NPTEL Questions

CP — July-Nov 2021

Week 1 — Ad-hoc and Implementation

1. (2 points) This question is about the reversort and engineering reversort problems discussed in class.

   (a) What is the cost of reversorting an array of numbers in descending order, i.e, the reverse of the desired sorted order? For example, the cost of reversorting the array $[7, 6, 5, 4, 3, 2, 1]$ is 12. Let N denote the length of the array, and assume that $N > 1$.

   A. $\frac{N(N+1)}{2} - 1$ B. $2N$ C. $2N - 1$ **D.** $2N - 2$

   Explanation: The first operation has cost N. Since this already sorts the array, each of the remaining $N - 2$ operations have cost one each, bringing the total cost to $2N - 2$

   (b) For which of the following combinations of values of C and N can we design an array of length N that has cost C?

   A. $N = 10, C = 8$ B. $N = 10, C = 55$ **C.** $N = 10, C = 25$ **D.** $N = 10, C = 9$

   Explanation: Recall that $N - 1 \leq C \leq \frac{N(N+1)}{2} - 1$.

2. (4 points) We define an array *a* of positive numbers to be MGOOD if it is not possible to split it into two subsequences such that the product of the elements in the first subsequence is equal to the product of the elements in the second subsequence and SGOOD if it is not possible to split the elements of the array into two subsequences such the sum of the elements in the first subsequence is equal to the sum of the elements in the second subsequence.

   A subsequence *s* of an array *a* is obtained by removing one or more elements from it. (Note that the size of a subsequence, as defined here, will never be zero.)

   Note: The subsequences obtained after splitting the array are mutually exclusive and exhaustive. Eg: For the array [1, 2, 2, 3]: {[1, 2], [2, 3]} is a valid split, however {[1, 2], [3]} and {[1, 2, 3], [2, 3]} are not.

   (a) Which of the following arrays are MGOOD?

   A. [1, 2, 3, 4, 6]

   B. [4, 4, 4]

   C. [4, 4, 4, 4]

   D. [1, 2, 3, 4, 5, 6]

   Explanation: For a) we can come up with the partition [1, 3, 4] and [2, 6]. For c) [4, 4] and [4, 4]. For the other two, such partitions are not allowed (in option (d) there is only one power of 5).

   (b) If the product of all the elements in a given array is not a perfect square, the array is necessarily MGOOD.

   A. True

   B. False

   Explanation: If the product of the array is not a perfect square, we can never achieve two partitions whose individual products are equal. Since, in that case, the product of the products of the partitions (also product of the entire array) would be a perfect square.

   (c) If the product of all the elements in a given array is a perfect square, the array is necessarily not MGOOD.

   A. True

   B. False

   Explanation: An example where the array is MGOOD despite the product of it being a perfect square- [2, 8, 32, 512]

   (d) Given an array *a* such that it is **not** SGOOD. What could the possible strategies to make this array SGOOD. You are allowed to remove some elements from the array. Specifically, you are expected to give the indices of the elements you will remove to make the array SGOOD.

   A. Remove all the smallest number in the array.

B. Remove all even numbers from the array.

C. Keep dividing all the elements of the array by 2, until an odd number appears. Store the index of any one odd number. Remove the number at this index in the original array.

D. Find the GCD (Greatest Common Divisor) of all numbers in the array, divide all elements by the GCD, and store indices of all even elements. Remove all numbers at these indices in the original array.

E. Find the GCD (Greatest Common Divisor) of all numbers in the array, divide all elements by the GCD, and store the index of any odd number. Remove the number at this index in the original array.

Explanation: Analogous to MGOOD, the sum of elements in a non-SGOOD array is even. Similarly, an array with odd sum of elements is necessarily SGOOD. To make a non-SGOOD array SGOOD, we can simply remove any odd element, thus making the sum of elements odd. If all elements are even, we can always factor out 2 from all sums, on both partitions. This is the same as dividing the array by 2. We keep repeating this until we find an odd element and we can no longer do this. This odd number can be removed to make the sum odd.

3. (3 points) You are required to construct an array of some size $N \geq 2$, such that it satisfies the following two properties-

1. All numbers in the array are pairwise non-coprime, that is,

$$GCD(a_i, a_j) \neq 1$$

$\forall i \neq j$, where GCD stands for greatest common divisor.

2. The GCD of all the numbers of the array is 1, that is,

$$GCD(a_1, a_2, \ldots, a_n) = 1$$

Answer the following questions-

(a) We can add a prime number as one of the N numbers.

