CS3231 Final Printed Notes

Chong Chin Herng

November 2024

9 Turing Machines

Turing Machine A Turing machine $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$ is an automaton where

- Q is a finite set of states
- Σ is a finite set of input symbols
- $\Gamma \supseteq \Sigma$ is a finite set of tape symbols
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is a transition function
- $q_0 \in Q$ is the starting state
- $B \in \Gamma - \Sigma$ is the blank symbol
- $F \subseteq Q$ is a finite set of accepting states

Instantaneous Description of a Turing Machine An instantaneous description of a Turing machine with set of states Q and set of tape symbols Γ is of the form $x_0x_1 \cdots x_{n-1}qx_nx_{n+1} \cdots x_m$ where $x_i \in \Gamma$ for all i, $q \in Q$, $n > 0 \implies x_0 \neq B$, and $m \geq n \implies x_m \neq B$.

Step of a Turing Machine Intuitively, a Turing machine has an infinite tape divided into cells, a head that can read/write and move/right in one step, and a finite number of states. The step-relation $\vdash$ on the set of instantaneous descriptions of M is defined accordingly. Moreover, $\vdash^*$ is the reflexive and transitive closure of $\vdash$.

Language accepted by a Turing Machine A language L is accepted by a Turing machine M with input symbol set Σ if and only if $L = L(M) = \{w \in \Sigma^* | q_0w \vdash^* \alpha q_f \beta \text{ for some } q_f \in F\}$.

Halting A machine M with starting state q_0 halts on input w if and only if there exists an instantaneous description ID such that $q_0w \vdash^* ID$ and for all instantaneous description ID' we have $ID \nvdash ID'$.

Function Computed by Turing Machine A function $f : S \to \Gamma^*$ is computed by a Turing machine M with input symbol set $\Sigma \supseteq S$ and tape symbol set Γ if and only if

1. $f(x)$ is defined $\iff$ M halts on input x.
2. The tape symbols on the instantaneous description of M when it halts on input x can be interpreted as $f(x)$.

Computably Enumerable (CE) Language A CE language is a language accepted by a Turing machine.

Decidable Language A decidable language is a language accepted by a Turing machine that halts on all inputs.

Partially Computable Function A partially computable function is a function computed by a Turing machine.

Computable Function A computable function is a function $f : \Sigma^* \to \Gamma^*$ computed by a Turing machine.

Theorem 9.1 A Turing machine is as powerful as the following automata:

- A Turing machine with multiple but fixed number of tapes/heads.
- A Turing machine whose head can stay where it is in one step.
- A Turing machine with a storage that can store a fixed finite number of tape symbols.
- A Turing machine whose tapes consist of a fixed finite number of tracks.
- A Turing machine that can use other Turing machines as subroutines.
- A Turing machine whose tapes are semi-infinite.
- A nondeterministic Turing machine.

Conjecture 9.2 (Church-Turing Thesis) A Turing machine is as powerful as any automaton.

Code of Strings The code for a string $w \in \{0, 1\}^*$ is $1x$ (in binary) -1. The string with code i is denoted as w_i.

Code of Turing Machines Let $M = (Q, \Sigma, \Gamma, \delta, q_1, X_3, \{q_2\})$ be a Turing machine where

- $Q = \{q_1, q_2, \cdots, q_{|Q|}\}$
- $\Gamma = \{X_1, X_2, \cdots, X_{|X|}\}$
- $X_1 = 0$ and $X_2 = 1$
- $\delta = \{\delta_1, \delta_2, \cdots, \delta_{|\delta|}\}$

Let $L = D_1$ and $R = D_2$. Let the code of $((q_i, X_j), (q_k, X_l, D_m)) \in \delta$ be $0^i 1 0^j 1 0^k 1 0^l 1 0^m$. The code of M is $C_1 11 C_2 11 \cdots C_{|\delta|}$ where C_i is the code of δ_i for all i. The code number of a Turing machine is the code of its code. The Turing machine with code number i is denoted as M_i. Moreover, denote $W_i = L(M_i)$

Universal Turing Machine The universal Turing machine is a Turing machine that on input (M, w), simulates M on input w and accepts $L_u = \{(M, w) : M \text{ accepts } w\}$. If i is not the code number of any Turing machine, then M_i is defined to be a Turing machine such that for all j, M_i on input w_j accepts/rejects/does not halt if the universal Turing machine accepts/rejects/does not halt on input (i, w_j).

