CSE596: Introduction to the Theory of Computation

Glossary

Jinghao Shi
jinghaos@buffalo.edu
Person #: `5009 4218`

December 5, 2013

This page intentionally left blank.

1 Preliminaries

1.1 Words and Language

Alphabet A finite set of symbols. $\Sigma = \{a_1, a_2, \dots, a_k\}$.

Word A finite sequence of symbols.

Language A set of words. $L \in \Sigma^*$.

1.2 Partial Functions

Partial Function $f : X' \to Y,$ where $X' \subset X$.

Total Function When $X' = X$.

Converge When $f(x)$ is defined. Noted as $f(x) \downarrow$.

Diverge When $f(x)$ is not defined. Noted as $f(x) \uparrow$.

1.3 Propositional Logic

Satisfiable A formula F is satisfiable if there exists an assignment to its variables that satisfies it.

Tautology A formula is valid (or is a tautology) if every assignment to its variables satisfies it

Conjunction $A_1 \wedge A_2 \wedge \cdots \wedge A_n$

Disjunction $A_1 \vee A_2 \vee \cdots \vee A_n$

Clause Disjunction of literals.

Conjunctive Normal Form (CNF) Conjunction of clauses.

1.4 cardinality

Same Cardinality $card(A) = card(B)$ iff. $\exists f : A \to B$ is a bijection.

Countable A set A is countable if $card(A) = card(N)$ or A is finite.

Countable Infinite $card(A) = card(N)$.

Enumerable A set is enumerable if it is the empty set or there is a function $f : N \to_{onto} A$, i.e., $A = range(f) = \{a_0, a_1, \dots\}$

Enumerable $\Rightarrow$ Countable Define h as follows:

$$h(0) = f(0)$$
$$h(n+1) = f(min\{x|f(x) \notin \{h(0), h(1), \dots, h(n)\}\})$$

- h is one-to-one since $h(n+1) \notin \{h(0), h(1), \dots, h(n)\}$.

- $range(h) \subseteq range(f) = S$.

- $f(0) = h(0)$, suppose by induction that $f(n) \in \{h(0), h(1), \dots, h(n)\}$ and $f(n+1) \notin \{h(0), h(1), \dots, h(n)\}$, then $n+1 = min\{x|f(x) \notin \{h(0), h(1), \dots, h(n)\}\}$, so $h(n+1) = f(n+1)$. So $\forall n, f(n) \in \{h(0), h(1), \dots, h(n)\}$, $S = range(f) \subseteq range(h)$.

Thus $S = range(f) = range(h)$, $h : N \rightarrow_{1-1} S$, $card(N) = card(S)$

Theorem 1.3. A set A is countable if and only if $card(A) \leq \aleph_0$.

$$
\begin{aligned}
& card(A) \leq \aleph_0 \\
\Rightarrow & \exists f : A \rightarrow_{1-1} N \\
\Rightarrow & f[A] \text{ doesn't have a largest number (otherwise, } A \text{ is finite.)} \\
\Rightarrow & a_0 = min\{f[A]\}, a_{n+1} = min\{f[A] - \{f(0), f(1), \ldots, f(n)\}\} \\
\Rightarrow & A \text{ is enumerable.} \\
\Rightarrow & A \text{ is countable.}
\end{aligned}
$$

Theorem 1.4. The set of all functions from N to N is not countable.
Let $A = \{f | f : N \rightarrow N\}$, suppose for contradiction that A is countable, then $A = \{f_1, f_2, \ldots\}$, define $g(x) = f_x(x) + 1$ and $g = f_k$ for some k, but $g(k) = f_k(k) + 1 \neq f_k(k)$.

Theorem 1.5. $\mathscr{P}(N)$ has cardinality greater than $\aleph_0$.
Let $A = \mathscr{P}(N) = \{S | S \subseteq N\}$ is power set of N. Suppose for contradiction that A is enumerable, then $A = \{S_0, S_1, \ldots\}$, define $T = \{k | k \notin S_k\}$ and $T \in A$. However, $\forall k\ T \neq S_k$ since $k \in T \Leftrightarrow k \notin S_k$.