A. True

B. False

Explanation: If we have a prime number at some position, and we satisfy the first condition, the GCD of all numbers in the array will be that prime number. We cannot satisfy both conditions at the same time if any one of the numbers is prime.

(b) Consider a number of the form

$$x = p_1^{\alpha_1} p_2^{\alpha_2} \ldots p_k^{\alpha_k}$$

where each p_i is a distinct prime number. What is the minimum value of k for which we may enter a valid x? Note that a "valid x" is any one of the N numbers that together satisfy the above two conditions.

A. 1

B. 2

C. $N - 1$

D. N

E. None of the above

Explanation: As explained in the previous question, we cannot have a single prime factor for any number. We can, however, have numbers as products of two prime numbers. For example, for $N =$ 3, this is a valid solution: [6, 10, 15] = [2 * 3, 2 * 5, 3 * 5].

(c) Let $p_1, p_2, \ldots, p_N$ be the first N prime numbers. Which of the following strategies would work ($N \geq 3$)?

A. Choose the i^{th} number as the product of the first i prime numbers.

B. Choose the first $N-1$ numbers as $p_1 * p_2, p_1 * p_3, p_1 * p_4, \ldots, p_1 * p_N$, and the N^{th} number as $p_2 * p_3 * p_4 * \cdots * p_N$

C. Choose the i^{th} number as p_i^i. Therefore the numbers chosen will be $p_1, p_2^2, p_3^3, \ldots, p_N^N$

D. Let $P = p_1 * p_2 * \cdots * p_N$, the N numbers can then be chosen as follows, $\frac{P}{p_1}, \frac{P}{p_2}, \ldots, \frac{P}{p_N}$.

Explanation: In a) the first number will be a prime, which does not work. b) works, since the GCD of any of the first $N-1$ numbers is p1, GCD of the N^{th} number with any other, say i^{th} number is p_{i+1}. However, the GCD of all numbers together is 1, since there is no prime that appears in all of the N numbers. In c), again the first number is a prime. d) works, since GCD of the i^{th} and j^{th} numbers is $g_{ij} = \frac{P}{p_i p_j} \neq 1$, however GCD of all numbers is 1, since the prime number p_i is missing in the i^{th} number.

4. (1 point) There are n pairs of old shoes that have been been kept in a straight line in your cupboard, in other words, there are $2n$ shoes altogether. You want to give them all away, but because you are superstitious, you will give away a pair only if both shoes are next to each other. It doesn't matter what the order between them is (the left shoe could be next to the right one or the other way round). You can take one of two possible actions:

   - You can swap any two shoes that are kept next to each other in the line.
   - If a pair of shoes are kept next to each other in the line, you can give them away. The remaining shoes are then shifted to fill the gap in the line.

   The actions can be performed in any order. For instance, you can make some swaps, then give away some shoes, and then go back to making swaps.

   Whenever you have two pairs of shoes say (X_0, X_1) and (Y_0, Y_1) which are kept in the row like this:

$$X_b \cdots Y_b \cdots X_{1-b} \cdots Y_{1-b}, \text{ where } b \in \{0,1\}$$

   you will, sooner or later, have to swap the inner Y and X. Before doing so, you won't be able to give away either the X-pair or the Y-pair away. Let k be the number of pairs which overlap in this way. What is the optimal number of actions that you need to perform to give away all pairs?

   A. $n+k$ B. $n+2k$ C. $2k$ D. k^2 E. Depends on the instance

   EXPLANATION. It is clear that at least $n+k$ actions are necessary. We will now show that there is always a solution in which the number of swap actions equals the number of overlapping pairs. Suppose we have a row of shoes. Let's find a pair (Z_0, Z_1) that's currently closest to each other, that is, the two shoes are at indices p and q and $|q-p|$ is as small as possible. From the minimality of $|q-p|$, it follows that the shoes who are standing between positions p and q must belong to distinct shoes, that is, there is no pair of shoes for which both shoes are in the interval between p and q. Each of those pairs of shoes therefore has an overlap with our chosen pair. We can now use swaps to bring the pair (Z_0, Z_1) together and give it away. The number of swaps we made is exactly equal to the number of overlaps that involved our pair of shoes. Once we are done, the other shoes are still in the same relative order, so the other overlapping pairs remained unchanged. Clearly, repeating this process will give us a valid solution, and the total number of swaps made will be exactly equal to the total number of overlapping pairs of shoes.