Theorem 9.3 $L_d = \{w_i : w_i \notin L(M_i)\}$ is not CE.

Theorem 9.4 $\overline{L_u}$ is not CE.

Theorem 9.5 If L is decidable, then $\overline{L}$ is decidable.

Theorem 9.6 L is decidable if and only if L and $\overline{L}$ are CE.

Many-One Reduction A language P_1 many-one reduces to a language P_2, or $P_1 \leq_m P_2$, if and only if there exists a computable function f such that $x \in P_1 \iff f(x) \in P_2$.

Theorem 9.7 If $P_1 \leq_m P_2$, then

- If P_2 is decidable, then P_1 is decidable.
- If P_2 is CE, then P_1 is CE.
- If P_1 is undecidable, then P_2 is undecidable.
- If P_1 is non-CE, then P_2 is non-CE.

Theorem 9.8 $L_{ne} = \{M_i : L(M_i) \neq \emptyset\}$ is CE.

Theorem 9.9 $L_e = \{M_i : L(M_i) = \emptyset\}$ is non-CE.

Corollary 9.10 L_{ne} is undecidable.

Theorem 9.11 (Rice's Theorem) For any property P about CE languages, if P is non-trivial, i.e. there exists a CE language that satisfies P and a CE language that does not, then $L_P = \{M_i : L(M_i) \text{ satisfies } P\}$ is undecidable.

Post's Correspondence Problem (PCP)

$$\begin{aligned} PCP = \{(A = w_1, w_2, \cdots, w_k, B = x_1, x_2, \cdots, x_k) : \\ \exists i_1, i_2, \cdots i_m (m > 0 \wedge w_{i_1} w_{i_2} \cdots w_{i_m} = x_{i_1} x_{i_2} \cdots x_{i_m})\} \end{aligned}$$

$$\begin{aligned} MPCP = \{(A = w_1, w_2, \cdots, w_k, B = x_1, x_2, \cdots, x_k) : \\ \exists i_1, i_2, \cdots i_m (w_1 w_{i_1} w_{i_2} \cdots w_{i_m} = x_1 x_{i_1} x_{i_2} \cdots x_{i_m})\} \end{aligned}$$

Theorem 9.12 $L_u \leq_m MPCP \leq_m PCP$.

Theorem 9.13 Let $A = w_1, w_2, \cdots, w_k$ and $B = x_1, x_2, \cdots, x_k$ by sequences of strings over Σ. Let $a_1, a_2, \cdots, a_k \notin \Sigma$. If G_A is defined to be the grammar

$$A \to w_i A a_i \text{ for all } i$$
$$A \to w_i a_i$$

and G_B is defined to be the grammar

$$B \to x_i B a_i \text{ for all } i$$
$$B \to x_i a_i$$

then $\overline{L(G_A)}$ and $\overline{L(G_B)}$ are context-free.

Theorem 9.14 The following languages are undecidable.

- $\{\text{CFG } G : G \text{ is ambiguous}\}$
- $\{\text{CFG } G_1, G_2 : L(G_1) \cap L(G_2) = \emptyset\}$
- $\{\text{CFG } G_1, G_2 : L(G_1) = L(G_2)\}$
- $\{\text{CFG } G_1, G_2 : L(G_1) \subseteq L(G_2)\}$
- $\{\text{CFG } G, \text{regular expression } R : L(R) = L(G)\}$
- $\{\text{CFG } G, \text{regular expression } R : L(R) \subseteq L(G)\}$
- $\{\text{CFG } G : L(G) = T^*\}$ where T is the underlying set of terminals

Unrestricted Grammar An unrestricted grammar (V, T, P, S) is a method to describe a language over T where

- V is a finite set of non-terminal symbols
- T is a finite set of terminal symbols such that $V \cap T = \emptyset$
- P is a finite set of productions of the form $\alpha \to \beta$, where $\alpha \in (V \cup T)^* V (V \cup T)^*$ and $\beta \in (V \cup T)^*$
- $S \in V$ is the start symbol

Steps and derivations of unrestricted grammar are similar to that of CFG.

