Chapter 1. 绪论 §1-1.

1. 集合是不重的. $\{7,21,57\}=\{7,21,7,57\}$.
无限集合: 有无限个元素的集合. 如 N, Z.

2. 序列: 一系列有序的元素. 如 $(7,21)\neq(21,7)$
元组(tuple): 有限个数的序列. k-tuple.

3. 幂集: $P(A)$ = A所有的子集组成的集合.
笛卡尔积: $A\times B=\{(a,b)\mid a\in A, b\in B\}$.
$A^k=A\times A\times\cdots\times A$. 共k个A.

4. 函数: $f: D\rightarrow R$. 映射关系.
D: 定义域 R: 值域.
k元函数: 定义域 $D=A_1\times A_2\times\cdots\times A_k$.
Predicate: 值域是{TRUE, FALSE}的函数.

5. 关系, 定义域 $D=A\times A\times\cdots A$ 的断言.
k元关系: $D=A\times A\times\cdots A$. 共k个.
二元关系的等价性:
① 自反性: 对 $\forall x\in A$. xRx.
② 对称性: 对 $\forall x,y\in A$ 若 xRy, 则 yRx.
③ 传递性: 对 $\forall x,y,z\in A$. 若 xRy 且 yRz 则 xRz

6. 图的描述: $G=(V,E)$. V是顶点集合. E是边集合. 每条边用二元组描述.

7. 路径: 图上的边连接起来的顶点序列.
简单路径: 无重复节点的路径.

8. 字符串序: 字典序和string order. string order 先按长度排, 同一长度内按字典序排.

9. 语言: 字符串的集合. prefix-free: 没有一个串是另一个串的前缀.

10. De Morgan 律. $P\vee(Q\wedge R)=(P\vee Q)\wedge(P\vee R)$.
$P\wedge(Q\vee R)=(P\wedge Q)\cup(P\wedge R)$.

Chapter 2. Automaton. 自动机

§2.1. Regular language. 正则语言.

1. Finite Automaton. 有限自动机
五元组: (Q,Σ,δ,q_0,F). 其中:
Q: 状态集. Σ: 字母表, $\delta: Q\times\Sigma\rightarrow Q$. 转移
$q_0\in Q$: 起始状态. $F\subseteq Q$. 接受状态.
FA M 识别的语言: $L(M)=\{w\mid M \text{接受} w\}$.

2. regular language: 被有限自动机识别的语言.

3. 正则运算. regular operations: A和B都是语言.
① union: $A\cup B=\{x\mid x\in A \text{ or } x\in B\}$
② concatenate: $A\circ B=\{xy\mid x\in A \text{ and } y\in B\}$
③ star: $A^*=\{x_1x_2\cdots x_k\mid k\geq 0 \text{ and } x_i\in A\}$.

4. 正则语言对并封闭: 设 $L(M_1)=A$. $L(M_2)=B$.
构造 FA M 识别 $A\cup B$: $Q=\{(r_1,r_2)\mid r_1\in A, r_2\in B\}$. $\delta((r_1,r_2),a)=(\delta(r_1,a),\delta(r_2,a))$.
$q_0=(q_1,q_2)$. $F=\{(r_1,r_2)\mid r_1\in F_1, r_2\in F_2\}$.

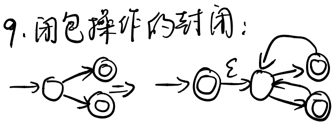
5. DFA: 下一个状态确定. NFA: 下一个状态不确定. Nondeterminism: 生成多份自身的拷贝进入下一阶段. 即并行地计算.

6. NFA 形式化定义: $\delta: Q\times\Sigma_\varepsilon\rightarrow P(Q)$. 其余不变.

7. 等价性: NFA和DFA等价. 将NFA转为DFA:
DFA 构造: $Q'=P(Q)$, $\delta'(R,a)=\{q\in Q\mid q\in\delta(r,a) \text{ for } r\in R\}$. $q_0'=\{q_0\}$. $F'=\{R\in Q'\mid R$ 包含NFA中的接受态$\}$.
对ε转移的处理: $E(R)=\{q\mid q$ 可以从R中状态经一个或数个ε转移得到$\}$. $\delta(r,a)\rightarrow E(\delta(r,a))$. $q_0'\rightarrow E(\{q_0\})$.