1.5 Misc

onto/surjection $f : A \rightarrow_{onto} B$ iff. $\forall b \in B, \exists a \in A$ s.t. $f(a) = b$

one-to-one/injection $f : A \rightarrow_{1-1} B$ iff. $f(a) = f(b) \Rightarrow a = b$

bijection Both one-to-one and onto.

2 Turing Machine and RAM

2.1 Turing Machine

Turing Machine $M = \langle Q, \Sigma, \Gamma, \delta, q_0, B, q_{accept}, q_{reject} \rangle$

TM-acceptable A language $L, L \subseteq \Sigma^*$, is Turing-machine-acceptable if there is a Turing machine that accepts L.

TM-decidable A language L is Turing-machine-decidable if L is accepted by some Turing machine that halts on every input.

$L(M)$ $L(M) = \{w \in \Sigma^* | M \text{ accepts } w\}$.

M **computes** ϕ M eventually enter an accepting configuration of $\phi(w_1, \ldots, w_n)q_{accept}$ iff. $\phi(w_1, \ldots, w_n) \downarrow$.

Partial Computable A partial function ϕ is partial computable if there is some Turing machine that computes it.

Total Computable $\phi(w_1, \ldots, w_n) \downarrow$ for all $w_1, \ldots, w_n$. If M computes a total computable function, $L(M) = \Sigma^*$.

Theorem 2.2. A language L is decidable if and only if both L and $\bar{L}$ are acceptable.
Parallel simulation.

RAM

1_j	X **add**$_j$ Y	append a_j to Y
2	X **del** Y	delete right most symbol of Y
3	X **clr** Y	$Y \rightarrow \lambda$
4	X $Y \leftarrow Z$	$Y = Z$
5	X **jmp** Y	
6_j	X Y **jmp**$_j$ X'	
7	X **continue**	

3 Undecidability

3.1 Undecidable Problems

Characteristic Function $f_S(x) = \begin{cases} 0 & \text{if } x \in S \\ 1 & \text{if } x \notin S \end{cases}$

w_M The word that encodes M.

Gödel Number e The code for a Turing machine M.

$\phi_e = \lambda x.U(e,x)$. ϕ_e is the partial function of one argument that is computed by M_e.

Theorem 3.1. The Program Termination problem (Example 3.2) is undecidable. There is no algorithm to determine whether an arbitrary partial computable function is total. Thus, there is no algorithm to determine whether a Turing machine halts on every input.
Define

$$\text{TEST}(i) = \begin{cases} \text{“yes”} & \text{if } \phi_i \text{ halts on every input.} \\ \text{“no”} & \text{otherwise} \end{cases}$$

$$\delta(k) = \begin{cases} \phi_k(k) + 1 & \text{if TEST(i)=”yes”} \\ 0 & \text{if TEST(k)=”no”} \end{cases}$$

Thus δ is total computable. Let $\delta = \phi_e$, then TEST(e) is “yes”. $\delta(e) = \phi_e(e) + 1 \neq \phi_e(e)$.

3.2 Pairing Functions

Paring function Computable one-to-one mapping $<,>: N \times N \to N$, whose inverse $\tau_1(< x, y >) = x$ and $\tau_2(< x, y >) = y$ are also computable.
Example, $< x, y >= \frac{1}{2}(x^2 + 2xy + y^2 + 3x + 1)$.

3.3 Computably Enumerable Sets

Computable enumerable (c.e.) A set S is c.e. if $S = \emptyset$ or $S = range(f)$ in which f is a total computable function.

index set Let $\mathscr{C}$ be any set of partial computable functions, then $P(\mathscr{C}) = \{e|\phi_e \in \mathscr{C}\}$ is called index set.

Homework 3.3 $A = \{(e,j)|L(M_e) = L(M_j)\}$ is not decidable.
Suppose by contradiction that A is decidable, let $L(M_j) = \Sigma^*$, then $\{e|L(M_e) = \Sigma^*\} = \{e|\phi_e \text{ is total computable}\}$ is decidable, this contradicts **Theorem 3.1**.

