Homework 4 Questions

Problem 1: (15 points) Answer each part TRUE or FALSE (Big O).

a) $2n = O(n)$
- True: As $g(n) = n$, any input n, a constant c can always be found such that $0 \leq 2n \leq cn$. For example, consider $c = 2$ and $n_0 = 1$,

$$f(n) \leq c\big(g(n)\big)$$
$$2n \leq 2(n)$$
$$2(1) \leq 2(1)$$
$$2 \leq 2 \text{ for all } n \geq 1$$

b) $n^2 = O(n)$
- False: Since n^2 grows faster than n, n^2 dominates the growth of $f(n) = n^2$. Therefore, n^2 cannot be bounded by $f(n)$, making $n^2 = O(n)$ false.

c) $3^n = 2^{O(n)}$
- True: According to the book, when the O symbol occurs in an exponent, as in the expression $f(n) = 2^{O(n)}$, the exponent dominates the expression, thus representing an upper bound of 2^{cn}. This means that 2^{cn} is an upper bound for 3^n, and $for\ all\ n \geq n_0$ there are $3^n <= 2^{\mathrm{cn}}$. Hence, it can be conclude that $3^n = 2^{O(n)}$.

Problem 2: (15 points) Answer each part TRUE or FALSE (Small o).

a) $n = o(2n)$
- False: n grows at the same rate as $2n$, which means that as n approaches infinity, the ratio $\frac{n}{2n}$ remains constant when the small o definition requires that as n approaches infinity, the ratio $\frac{f(n)}{g(n)}$ approaches zero. This would indicate that $f(n) = cg(n)$ when the definition requires $f(n) < cg(n)$ for any real number $c > 0$, there exists a real number n_0 for all $n > n_0$. However, this can be violated with $c = \frac{1}{2}$ for any value of n.

b) $2n = o(n^2)$
- True: $2n$ grows slower than n^2, which means that as n approaches infinity, the ratio ratio $\frac{2n}{n^2}$ approaches zero, which satisfies the small o definition.

c) $n = o(log\ n)$
- False: $log\ n$ grows slower than n, which means that as n approaches infinity, the ratio $\frac{n}{\log(n)}$ approaches infinity, not zero as required by the small o definition.

Problem 3: (10 points) Is the following pair of numbers relatively prime? Show the calculations that led to your conclusion.

a) 1274 and 10505

- Yes, this pair of values are relatively prime. Since Two numbers are relatively prime if 1 is the largest integer that evenly divides them both. We can calculate this using the Euclidean algorithm to find $\gcd(1274, 10505)$. Our Process is as follows:
  1) Using Division Algorithm, find q and r to write a=bq+r
  2) Do step 1 again, but now use b as your new a, and use r as your new b
  3) Stop when your $r = 0$. Your b from the last step is your final answer.
- Proof:

Step 1: Using Division Algorithm, find q and r to write a = bq + r
- $a = bq + r$
- $1274 = 10505q + r$
- $1274 = 10505(0) + 1274$

Step 2: Do step 1 again, but now use b as your new a, and use r as your new b
- $a = bq + r$
- $10505 = 1274q + r$
- $10505 = 1274(8) + 313$

Step 3: Do step 1 again, but now use b as your new a, and use r as your new b
- $a = bq + r$
- $1274 = 313q + r$
- $1274 = 313(4) + 22$

Step 4: Do step 1 again, but now use b as your new a, and use r as your new b
- $a = bq + r$
- $313 = 22q + r$
- $313 = 22(14) + 5$

Step 5: Do step 1 again, but now use b as your new a, and use r as your new b
- $a = bq + r$
- $14 = 5q + r$
- $14 = 5(2) + 4$

Step 6: Do step 1 again, but now use b as your new a, and use r as your new b
- $a = bq + r$
- $5 = 4q + r$
- $5 = 4(1) + 1$

Step 7: Do step 1 again, but now use b as your new a, and use r as your new b
- $a = bq + r$
- $4 = 1q + r$
- $4 = 1(4) + 0$

Step 8: Stop when your $r = 0$. Your b from the last step is your final answer.
- $\gcd(1274, 10505) = 1$

❖ By definition, two numbers are relatively prime if 1 is the largest integer that evenly divides them both. Thus, in order to prove that 1274 and 10505 are relatively prime, I calculated their greatest common divisor (gcd) using the Euclidean algorithm, and obtained $\gcd(1274, 10505) = 1$. As such, I have successfully prove that 1274 and 10505 have no common divisors other than 1, and therefore they are relatively prime.

Problem 4: (10 points) Is the following formula satisfiable? Why?

$$(x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$$

First, lets review the Boolean truth values

∧ and *conjunction (intersection)*		∨ or *disjunction (union)*	
0 ∧ 0	0	0 ∨ 0	0
0 ∧ 1	0	0 ∨ 1	1
1 ∧ 0	0	1 ∨ 0	1
1 ∧ 1	1	1 ∨ 1	1

¬ not *negation (complement)*	
¬0	1
¬1	0

According to our textbook, a Boolean formula is satisfiable if some assignment of 0s and 1s to the variables makes the formula evaluate to 1.

Therefore, to test the satisfiability of this equation, we can try different combinations of 0 and 1 to check if any of them evaluate to the value of 1. There are two possible scenarios to consider: $x = 1$ and $y = 0$, or $x = 0$ and $y = 1$. It is not necessary to test the scenarios where x and y have the same value since they are represented by the same symbol.

