Turing Machine: 7-tuple (Q, ,, , , , ) such that: Q is set of all states. is the input alphabet not containing blank symbol; is the tape alphabet containing the blank symbol; is the transition function; and the rest are start, accept and reject state respectively.

Configuration: current state, tape contents, head location

Turing-Recognizable: a Turing Machine recognizes (accept) it

Turing-decidable: for some language, if some Turing Machine decides (does not loop) on it.

Variants of Turing Machine: to show equivalency, simulate each possible operation (two ways).

Nondeterministic Turing Machines: Can try many possible branches at one point of execution.

Every Nondeterministic TM as an equivalent deterministic TM: Try all possible options, accept if reaches , else does not terminate.

Enumerator: TM that starts with blank input on tape, prints out the language it accepts.

A language is Turing Recognizable if and only if some enumerator enumerates it.

- Decidable by directly testing

Similarly, , (for regular expressions) are also decidable.

Reason: NFA has equivalent DFA, REX has equivalent NFA.

- Decidable by BFS

- Decidable by constructing such that

Every Context-free language is decidable.

– Turing Recognizable, not decidable.

Two sets are same-size if there’s a both one-to-one and onto function between these sets.

A Set is countable if it is finite or same size with N = {1, 2, 3…}

R (set of real numbers) is uncountable.

co-Turing-recognizable: if (for some language) is the complement of a Turing recognizable language.

A language is decidable if and only if it is Turing-recognizable and co-Turing-recognizable.

-- undecidable. Prove by assume decidable and show is decidable based on assumption.

is undecidable. Prove by reducing from .

-- undecidable.

-- undecidable, prove by reducing from (with one input TM being a TM that rejects all strings).

Computable function: a function is computable function if there’s some Turing Machine M, on every input , halts with on tape only.

Language A is mapping reducible to Language B (), if there’s a computable function such that . (The function is called reduction).

If is decidable and , then is decidable.

If is undecidable and , then is undecidable.

If is Turing-recognizable and , then is Turing-recognizable.

If is not Turing-recognizable and , then is not Turing-recognizable.

To prove is unrecognizable, we can prove .

is neither Turing-recognizable nor co-Turing-recognizable.

Rice’s theorem: let P be any non-trivial property of the language of a TM, determining whether a given TM’s language has property P is undecidable.

: time complexity class -> collections of all languages that are decidable by an time Turing Machine.

: class of all languages that are decidable in polynomial time on single-tape Turing Machine. In other words: .

Every CFG is a member of P.

A verifier for a language is an algorithm , where . A polynomial-time verifier runs in polynomial time in the length of . is called certificate.

is the class of languages that have polynomial time verifier.

: languages decided by time nondeterministic Turing Machine.

polynomial time (mapping) reducible: if polynomial time reduction function exists. (written )

If and , then .

: a problem is if , and every is polynomial time reducible to .

If , and for some , then is .

problems:

Hamiltonian path: go through each vertex in graph exactly once.

by reducing from

Clique: subgraph that every pair of node is connected via an edge.

Cook-Levin Theorem.

subset of nodes which every edge of G touches one of these nodes.

Space Complexity is a function which value is:

For deterministic TMs: maximum number of tape cells that TM scans on any input of length .

For non-deterministic TMs: max number of tape cells that TM scans on any branch.

is language decided by space deterministic TMs

is similar but by nondeterministic TMs.

Savitch’s Theorem: for any function where , .

is collection of languages decidable in polynomial space on a deterministic TM.

is similar but by nondeterministic TMs. (­­ by Savitch)

Generally: .

A language is if:

* It is in .
* It is (every problem is polynomial-time reducible to it).

problems:

is a true fully quantified Boolean formula

Player E has a winning strategy in the formula game with .

Formula game: Player A assigns value for variables, and Player E assigns value for , and Player E can always win no matter which values Player A assigns.

Player I has winning strategy for a GG game starting at node in graph

GG game: take turns to extend a simple path until one fails to extend the path anymore.

is set of language decidable in logarithmic space on deterministic TM.

is similar but on nondeterministic TM. ( is in NL)

A Turing Machine that uses space runs in time .

Log-space reducible function: starts with and halt with only on the tape in space.

Log-space reducible (): mapping reducible with log-space reducible function.

A Language is if:

* It is in .
* Every language in is log-space reducible to it.

If and , then .

is . Thus every problem is log-space (thus polynomial time) reducible to . Since , we have .

Interactive Proof System:

* Verifier: a function with following inputs:
* Input String (the objective is to check if input string is part of a language).
* Random input (equivalent of letting verifier do random moves).
* Partial message history (between verifier and prover).
* Prover: party with unlimited computational ability, a function with following inputs:
* Input String
* Partial message history

Prover’s output is verifier’s next input. means that prover sends to verifier after exchanging first messages so far.

* Interaction: if there exists which are a legal message exchange history and the last state is accept.

[Assuming the message length and amount of message exchange are the same, and polynomial bounded of input length.]

: set of language (say ) if there exists some polynomial-time computable function such that for some (arbitrary) function , and for all function  :

is a cnf formula with exactly satisfying assignments.