Context Sensitive Grammar A context sensitive grammar is an unrestricted grammar with set of productions P such that for every production $\alpha \to \beta \in P$, we have $|\alpha| \leq |\beta|$.

Theorem 9.15 If G is an unrestricted grammar, then $L(G)$ is CE.

Theorem 9.16 If L is CE, then there exists an unrestricted grammar describing L.

10 Complexity

Model of Computation We assume the model of Turing machine with multiple but fixed number of tapes. The input tape is read only and the output tape is one-way write only. All other tapes are known as worktapes.

Time Complexity For deterministic Turing machines, $Time_M(x)$ is defined to be the time/number of steps used by a Turing machine M on input x before halting. If M does not halt on input x, then $Time_M(x)$ is defined to be ∞.

For nondeterministic Turing machines, $Time_M(x)$ is defined to be the maximum time used by a Turing machine M on any path on input x before halting. If M does not halt on some path on some input x, then $Time_M(x)$ is defined to be ∞.

A Turing machine M is $T(n)$ time bounded if and only if for any input x of length n, $Time_M(x) \leq T(n)$.

Space Complexity $Space_M(x)$ is defined to be the maximum number of cells touched by a Turing machine M on input x on any of its worktapes before halting. If M does not halt on input x, then $Space_M(x)$ is defined to be ∞.

For nondeterministic Turing machines, $Space_M(x)$ is defined to be the maximum number of cells touched by a Turing machine M on any path on input x before halting. If M does not halt on some path on some input x, then $Space_M(x)$ is defined to be ∞.

A Turing machine M is $S(n)$ space bounded if and only if for any input x of length n, $Space_M(x) \leq S(n)$.

Complexity Classes Let Σ be the underlying alphabet set.

$$DSPACE(S(n)) = \{L \subseteq \Sigma^* : \exists S(n) \text{ space bounded deterministic Turing machine accepting } L\}$$

$$DTIME(T(n)) = \{L \subseteq \Sigma^* : \exists T(n) \text{ time bounded deterministic Turing machine accepting } L\}$$

$$NSPACE(S(n)) = \{L \subseteq \Sigma^* : \exists S(n) \text{ space bounded nondeterministic Turing machine accepting } L\}$$

$$NTIME(T(n)) = \{L \subseteq \Sigma^* : \exists T(n) \text{ time bounded nondeterministic Turing machine accepting } L\}$$

11 NP Completeness

Efficient Complexity Classes Let Σ be the underlying alphabet set.

$$P = \cup_{k \in \mathbb{N}} DTIME(n^k)$$

$$NP = \cup_{k \in \mathbb{N}} NTIME(n^k)$$

$$coNP = \{L \subseteq \Sigma^* : \overline{L} \in NP\}$$

Theorem 11.1 $P \subseteq NP$.

Theorem 11.2 Let Σ be the underlying alphabet set. $L \in NP$ if and only if there exists a predicate $P : \Sigma^* \to \{\text{true}, \text{false}\}$ computed by a poly time bounded Turing machine and a polynomial $q(\cdot)$ such that $x \in L \iff \exists y \in \Sigma^*(|y| \leq q(|x|) \wedge P(x, y))$. y is known as a proof that $x \in L$.

Poly Time, Many-One, Reduction Let L_1 and L_2 be languages over Σ and Γ respectively. L_1 is poly time, many-one, reducible to L_2, or $L_1 \leq_m^p L_2$ if and only if there exists a function $f : \Sigma^* \to \Gamma^*$ computed by a poly time bounded Turing machine such that $x \in L_1 \iff f(x) \in L_2$.

Poly Time, Turing, Reduction Let L_1, L_2 be languages and M be an oracle deciding L_2. L_1 is poly time, Turing, reducible to L_2, or $L_1 \leq_T^p L_2$ if and only if there exists a poly time bounded Turing machine deciding L_1 that uses M as a subroutine polynomial number of times.