- Scenario 1: $x = 1$ and $y = 0$

$$(x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$$

$$(1 \vee 0) \wedge (1 \vee \bar{0}) \wedge (\bar{1} \vee 0) \wedge (\bar{1} \vee \bar{0})$$

$$(1 \vee 0) \wedge (1 \vee 1) \wedge (0 \vee 0) \wedge (0 \vee 1)$$

$$(1) \wedge (1) \wedge (0) \wedge (1)$$

$$(1) \wedge (0) \wedge (1)$$

$$(0) \wedge (1)$$

$$(0)$$

Scenario 1 evalutates to 0 and, as such, we know that $x = 1$ and $y = 0$ can not be the correct values to prove this formulas satisfiability.

- Scenario 2: $x = 0$ and $y = 1$

$$(x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$$

$$(0 \vee 1) \wedge (0 \vee \bar{1}) \wedge (\bar{0} \vee 1) \wedge (\bar{0} \vee \bar{1})$$

$$(0 \vee 1) \wedge (0 \vee 0) \wedge (1 \vee 1) \wedge (1 \vee 0)$$

$$(1) \wedge (0) \wedge (1) \wedge (1)$$

$$(0) \wedge (1) \wedge (1)$$

$$(0) \wedge (1)$$

$$(0)$$

After testing Scenario 2, I have shown that it also evaluates to 0. Therefore, none of the possible assignments of 0s and 1s to the variables can make the formula evaluate to 1. This proves that the Boolean equation $(x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$ is not satisfiable.

Problem 5: (10 points) What is P-Problem? What is NP-Problem? What is NP-Complete problem?

A P-Problem is a problem that is efficiently decidable in polynomial, such as n^k, time on a deterministic single-tape Turing machine. These problems are considered "tractable" and the class of problems that are realistically solvable can computed using the function:

$$P = \bigcup_k TIME(n^k)$$

"union of all polynomial time functions"

An NP-Problem stands for "nondeterministic polynomial time" and it is a decision problem that can be solved by a non-deterministic Turing machine in polynomial time. In other words, given a potential solution to an NP problem, it can be verified to be correct or incorrect in polynomial time. However, finding a solution may require a non-polynomial amount of time.

NP-Complete refers to a class of problems whose "individual complexity is related to that of the entire class." In other words, if an efficient, polynomial time algorithm exists for solving a problem within the class, then an efficient algorithm exists for solving all problems in NP. An example of an NP-Complete problem includes the Boolean satisfiability problem.

Problem 6: (20 points) Show that NP is closed under union

(Hint: For any two languages L_1 and L_2, let M_1 and M_2 be the NTM that decides them in polynomial time. Construct a NTM M' that decided $L_1 \cup L_2$ in polynomial time.)

To show that NP is closed under union, I need to prove that for any two languages $L_1$and L_2 in NP, their union $L_1 \cup L_2$ is also in NP. To do this, I will begin by constructing a non-deterministic Turing machine (NTM) M' that decides $L_1 \cup L_2$ in polynomial time. This process is shown below:

- M' non-deterministically guesses whether an input string, w, belongs to L_1 or L_2
  - Instead of following a set of predefined steps, the M' select one of the languages and verify whether the guess is correct or not.

- **if** w belongs to L_1, M' simulates the NTM M_1:
  if M_1 accepts w:
  then M' accepts w
  else M' rejects w.
- **else if** w belongs to L_2, M' simulates the NTM M_2:
  if M_2 accepts w:
  then M' accepts w
  else M' rejects w.
- **else** w is not in either L_1 or L_2, M' rejects w.

Since M_1 and M_2 decide their languages in NP, and the non-deterministic guess of L_1 or L_2 can be made in NP, I can conclude that M' also runs in NP. This process shows that M' decides $L_1 \cup L_2$ since it accepts a string w iff either M_1 or M_2 accepts w. Since this implies that $L_1 \cup L_2$ is in NP, I have shown that NP is closed under union.

Problem 7: (10 points) Prove that 3SAT is polynomial reducible to CLIQUE.

To prove that 3SAT is polynomial reducible to CLIQUE, I can utilize the polynomial time reduction to construct a graph G that has a k-clique iff the 3SAT formula ϕ is a satisfiable 3cnf-formula; Let ϕ be a formula with k clauses such as:

$$\phi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \cdots (a_k \vee b_k \vee c_k)$$

Assuming that ϕ is satisfiable, we can form a k-clique in G by selecting one node from each triple that corresponds to literal in a clause of ϕ. Each triple of nodes in the graph corresponds to a clause in the formula. For example, if the 3SAT formula contains the clauses $(x_1 \vee x_1 \vee x_2)$ and $(\neg x_1 \vee \neg x_2 \vee \neg x_2)$, then there would be two corresponding groups of triple nodes in the graph: x_1, x_1, x_2 and $\neg x_1, \neg x_2, \neg x_2$, The edges of G connect nodes that do not belong to the same triple, such that x_1 would not connect to x_2, and do not have contradictory labels, such that x_1 would not connect to a node $\neg x_1$. By selecting one node from each triple, one true literal from each clause is selected, which results in a satisfying assignment for ϕ.

Furthermore, if G has a k-clique, then each triple must contain exactly one node from the k-clique. Since each node corresponds to a literal in ϕ, we can assign truth values to the literals in ϕ by setting each one corresponding to a node in the k-clique to true. Therefore, each clause contains a literal that is assigned True, and we have a satisfying assignment for ϕ.