4 Problems

1. Is a given number $N$ prime?
2. What is a greatest common divisor (GCD) of given numbers $N$ and $M$?
3. Given a number $N$, factor it.
4. Given a polynomial eq'n (eg, $x^2+y^2=z^2$), does it have an integral solution?
1. Eratosthenes's Sieve

Write 1, 2, 3, 4, ..., N. Cross out all multiples of 2, then 3, then 5, ...

The survivors are primes.

Agrawal, Kayal, Saxena 2003: Polytime Primality Testing!

2. Euclid's Algorithm

\[ \text{GCD}(M, N) \]

\[
\begin{align*}
\rho_0 &= M \\
\rho_1 &= N \\
n &= 1 \\
\text{while } \rho_n \neq 0 &\quad \begin{cases} 
\rho_{n+1} &= \rho_n \text{ rem } \rho_i \\
n &= n+1
\end{cases} \\
\text{return } \rho_{n-1}
\end{align*}
\]
3. Factoring not known in PolyTime

Peter Shor 1994:
Quantum PolyTime Algo for Integer Factoring

RSA assumes Factoring is hard (for security)!

4. Solving Diophantine Equations is undecidable

Yuri Matiyasevich 1970
(based on [Robinson, Davis, Putnam])

Alan Turing:
- Turing machine
- Undecidable problems exist!
Efficient Computation

ENIAC 1940's

1960s: \( P = \text{Polynomial Time Solvable} \approx \text{ Efficient} \)

**Why P?**

- Independent of the computer architecture.
- Can be defined without computers at all (say, using logic).
- Captures many natural problems.
Complexity Theory = Computability Theory with Limited Resources (time, space, ...)

Goals of Complexity Theory

- Understand what can & can't be efficiently computed.
- Taxonomy of problems according to required resources (time, space, ...)
- Study relations among complexity-theoretic concepts (algorithms, lower bounds, ...)
Some recent successes

- Primality in P: [AKS 2004]
- ST-connectivity for undirected graphs in deterministic Logspace: [Reingold '05]
- PCP Theorem: [AS '98, ALMSS '98]
- Hardness \Rightarrow Pseudorandomness: [BM '89, Y '82, NW '94, ...]

This Course

- What is known?
- What is conjectured?
- Open questions
- Ideas & Algorithmic techniques behind complexity results.
Review

1. Problems & Languages
2. Turing machines
3. Complexity Classes
4. Reductions
5. Completeness

\[ f : \Sigma^* \rightarrow \Sigma^* \]

\[ \Sigma = \{ 0, 1 \} \]

\[ \Sigma^* = \{ \text{all finite strings over } \Sigma \} \]
Decision problem

\[ f : \Sigma^* \rightarrow \{0, 1\} \]

Language \( L \subseteq \Sigma^* \)

decision problem \( f(x) = \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{o.w.} \end{cases} \)

e.g., SAT = \{ \Phi \mid \Phi \text{ is satisfiable} \}
Turing machine

1930's: Human computers

In each step:
- scan the current sheet
- update it (based on her internal state)
- move to the sheet on your left or right
- possibly change your internal state
TM $M = (Q, \Sigma, \delta, q_0, q_{acc}, q_{rej}, \Gamma)$

- $Q = \text{set of states}$ (finite)
- $\Sigma = \text{input alphabet}$
- $\Gamma = \text{tape alphabet}$

  $\Gamma \supseteq \Sigma$

  e.g., $\Sigma = \{0, 1\}$
  $\Gamma = \{0, 1, \_, \#, \}$

- $q_0 = \text{initial state}$
- $q_{acc} = \text{accepting state}$
- $q_{rej} = \text{rejecting state}$

  $S : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, -\}$
Computation of a TM

\[ u \ x_1 \ x_2 \ x_3 \ \ldots \ x_n \ u \]

Input tape

Input \( x = x_1 \ldots x_n \in \Sigma^* \)

\( q_0 \) follows 8 sequence of steps

Step 1

\[ q_0 \]

\( s(q_0, 0) = (q_1, 1, 2) \)

Step 2

\[ q_1 \]

Possible for TM to run forever

Impossible to tell if a given TM halts
on a given input:
The Halting Problem!
Multi-tape TMs

\[ S : Q \times \Gamma^k \rightarrow \Delta \times \Gamma^k \times Q \times \Gamma^k \times \{L, R, -\} \]

Thm: Every k-tape TM running in time t
can be simulated by a 1-tape TM in time $O(t^2)$.  

Idea:

Is there a faster way?  

No!
Palindromes:

\[ PAL = \{ w \in \{0, 1\}^* \mid w = w^R \} \]

Examples:
- 0110
- 010
- 11
- 01

Want a 1-tape TM to decide \( PAL \).

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow \ldots \rightarrow q_n \rightarrow \]

Want to know: \( \text{Is } a = a^R? \)

- \( q_0 \) remembers 0
- \( (p_1, -11-1) \)
- \( q_0 \rightarrow (p_0, L, R) \)
(p₀, 0) → (p₀, 0, R)
(p₀, 1) → (p₀, 1, R)
(p₀, U) → (p₀, U, L)
(r₀, 1) → reject
(r₀, 0) → (s, U, L)
(s, 0) → (s, 0, L)
(s, 1) → (s, 1, L)
(s, U) → (q₀, U, R)
(q₀, U) → 2 acc

+ take care of single symbol settings!
2-tape TMs

Set $O(n)$ (Linear) time!

Thm: PAL on 1-tape TM requires time $\Omega(n^2)$. 

Exercise!
There: Any $k$-tape TM in time $t$ can be simulated by a 2-tape TM in time $O(t \cdot \log t)$.

Universal TM

$U$ given $(M, w)$, $M$ is some TM, $w$ input,

$U$ simulates $M$ on $w$, time $O(t \cdot \log t)$.