CS 505 Spring 2025 — Homework 1

YOUR NAME HERE (FIRST AND LAST) (UID: YOUR UID HERE)

Due Date: February 6, 2025, no later than 2:00pm Central Time.

Collaboration Policy

Collaboration between students is encouraged. However, **all collaborations** need to be acknowledged (whether they are in this class or not in this class). You **MUST** list all collaborators for homework assignments. Moreover, collaborating **does not mean** you can copy-paste work from each other. Each submission needs to be in the students own words, otherwise it will be considered plagiarism.

Moreover, you are allowed to look to other resources for help with the homework. Please correctly cite such resources by using the `\cite` command, and including the correct citations in `local.bib`.

Finally, please acknowledge any other discussions that helped you complete this assignment. This can include "office hours," "Piazza," or other discussions where a direct collaboration did not happen.

Collaborator and Discussion Acknowledgements

Please list your collaborators below. Include their First and Last names, along with their UID if they are a UIC student. If you did not collaborate with others for this assignment, please copy-paste the following line into the first item of the `itemize` below.

I worked on this assignment individually and did not collaborate with others.

- Collaborator 1...

1 Problem 1 (Turing Machine Equivalences) (20 Points)

1.1 Part 1 (10 Points)

A *single-tape* Turing machine is a TM with a single read/write tape that is infinite in one direction. This tape is used as the input, output, and work tape. Formally prove that for every function $f\colon \{0,1\}^* \to \{0,1\}^*$ and time constructible $T\colon \mathbb{N} \to \mathbb{N}$, if f is computable in time T by a k-tape Turing machine (for $k \geq 3$), then it is computable in time $O(k \cdot T^2)$ on a single-tape Turing machine.

Note. You are expected to give the full formal description of the single-tape Turing machine. This includes specifying its set of states, and its transition function $\delta\colon Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$. You are also expected to give the run-time analysis of your single-tape Turing machine.

Proof of Problem 1 Part 1. Your answer here... □

1.2 Part 2 (10 Points)

Define a *RAM Turing machine* (i.e., random access memory Turing machine) to be a Turing machine that has *random access memory*. We formalize this as follows. A RAM Turing machine has 4 tapes: an input tape, an output tape, and 2 work tapes. One work tape is the *address tape*, and the other is the *memory tape*. Let $A\colon \mathbb{N} \to \mathbb{N}$ be the array denoting the memory tape. The RAM Turing machine has random access to the memory tape; that is, in a single step of the computation, the RAM Turing machine can move the memory tape head to any location i for $i \in \mathbb{N}$ (i.e., move the tape head to $A[i]$).

The RAM Turing machine has two additional tape alphabet symbols: `R` and `W`, along with an additional state q_{access}. Whenever the machine enters the state q_{access}, it reads from the address tape. If the read contents have the form $\langle i \rangle_2$`R`, then the machine reads symbol $A[i]$ and writes it to the cell directly to the right of the `R` symbol. If the read contents have the form $\langle i \rangle_2$`W`γ, then the Turing machine writes γ to $A[i]$. Here, $\langle i \rangle_2$ denotes the binary representation of i.

Show that any function $f\colon \{0,1\}^* \to \{0,1\}$ that is computable on a time T RAM Turing machine (for time constructible T), then $f \in \mathbf{DTIME}(T^2)$.

Note. You do not need to give the super formal Turing machine description for this problem. Also remember that the number of tapes in the Turing machine cannot grow with the input length (it must be a constant).

Proof of Problem 1 Part 2. Your answer here... □

2 Problem 2 (Undecidability) (15 Points)

2.1 Part 1 (5 Points)

In class, we showed that the halting problem, specified by the language

$$L_H = \{(\alpha, x) \colon M_\alpha(x) \text{ halts in a finite number of steps}\}$$

was undecidable via a reduction to another language. Prove that L_H is undecidable directly. I.e., do not reduce undecidablitilty of L_H to another language; show the proof (via contradiction) directly.

Proof of Problem 2 Part 1. Your answer here... □

2.2 Part 2 (5 Points)

A function $f \colon \{0,1\}^* \to \{0,1\}^*$ is a *partial function* if it is not necessarily defined on all inputs. We say that a Turing machine M computes partial function f if for every x that $f(x)$ is defined on, we have $M(x) = f(x)$, and if x is not defined on f, then $M(x)$ loops infinitely.

Let S be a set of partial functions. Define f_S to be the function defined as follows.

- $f_S(\alpha) = 1$ if and only if M_α computes some function $g \in S$.
- $f_S(\alpha) = 0$ if and only if M_α does not compute any function $g \in S$.

(Rice's Theorem): Prove that if $S \neq \emptyset$ and if S is not equal to the set of all partial functions computable by some Turing machine, then f_S is uncomputable. Equivalently, the language $L_S = \{\alpha \colon f_S(\alpha) = 1\}$ is undecidable.

Hint: can you reduce this problem to the Halting problem?

Proof of Problem Part 2. Your answer here... □

2.3 Part 3 (5 Points)

Given a Turing machine M, let $\mathcal{L}(M) \subset \{0,1\}^*$ denote the language that M recognizes (i.e., for all $x \in \mathcal{L}(M)$, $M(x) = 1$). Define a new language

$$L_{\text{ETM}} = \{\alpha \colon \mathcal{L}(M_\alpha) = \emptyset\}.$$

Prove that L_{ETM} is undecidable.

