Turing Recognizable Languages.

Given a TM $M$ and input $x \in \Sigma^*$, computation of $M$ on $x$ may never halt!

Example TM that just keeps moving its head to the right (without any other change).

$$S(q_{\text{start}}, a) = (q_{\text{start}}, a, R) \forall a \in \Gamma.$$ 

Def A language $L$ is called Turing recognizable if there is a TM $M$ such that

$x \in L \implies M$ on $x$ halts and accepts.

$x \notin L \implies M$ on $x$ either halts and rejects or never halts.

Example $L_{\text{Hilbert}} = \{ \langle p \rangle \mid P \text{ is a multivariate polynomial w/ integer coefficients that has an integer solution} \}$.

E.g. $x^2 - 5 = 0 \in L_{\text{Hilbert}}.$

$x^3 + y^3 + xy + 8 = 0 \in L_{\text{Hilbert}}.$
Claim: \( L_{\text{Hilbert}} \) is Turing recognizable.

Proof: Design a TM that

Given a polynomial \( P = P(x_1, x_2, \ldots, x_k) \), for \( n = 1, 2, 3, \ldots \),

tries all candidate solutions \( x_1, \ldots, x_k \in [-n, n] \),

and accepts if it finds a solution.

Fact!!

Diagram:

Theorem:

1. There exists a language that is not Turing recognizable.
2. \( L_{\text{Hilbert}} \) is TR but not decidable.
Def: A language $L$ is called **decidable** if there is a TM $M$ that **halts** on every input and for

\[ \forall x \in L \implies M \text{ accepts } x. \]

\[ \forall x \notin L \implies M \text{ rejects } x. \]

**Example** - $L_{\text{primes}}$, $L_{\text{conn}}$, $L_{\text{bipartite}}$ are all **decidable**.

- In fact, not easy to come up with (explicit) language that is **not decidable**.

**Encoding Turing machine descriptions**

For a TM $M = (Q, \Sigma, \Gamma, \delta, q_{\text{start}}, q_{\text{accept}}, q_{\text{reject}})$

\[ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \]

\[ k \quad l \quad m \quad 1 \quad 2 \quad 3 \]

we can write its complete description as a string, denoted as $\langle M \rangle$. (over an alphabet of fixed size say with 10 symbols)
E.g. We could write \( \langle M \rangle \) as
\[ k \neq l \neq m \neq \langle \delta \rangle \]
where \( \langle \delta \rangle \) is a list of \( k \cdot m \) instructions of the type \( (q, a) \rightarrow (q', a', L) \).

Language that is \text{TR} but not \text{decidable}

\[ L_{\text{Halt}} = \{ \langle M \rangle \mid M \text{ halts on } \varepsilon \} \]

Claim \( L_{\text{Halt}} \) is \text{Turing recognizable}.

Proof Let \( M_{\text{sim}} \) be the TM that
\[ M_{\text{sim}} := \text{"Given } \langle M \rangle \text{ as input,}
\text{Simulate } M \text{ on } \varepsilon.
\text{If } M \text{ halts then accept."} \]
Clearly,
\[ \langle M \rangle \in L_{\text{Halt}} \Rightarrow M_{\text{sim}} \text{ on } \langle M \rangle \text{ halts & accepts.} \]
\[ \langle M \rangle \notin L_{\text{Halt}} \Rightarrow M_{\text{sim}} \text{ on } \langle M \rangle \text{ never halts.} \]
Claim: $L_{halt}$ is not decidable.

Proof: Diagonalization method.

Will not cover in this class.

Ref: Text on Theory of Computation.

Henceforth:

- We'll restrict to only decidable languages.
- Let's first convince ourselves that "TM can do everything a general purpose computer can do."

Roughly: General purpose computers

$\equiv$ C-program + memory

$\equiv$ Assembly lang. + memory

$\equiv$ Fixed number of registers of fixed size and fixed set of elementary instructions

$\equiv$ TM program + infinite tape.
Convince yourselves that following languages are decided by a TM.

\[ L_{\text{Add-Unary}} = \{ a^i b^j c^{i+j} | i, j \geq 0 \} \]

\[ L_{\text{Mult-Unary}} = \{ a^i b^j c^{i+j} | i, j \geq 0 \} \]

\[ L_{\text{Unary-Binary}} = \{ a^i \neq \chi | i \geq 0, \chi \in \{0,1\}^*, \chi \text{ represents } i \text{ in binary} \} \]

**Running Time**

**Def** A language \( L \) is decided by a TM \( M \) in time \( t(n) \), \( t: N \rightarrow N \) if

- \( L \) is decided by \( M \),
- \( \forall \xi \in \Sigma^*, |\xi| = n \), \( M \) halts in at most \( t(n) \) steps.

**Example**

\[ L_{\text{Palindrome}} = \{ w | w = w^R \} \]

is decided in time \( O(n^2) \).
Def Let $t(n): \mathbb{N} \to \mathbb{N}$. "reasonable".

$\text{DTIME}(t(n))$ is the class of all languages decidable in time $t(n)$.

Def The class $P$.

$$P = \bigcup_{k=1,2,3,\ldots} \text{DTIME}(n^k).$$

- $P$ is thought of as the class of all problems that have "efficient" algorithm.
- TM model is polynomial time equivalent to
  - its own variants (e.g., k-tape TM)
  - "all" known models that are "realistic".
- Church-Turing thesis.
- Quantum computer? Hmm...
Examples of problems in $\mathcal{P}$

- **PALINDROMES** = $\{w \mid w = w^R\}$

- **PATH** = $\{<G, s, t> \mid G$ is a directed graph, $s, t$ are nodes, there is a $s \rightarrow t$ path.$\}$

- All problems in the course so far for which we designed polyme algorithms. (Recast as decision problems, e.g.

  $\text{MST} = \{<G, \text{cost}, C> \mid G$ is a graph with a cost function on edges, has a spanning tree of cost $\leq C.$ $\}.$

- **Unary Prime** = $\{#^n \mid n$ is a prime.$\}$

- **PRIME** = $\{<n> \mid <n>$ is the binary representation of $n,$ $n$ is a prime.$\}$

- **REL. PRIME** = $\{<m, n> \mid m, n$ in binary, $\gcd(m, n) = 1.$ $\}$

  Euclid's Algorithm!