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).

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

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

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

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

Example

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

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

$x^3 + y^3 + xy + 8 = 0 \in L_{Hilbert}$

Claim $L_{Hilbert}$ is Turing recognizable.

Proof Design a TM that

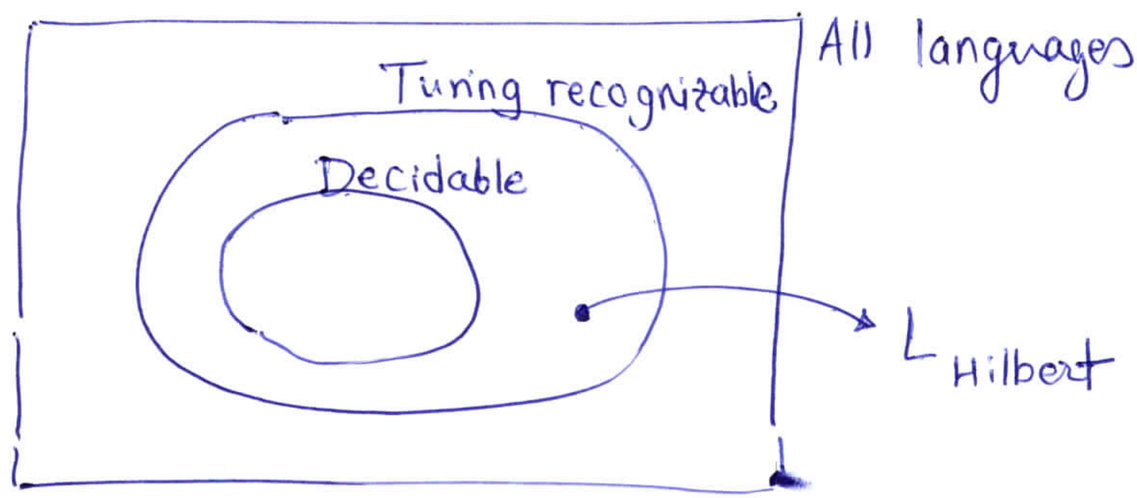
Given a polynomial $p = p(x_1, x_2, .., x_k)$,

for $n = 1, 2, 3, ...$

tries all candidate solutions $x_1, .., x_k \in [-n, n]$,

and accepts if it finds a solution. ▨

Fact !!!



Theorem

① There exists a language that is not Turing recognizable.

② $L_{Hilbert}$ 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

$x \in L \Rightarrow M$ accepts x.

$x \notin L \Rightarrow M$ rejects x.

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

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

Encoding Turing machine descriptions

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

$$\begin{array}{ccccccc} Q & \Sigma & \Gamma & & q_{start} & q_{accept} & q_{reject} \\ \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow & \downarrow \\ k & \ell & m & & 1 & 2 & 3 \end{array}$$

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

$$k \# l \# m \# \langle \delta \rangle$$

where $\langle \delta \rangle$ is a list of $k \cdot m$ instructions of the type $(q, a) \to (q', a', L)$.

Language that is TR but not decidable

Halting problem!

$$L_{Halt} = \{ \langle M \rangle \mid M \text{ halts on } \varepsilon \}.$$

empty string input

Claim L_{Halt} is Turing recognizable.

Proof Let M_{sim} be the TM that

M_{sim} := " Given $\langle M \rangle$ as input,

Simulate M on ε.

If M halts then accept. "

Clearly,

$\langle M \rangle \in L_{Halt} \Rightarrow M_{sim}$ on $\langle M \rangle$ halts & accepts.

$\langle M \rangle \notin L_{Halt} \Rightarrow M_{sim}$ on $\langle M \rangle$ never halts.

$\blacksquare$

<u>Claim</u> L_{Halt} is not decidable !

<u>Proof</u> Diagonalization method.

Will not cover in this class.

Ref: Text on Theory of Computation. ▨

<u>Henceforth</u>

- 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 computer

$\equiv$ C - program + memory

$\equiv$ Assembly lang. program + memory

$\equiv$ Fixed number of registers of fixed size <u>and</u> fixed set of elementary instructions + memory

$\equiv$ TM program + infinte tape. ▨

Convince yourselves that following languages are decided by a TM.

$$L_{\text{Add-unary}} = \{ a^i b^j c^{i+j} \mid i,j \geq 0 \}.$$

$$L_{\text{Mult-Unary}} = \{ a^i b^j c^{i \cdot j} \mid i,j \geq 0 \}.$$

$$L_{\text{Unary-binary}} = \left\{ a^i \# x \;\middle|\; \begin{array}{l} i \geq 0,\ x \in \{0,1\}^*, \\ x \text{ represents } i \text{ in binary} \end{array} \right\}.$$

Running Time

Def A language L is decided by a TM M in time $t(n)$, $t: \mathbb{N} \to \mathbb{N}$ if

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

Example $L_{\text{palindrome}} = \{ w \mid w = w^R \}$

is decided in time $O(n^2)$.

Def Let $t(n) : \mathbb{N} \to \mathbb{N}$ "reasonable".

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

Def The class P.

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

- P is thought of as the class of all problems that have "efficient" algorithm.
- TM model is polynonial 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 P

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

- PATH $= \{ \langle G, s, t \rangle \mid$ G is a directed gr, s, t are nodes, there is $s \rightsquigarrow t$ path. $\}$

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

  MST $= \{ \langle G, \underline{cost}, C \rangle \mid$ G is a graph with $\underline{cost}$ function on edges, has spanning tree of cost $\leq C$. $\}$)

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

- PRIME $= \{ \langle n \rangle \mid \langle n \rangle$ is the binary rep. of n, n is a prime. $\}$

- REL-PRIME $= \{ \langle m, n \rangle \mid$ m, n in binary. $\gcd(m, n) = 1 \}$. !!!

  Euclid's Algorithm!