Theorem 3.2. $\{e|\phi_e is \text{ total computable}\}$ is not computably enumerable.
Let $S = \{e|\phi_e \text{ is total computable}\}$ and suppose for contradiction that S is c.e., then $S = range(g)$ for some total computable function g. Define $U_S(e,x) = \phi_{g(e)}(x)$ and $h(x) = U_S(x,x) + 1$, so $\exists k \in S$ s.t. $h = \phi_k$ and $\exists e$ s.t. $k = g(e)$. Finally,

$$\begin{aligned} \phi_k(e) &= h(e) \\ &= U_S(e,e) + 1 \\ &= \phi_{g(e)}(e) + 1 \\ &= \phi_k(e) + 1 \end{aligned}$$

Theorem 3.3. A set S is computably enumerable if and only if there is a decidable relation $R(x, y)$ such that

$$x \in S \Leftrightarrow \exists y R(x, y).$$

Theorem 3.4. A set S is computably enumerable if and only if it is Turing-machine- acceptable.

Corollary 3.1. A set S is decidable if and only if S and $\bar{S}$ are both computably enumerable.

Corollary 3.2. A set S is computably enumerable if and only if S is the domain of some partial computable function.

Homework 3.4 Prove that an infinite set is decidable if and only if it can be enumerated in increasing order by a one-to-one total computable function.

Homework 3.6 Prove that every infinite c.e. set contains an infinite decidable subset.

$$\begin{aligned} h(0) &= f(0) \\ h(n+1) &= f(min\{x | f(x) \notin \{h(0), h(1), \ldots, h(n)\}\}) \end{aligned}$$

$W_e = dom(\phi_e)$

3.4 Complete Set

Diagonal Set $K = \{x | \phi_x(x) \downarrow\} = \{x | U(x, x) \downarrow\}$=\{TM that accepts its own code.\}. Since $\lambda x.U(x, x)$ is partial computable, K is c.e.. However, K is not decidable, in particular, $\bar{K}$ is not c.e.
Suppose $\bar{K} = W_e = dom(\phi_e)$, then $e \in \bar{K} \Leftrightarrow \phi_e(e) \downarrow \Leftrightarrow e \in K$.

Many-one reducible $A \leq_m B$ if there is a total computable function s.t. $x \in A \Leftrightarrow f(x) \in B$

Lemma 3.2. 1. If $A \leq_m B$ and B is c.e., then A is c.e.
2. If $A \leq_m B$ and B is decidable, then A is decidable.

Theorem 3.6. The Halting problem is undecidable. Specifically, the set $L_U = \{(e, w) | M_e \text{ accepts } w\}$ is not decidable.
$x \in K \Leftrightarrow (x, x) \in L_U$, $x \mapsto (x, x)$ is total, so $K \leq_m L_U$.

3.4.1 Complete Problems

Many-one complete L is many-one complete if

1. L is c.e.
2. For every c.e. set A, $A \leq_m L$

Homework 3.8 Show that K is a many-one complete set. Note that it suffices to show that $L_U \leq_m K$. Need to show $(e, w) \in L_U \Leftrightarrow f((e, w)) \in K$ for some total computable function f. Define $f((e, w)) = e'$ where M'_e is defined as follows.

on input x;
if M_e *accepts* w **then**
| ACCEPT;
else
| REJECT;
end

Then we have

$$\begin{aligned}(e,w) \in L_U &\Leftrightarrow L(M'_e) = \Sigma^* \\ &\Leftrightarrow e' \in L(M'_e) \\ &\Leftrightarrow e' \in K\end{aligned}$$

3.5 S-m-n Theorem

Corollary 3.3. For every partial computable function $\lambda x.\Psi(e,x)$, there is a total computable function f so that $\phi_{f(e)}(x) = \Psi(e,x)$.

Theorem 3.9. There is a total computable function f such that $range\phi_{f(e)} = dom\phi_e$.
Define

$$\Psi(e,x) = \begin{cases} x & \text{if } x \in dom\phi_e \\ \uparrow & \text{otherwise.} \end{cases}$$

So $range(\lambda x.\Psi(e,x)) = dom\phi_e$, and $\phi_{f(e)} = \Psi(e,x)$, so $range(\phi_{f(e)}) = dom\phi_e$.

Homework 3.9 Prove that there is a total computable function g such that $dom\phi_{g(e)} = range\phi_e$.
Define

$$\Psi(e,x) = \begin{cases} 1 & \text{if } x \in range\phi_e \\ \uparrow & \text{otherwise.} \end{cases}$$

So $range(\Psi(e,x)) = range(\phi_{g(e)}(x) = range(\phi_e)$

3.6 Recursion Theorem

Theorem 3.10. For every total computable function f there is a number n such that $\phi_n = \phi_{f(n)}$. A number n with this property is called a fixed point of f .