Log-Space Many-One Reduction Let L_1 and L_2 be languages over Σ and Γ respectively. L_1 is log-space many-one reducible to L_2, or $L_1 \leq_m^{\text{log space} L_2}$ if and only if there exists a function $f : \Sigma^* \to \Gamma^*$ computed by a $\log n$ space bounded Turing machine such that $x \in L_1 \iff f(x) \in L_2$.

NP-Hardness A language L is NP-hard if and only if $\forall L' \in NP(L' \leq_m^p L)$.

NP-Completeness A language L is NP-complete if and only if $L \in NP$ and L is NP-hard.

Theorem 11.3 $\leq_m^p$ is reflexive and transitive.

Corollary 11.4 If L is NP-complete, $L' \in NP$ and $L \leq_m^p L'$, then L' is NP-complete.

Graph A graph is an ordered pair (V, E) where

- V is a set of vertices
- $E \subseteq V \times V$ is a set of (directed) edges

Undirected Graph A graph (V, E) is undirected if and only if E is symmetric.

Cycle Let $G = (V, E)$ be a graph. A cycle of G is a sequence $v_1, v_2, \cdots, v_{k-1}, v_k \in V$ such that $v_k = v_1$ and for all i, $(v_i, v_{i+1}) \in E$ and are pairwise distinct.

Acyclicity A graph is acyclic if and only if it has no cycle.

Child and Parent Let $G = (V, E)$ be a directed graph. v is a child of u and u is a parent of v if and only if $(u, v) \in E$.

Satisfiability A literal is a Boolean variable or its negation. A clause is of the form $(l_1 \vee l_2 \vee \cdots \vee l_k)$ where l_i is a literal for all i.

$$\text{Satisfiability} = \{(\text{underlying set of Boolean variables } U, \text{set of clauses } C) : \bigwedge_{c \in C} c \equiv \text{ true}\}$$

$$\text{3-SAT} = \{(\text{underlying set of Boolean variables } U, \text{set of clauses with 3 literals } C) : \bigwedge_{c \in C} c \equiv \text{ true}\}$$

3-Dimensional Matching

$$\begin{aligned}\text{3-Dimensional Matching} = \{(X, Y, Z, S \subseteq X \times Y \times Z) : &\\ &X, Y, Z \text{ are pairwise disjoint}, |X| = |Y| = |Z| = n,\\ &\exists S' \subseteq S(|S'| = n \wedge \text{no two elements of } S' \text{ agree in any coordinate})\}\end{aligned}$$

Vertex Cover

$$\text{Vertex Cover} = \{(V, E \subseteq V \times V, k \in \mathbb{Z}^+) : \exists V' \subseteq V(|V'| \leq k \wedge \forall (u, v) \in E(u \in V' \vee v \in V'))\}$$

MAX-CUT

$$\text{MAX-CUT} = \{(V, E \subseteq V \times V, k \in \mathbb{Z}^+) : \exists \text{partition } (X, Y) \text{ of } V(|(X \times Y) \cap E| > k)\}$$

Clique

$$\text{Clique} = \{(V, E \subseteq V \times V, k \in \mathbb{Z}^+) : \exists V' \subseteq V(|V'| \geq k \wedge \forall u, v \in V'(u \neq v \implies (u, v) \in E))\}$$

Independent Set

$$\text{Independent Set} = \{(V, E \subseteq V \times V, k \in \mathbb{Z}^+) : \exists V' \subseteq V(|V'| \geq k \wedge \forall u, v \in V'(u \neq v \implies (u, v) \notin E))\}$$

Hamiltonian Circuit A Hamiltonian circuit of a graph G is a cycle $v_1, v_2 \cdots v_k, v_1$ of G such that every vertex except v_1 appears exactly once.

$$\text{Hamiltonian Circuit} = \{\text{graph } G : G \text{ has a Hamiltonian circuit}\}$$

Partition

$$\text{Partition} = \left\{(\text{finite } A, s : A \to \mathbb{R}_{\geq 0}) : \exists A' \subseteq A \left(\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)\right)\right\}$$