Proof of Problem Part 3. Your answer here... □

3 Problem 3 (P and NP) (25 Points)

3.1 Part 1 (10 Points)

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set $E \subseteq V \times V$. A *path of length* n in G is an ordered tuple of $n+1$ vertices $(v_1, \ldots, v_{n+1})$ such that $(v_i, v_{i+1}) \in E$ for all $i \in [n]$. We say that a path is *simple* if every vertex in the path is unique (i.e., $v_i \neq v_j$ for all $i \neq j$). Define two languages

$$\text{SPATH} = \{(G, u, v, k) \colon G \text{ contains a simple path from } a \text{ to } b \text{ of length at most } k\};$$

and

$$\text{LPATH} = \{(G, u, v, k) \colon G \text{ contains a simple path from } a \text{ to } b \text{ of length at least } k\};$$

1. Show that SPATH $\in \mathbf{P}$.
2. Show that LPATH $\in \mathbf{NP}$.

Proof of Problem 3 Part 1. Your answer here... □

3.2 Part 2 (5 Points)

Define a function $\text{pad} \colon \{0,1\}^* \times \mathbb{N} \to \{0,1\}^* \times \{\#\}^*$ as follows. For $x \in \{0,1\}^*$ and $k \in \mathbb{N}$, $\text{pad}(x, k) = (x, \#^j)$, where $j = \max\{0, (k - |x|)\}$ and $\#^j$ denotes writing the symbol $\#$ j times (e.g., $\#^3 = \#\#\#$).

Let $A \subseteq \{0,1\}^*$ be any language and let $T \colon \mathbb{N} \to \mathbb{N}$ be a function. Let $\mathbf{pad}(A, T)$ denote the following language:

$$\mathbf{pad}(A, T) = \{\text{pad}(x, T(|x|)) \mid x \in A\}.$$

Prove that if $A \in \mathbf{DTIME}(n^6)$ then $\mathbf{pad}(A, n^2) \in \mathbf{DTIME}(n^3)$.

Proof of Problem 3 Part 3. Your answer here... □

3.3 Part 3 (10 Points)

1. Prove that $\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{coNP}$.
2. Prove that if $\mathbf{P} = \mathbf{NP}$ then $\mathbf{NP} = \mathbf{coNP}$.

Proof of Problem 3 Part 3. Your answer here... □

4 Problem 4 (Reductions) (20 Points)

4.1 Part 1 (10 Points)

The *subset sum* problem asks the following: given n non-negative integers $A_1, \ldots, A_n$ and a target $t \in \mathbb{N}$, decide whether there exists a subset $S \subset [n]$ such that $\sum_{i \in S} A_i = t$. Formally, it is given by the language

$$\text{SUBSET-SUM} = \{(A_1, \ldots, A_n; t) \ : \ \exists S \subset [n] \text{ s.t. } \sum_{i \in S} A_i = t\}.$$

Prove that SUBSET-SUM is **NP**-complete via a reduction from 3SAT (i.e., show 3SAT $\leq_p$ SUBSET-SUM).

Proof of Problem 4 Part 1. Your answer here... □

4.2 Part 2 (10 Points)

The *k-independent set* problem for an undirected graph $G = (V, E)$ asks one to find a subset $S \subset V$ such that (1) $|S| \geq k$ and (2) for all $u, v \in S$, we have $(u, v) \notin E$. Let INDSET be the language

$$\text{INDSET}_k = \{(G = (V, E), k) \colon \exists S \subset V \text{ s.t. } |S| \geq k \text{ and } (u, v) \notin E \ \forall u, v \in S\}.$$

We showed that INDSET_k is **NP**-complete in class. Prove that the following language is **NP**-complete by reducing it to INDSET_k:

$$\text{CLIQUE} = \{(G, k) \colon G \text{ has a } k\text{-clique}\},$$

where a k-clique is a set of vertices $S \subset V$ such that $|S| = k$ and $\forall u, v \in S$, it holds that $(u, v) \in E$.

Hint: if G has an independent set of size k, think about what how these vertices are connected in the complement graph, where $\overline{G} = (V, \overline{E})$. That is, $\overline{G}$ contains all edges which are not present in G.

Proof of Problem 4 Part 2. Your answer here... □

5 Problem 5 (One More Reduction) (20 Points)

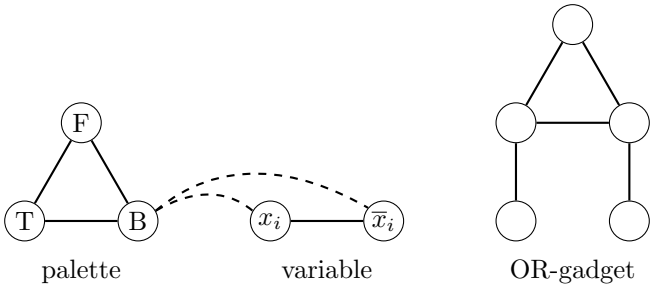
Let $G = (V, E)$ be an undirected graph. A *k-coloring* of G is an assignment of k colors to nodes/vertices such that no adjacent nodes share a color. In other words, given k distinct colors (i.e., labels), label each node/vertex of G such that any two adjacent nodes (i.e., nodes which share an edge) do not share a color.

Let

$$3COL = \{G = (V, E) \colon G \text{ is 3-colorable}\}$$

be the set of all graphs that have a valid 3-coloring. Prove that $3COL$ is **NP**-complete. For proving **NP**-hardness, reduce from 3SAT (i.e., 3SAT $\leq_p$ $3COL$).

Hint: let $\{T, F, B\}$ be your three colors ("True", "False", and "Base"). Use the following three subgraphs to help in your reduction. The OR-gadget can help you capture satisfiability of clauses.



Proof of Problem 5. Your answer here... □