CSCI 3104 Fall 2021
Instructors: Profs. Grochow and Waggoner

Problem Set 12

Due Date ..... Dec / 8^{th} / 2021
Name ..... **Michael Ghattas**
Student ID **109200649**
Collaborators ..... **Me, Myself & I**

Contents

1 Instructions

- The solutions **must be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.

- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign)

Problem 1.

- My submission is in my own words and reflects my understanding of the material.
- I have not collaborated with any other person.
- I have not posted to external services including, but not limited to Chegg, Discord, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

I agree to the above, Michael Ghattas. □

3 Standard 26- Computational Complexity: Formulating Decision Problems

Problem 2. Recall the Maximum Bipartite Matching problem from class, where we took as input a bipartite graph $G(L\dot{\cup}R, E)$ and asked for a maximum cardinality matching. Formulate the decision variant of this problem using the Instance/Decision format from class. [**Note:** See Example 172 of M. Levet's Lecture Notes.]

Answer:

- Instance: Let $G(L\dot{\cup}R, E)$ be a bipartite graph, and $k \in \mathbb{N}$.
- Decision: Does there exist a Matching value (k), of edges from graph G?

□

4 Standard 27- Computational Complexity: Problems in P

Problem 3. Consider the decision variant of the Interval Scheduling problem.

- Instance: Let $\mathcal{I} = \{[s_1, f_1], \ldots, [s_n, f_n]\}$ be our set of intervals, and let $k \in \mathbb{N}$.
- Decision: Does there exist a set $S \subseteq \mathcal{I}$ of at least k pairwise-disjoint intervals?

Show that the decision variant of the Interval Scheduling problem belongs to P. You are welcome and encouraged to cite algorithms we have previously covered in class, including known facts about their runtime. [**Note:** To gauge the level of detail, we expect your solutions to this problem will be 2-4 sentences. We are not asking you to analyze an algorithm in great detail.]

Answer:

The Greedy algorithm will first compute the maximum *size* of the non-overlapping intervals and then check if $(size \geq k)$. Using **Quick-Sort** for sorting the intervals array, the algorithm takes $O(n^2)$ time. Thus, a polynomial time algorithm for the decision problem exists, and therefore **Interval Scheduling** $\in$ P.

□

5 Standard 28- Computational Complexity: Problems in NP

Problem 4. We consider an algebraic structure $Q = (S, \star)$, where S is our set of elements and $\star : S \times S \to S$ is our "multiplication" operation. That is, for any $i, j \in S$, the product $i \star j \in S$. When S is finite, we can write out its "multiplication table": the rows are indexed by the elements of S, the columns are indexed by the elements of S, and the entry in the (i, j) position in the table is $i \star j$. We say that $(S, \star)$ is a *quasigroup* if the multiplication table is a Latin square; that is, if each element of S appears exactly once in each row and exactly once in each column.

The Latin Square Isotopy problem is defined as follows.

- Decision: Let $Q_1 = (S_1, \star)$ and $Q_2 = (S_2, \diamond)$ be quasigroups, where S_1, S_2 are n-element sets. Suppose that $(S_1, \star)$ and $(S_2, \diamond)$ are given by their multiplication tables.
- Decision: Do there exist one-to-one functions $\alpha, \beta, \gamma : S_1 \to S_2$ such that for all $x, y \in S$:

$$\alpha(x) \diamond \beta(y) = \gamma(x \star y).$$

  Here, $\alpha(x) \diamond \beta(y)$ is a product being considered in Q_2, while $x \star y$ is a product being considered in Q_1.

Show that the Latin Square Isotopy problem belongs to NP. *

Answer:

We first need to show that each of the α, β and γ functions are $one-to-one$ by verifying that every possible pair $(i, j) \in S_1$, $\alpha(i) \neq \alpha(j)$. This part of the process has a polynomial algorithm that will take $O(n^2)$ time. Now we will need to verify that the given equality of $\alpha(x)$, $\beta(y)$, $\gamma(x, y)$ holds true. We note that this part of the process has a polynomial algorithm that will take $O(n^3)$ time. Thus the **Latin Square Isotopy** $\in$ **NP**.

□

*It remains open whether the Latin Square Isotopy problem is in P. The Latin Square Isotopy problem, as well as closely related algebraic problems such as Group Isomorphism in the multiplication table model, serve as key barriers for placing Graph Isomorphism into P. The best known algorithm for Graph Isomorphism, due to Babai in 2016, runs in time $n^{\Theta(\log^2(n))}$. Babai's algorithm combines combinatorial techniques such as color-refinement (Weisfeiler–Leman) and algebraic techniques (e.g., permutation group algorithms, the Classification of Finite Simple Groups). This is a very significant result. There has been considerable work on Group Isomorphism in the last 10 years. Isomorphism testing is an active area of research in our CS Theory group!