8. 用NFA证明连接操作的封闭性:
构造NFA: $Q=Q_1\cup Q_2$,
$\delta(q,a)=\delta_1(q,a)$, $q\in Q_1$ and $q\notin F_1$.
$\delta_1(q,a)$, $q\in F_1$ and $a\neq\varepsilon$.
$\delta_1(q,a)\cup\{q_2\}$, $q\in F_1$ and $a=\varepsilon$.
$\delta_2(q,a)$. $q\in Q_2$

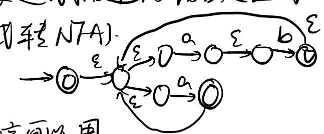
9. 闭包操作的封闭:



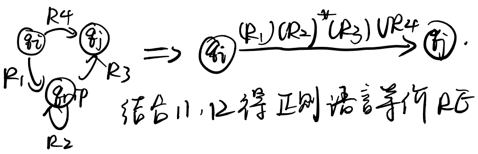
总之, 正则语言对并, 连接, 闭包操作都封闭.

10. 正则表达式的递归定义: $\{a\}$, $\{\varepsilon\}$, ϕ, 或 $(R_1\cup R_2)$, $R_1\circ R_2$, R_1^*.

11. 由正则表达式描述的语言是正则语言. (正则表达式转NFA).
$(ab\cup a)^*$:



12. 正则语言可以用正则表达式描述. (DFA转正则表达式).
step1: DFA 转 GNFA. 添加新的初始状态和接受状态. 合并相同状态的箭头(∪).
step2: 消去GNFA中的状态, 直到只剩 s, a.



结合11, 12 得正则语言等价 RE

13. 正则语言的泵引理: 对于正则语言A, 存在一泵长度p, 对于任一 $w\in A$, $|w|\geq p$, w可以被分成 $w=xyz$. 使得: ① $xy^iz\in A$. $i\geq 0$ ② $y>0$ ③ $|xy|\leq p$.
泵引理的证明: 使用鸽巢原理. 令 $w=|Q|$.
w的前p个字符在FA上识别时存在重复状态. 重复状态前的为x, 重复状态为y, 后为z.

§2.2. context-free language. 上下文无关语言.

1. CFG, 上下文无关文法: 四元组 (V,Σ,R,S)
V: 变元集合. Σ: 终结符集合 (不包括ε). R: 替换规则
S: 起始变元

2. 生成: 应用一步替换规则. 派生: $u\overset{*}{\Rightarrow}v$.
CFG的语言: $\{w\in\Sigma^*\mid S\overset{*}{\Rightarrow}w\}$.

3. CFG的歧义性: 对同一个串w, 存在两种从S到w的最左派生.

4. 乔姆斯基范式 CNF: 所有替换规则形如 $A\rightarrow BC$ 或 $A\rightarrow a$. B, C: 变元(非起始). CNF允许 $S\rightarrow\varepsilon$. S为起始变元
CNF的转换规则:
step1: 添加新的起始变元 $S_0\rightarrow S$.
step2. 删除所有空替换 $A\rightarrow\varepsilon$, $A\neq S_0$. 对 $R\rightarrow uAv$, 添加 $R\rightarrow uv$. $R\rightarrow A$: $R\rightarrow\varepsilon$. (找A在右边)
step3. 删除一元替换 $A\rightarrow B$. (找B在左边)
step4. 替换≥3的规则.

5. 下推自动机PDA: 六元组 $(Q,\Sigma,\Gamma,\delta,q_0,F)$
Q: 状态集 Σ: 输入串的字母表, Γ: 栈字母表.
$\delta: Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon\rightarrow P(Q\times\Gamma_\varepsilon)$ 转移函数.
$q_0\in Q$: 起始状态. $F\subseteq Q$: 接受状态.