Corollary 3.5. There is a number (i.e., program) n such that ϕ_n is the constant function with output n.
Define $\Psi(e,x) = e$, then $\Psi(e,x) = \phi_{f(e)}(x) = \phi_e(x) = e$.

$W_n = \{n\}$ Define

$$\Psi(e,x) = \begin{cases} e & \text{if } x = e \\ \uparrow & \text{otherwise} \end{cases}$$

$\Psi(e,x) = \phi_{f(e)}(x) = \phi_e(x)$, $dom\phi_e = \{e\}$

$W_n = \{n^2\}$ Define

$$\Psi(e,x) = \begin{cases} e & \text{if } x = e^2 \\ \uparrow & \text{otherwise} \end{cases}$$

$\Psi(e,x) = \phi_{f(e)}(x) = \phi_e(x)$, $dom\phi_e = \{e^2\}$

Homework 3.11 Show that there is no algorithm that given as input a Turing machine M, where M defines a partial function of one variable, outputs a Turing machine M' such that M' defines a different partial function of one variable.
Suppose for contradiction that such $\exists f \forall n \phi_n \neq \phi_{f(n)}$, and f is total.

3.7 Rice's Theorem

Theorem 3.12. An index set $P_{\mathscr{C}}$ is decidable if and only if $P_{\mathscr{C}} = 0$ or $P_{\mathscr{C}} = N$.
Suppose $P_{\mathscr{C}} \neq \emptyset$ and $P_{\mathscr{C}} \neq N$, let $j \in P_{\mathscr{C}}$ and $k \notin P_{\mathscr{C}}$, define

$$f(x) = \begin{cases} k & \text{if } x \in P_{\mathscr{C}} \\ j & \text{if } x \notin P_{\mathscr{C}} \end{cases}$$

Suppose for contradiction that $P_{\mathscr{C}}$ is decidable, then f is total, then f has a fixed point n such that $\phi_n = \phi_{f(n)}$. Since n and $f(n)$ is the code for same partial functions, either they both belong to $P_{\mathscr{C}}$ or both belong to $\overline{P_{\mathscr{C}}}$, but $x \in P_{\mathscr{C}} \Leftrightarrow f(x) \notin P_{\mathscr{C}}$.

3.8 Turing Reductions and Oracle Turing Machines

M^A an oracle TM with A as its oracle.

Definition 3.5. A is decidable in B if $A = L(M^B)$, where M^B halts on every input.

Definition 3.6. A is Turing-reducible to B if and only if A is decidable in B. In notation: $A \leq_T B$.

Homework 3.13 Prove each of the following properties:
1. $\leq_T$ is transitive;
2. $\leq_T$ is reflexive;
3. For all sets A, $A \leq_T \bar{A}$;
4. If B is decidable and $A \leq_T B$, then A is decidable;
5. If A is decidable, then $A \leq_T B$ for all sets B;
6. $A \leq_m B \Rightarrow A \leq_T B$;
7. $\exists A, B[A \leq_T B$ and $A \nleq_m B]$;
$\bar{K} \leq_T K$ but $\bar{K} \nleq_m K$.
8. $\exists A, B[A \leq_T B$ and B is c.e. and A is not c.e.]. $\bar{K} \leq_T K$, K is c.e., $\bar{K}$ is not c.e.

4 Introduction to Complexity Theory

4.1 Complexity Classes and Complexity Measures

Online TM An online Turing machine is a multitape Turing machine whose input is written on one of the work tapes, which can be rewritten and used as an ordinary work tape.

Time-bounded M is a $T(n)$ time-bounded Turing machine if for every input of length n, M makes at most $T(n)$ moves before halting.

$DTIME(T(n))$ to be the set of all languages having time complexity $T(n)$.

$NTIME(T(n))$ to be the set of all languages accepted by nondeterministic $T(n)$ time-bounded Turing machines.

Offline TM An off-line Turing machine is a multitape Turing machine with a separate read-only input tape. The Turing machine can read the input but cannot write over the input.

Space-bounded M is an $S(n)$ space-bounded Turing machine if, for every word of length n, M scans at most $S(n)$ cells over all storage tapes.

