**Homework 4 Questions**

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

* True: As , any input , a constant can always be found such that . For example, consider and ,

*for all*

* False: Since grows faster than , dominates the growth of . Therefore, cannot be bounded by , making false.
* True: According to the book, when the symbol occurs in an exponent, as in the expression , the exponent dominates the expression, thus representing an upper bound of . This means that is an upper bound for , and there are . Hence, it can be conclude that .

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

* False: grows at the same rate as , which means that as approaches infinity, the ratio

remains constant when the small definition requires that as approaches infinity, the ratio approaches zero. This would indicate that when the definition requires for any real number , there exists a real number for all . However, this can be violated with for any value of .

* True: grows slower than , which means that as approaches infinity, the ratio

ratio approaches zero, which satisfies the small definition.

* False: grows slower than , which means that as approaches infinity, the ratio approaches infinity, not zero as required by the small definition.

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

## and

* 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 . 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 as your new , and use as your new
  3. Stop when your . Your from the last step is your final answer.
* Proof:

#### Using Division Algorithm, find and to write

#### Do step 1 again, but now use as your new , and use as your new

#### Do step 1 again, but now use as your new , and use as your new

#### Do step 1 again, but now use as your new , and use as your new

#### Do step 1 again, but now use as your new , and use as your new

#### Do step 1 again, but now use as your new , and use as your new

#### Do step 1 again, but now use as your new , and use as your new

#### Stop when your . Your from the last step is your final answer.

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

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

First, lets review the Boolean truth values

|  |  |  |  |
| --- | --- | --- | --- |
| and conjunction (intersection) | | or disjunction (union) | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| not *negation (complement)* | | | |
|  | |  | |
|  | |  | |

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

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

## Scenario 1: and

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

## Scenario 2: and

After testing Scenario 2, I have shown that it also evaluates to . Therefore, none of the possible assignments of s and s to the variables can make the formula evaluate to 1. This proves that the Boolean equation 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 , 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:

*“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 L1 and L2, let M1 and M2 be the NTM that decides them in polynomial time. Construct a NTM M’ that decided L1UL2 in polynomial time.)

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

* non-deterministically guesses whether an input string, , belongs to or
  + Instead of following a set of predefined steps, the select one of the languages and verify whether the guess is correct or not.
* **if** belongs to , simulates the NTM :

**if** accepts :

then accepts

**else** rejects .

* **else if** belongs to , simulates the NTM :

**if** accepts :

then accepts

**else** rejects .

* **else** is not in either or , rejects .

Since and decide their languages in NP, and the non-deterministic guess of or can be made in NP, I can conclude that also runs in NP. This process shows that decides since it accepts a string iff either or accepts . Since this implies that 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 that has a -clique iff the 3SAT formula is a satisfiable 3cnf-formula; Let be a formula with clauses such as:

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

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