Theory of Computation: Assignment 10

Arjun Chandrasekhar

Due 04/14/2022 at 11:59 pm (50 points)

1. This problem is based on problem 7.6 in Sipser. We will prove that P is closed under complement, union, and concatenation. Formally, let L_1, L_2 be formal languages with $L_1, L_2 \in$ P - that is, both languages can be recognized in polynomial time. In particular, assume that $L_1, L_2 \in \text{TIME}(n^c)$
   (a) (5 points) Prove that $\overline{L_1} = \{w | w \notin L_1\} \in \text{TIME}(n^c)$.
   (b) (5 points) Prove that $L_1 \cup L_2 = \{w | w \in L_1 \text{ or } w \in L_2\} \in \text{TIME}(n^c)$.
   (c) (5 points) Prove that $L_1 \circ L_2 = \{w | w = xy \text{ for some } x \in L_1, y \in L_2\} \in \text{TIME}(n^{c+1})$.
2. (5 points) Explain why the previous problem establishes that P is closed under complement, union, and concatenation.
3. For two numbers a and b, the **least common multiple (lcm)** is the smallest number m such that both a and b are factors of m. For example $\text{lcm}(18, 12) = 36$ because it's the smallest number that has both 18 and 12 as factors.

   Formally, we will work with the following decision problem

   $$\text{LCM} = \{\langle a, b, k\rangle | \, \text{lcm}(a, b) = k\}$$

   Note that a, b, k are all represented in binary.

   (a) (5 points) Explain why the following algorithm to decide L does not run in polynomial time
      1. Check if k is a multiple of both a and b; if it's not, reject $\langle a, b, k\rangle$
      2. For $i = 1, 2, \ldots k - 1$ do the following:
         a. If i is a multiple of both a and b, we've found a smaller multiple. Reject $\langle a, b, k\rangle$
      3. If we reach finish the loop without finding a smaller multiple, accept $\langle a, b, k\rangle$

   (b) (5 points) Prove that LCM $\in$ P. You may use without proof the fact that $\text{lcm}(a, b) = \dfrac{a \cdot b}{\text{gcd}(a, b)}$
4. (10 points) Consider the following language

   $$\text{UNARY-MULTISET-SUM} = \left\{ \langle B, x_1, \ldots x_n\rangle | \begin{array}{c}\text{B is unary} \\ \text{There is a combination of } x_i\text{'s (repeats allowed)} \\ \text{that add up to B}\end{array} \right\}$$

   As an example:

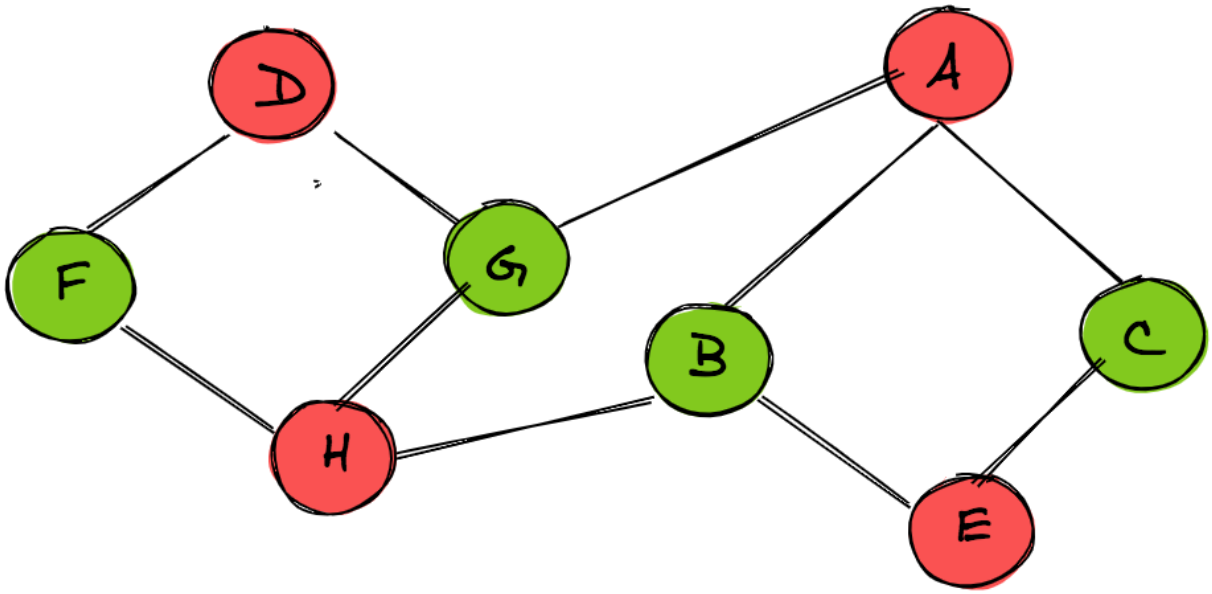
   - $\langle 31, 7, 4, 9\rangle \in$ UNARY-MULTISET-SUM because $31 = 9 + 7 + 7 + 4 + 4$. Note how we can repeat numbers, unlike the subset sum problem we studied in class
   - $\langle 101, 4, 6, 8\rangle \notin$ UNARY-MULTISET-SUM$\rangle$ because 101 is odd, but $4, 6, 8$ are all even

Note that technically 31 and 101 would be represented in unary, i.e. $\underbrace{1\ldots1}_{31}$ and $\underbrace{1\ldots1}_{101}$.

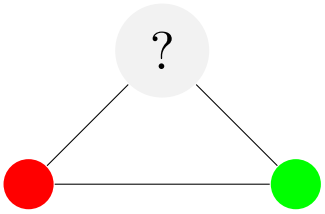
Prove that UNARY-MULTISET-SUM $\in$ P. (**Hint:** Start by making a 1-D array A of size $B+1$. Use dynamic programming to fill in the array.)

5. (10 points) A graph G is said to be **2-colorable** if we can assign every vertex one of two colors (say, red or green), with the constraint that two vertices that are connected by an edge must have opposite colors.

   The following image illustrates the problem. Notice how there is no edge connecting two vertices with the same color. All of the adjacent vertices have opposite colors. Thus, this graph is 2-colorable.



By contrast, a triangle graph is not 2-colorable



Formally we will work with the following language

$$\text{2-COLORING} = \{\langle G\rangle | G \text{ is a 2-colorable graph}\}$$

Prove that 2-COLORING $\in$ P. (**Hint:** One approach is to use a greedy coloring algorithm. Another - elegant - approach is to convert the graph into a 2-SAT formula).