Complexity classes

1. L = DSPACE($log(n)$)
2. NL = NSPACE($log(n)$)
3. POLYLOGSPACE = $\cup\{$DSPACE($(logn)^k$)$|k \geq 1\}$
4. DLBA = $\cup\{$DSPACE(kn)$|k \geq 1\}$
5. LBA = $\cup\{$NSPACE(kn)$|k \geq 1\}$
6. P $= \cup\,\{$DTIME(n^k)$|k \geq 1\}$
7. NP = $\cup\{$NTIME(n^k)$|k \geq 1\}$
8. E $= \cup\,\{$DTIME(k^n)$|k \geq 1\}$
9. NE = $\cup\{$NTIME(k^n)$|k \geq 1\}$
10. PSPACE = $\cup\{$DSPACE(n^k)$|k \geq 1\}$
11. EXP = $\cup\{$DTIME($2^{p(n)}$)$|p$ is a polynomial$\}$
12. NEXP = $\cup\{$NTIME($2^{p(n)}$)$|p$ is a polynomial$\}$

5 Basic Results of Complexity Theory

5.1 Linear Compression and Speedup

Big-Oh Notation $g(n) \in O(f(n)) \Leftrightarrow \exists c > 0, \forall n, g(n) \leq cf(n)$.

Theorem 5.1 (Space Compression with Tape Reduction). For every k-tape $S(n)$ space-bounded off-line Turing machine M and constant $c > 0$, there exists a one- tape $cS(n)$ space-bounded off-line Turing machine N such that $L(M) = L(N)$. Furthermore, if M is deterministic, then so is N.

Corollary 5.1. The following identities hold:

$$\text{DSPACE}(S(n)) = \text{DSPACE}(O(S(n)))$$
$$\text{NSPACE}(S(n)) = \text{NSPACE}(O(S(n)))$$

This implies DLBA = DSPACE(n), and LBA = NSPACE(n).

Theorem 5.2 (Linear Speedup). If L is accepted by a k-tape $T(n)$ time- bounded Turing machine M, $k > 1$, and if $n \in o(T(n))$, then for any $c > 0$, L is accepted by a k-tape $cT(n)$ time-bounded Turing machine N. Furthermore, if M is deterministic, then so is N.

5.2 Constructible Functions

Space-constructible There is an $S(n)$ space-bounded Turing machine M such that for each n there is some input of length n on which M uses exactly $S(n)$ cells.

Property of Space-constructible

- Space-constructible implies fully space-constructible for space bounds $S(n)$ such that $S(n) \geq n$.
- If $S_1(n)$ and $S_2(n)$ are space-constructible, then so are $S_1(n)S_2(n)$, $2^{S_1(n)}$, and $S_1(n)^{S_2(n)}$

5.3 Tape Reduction

Theorem 5.5 Let M be a k-tape $T(n)$ time-bounded Turing machine such that $n \in o(T(n))$. There is a one-tape $T^2(n)$ time-bounded Turing machine N such that $L(N) = L(M)$. Furthermore, if M is deterministic, then so is N.

Oblivious TM A Turing machine is oblivious if the sequence of head moves on the Turing machine's tapes is the same for all input words of the same length. That is, for $t \geq 1$, the position of each of the heads after t moves on an input word x depends on t and $|x|$, but not on x.

Theorem 5.6. If L is accepted by a k-tape $T(n)$ time-bounded Turing machine M, then L is accepted by an oblivious two-tape Turing machine N in time $O(T(n)logT(n))$. Furthermore, if M is deterministic, then so is N.

Theorem 5.7. If L is accepted by a k-tape $T(n)$ time-bounded non- deterministic Turing machine M, then there are a constant $c > 0$ and a two-tape nondeterministic Turing machine N that accepts L such that for each word $x \in L$, the number of steps in the shortest computation of N on x is at most $cT(n)$.

5.4 Inclusion Relationships

Theorem 5.8. For every function f , DTIME(f) $\in$ DSPACE(f) and NTIME(f) $\in$ NSPACE(f)

Theorem 5.9. If L is accepted by an $S(n)$ space-bounded Turing machine, $S(n) \geq logn$, then L is accepted by an $S(n)$ space-bounded Turing machine that halts on every input.

