, where the w_i's are given positive constants. How are the particles distributed between the sites if the energy of the system is minimal? (Hint: Use a greedy approach. Namely, distribute the particles between the sites one at a time, and at each step decide greedily where to place the next particle. Prove that you indeed achieve the minimal energy configuration.)

6. Consider the problem, discussed in class, of a server who needs to satisfy n orders of known duration times. Suppose that, instead of a single server, there are two servers who work simultaneously. Suggest a greedy algorithm for solving the problem and prove its optimality.

7. A device contains n programs of lengths $L_1, L_2, \ldots, L_n$. The relative frequency with which program i is employed is p_i, $1 \leq i \leq n$. Each time when a program is to be used, it is necessary to start reading the device from the beginning. (The reading is done at some constant speed.) Thus, if the programs reside on the device at some order $i_1, i_2, \ldots, i_n$, then the (weighted) average time required for using a program is

$$\bar{T} = C \sum_{j=1}^{n} p_{i_j} \sum_{k=1}^{j} L_{i_k} ,$$

for an appropriate constant C.

(a) Show that $\bar{T}$ is minimal if the programs are ordered according to decreasing order of the ratios p_i/L_i.

(b) Show that by either ordering the programs according to decreasing order of the p_i's or according to increasing order of the L_i's, we do not necessarily minimize $\bar{T}$.

8. Consider the problem, discussed in class, of a server who needs to perform certain jobs of one unit time duration, each having some deadline. Show that a set of jobs is acceptable if and only if, when ordering them by decreasing order of profit, and scheduling each job in turn for the latest possible time, given its deadline and the scheduling of earlier jobs, we obtain an acceptable ordering.

3 Divide-and-Conquer

9. Suggest a non-recursive algorithm for finding both the minimum and the maximum of a set of size n using $\lceil 3n/2 \rceil - 2$ comparisons.

10. Suggest an algorithm for finding the second least element of a set of size n using $n + \lceil \log_2 n \rceil - 2$ comparisons. (Hint: Find the least element also.)

11. In the selection algorithm for finding the k-th element of a set, we divided the set into subsets of five elements each.

(a) Why did we not divide the set into subsets of three elements each?

(b) Would the algorithm work with similar runtime if we divided the set into subsets of size seven?

12. As is well known, the `Quicksort` algorithm works in $O(n \log n)$ average time, but takes $O(n^2)$ time in the worst case. How can you improve it so that it will work in $O(n \log n)$ in the worst case?

13. A non-decreasing sequence $(a_n)_{n=1}^{\infty}$ satisfies $a_n \leq a_{\lceil \alpha_1 n \rceil} + \ldots + a_{\lceil \alpha_r n \rceil} + Cn$, where $\alpha_1 + \ldots + \alpha_r < 1$ and C is some constant. Show that there exists a constant K such that $a_n \leq Kn$ for each n.

4 Dynamic Programming

14.

(a) Bound the Catalan numbers from above and below by showing in an elementary way that the middle binomial coefficients $\binom{2n}{n}$ satisfy the inequalities

$$\frac{4^n}{2n+1} \leq \binom{2n}{n} \leq 4^n \,.$$

(b) Estimate the Catalan numbers employing Stirling's approximation for $n!$.

15. Recall the algorithm, presented in class, for determining whether a group of experts should be augmented by two decision makers. The main issue there was calculating $S(k)$ and $S(k-1)$ for $k = \frac{n+1}{2}$, where $S(r) = \sum_{1 \leq i_1 < \ldots < i_r \leq n} \alpha_{i_1} \ldots \alpha_{i_r}$, where $\alpha_i = \frac{p_i}{q_i}$.

The other problem mentioned there was to determine the probability of the majority rule to be correct, which is the sum of the probabilities of all subsets of $\frac{n+1}{2}$ or more experts being correct. For example, when $n = 3$, it is $P_M(3) = p_1p_2q_3 + p_1q_2p_3 + q_1p_2p_3 + p_1p_2p_3$. Note that the first three terms express the probability that exactly

two out of the three experts are correct, and the last term represents the probability of all experts being correct. Suggest a procedure for calculating $P_m(n)$ with running time $O(n^2)$.

16. A relational database (such as MS-Access, SQL-server, Oracle) holds its data in tables. A table may be thought of as a file containing records of the same structure. For example, a university database holds its data in several tables, such as: lecturer table, course table, courses-taught-by-lecturer table, and some other tables.

Frequently, users pose queries to the database, queries which require gathering information from several tables, based on some criteria. An important query operator is the (natural) *join*. Let T_1 and T_2 be tables. Their join, denoted by T_1T_2, is a new table, in which each record is a concatenation of a record from T_1 and a record from T_2, depending on whether they have a common attribute having the same value (for example, the same i.d. number).

