October 21, 2025

Problem Set 3

Exercise 1. TMs are Robust. Turing Machine's were defined in Problem Set 2. In there, they had a very specific definition. However, it was also stated that they define computation, which seems have not changed in "power" since the inception of TMs.
While these two sentences may seem contradictory, TMs are also resistant to such changes. This resistance property is called being robust.
All of the following questions tackle the robustness of TMs against many different modifications to their definitions:

1. ($\star$) Consider a TM model with a larger input alphabet, show that a TM with $\Sigma = \{0, 1\}$ can simulate the one with the larger input alphabet.

2. ($\star$) Consider a TM model with multiple heads on the same work tape. Where each acts as a regular TMs head, and can move independent of one another. Show that a TM with m heads can be simulated by the regular TM.

3. ($\star$) Consider a TM model with multiple tapes, where each tape has its own head. Only the first tape starts with the input on it, (the rest start off blank).

Similar to the difference between a DFA and an NFA, a nondeterministic TM accepts if and only if there is a potential sequence of choices that lead to the TM accepting (out of many possible choices).

4. ($\star\star$) Show that any NTM can be simulated by a regular TM.

Our computers have random access memory, where the all of the memory locations can be accessed in roughly the same amount of time. You may have noticed that this is not the case with TMs, as the head needs to scan across to reach a cell.
Let a Random Access TM be a machine that has a special address tape, where it can write the index of a cell (in binary) and then the head will immediately move to that cell in the next time step. Note that it still takes time to write the address on the address tape.

5. ($\star\star$) Show that any RATM can be simulated by a regular TM.

Let an enumerator be a TM that has a special output tape and no halting state. When an enumerator starts it will start printing strings, separated by b onto the output tape. The language of an enumerator is the set of strings that will get written to the output tape at some point in time.

6. ($\star\star\star$) Show that a language is recognized by an enumerator if and only if it is recognized by a TM.

Exercise 2: Geometry of Numbers. *Lattices* are powerful mathematical objects that have deep connections to coding theory, homomorphic encryption, and post-quantum cryptography. In this exercise, we explore some properties and results about lattices over $\mathbb{R}^n$.

Let $\mathbf{B} = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \end{pmatrix}$ be an $n \times n$ matrix whose columns $\mathbf{b}_i$ are linearly independent. The set of integer combinations of the $\mathbf{b}_i$ is called a *lattice* with basis $\mathbf{B}$, denoted by $\Lambda(\mathbf{B})$:

$$\Lambda(\mathbf{B}) := \left\{ \sum_{i=1}^{k} a_i \mathbf{b}_i : a_i \in \mathbb{Z} \right\}.$$

1. ($\square$) Let $\mathbf{B} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Draw a plot of $\Lambda(\mathbf{B})$. Do the same for $\mathbf{C} = \begin{pmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix}$.

2. ($\star$) Let $\mathbf{B}' = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix}$. Draw a plot of $\Lambda(\mathbf{B}')$ and prove that $\Lambda(\mathbf{B}) = \Lambda(\mathbf{B}')$.

Problem 2 tells us that distinct bases can generate the same lattice. There are nice theorems that allow us to tell precisely when two bases have the same lattice: for example, one *algebraic* characterization is $\Lambda(\mathbf{A}) = \Lambda(\mathbf{B})$ if and only if $\mathbf{A} = \mathbf{B}\mathbf{U}$ for some unimodular matrix $\mathbf{U}$. We give a *geometric* characterization in the next two problems.

3. ($\star$) Define the *fundamental domain* of $\mathbf{B}$ to be the set

$$\Pi(\mathbf{B}) := \left\{ \sum_{i=1}^{n} \xi_i \mathbf{b}_i : \xi_i \in [0, 1) \right\}.$$

   With $\mathbf{B}$ and $\mathbf{B}'$ from Problem 1 and 2, respectively, draw $\Pi(\mathbf{B})$ and $\Pi(\mathbf{B}')$. Does $\Pi(\mathbf{B}')$ contain any lattice points of $\Lambda(\mathbf{B})$?

4. ($\star\star$) Let $\mathcal{L}$ be a lattice and $\mathbf{B} = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \end{pmatrix}$, where the $\mathbf{b}_i$ are linearly independent and each $\mathbf{b}_i$ belongs to $\mathcal{L}$. Prove that $\Lambda(\mathbf{B}) = \mathcal{L}$ if and only if $\Pi(\mathbf{B})$ has no nonzero points from $\mathcal{L}$, i.e., $\Pi(\mathbf{B}) \cap \mathcal{L} = 0$. Hint: pick a lattice point $\mathbf{a} \in \mathcal{L}$ and write $\mathbf{a}$ as an $\mathbb{R}$-linear combination of the $\mathbf{b}_i$. What does it mean to say that $\mathbf{a}$ is in $\Lambda(\mathbf{B})$?

