Problem Set 3 Solutions

This problem set covers the material of the third quarter of the course: lectures 13-19. This problem set is not due and it will not be graded, but you are welcome to come to office hours to discuss your solutions.

---

1. Suppose that we are given the use of two 3-D printers for T minutes each, and that we have n objects that we would like to print, to sell later. Object i would take t_i minutes to print, and would earn us v_i Euro in profit once sold. Devise an algorithm that runs in time polynomial in T and n and that, Given T, $t_1, \ldots, t_n$ and $v_1, \ldots, v_n$ decides a schedule of which print jobs to run on the first printer and which print jobs to run of the second printer, so that both printers can complete their assigned jobs in $\leq T$ minutes, and that the total profit is maximized (possibly, not all n objects will be printed).

   Note: these are one-of-a-kind object, so you should not print more than one copy of the same object, or else you will only be able to sell one of them.

   Solution. For every $0 \leq T_1 \leq T$, $0 \leq T_2 \leq T$, and $0 \leq k \leq n$, we consider the problem of finding the schedule of maximum profit if we can use up to T_1 hours on printer 1, up to T_2 hours on printer 2, and we can only print a subset of the first k jobs. If we call $P[T_1, T_2, k]$ the above maximum profit, then the answer to the question is the value $P[T, T, n]$.

   We have

   $$\forall T_1, T_2 \quad P[T_1, T_2, 0] = 0$$

   and

   $$\forall T_1, T_2, k \quad P[T_1, T_2, k+1] = \min \begin{cases} P[T_1, T_2, k] & \\ v_k + P[T_1 - t_{k+1}, T_2, k] & \text{if } T_1 \geq t_{k+1} \\ v_k + P[T_1, T_2 - t_{k+1}, k] & \text{if } T_2 \geq t_{k+1} \end{cases}$$

2. Suppose that we redefine Edit Distance to allow one operation to delete an arbitrary sequence of consecutive letters, in addition to the operations of inserting one letter and of deleting one letter. For example, according to this definition, the Edit Distance between *artisan* and *cart* is 2.

   Describe and analyze an algorithm that, given two strings both of length $\leq n$, runs in time polynomial in n and computes the modified Edit Distance between them.

Solution. This is very similar to the algorithm for the standard edit distance problem, except that, in handling the case of deletions, we allow arbitrarily many consecutive characters to be deleted at once.

Given two strings A and B we define an array $ED[i,j]$ that stores the (modified) edit distance between $A_1,\ldots,A_i$ and $B_1,\ldots,B_j$. We have

$$\forall i \quad ED[i,0] = i$$
$$\forall j \quad ED[0,j] = j$$

and then

$$\forall i \geq 1,\ \forall j \geq j \quad ED[i,j] = \min \begin{cases} ED[[i-1,j-1] & \text{if } A_i = A_j \\ \min\limits_{1\leq k\leq i} ED[i-k,j]+1 & \\ ED[i,j-1]+1 & \end{cases}$$

Solution. First we note that the problem of deciding if two circuits are *not* equivalent is in NP, because it is the problem of deciding if there exists a solution (an input x) with a polynomial time checkable property (that $C(x) \neq C'(x)$).$Thismeansthat, if P = NP, theproblemofcheckingiftwocircuitsarenotequivalentissolvableinpolynomialtime, andhencesoisthepro$

Now, given a circuit C and an integer parameter k, the problem of finding a circuit C' equivalent to C of size at most k is a search problem in NP, because the property of being a valid solution (being equivalent to C and having at most k gates) is solvable in polynomial time. By using again our assumption that P=NP, and the fact that it implies that all NP search problems are solvable in polynomial time, we have that there is a polynomial time algorithm that, given C and k, finds a circuit C' equivalent to C of size at most k, if one exists. We apply this polynomial time algorithms for $k = 0, 1, \ldots$ until we find the smallest k for which the algorithm finds a circuit C' equivalent to C.

3. Say that two Boolean circuits C and C' are *equivalent* if, for every input they have the same output, that is

$$\forall x \quad C(x) = C'(x)$$

Show that if P=NP there is a polynomial time algorithm that, given a boolean circuit C, finds a smallest circuit C' such that C' is equivalent to C. By "smallest" we mean the circuit with the fewest gates.

4. We say that the input x_1 of a Boolean circuit $C(x_1,\ldots,x_n)$ is *influential* if there is a setting $a_2,\ldots,a_n$ for the other variables such that

$$C(0,a_2,a_3,\ldots,a_n) \neq C(1,a_2,a_3,\ldots,a_n)$$

Prove that problem of deciding if the first input of a Boolean circuit is influential is NP-complete

[Hint: reduce from Circuit SAT]

Solution. Given a circuit C with n inputs, we define a new circuit C' with $n+1$ inputs as $C'(x_1, \ldots, x_{n+1}) = x_1 \wedge C(x_2, \ldots, x_{n+1})$, where $\wedge$ represents an AND gate.

If C is satisfiable, and $C(a_1, \ldots, a_n) = 1$, then $C'(0, a_1, \ldots, a_n) = 0$ and $C'(1, a_1, \ldots, a_n) = 1$, so that the C' is a circuit for which the first input is influential. If C is not satisfiable, then for every $a_1, \ldots, a_n$ we have $C(a_1, \ldots, a_n) = 0$, so that $C'(0, a_1, \ldots, a_n) = C'(1, a_1, \ldots, a_n) = 0$ and C' is a circuit for which the first input is not influential.