The result may vary from a null set of records to the whole cartesian product, depending on the data given to the query and the values of the appropriate attribute. A query may be composed of join operators among many tables, simply by performing the first join between some two tables (resulting in a new table), then performing a join between two of the tables we have now (i.e., two of the original tables or the newly-created table and one of the original tables), and so on. The order in which the joins are performed does not change the result. For example, let T_1, T_2, T_3 and T_4 be tables. Then the calculations of $((T_1T_2)T_3)T_4$, $(T_1T_2)(T_3T_4)$, $(T_2T_1)(T_3T_4)$ and all other arrangements will yield the same result. However, they may vary considerably in the amount of time needed to perform. Thus, a database employs a query analyzer to determine, among all possible arrangements, which is likely to consume the least amount of time.

We are required to form the join of n given tables $T(1), \ldots, T(n)$. Let a_n denote the number of essentially distinct ways of calculating this join.

(a) Find a recurrence equation and initial conditions determining the sequence $(a_n)_{n=1}^{\infty}$.

(b) Show that the sequence grows faster than any exponential.

(c) Prove that $a_n = 2^{n-1}(2n-3)!!$ $(= 2^{n-1} \cdot 1 \cdot 3 \cdot 5 \cdot \ldots \cdot (2n-3))$ for $n \geq 2$.

(d) Suggest an algorithm for calculating the optimal way for performing the join, which requires only exponential time. (For the purposes of this exercise, assume that we are able to compute in advance, within almost negligible time, how long it takes to perform the join between any two tables, obtained by joining any two sets of $T(i)$'s.)

17. Prove that, given a tree and any vertex of the tree, there exists a path which starts and ends at that vertex and passes through each edge of the tree exactly twice (in opposite directions).

18. Prove that, given a graph with a Euclidean matrix of distances, the traveling salesman problem cannot possibly have a solution with a strictly smaller total distance if we allow paths passing through some vertices several times.

5 The Fast Fourier Transform

19. Prove that, given distinct real numbers $x_1, x_2, \ldots, x_n$ and any real numbers $y_1, y_2, \ldots, y_n$, there exists at most one polynomial P of degree not exceeding $n-1$ such that $P(x_i) = y_i$ for $1 \leq i \leq n$. (Hint: The conditions on the required polynomial lead to a system of linear equations, whose coefficient matrix is a Vandermonde.)

20. The *Cartesian sum* $A+B$ of two multisets A, B of real numbers is the multiset consisting of all sums of the form $a+b$ with $a \in A, b \in B$. Here the multiplicity of each element of $A+B$ is the number of possibilities of writing it in the above form. Explain how you can calculate the cartesian sum of two multisets of integers. Your algorithm should work in time $O(n \log n)$ for multisets with up to n distinct elements in each, all being bounded in absolute value by cn for some arbitrary fixed constant c.

21. Design an algorithm which, given n (not necessarily distinct) numbers $x_1, x_2, \ldots, x_n$, calculates the polynomial $\prod_{i=1}^{n}(x - x_i)$ in time $O(n \log^2 n)$.

6 Introduction to Cryptography

22. We are given two affine transformation ciphers designed for English texts:

$$\begin{aligned} E_1(x) &= ax + b \bmod 26, \\ E_2(x) &= cx + d \bmod 26. \end{aligned}$$

Recall that this implicitly means that $\gcd(a, 26) = \gcd(b, 26) = 1$.

(a) Explain what goes wrong if a (or c) is not assumed to be relatively prime to 26. Provide an example to demonstrate your claim.

(b) What is the size of the set of all possible keys for E_1?

(c) Consider the encryption scheme given by $E_3 = E_1(E_2(x))$. Is E_3 a better encrypting function than any of the others? If so – what is the size of the set of all possible keys for E_3? If not – explain why.

23. In a certain language, the alphabet consists of the three letters $0, 1$ and 2, and the frequencies are $0.5, 0.2$ and 0.3, respectively. Find the expected value of the index of coincidence for texts of length N in the language, as well as for texts of the same length, encrypted by Vigenere's cipher with the key $(2, 1)$.

24.

(a) Prove that, when using any polyalphabetic encryption system, the expected value of the index of coincidence for encrypted texts does not exceed that of regular texts.

(b) Find a language (namely, an alphabet and frequencies p_i of letters) such that there exist non-trivial Vigenere ciphers which do not reduce the expected value of the index of coincidence.

(c) Prove that for every language there exists a Vigenere cipher such that the expected value of the index of coincidence for encrypted texts is $1/n$ (where n is the size of the alphabet).

25. The file `ciphertext1.txt` contains a text, enciphered using Vigenere's cipher.

(a) Employ Kasiski's method to find all possible values for the period of the cipher.

(b) Find the value of the index of coincidence to estimate the period of the cipher.

(c) Decipher the message.

(d) Find the value of the index of coincidence for the original text.