6. PDA和上下文无关语言的等价性.

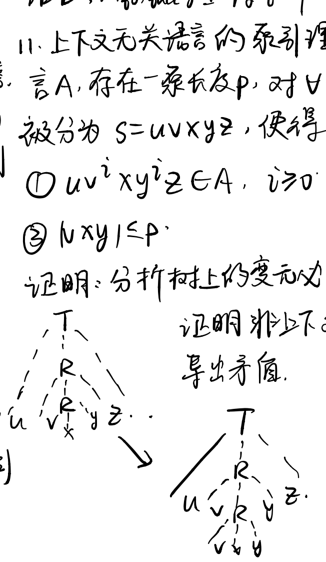
7. 语言上下文无关 ⇒ 存在PDA识别它(CFG转PDA).
一个四状态的PDA: $\rightarrow q_1\xrightarrow{\varepsilon,\varepsilon\rightarrow\$}q_2\xrightarrow{\varepsilon,\varepsilon\rightarrow S}q_{loop}\xrightarrow{\varepsilon,\$\rightarrow\varepsilon}q_{acc}$
q_{loop} 处理所有替换规则. 逆序压栈.
PDA的 $\Sigma=\{terminals\}$. $\Gamma=\{terminals\}\cup\{vars\}$.

8. PDA识别该语言 ⇒ 语言上下文无关(PDA→CFG).
对PDA P中的每一对状态 p, q, 变元 A_{pq} 产生所有将p+空栈带到q+空栈的串.
构造替换规则的两种情况: ① 最后出栈字符和最早进栈字符相同. 即中间状态栈不空. 有规则 $A_{pq}\rightarrow aA_{rs}b$. ② 不同. 中间有空栈. 有规则 $A_{pq}\rightarrow A_{pr}A_{rq}$. r状态下栈空.
形式证明: 给定PDA $P=(Q,\Sigma,\Gamma,\delta,q_0,\{q_{acc}\})$.
构造CFG G的变元 $V=\{A_{pq}\mid p,q\in Q\}$.
$S=A_{q_0,q_{acc}}$.
① 若 $\exists(r,u)\in\delta(p,a,\varepsilon)$ 且 $(q,\varepsilon)\delta(q,b,u)$.
加入规则 $A_{pq}\rightarrow aA_{rs}b$.
② 对所有 $p,q,r\in Q$, 加入规则 $A_{pq}\rightarrow A_{pr}A_{rq}$.
③ 加入规则 $A_{pp}\rightarrow\varepsilon$.

9. 若 A_{pq} 产生串x, 则x将p+空栈带到q+空栈
对 A_{pq} 派生x的步数归纳. ① 1步: $A_{pp}\rightarrow\varepsilon$.
② k+1步: 分 $A_{pq}\rightarrow aA_{rs}b$ 和 $A_{pq}\rightarrow A_{pr}A_{rq}$.

10. 若x将p+空栈带到q+空栈, 则 A_{pq} 产生串x.
对P的计算步数归纳. ① 0步: $A_{pp}\rightarrow\varepsilon$.
② k+1步: 分中间状态有没有空栈
结合, $A_{q_0,q_{acc}}$ 产生将 q_0 带到 q_{acc} 的串. 8证毕.

11. 上下文无关语言的泵引理: 对于上下文无关语言A, 存在一泵长度p, 对 $\forall s\in A$, $|s|\geq p$, s可以被分为 $s=uvxyz$, 使得
① $uv^ixy^iz\in A$, $i\geq 0$ ② $|vy|>0$
③ $|vxy|\leq p$.
证明: 分析树上的变元必会重复.



证明非上下文无关: 使用泵引理导出矛盾.

Chapter 3. Computability. 可计算性.

§3.1. Church-Turing 论题.

1. 图灵机：$(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$.
Σ：输入字母表，$\sqcup \notin \Sigma$. Γ：纸带字母表，$\sqcup \in \Gamma$.
$\delta: Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$.
2. 图灵机接受(进入接受态)的串集合是该图灵机的语言，或图灵机识别该语言.
3. 图灵可识别：如果一个图灵机识别一个语言，则该语言是图灵可识别的.
4. 判定器：总是进入 q_{accept} 和 q_{reject} 的图灵机.
5. 可判定：如果一个判定器识别这个语言，则该语言是可判定的.
6. 多纸带图灵机：用单纸带图灵机模拟，用#作为多条纸带的分隔. $\delta: Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$.
7. 非确定图灵机：每次计算产生分支.
$\delta: Q \times \Gamma \to P(Q \times \Gamma \times \{L, R\})$. 用多带图灵机模拟：输入tape，模拟tape，地址tape(分支).
8. 枚举器：带有printer的TM. 枚举即将串输出到printer. 图灵可识别 $\Leftrightarrow$ 枚举器枚举.
9. Church-Turing 论题：图灵机和算法等价.
10. implement-level：有具体的读写头和纸带描述. high-level：只需描述算法.

