**Problems for Recognizers**

Acceptance:
\[ A_M = \{ \langle M, w \rangle | M \text{ is a machine that accepts } w \} \]

Emptiness
\[ E_M = \{ \langle M \rangle | M \text{ is a machine with } L(M) = \emptyset \} \]

Universality
\[ ALL_M = \{ \langle M \rangle | M \text{ is a machine with } L(M) = \Sigma^* \} \]

Equivalence
\[ EQ_M = \{ \langle A, B \rangle | A, B \text{ are machines with } L(A) = L(B) \} \]

All decidable for DFA/NFA/REX (convert to minimized DFA)
\[ A_{CFG}, E_{CFG} \text{ decidable, } ALL_{CFG}, EQ_{CFG} \text{ not decidable} \]

\[ A_{TM}, HALT_{TM}, EQ_{TM}, \text{ etc. not decidable} \]

**Proving Undecidability of a Language L**

**Diagonalization** (directly)

- e.g., for proving \( A_{TM} \) undecidable

Reducing from \( A_{TM} \). Example: \( E_{TM} \).

Assume decider \( D \) for \( E_{TM} \). Build decider for \( A_{TM} \).

Construct a TM \( M_1 \) that will either have an empty language or not, depending on whether \( M \) accepts \( w \).

- won’t ever run \( M_1 \), but feed as input to \( D \)

On input \( x \):
- if \( x \neq w \), reject
- otherwise, run \( A \) on \( w \) (= \( x \)), accept if \( A \) does

Thus, \( L(M_1) = \{ w \} \) if \( A \) accepts \( w \), \( \emptyset \) otherwise

Could use \( D \) to decide \( A_{TM} \).

**Proving a Language is not Turing-recognizable**

Similar idea, but reduce from \( \overline{A_{TM}} \).

\( EQ_{TM} \) not Turing-recognizable nor co-Turing-recognizable.

On input \( \langle M, w \rangle \), construct two machines:

\[ M_1: \text{ rejects any input / } M_{all}: \text{ accepts any input} \]

\[ M_{eq} \text{ accept all } \emptyset \text{, according to run of } M \text{ on } w \]

\[ M_{eq} \text{ not } EQ \text{ } M_{w} \text{ iff } M \text{ accepts } w \]

\[ \overline{A_{TM}} \leq_m EQ_{TM}, A_{TM} \leq_m EQ_{TM} \]

\[ M_{eq} \text{ EQ } M_{w} \text{ iff } M \text{ accepts } w \]

\[ \overline{A_{TM}} \leq_m EQ_{TM}, \overline{A_{TM}} \leq_m EQ_{TM} \]

Mapping reduction: \( f(\langle M, w \rangle) = \langle M_{eq}, M_{w} \rangle \) or \( \langle M_{all}, M_{w} \rangle \)

Homework 4, accepting precisely all strings with even length.
### Reduction via Computation Histories

Linear Bounded Automata: $A_{LBA}$ decidable (finite number of configurations), but $E_{LBA}$ is not.

Set of accepting computation histories of a TM can be checked by an LBA.

Another use: all strings that are not accepting computation histories on a string $w$.
- can generate with a PDA / CFG $\Rightarrow$ ALL_CFG = undecidable (deciding $\neq \Sigma^*$ $\iff$ deciding $A_{TM}$)
- can generate via extended regular expressions
  use to prove that equivalence of regular expressions with exponentiation is EXPSPACE-hard.

### Mapping Reducibility

**Def.** A function $f : \Sigma^* \rightarrow \Sigma^*$ is a **computable function** if some Turing machine $M$, on input $w$, halts with just $f(w)$ on tape.

**Def.** A language $A$ is **mapping reducible** to language $B$ (written $A \leq_m B$) if there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$ where for every $w$, $w \in A \iff f(w) \in B$.

YES for $A$ means YES for $B$
NO for $A$ means NO for $B$

### Using Mapping Reducibility

**Decidability**

- If $A \leq_m B$ and $B$ is decidable, then $A$ is decidable.
- If $A \leq_m B$ and $A$ is undecidable then $B$ is undecidable.

**Turing-recognizability**

- If $A \leq_m B$ and $B$ is Turing-recognizable, then $A$ is Turing-recognizable.
- If $A \leq_m B$ and $A$ is not Turing-recognizable then $B$ is not Turing-recognizable.

### Recursion Theorem

A TM can obtain and execute its own description.