Theorem 5.10. NTIME($T(n)$) $\in$ DSPACE($T(n)$).

Corollary 5.7. NP $\in$ PSPACE.

Theorem 5.11. NTIME($T(n)$) $\in \cup\{$DTIME$(c^{T(n)}|c \geq 1\}$.

Corollary 5.8. If S is fully time-constructible and $S(n) \geq log(n)$, then NSPACE($S(n)$) $\subseteq \{$DTIME$(c^{S(n)}|c \geq 1\})$.

Theorem 5.13 (Savitch). If S is fully space-constructible and $S(n) \geq log(n)$, then NSPACE($S(n)$) $\subseteq$ DSPACE($S^2(n)$).

Corollary 5.9.

$$\begin{aligned}\text{PSPACE} &= \cup\{\text{DSPACE}(n^c)|c \geq 1\}\\ &= \cup\{\text{NSPACE}(n^c)|c \geq 1\}\\ \text{POLYLOGSPACE} &= \cup\{\text{DSPACE}(log(n)^c)|c \geq 1\}\\ &= \cup\{\text{NSPACE}(log(n)^c)|c \geq 1\}\end{aligned}$$

Corollary 5.10.

$$\begin{aligned}\text{NSPACE}(n) &\subseteq \text{DSPACE}(n^2)\\ \text{NL} &\subseteq \text{POLYLOGSPACE}.\end{aligned}$$

5.5 Separation Results

Theorem 5.15 (Space Hierarchy Theorem). Let $S(n)$ be fully space- constructible. There is a language $L \in$ DSPACE($S(n)$) such that for every function $S'(n)$, if $S'(n) \in o(S(n))$, then $L \notin$ DSPACE($S(n)$).

Corollary 5.13. $L \subset$ POLYLOGSPACE, POLYLOGSPACE $\subset$ DLBA, and DLBA $\subset$ PSPACE.

$$\begin{aligned}\text{POLYLOGSPACE} &= \cup\{\text{DSPACE}((logn)^k)|k \geq 1\}\\ &\subseteq \text{DSPACE}(n^{\frac{1}{2}})\\ &\subseteq \text{DLBA}.\end{aligned}$$

Corollary 5.14. LBA $\subset$ PSPACE.

$$\begin{aligned}\text{LBA} &= \text{NSPACE}(n), \text{by Corollary 5.1}\\ &\subseteq \text{DSPACE}(n^2), \text{by Theorem 5.13}\\ &\subset \text{DSPACE}(n^3), \text{by Theorem 5.15}\\ &\subseteq \text{PSPACE}.\end{aligned}$$

Theorem 5.16 (Time Hierarchy Theorem). Let T be a fully time-constructible function and assume that there exists a function $T'(n)$ so that $T'(n)log(T'(n)) \in o(T(n))$. Then there is a language $L \in$ DTIME($T(n)$) such that for every function $T'(n)$ such that $T'(n)log(T'(n)) \in o(T(n))$, $L \notin$ DTIME($T'(n)$).

Corollary 5.15. For every constant $c > 0$, DTIME(n^c) $\subset$ DTIME(n^{c+1}) and DTIME(2^{cn}) $\subset$ DTIME($2^{(c+1)n}$).

Corollary 5.16. P $\subset$ E and E $\subset$ EXP.

5.6 Translation Techniques and Padding

Lemma 5.2. Let $S(n)$ and $f(n)$ be fully space-constructible functions, where $S(n) \geq n$ and $f(n) \geq n$. For a language L, define $p(L) = \{x10^i|x \in L \text{ and } |x10^i| = f(|x|)\}$. Then $L \in$ NSPACE($S(f(n))$) $\Leftrightarrow p(L) \in$ NSPACE($S(n)$).

Theorem 5.17. Let $S_1(n), S_2(n)$, and $f(n)$ be fully space-constructible functions, where $S_1(n) \geq n, S_2(n) \geq n$ and $f(n) \geq n$. Then $\text{NSPACE}(S_1(n)) \subseteq \text{NSPACE}(S_2(n))$ implies $\text{NSPACE}(S_1(f(n))) \subseteq \text{NSPACE}(S_2(f(n)))$.

