CSEP521: Algorithms	December 2, 2022
Homework 4	
Anup Rao	Due: December 11, 2022

Read the fine print[1]. An algorithm is said to run in polynomial time if it runs in time $O(n^d)$ for some constant d on inputs of size n. Each problem is worth 10 points:

1. Show that if P=NP, then there is a polynomial time algorithm for factoring. Here you are given an n-bit number N, and you need to find a factor a that divides N, with $a \neq 1$, and $a \neq N$, if such an a exists.

   Solution. Let $\mathsf{IsFactor}(N, x, y)$ be a procedure with polynomial runtime that returns True if there is an a that divides N and $x \leq a \leq y$. (It is easy to verify that the language corresponding to $\mathsf{IsFactor}(N, x, y)$ is in NP.) The following algorithm outputs the smallest a that divides N such that $1 < a < N$, if such an a exists.

Input: N
Result: a that divides N, satisfying $1 < a < N$
if $\mathsf{IsFactor}(N, 2, N-1) == \textit{False}$ **then**
  return No such divisor exists
end
$\ell = 2$, $r = N - 1$
while $\ell \leq r$ **do**
  if $\ell == r$ *and* ℓ *divides* N **then**
    return ℓ
  end
  $m = \lfloor (\ell + r)/2 \rfloor$
  if $\mathsf{IsFactor}(N, \ell, m) ==$ *True* **then**
    $r == m$
  end
  else
    $\ell = m$
  end
end

---

[1] In solving the problem sets, you are allowed to collaborate with fellow students taking the class, but **each submission can have at most one author**. If you do collaborate in any way, you must acknowledge, for each problem, the people you worked with on that problem. The problems have been carefully chosen for their pedagogical value, and hence might be similar to those given in past offerings of this course at UW, or similar to other courses at other schools. Using any pre-existing solutions from these sources, for from the web, constitutes a violation of the academic integrity you are expected to exemplify, and is strictly prohibited. Most of the problems only require one or two key ideas for their solution. It will help you a lot to spell out these main ideas so that you can get most of the credit for a problem even if you err on the finer details. Please justify all answers. Some other guidelines for writing good solutions are here: `http://www.cs.washington.edu/education/courses/cse421/08wi/guidelines.pdf`.

Note that the while loop iterates at most $\log N \leq n$ times. Since each iteration has a polynomial runtime, the overall runtime of the algorithm is also polynomial. For the proof of correctness, observe that if $\mathsf{IsFactor}(N, 2, N-1)$ is True, then every iteration of the while loop considers a range from x to y such that $\mathsf{IsFactor}(N, x, y)$ is True. The distance between x and y decrease in each iteration, eventually finding the number that divides N.

2. Compute the dual of the following program:

$$\begin{array}{ll} \text{maximize} & x_1 - 3x_2 + 4x_3 \\ \text{subject to} & 5x_1 + 3x_2 \leq 0 \\ & 4x_1 - x_2 \leq 3 \\ & -x_2 + 3x_3 \leq 2 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{array} .$$

Solution:

The dual problem of a primal problem of the form

$$\text{maximize } c^T x, \text{ subject to } Ax \leq b, x \geq 0$$

is given by

$$\text{minimize } b^T y, \text{ subject to } A^T y \geq c, y \geq 0$$

We first substitute $x_3 = x_3^+ - x_3^-$, to get the standard form for the primal problem

$$\begin{array}{ll} \text{maximize} & x_1 - 3x_2 + 4x_3^+ - 4x_3^- \\ \text{subject to} & 5x_1 + 3x_2 \leq 0 \\ & 4x_1 - x_2 \leq 3 \\ & -x_2 + 3x_3^+ - 3x_3^- \leq 2 \\ & x_1, x_2, x_3^+, x_3^- \geq 0 \end{array}$$

where

$$A = \begin{pmatrix} 5 & 3 & 0 & 0 \\ 4 & -1 & 0 & 0 \\ 0 & -1 & 3 & -3 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 3 \\ 2 \end{pmatrix}, \quad c = \begin{pmatrix} 1 \\ -3 \\ 4 \\ -4 \end{pmatrix}.$$

Therefore, the dual problem is

$$\begin{array}{ll} \text{minimize} & 3y_2 + 2y_3 \\ \text{subject to} & 5y_1 + 4y_2 \geq 1 \\ & 3y_1 - y_2 - y_3 \geq -3 \\ & 3y_3 \geq 4 \\ & -3y_3 \geq -4 \\ & y_1, y_2, y_3 \geq 0, \end{array}$$

Note: You can further simplify the above equations and unequations by putting $y_3 = \frac{4}{3}$ and making subsequent modifications. However, it is not needed to get full credit for this problem.

3. You are given the following 4 points in the plane:

$$(a_1, b_1) = (1, 3), (a_2, b_2) = (2, 7), (a_3, b_3) = (3, 5), (a_4, b_4) = (4, -1).$$

You want to find a line that approximately passes through these points. A line

$$\ell_{\alpha,\beta} = \{(x, y) : y = \alpha x + \beta\}$$

is specified by the numbers α, β. The goal is to find the line that minimizes its error from the point farthest from it. Write a linear program to find the parameters α, β to minimize the error

$$\max_{i=1,2,3,4} |b_i - \alpha \cdot a_i - \beta|.$$

The program need not be in standard form.

Solution:

$$\begin{aligned} &\text{minimize } c \\ &\text{subject to} \\ &\text{for all } i = 1, \dots, 4, \\ &\qquad c \geq b_i - \alpha a_i - \beta \\ &\qquad c \geq -b_i - \alpha a_i - \beta \end{aligned}$$

4. Consider a special version of the 3SAT problem, where every clause has exactly 3 literals, and each variable appears at most 3 times. Show that this version of 3SAT can be solved in polynomial time, by giving a polynomial time algorithm that finds a satisfying assignment. HINT: Consider the bipartite graph with clauses on the left, and variables on the right. Connect a clause to a variable if the variable appears in the clause. Argue that this graph has a perfect matching. Then give an algorithm to find the perfect matching and find a satisfying assignment.

Solution Let $x_1, \dots, x_n$ and $c_1, \dots, c_m$ be the the variables and clauses, respectively. As in the hint, construct a bipartite graph with the clauses on the left and the variables on the right, in which there is an edge between c_i and x_j if x_j or its negation appears in c_i. We claim that this graph has a matching in which all clauses are matched.

Claim 1. *There is an injective map $f : \{1, 2, \dots, m\} \to \{1, 2, \dots, n\}$ such that the edges $(c_1, x_{f(1)}), (c_2, x_{f(2)}), \dots, (c_m, x_{f(m)})$ form a matching.*

Proof We will prove this claim using Hall's theorem. We want to show that for every subset of clauses C, its neighborhood $N(C) \subseteq \{x_1, \dots, x_n\}$ satisfies $|N(C)| \geq |C|$. For the sake of contradiction, assume that there is a C such that $|N(C)| < |C|$. As $|N(C)| \leq 3|C|$, it must be the case that there is a variable in $N(C)$ with an edge to more than 3 clauses, which is a contradiction to the assumption that each variable appears three times. ■

Let f be the injective map given by Claim 1. Now, for each $i \in \{1, 2, \ldots, n\}$, if $f^{-1}(i)$ exists, then set x_i such that the clause $c_{f^{-1}(i)}$ is satisfied; otherwise, set $x_i = 0$. By Claim 1, we know that $(c_1, x_{f(1)}), (c_2, x_{f(2)}), \ldots, (c_m, x_{f(m)})$ form a matching. Hence, all clauses are satisfied by this assignment.