§3.2. Decidability 可判定性.

1. $A_{DFA} = \{\langle B, w\rangle \mid B$ 是DFA 且接受串 $w\}$ 是可判定的. 语言 A_{DFA} 是可判定的，直接构造 TM 证明.
2. A_{NFA}, A_{REX} 都是可判定的. 构造或转化.
3. $E_{DFA} = \{\langle A\rangle \mid A$ 是 DFA 且 $L(A) = \phi\}$ 是可判定的. 使用标记状态方法证明.
4. $EQ_{DFA} = \{\langle A, B\rangle \mid A, B$ 是 DFA 且 $L(A) = L(B)\}$ 是可判定的. $L(C) = (\overline{L(A)} \cap L(B)) \cup (L(A) \cap \overline{L(B)})$. 转化为 E_{DFA}.
5. $A_{CFG} = \{\langle G, w\rangle \mid G$ 是 CFG 且 G 生成 $w\}$ 是可判定的. 将 G 转为 CNF，则 w 的任何派生都在 $2n-1$ 步内，枚举这 $2n-1$ 步.
6. $E_{CFG} = \{\langle G\rangle \mid G$ 是 CFG 且 $L(G) = \phi\}$. 是可判定的. 从terminal开始标记，直到没有新的var被标记. 考察S有没有被标记.
7. EQ_{CFG} 是不可判定的. CFL上的操作不封闭.
8. $A_{TM} = \{\langle M, w\rangle \mid M$ 是一个TM 且 M 接受 $w\}$ 是不可判定的.
U = "对输入 $\langle M, w\rangle$，M是一个TM：
   1. 在M上模拟w.
   2. 如果M接受，则接受. 如果M拒绝，则拒"
9. 对角化方法：衡量两个无限集合的大小.
A和B有相同大小：存在双射 $f: A \to B$.
10. 可数集合：和 $N = \{1, 2, 3 \ldots\}$ 有相同大小或有限. 实数集合不可数.
11. 一些语言不是图灵可识别的. TM的全集是可数的，语言的全集是不可数的.
12. 证明 A_{TM} 是不可判定的：假设H是 A_{TM} 的判定器. $H(\langle M, w\rangle) = \begin{cases} acc, & \text{如果M接受w} \\ rej & \text{如果M拒绝w} \end{cases}$

$$D(\langle M\rangle) = \overline{H}(\langle M, \langle M\rangle\rangle) = \begin{cases} acc & \text{如果M拒绝}\langle M\rangle \\ rej & \text{如果M接受}\langle M\rangle \end{cases}$$

$$D(\langle D\rangle) = \begin{cases} acc & \text{如果D拒绝}\langle D\rangle \\ rej & \text{如果D接受}\langle D\rangle \end{cases}$$ 出现矛盾.

13. 补图灵可识别：图灵可识别语言的补集.
14. 一个语言是可判定的 iff 它是图灵可识别的，也是补图灵可识别的. 并行运行识别器.

§3.3. Reducibility. 可归约性.