Example 5.5. $\text{NSPACE}(n^2) \subset \text{NSPACE}(n^3)$.
Suppose for contradiction that $\text{NSPACE}(n^3) \subseteq \text{NSPACE}(n^2)$, then we have

$$\text{NSPACE}(n^6) \subseteq \text{NSPACE}(n^4) \text{ with } f(n) = n^2$$
$$\text{NSPACE}(n^9) \subseteq \text{NSPACE}(n^6) \text{ with } f(n) = n^3$$

Then we have the following.

$$\begin{aligned}
\text{NSPACE}(n^9) &\subseteq \text{NSPACE}(n^6) \\
&\subseteq \text{NSPACE}(n^4) \\
&\subseteq \text{DSPACE}(n^8), \text{ by Savitch theorem} \\
&\subset \text{DSPACE}(n^9), \text{by space hierarchy theorem} \\
&\subseteq \text{NSPACE}(n^9)
\end{aligned}$$

Example 5.6. We use the analog of Theorem 5.17 for deterministic time to show that $\text{DTIME}(2^n) \subset \text{DTIME}(n2^n)$.
Suppose for contradiction that $\text{DTIME}(n2^n) \subseteq \text{DTIME}(2^n)$, then we have

$$\text{DTIME}(2^n 2^{2^n}) \subseteq \text{DTIME}(2^{2^n}) \text{ with } f(n) = 2^n$$
$$\text{DTIME}((n+2^n)2^{n+2^n}) \subseteq \text{DTIME}(2^{n+2^n}) \text{ with } f(n) = n + 2^n$$
$$\text{DTIME}((n+2^n)2^n 2^{2^n}) \subseteq \text{DTIME}(2^{2^n}) \text{ combine above two.}$$

Which violate the time hierarchy theorem.

5.6.1 Tally Languages

Definition For $L \in \Sigma^*$, let Tally(L) $= \{1^{n(w)} | w \in L\}$.

Theorem 5.18. NE $\subseteq$ E if and only if every tally language in NP belongs to P.

Corollary 5.17. P = NP implies E = NE.

6 Nondeterminism and NP-Completeness

6.1 Characterizing NP

Theorem 6.1. A set A belongs to NP if and only if there exist a polynomial p and a binary relation R that is decidable in polynomial time such that for all words in Σ^*, $x \in A \Leftrightarrow \exists y[|y| \leq p(|x|) \wedge R(x, y)]$.

Verifier Define a verifier for a language A to be an algorithm V such that $A = \{x | \exists y[V \text{ accepts } \langle x, y\rangle]\}$.

Corollary 6.1. NP is the class of all languages A having a polynomial-time verifier.

6.2 The Class P

6.3 Enumerations

Definition 6.1. A class of sets $\mathscr{C}$ is effectively presentable if there is an effective enumeration $\{M_i\}_i$ of Turing machines such that every Turing machine in the enumeration halts on all inputs and $\mathscr{C} = \{L(M_i) | i \geq 0\}$.

Theorem 6.2. There is no effective enumeration of the class of all deterministic Turing machines that operate in polynomial time. That is, $S = \{i | \mathrm{DM}_i \text{operates in polynomial time}\}$ is not a computably enumerable set.

Theorem 6.3. P and NP are effectively presentable:
$\mathrm{NP} = \{\mathrm{L}(\mathrm{NP}_i) | i \geq 0\}$;
$\mathrm{P} = \{\mathrm{L}(\mathrm{P}_i) | i \geq 0\}$;

6.4 NP-Completeness

Definition 6.2. A set A is many-one reducible in polynomial time to a set B (notation: $A \leq_m^P B$) if there exists a function f that is computable in polynomial time so that $x \in A \Leftrightarrow f(x) \in B$.

Theorem 6.4. $\mathrm{NP} \neq \mathrm{E}$.

Definition 6.3. A set A is $\leq_m^P$-complete for NP (commonly called NP-complete) if

1. $\mathrm{A} \in \mathrm{NP}$;
2. for every set $\mathrm{L} \in \mathrm{NP}$, $\mathrm{L} \leq_m^P \mathrm{A}$.

Theorem 6.5. If A is NP-complete, then $A \in P$ if and only if $\mathrm{P} = \mathrm{NP}$.

Universal set for NP $\mathscr{U} = \{\langle i, x, 0^n\rangle |$ some computation of NP_i accepts x in fewer than n steps$\}$