Let $\mathcal{L}$ be a lattice and $\mathbf{B}$ be a basis for $\mathcal{L}$. We define the *determinant* of the lattice, denoted $\det(\mathcal{L})$, to be the volume of $\Lambda(\mathbf{B})$. As the notation suggests, the determinant of $\mathcal{L}$ is invariant under the choice of basis. Here is a celebrated result of Hermann Minkowski:

Theorem 1. *(Minkowski). Let $\mathcal{L}$ be a lattice and S be a convex subset of $\mathbb{R}^n$ that is symmetric about the origin. If $\mathrm{Vol}(S) > 2^n \det(\mathcal{L})$, then S contains a nonzero point of $\mathcal{L}$.*

5. ($\star\star$) Prove that every prime $p \equiv 1 \pmod 4$ can be written as the sum of two squares. Hint: if $p \equiv 1 \pmod 4$ then -1 is a quadratic residue mod p, i.e., -1 is the square of some number mod p. Consider the lattice in $\mathbb{R}^2$ generated by $(1, r)$ and $(0, p)$, where $r^2 \equiv -1 \pmod p$, and apply Minkowski's theorem to the ball centered at the origin with judicious choice of radius.

6. ($\star\star\star$) Prove that every prime can be written as the sum of four squares.

Exercise 3: Matrix Exponentiation. We previously considered how to efficiently multiply matrices, so it's a natural next step to consider how to efficiently exponentiate matrices. Raising matrices to a power is useful—for instance, if A is the adjacency matrix for a graph G, computing A^k will yield all paths of length k in graph G (we encourage you to verify this yourself; it's a useful property).

Assume $A \in R^{n \times n}$ and $k \in \mathbb{Z}^+$, where R is any commutative ring (for simplicity, you can let it be $\mathbb{Z}$ or $\mathbb{R}$). Assume that arithmetic operations in R cost $O(1)$, and matrix multiplication in $R^{n \times n}$ can be done in $O(n^\omega)$ where $2 \leq \omega \leq 3$.

1. ($\star$) What is the most straightforward algorithm to compute A^k? What is the time complexity of this approach?

2. ($\star\star$) Show that we can use divide and conquer to compute A^k more efficiently.

However, when k is sufficiently larger than n, there is a more efficient approach, relying more heavily on the structure of matrices. It will be helpful (read necessary) to use the following result.

Theorem 2. *(Cayley-Hamilton) Consider $A \in R^{n \times n}$. Let its characteristic polynomial $p_A \in R[x]$ be defined as:*

$$p_A(\lambda) = \det(A - \lambda I) = \lambda^n + c_{n-1}\lambda^{n-1} + \ldots + c_0$$

Then A satisfies its characteristic polynomial. That is to say:

$$p_A(A) = A^n + c_{n-1}A^{n-1} + \ldots + c_0 I = \mathbf{0}$$

You may assume that we can compute the characteristic polynomial of a matrix in $O(n^\omega)$.

3. ($\star$) Verify on some reasonably small matrix that $p_A(A) = 0$.

4. ($\star$) Convince yourself that λ^k can be expressed as $q(\lambda)p_A(\lambda) + r(\lambda)$, for polynomials q and r, where the degree of r is less than n. Note that p_A is **monic** by definition!

5. ($\star\star$) Apply Cayley-Hamilton to the previous expression. Explain why this helps us compute A^k. Does this generalize to any polynomial of our matrix instead of just exponentiation?

6. ($\star\star$) How can we find our polynomial $r(\lambda)$? How do we evaluate $r(A)$? What would the total time complexity be? Find a condition on n and k where this would give a better bound than the algorithm found in part 2.

7. ($\star\star\star$) Assume that $R = \mathbb{C}$, and we somehow still have infinite-precision arithmetic in $O(1)$ time (this is not too realistic, but analyzing numerical stability here is not the point). Let f be any function that is analytic around the eigenvalues λ_i of A. For example, the exponential function $f(x) = e^x$ converges everywhere in the complex plane, and can be expressed as:

$$f(x) = \sum_{i=0}^{\infty} \frac{1}{n!} x^n$$

Assume that you are given access to $f(\lambda_i)$ of A and know all λ_i are unique. How can you compute $f(A)$ efficiently?