Use: e.g., in proofs by contradiction (do something else than description says)
- e.g. assume $A_{TM}$ has a decider $H$

Construct a TM $B$:

1. Obtain own description $\langle B \rangle$
2. Run $H$ on input $\langle B, w \rangle$
3. Do the opposite of $H$ (accept/reject)

### Descriptive Complexity

The **minimal description** of a binary string $x$ is the shortest string $\langle M, w \rangle$ where $M$ halts on input $w$ with $x$ on tape.

The **descriptive complexity** (Kolmogorov complexity) is the length of the minimal description: $K(x) = |d(x)|$

**Def.** A string $x$ is $c$-compressible if $K(x) \leq |x| - c$.

- incompressible = not 1-compressible.
- Most strings are close to incompressible.
- Incompressible strings are undecidable.
- Can only enumerate a finite subset.

### Polynomial Verifiers and NP

**Def.** A **verifier** for a language $A$ is an algorithm $V$, where

$$A = \{ w | V \text{ accepts } \langle w, c \rangle \text{ for some string } c \}$$

A polynomial-time verifier runs polynomial in the length of $v$.

A language is **polynomially verifiable** if it has a polynomial time verifier.

**NP** is the class of languages that have polynomial-time verifiers.

Equivalent: A language is in NP if it is decided by some nondeterministic polynomial time Turing machine.
NP-completeness

Def. A language $B$ is NP-complete iff
1. $B$ is in NP
2. for any $A$ in NP, we have $A \leq_P B$
   - If $B$ is NP-complete and $B \leq_P C$, then $C$ is NP-complete
     - reduce known NP-complete problem $B$ to target $C$
     - reduce target problem $C$ from NP-complete problem $B$
   - If $B$ is NP-complete and $B \in P$, then $P = NP$
All NP-complete problems are polynomially reducible to one another (the hardest problems in NP)

Time Complexity

Time complexity class $\text{TIME}(t(n)) = \text{all languages that are decidable by an } O(t(n)) \text{ (deterministic, single-tape) Turing machine.}$

A $t(n)$ multivariate TM has an equivalent $O(t^2(n))$ single-tape TM.

Every $t(n)$ nondeterministic TM has an equivalent $2^{O(t(n))}$ deterministic single-tape TM.

Savitch’s Theorem

For any function $f : N \rightarrow R^+$, where $f(n) \geq n$,

$\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$

$\implies \text{NPSPACE} = \text{PSPACE}$

Space Hierarchy Theorem: For any space constructible function $f : N \rightarrow N$, there exists a language that is decidable in $O(f(n))$ space but not $o(f(n))$ space.

$\text{SPACE}(n^{c_1}) \subseteq \text{SPACE}(n^{c_2})$ for any real $c_1, c_2 > 0$

Every $t(n)$ nondeterministic TM has an equivalent $2^{O(t(n))}$ deterministic single-tape TM.

Time Hierarchy Theorem: For any time constructible function $t : N \rightarrow N$, there exists a language that is decidable in $O(t(n))$ time but not in time $o(t(n)/\log t(n))$.

$\text{TIME}(n^{c_1}) \subseteq \text{TIME}(n^{c_2})$ for any reals $1 \leq c_1 < c_2$

Satisfiability problems

"Most general": satisfy all constraints
- Circuit-SAT
- SAT
- 3-SAT
- NAE-SAT (not-all-equal)
### Covering Problems
Achieve some global goal with few elements
- **Vertex Cover**: cover edges with vertices
- **Set Cover**: cover entire set with subsets
- **Hitting Set**: cover subsets with elements
- **Dominating Set**: cover self and neighbor vertices

### Packing Problems
Choose many elements while avoiding conflicts
- **Independent Set**: vertices with no edges
- **Set Packing**: non-intersecting subsets

**Polynomial**
Matching (edges with no common endpoints)
case of bipartite graphs (network flow)

### Sequencing problems
- **Hamiltonian Path** (all nodes)
  reduction to cycle: extra node, connected to all others
- **Hamiltonian Cycle**
  reduction to path: split a node, add an endpoint to each half
- **Traveling Salesman Problem** minimum-length tour
  reduce from HAM-CYCLE

### Numerical Problems
- **Subset-Sum**: numbers with precise sum
  reduce from SAT: construct numbers digit-by-digit

### Partitioning / Coloring Problems
- **3-coloring**: no edge with same-color nodes
- **$k$-coloring**

**Polynomial**
- **2-coloring**: (bipartite graph)