6 Standard 30- Structure and Consequences of P vs. NP

6.1 Problem 5

Problem 5. A student has a decision problem L which they know is in the class NP. This student wishes to show that L is NP-complete. They attempt to do so by constructing a polynomial time reduction from L to SAT, a known NP-complete problem. That is, the student attempts to show that $L \leq_p \mathsf{SAT}$. Determine if this student's approach is correct and justify your answer.

Answer:

The student need to show the process of deducing a polynomial time reduction from **SAT** to **L**, in order to show that ($SAT \leq_p L$). Thus the student's approach is **incorrect**. □

6.2 Problem 6

Problem 6. The underlying model of computation used to define the complexity classes P and NP is the Turing Machine model. A Turing Machine can only read or write a single bit from memory at a given computation step. This will be the only fact about Turing Machines that we need.

Now define the complexity class PSPACE to be the set of languages L, where for a given string $x \in \Sigma^*$, there exists an algorithm (Turing Machine) A and polynomial $p(n)$ (depending on L) such that:

- $A(x)$ uses at most $p(|x|)$ bits,
- $A(x) = 1$ if and only if $x \in L$, and
- $A(x) = 0$ if and only if $x \notin L$.

We will show that NP $\subseteq$ PSPACE. The goal of this problem is to practice with the simulation technique.

(a) Suppose we have an algorithm (Turing Machine) that runs in time $T(n)$. Show that our algorithm uses at most $T(n)$ bits.

Answer:

If each time step can only read or write one bit, then the process takes at most $T(n)$ bits. □

(b) In light of part (a), explain why P $\subseteq$ PSPACE. [**Note:** It is likely that your answer will only require a couple sentences.]

Answer:

If $T(n)$ is a polynomial, then we know that $T(n)$ takes at most $T(n)$ bits, as per part (A). Thus **P** $\in$ **PSPACE**. □

(c) The Hamiltonian Cycle problem takes as input a simple, undirected graph G. We ask whether G has a cycle that visits each vertex. It is well-known that the Hamiltonian Cycle problem is NP-complete. Show that the Hamiltonian Cycle problem belongs to PSPACE. [**Hint:** It suffices to show that a brute-force approach can be computed using a polynomial amount of space in the number of vertices. Note that if our graph has n vertices, then each vertex is represented using a binary string of length $\lceil \log_2(n) \rceil$.]

Answer:

First we will need to go through every possible permutation of the vertices to verify if there is a **Hamiltonian Cycle**. Going through the possible permutations will take $(n \log n)$ memory. Once a **Hamiltonian Cycle** is identified, we end the run and index it accordingly. If a **Hamiltonian Cycle** is not found, we delete the memory of permutation run and index it accordingly. Since there are (n^n) possible permutations, storing the indices of these permutations will take $(n \log n)$ memory. Thus the **Hamiltonian Cycle** $\subseteq$ **PSPACE**. □

(d) Here, we show that NP $\subseteq$ PSPACE. Let $L \in$ NP, and let $\omega \in \Sigma^*$ be an input string. We want to decide whether $\omega \in L$. In light of parts (b) and (c), give a PSPACE algorithm to decide whether $\omega \in L$. This establishes that $L \in$ PSPACE, and so NP $\subseteq$ PSPACE. [**Hint:** Start by reducing L to Hamiltonian Cycle in polynomial-time. Is this reduction PSPACE-computable?]

Answer:

Since it is known that **Hamiltonian Cycle** is NP-Complete, there there exists a reduction of polynomial time from L to the **Hamiltonian Cycle**, such that if $(\omega \in L)$ there exists a **Hamiltonian Cycle**, and if $(\omega \notin L)$ there is no **Hamiltonian Cycle**. Since it take a polynomial amount of time to compute the reduction, it takes a polynomial amount of bits from part (b). From part (c), we can see that the **Hamiltonian Cycle** $\subseteq$ **PSPACE**. Therefore using this algorithm we will make a reduction from L to **Hamiltonian Cycle** in order to show that $(Hamiltonian - Cycle \leq_p L)$. Since L can be reduced into **Hamiltonian Cycle**, then $L \subseteq$ **PSPACE**. Given L is arbitrary, we can conclude that **NP** $\subseteq$ **PSPACE**.

□