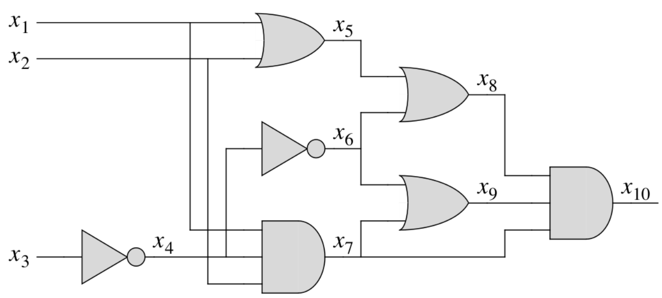
ECE 606, Fall 2021, Assignment 12
Due: Tuesday, December 7, 11:59pm

Submission: submit your written solutions to crowdmark. There are no [**python3**] problems in this assignment.

This is a kind of "capstone" assignment that includes problems from various pieces of content we have covered.

1. Consider the following proposed reduction from CIRCUIT-SAT to SAT. Label every wire in the input circuit $x_1, x_2, \ldots, x_n$, where x_n is the output wire. Starting with the output wire, inductively rewrite each x_i based on the gate of which x_i is output, so x_i is expressed as a boolean function of the inputs to the gate. Do this repeatedly till only the variables that correspond to the input wires remain.



For example, consider the above circuit and labels on the wires. We would progressively rewrite as follows:

$$
\begin{array}{lll}
 & x_{10} & \\
\to & x_8 \wedge x_9 \wedge x_7 & \because \text{rewrite } x_{10} \\
\to & (x_5 \vee x_6) \wedge x_9 \wedge x_7 & \because \text{rewrite } x_8 \\
\to & (x_5 \vee x_6) \wedge (x_6 \vee x_7) \wedge x_7 & \because \text{rewrite } x_9 \\
\to & (x_5 \vee x_6) \wedge (x_6 \vee (x_4 \wedge x_2 \wedge x_1)) \wedge (x_4 \wedge x_2 \wedge x_1) & \because \text{rewrite } x_7 \\
 & \text{and so on, till we are left with an expression in } x_1, x_2, x_3 \text{ only.} &
\end{array}
$$

Show, via counterexample, that the above reduction is not guaranteed to be polynomial-time.

2. Consider the following problem, call it INDSETVERTEXCOVER: given inputs an undirected graph G and an integer k, does G have an independent set of size k and a vertex cover of size k?

   Show that INDSETVERTEXCOVER is **NP**-complete. To show that it is **NP**-hard, you are allowed to reduce from any problem we have discussed in the course.

3. Consider the problem: given input an integer $n > 2$, is n prime? Consider also the following algorithm for it, which leverages the fact that n is composite if and only if it has a divisor d such that $1 < d \leq \lfloor \sqrt{n} \rfloor$.

isPrime(n)

  foreach i *from* 2 *to* $\lfloor\sqrt{n}\rfloor$ **do**

    if i *divides* n **then return** false

  return true

Is isPrime a polynomial-time algorithm? Justify.

4. True or not necessarily true?

   (a) if the time-efficiency of an algorithm in the worst-case is $O(f(n))$, then the time-efficiency (without further qualification as to worst-/best-/expected-case) of that algorithm is $O(f(n))$.

   (b) if the time-efficiency of an algorithm in the worst-case is $\Theta(f(n))$, then the time-efficiency (without further qualification as to worst-/best-/expected-case) of that algorithm is $\Theta(f(n))$.

5. Suppose we know that the only arrays we want to sort are those that contain integers in a range $[1, k]$, where k is some constant positive integer. Consider the following algorithm for this problem.

   SortRange($A[1, \ldots, n]$)

   Allocate a new array $B[1, \ldots, k]$
   foreach i *from* 1 *to* k **do** $B[i] \leftarrow 0$
   foreach i *from* 1 *to* n **do** $B[A[i]] \leftarrow B[A[i]] + 1$
   $j \leftarrow 1$
   foreach i *from* 1 *to* k **do**
     while $B[i] > 0$ **do**
       $A[j] \leftarrow i$
       $j \leftarrow j + 1$
       $B[i] \leftarrow B[i] - 1$

   What is a characterization of the running-time of SortRange in $\Theta(\cdot)$ notation as a function of n, the number of items in the input array A?