1. A归约到B：A不可能比B更难.
若B可判定，则A一定可判定. 若A不可判定，则B一定不可判定.
2. $HALT_{TM} = \{\langle M, w\rangle \mid M$ is TM 且在w上停机$\}$ 不可判定. A_{TM} 可以归约到 $HALT_{TM}$.
3. $E_{TM} = \{\langle M\rangle \mid M$ is TM and $L(M) = \phi\}$ 不可判定. 构造另一个TM，输入 $x = w$ 且M接受w才接受，A_{TM} 可以归约到 E_{TM}.
4. EQ_{TM} 不可判定. E_{TM} 可以归约到 EQ_{TM}.
5. $PCP = \{\langle P\rangle \mid P$ 是有匹配的一个实例$\}$ 是不可判定的. MPCP：PCP + 匹配从第一张开始. A_{TM} 可以归约到 MPCP，$MPCP \leq_m PCP$
6. 可计算函数：$f: \Sigma^* \to \Sigma^*$. 如果一个TM M对所有输入w，停机时纸带上是 $f(w)$.
7. 映射可归约性：$A \leq_m B$，如果有一个可计算函数对 $\forall w \in A \Leftrightarrow f(w) \in B$.
8. $A \leq_m B \Leftrightarrow \overline{A} \leq_m \overline{B}$
9. 如果 $A_{TM} \leq_m \overline{B}$，则B不是图灵可识别的.
10. EQ_{TM} 不是图灵可识别的，也不是补图灵可识别的. $A_{TM} \leq_m \overline{EQ_{TM}}$ $A_{TM} \leq_m EQ_{TM}$.

Chapter 4. Complexity Theory. 复杂度理论.

1. 大O记号：$f(n) = O(g(n))$. $\exists n_0, c > 0$. 对 $\forall n \geq n_0$ 有 $f(n) \leq cg(n)$. $g(n)$：渐近上界.
小o记号：$f(n) = o(g(n))$. 如果 $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$.
2. 时间复杂度类：$TIME(t(n))$，能在 $O(t(n))$ 时间被TM判定的语言类.
3. $t(n)$ 时间的多带TM等价于 $O(t^2(n))$ 时间的单带TM. 证明：模拟时间为 $O(t(n))$，最大长度为 $t(n)$ $t(n) \cdot O(t(n)) = O(t^2(n))$.
4. $t(n)$ 时间的非确定TM等价于 $2^{O(t(n))}$ 时间的单带TM. $b^{t(n)}$：最多的叶子结点数. $O(t(n))$：最大长度. $O(t(n) \cdot b^{t(n)}) = 2^{O(t(n))}$
5. P类问题：确定型单带TM在多项式时间内可判定的问题.
6. 路径，互素，CFL都属于P.
7. Verifier验证器，语言A的验证器是一个算法V. $A = \{w \mid V$ 接受 $\langle w, c\rangle$，对某一串 $c\}$.
多项式时间可验证：有一个多项式时间的V.
c：证书. 如：哈密顿路.
8. NP定义：有多项式时间的验证器.
9. 定理：一个语言 $\in NP$ iff 该语言被某个非确定TM在多项式时间内判定.
$\Rightarrow$：非确定地选取证书c. 用V判定c.
$\Leftarrow$：用NTM N构造V.
10. $CLIQUE = \{\langle G, k\rangle \mid G$ 是一个有k-clique的图$\}$
$CLIQUE \in NP$. 证明：构造V或NTM N.
11. $SUBSET\text{-}SUM = \{\langle S, t\rangle \mid S = \{x_1, \ldots, x_k\}$, and for some $\{y_1, \ldots, y_l\} \subseteq S$, $\sum y_i = t\} \in NP$.
12. NP-完全：如果一个NP-完全算法存在多项式时间算法，则所有NP问题多式内可解.
13. 多项式时间归约 $\leq_p$：$\leq_m$ 的可计算函数是多项式时间的. $A \leq_p B$ 且 $B \in P$，则 $A \in P$.
14. $3SAT = \{\langle \varphi\rangle \mid \varphi$ 是一个可满足的3cnf-formula$\}$.
$3SAT \leq_p CLIQUE$.
15. NP-完全：$B \in$ NP-complete，满足：
① $B \in NP$. ② $\forall A \in NP$ 可以 $A \leq_p B$.
若只有②，则 $B \in$ NP-Hard.
16. $SAT \in$ NP-Complete.
$3SAT \in$ NP-complete.

$$\left.\begin{array}{l} 3SAT \leq_p HAMPATH \\ 3SAT \leq_p \text{Vertex cover} \\ 3SAT \leq_p \text{SUBSET-Sum} \end{array}\right\} \text{NP-Complete}$$