Set Cover

$$\begin{aligned}\text{Set Cover} = \{(\text{finite } A, \{S_1, S_2, \cdots, S_m\} \in \mathcal{P}(\mathcal{P}(A)), k \in \mathbb{Z}^+) :&\\ &\exists Y \subseteq \{1, 2, \cdots, m\}(|Y| \leq k \wedge A \subseteq \cup_{i \in Y} S_i)\}\end{aligned}$$

Traveling Salesman Problem

$$\begin{aligned}\text{Traveling Salesman Problem} = \{(V, wt : V \times V \to \mathbb{R}, B \in \mathbb{R}) : &\\ &(V, V \times V) \text{ has a Hamiltonian circuit } v_1 v_2 \cdots v_k v_1\\ &\text{such that } \sum wt(v_i, v_{i+1}) + wt(v_k, v_1) \leq B\}\end{aligned}$$

Theorem 11.5 3-SAT, 3-Dimensional Matching, Vertex Cover, MAX-CUT, Clique, Independent Set, Hamiltonian Circuit, Partition, Set Cover and Traveling Salesman Problem are NP-complete.

Theorem 11.6 If there exists an NP-complete language L such that $L \in P$, then for all $L \in NP$ we have $L \in P$. In other words, $P = NP$.

Corollary 11.7 If $P \neq NP$, then for all NP-complete languages L we have $L \notin P$.

Tutorial 8

Question 3 $f : X \to Y$ is partially computable if and only if the graph of f, $L_f = \{(x, y) : x \in X \text{ and } f(x) = y\}$ is CE.

Question 4 A 2 stack NPDA is as powerful as a Turing machine for language acceptance.

Tutorial 9

Question 1 A Turing machine M enumerates a language $L = \{x_1, x_2, \cdots\}$ if and only if M can write $x_1, x_2, \cdots$ on a one-way write only tape. L is CE if and only if some Turing machine enumerates L. If some Turing machine enumerates $L = \{x_1, x_2, \cdots\}$ such that $x_1 \leq x_2 \leq \cdots$ for some total order $\leq$, then L is decidable.

Question 2 Elements of complexity classes can change if we assume the model of Turing machine with one tape.

Question 3 CE languages are closed under union and intersection.

Question 4 If L_1 is decidable and L_2 is CE, then $L_2 - L_1$ is CE and $L_1 - L_2$ is either non-CE or decidable.

Question 5 A co-finite language is a language whose complement is finite. Every finite or co-finite language is decidable. If L is decidable and D is finite, then the symmetric difference of L and D, or $L\boldsymbol{\Delta}D = (L - D) \cup (D - L)$, is decidable. If $L = L(M)$ is CE and undecidable, then the set of inputs on which M does not halt is infinite.

Question 6 If L is a decidable language over Σ, then $\{x \in \Sigma^* : xx \in L\}$ and $\{x \in \Sigma^* : \text{some prefix of } x \text{ is in } L\}$ is decidable.

Question 7 $\{(M_i, w_j, t \in \mathbb{N}) : M_i \text{ accepts } w_j \text{ in } \leq t \text{ time}\}$ is decidable.

Question 8 (Halting Problem) $\{(M_i, w_j) : M_i \text{ halts on input } w_j\}$ is undecidable.

Tutorial 10

Question 1 (State Entry Problem for Turing Machines) $\{(M = (Q, \Sigma, \Gamma, \delta, q_0, B, F), q \in Q, w \in \Sigma^*) : M \text{ will enter } q \text{ on input } w\}$ is undecidable.

Question 2 $L_{fin} = \{M_i : W_i \text{ is finite}\}$ and $L_{inf} = \{M_i : W_i \text{ is infinite}\}$ are non-CE.

Question 3 The following languages are decidable:

- $\{(M, M') : M, M' \text{ are DFAs and } L(M) \cap L(M') = \emptyset\}$
- $\{(G, M) : G \text{ is a CFG, } M \text{ is a DFA and } L(G) \cap L(M) = \emptyset\}$

Moreover, $\{(M_i, M_j) : W_i \cap W_j = \emptyset\}$ is undecidable.