Formal Languages and Automata Theory.

不保证正确性 ~

期末考试范围

没复习前的状态

2

语言与有穷自动机DFA的相互转换 联DFA

3

✓ NFA转换为DFA-两种方法 一个是不断加状态,新到没新状态为止

~~识别关键字的NFA~~

4

✓ 正则表达式与NFA之间的相互转换 GNFA?

※5

✓ **正则语言的泵引理** 指出矛盾.

✓ 最小化问题 5.5 最小化.证明.

✓ M-N定理，证明语言正则的充要条件 仅限听说.

6

✓ 不太考：上下文无关语言-设计上下文无关文法 CFL设计.

✓ 上下文无关语言转为PDA

7

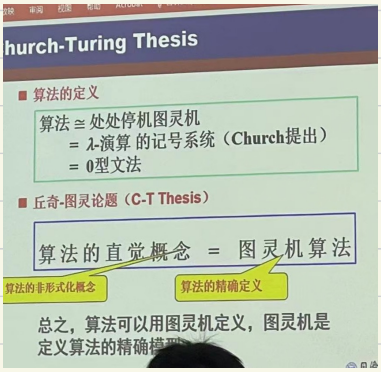
✓ 上下文无关文法的化简-为CNF CFL化简+乔姆斯基

~~不考：上下文无关语言泵引理~~

8

✓ **church-turing thesis** 那一页PPT.

8. C-T Thesis



算法

C-T Thesis.

算法 ⇄ 图灵机

7. 0上下文无关文法化简

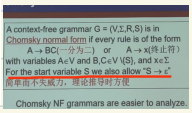
1. 消除 ε -产生式
2. 消除单一产生式
3. 消除不可产生的无用符号
4. 消除不可达的无用符号

} 这两点不能调换

消除ε-产生式的方法：

- 对形如 $A\to X_1X_2\cdots X_m$ 的产生式进行考察，
  找出文法的可空变量集U，
  然后对于 $\forall H\subseteq U$，从 $A\to X_1X_2\cdots X_m$ 中删除 H 中的变量。
  对于不同的子集H，得到不同的 A 产生式，用这组 A 产生式替代产生式 $A\to X_1X_2\cdots X_m$。
- 必须避免在这个过程中产生新的 ε -产生式：
  当 $\{X_1, X_2, \cdots, X_m\}\subseteq U$ 时，不可将 $X_1, X_2, \cdots, X_m$ 同时从产生式 $A\to X_1X_2\cdots X_m$ 中删除。

开始变元可以空（原语言中有ε）



$\emptyset\subseteq U$ 也就是原来的不动才下来

一次不用一整个全删，

比如 cBC，可变为 cB，Bc.

消除单一产生式

确定单元对

①如果有A→B，则称[A，B]为单元对；
②若[A，B]和[B，C]是单元对，则[A，C]是单元对。

消除单元对

①删除全部形如A→B的单元产生式；
②对每个单元对[A，B]，将B复制给A。

就把能取空的全取空了.

例7.1 给定文法：

G_1：S→ 0 | 0A | E
A→ε| 0A | 1A |B
B→ _C
C→0 | 1 | 0C | 1C
D→1 | 1D | 2D
E→0E2 | E02

去掉无用符号

S→0|0A
A→ε|0A|1A|B
B→_C
C→0|1|0C|1C

去除ε

S→0|0A
A→0A|1A|B|0|1
B→_C
C→0|1|0C|1C

去掉单个产生式

S→0A|0A
A→0A|1A|_C|0|1
B→_C
C→0|1|0C|1C

不可达不可达 $S\Rightarrow \alpha X\beta$，X是可达的

不可空与不可空 $\alpha X\beta \Rightarrow w$，X是不可空的

X有用：X可空且X可达 即 $S\Rightarrow \alpha X\beta \Rightarrow w$

例7.2 消除文法中的无用符号

S →AB | a

A →b

先消 $\alpha X\beta \overset{*}{\Rightarrow} w$ 形的

S → a

A → b

再消 $S \overset{*}{\Rightarrow} \alpha X\beta$ 形的

S → a.

<先消最后没结果的 再消到都没以了的>

注意：

- 先消除非“可产生的”符号；
- 后消除非“可达的”符号；

例 7-3 设有如下文法，消除无用符号

S→AB | *a* | BB，A→*a*，C→*b* | AB*a*

Q又而言之

S → a

A → a

C → b

Q又可达.

S → a.

例7.4 有如下文法，求可空变量集。

S → ABS | ABO

A → CA | CBC

B → 1C | ε

C → 1C | ε

可空变量集 A、B、C.

{A, B, C}

但是不能简单地将 S→ABS 中的A 删去，而是要考虑表达式 A 产生 ε 和A 不产生 ε 的情况。

例7.4 有如下文法，求可空变量集

S → ABS | ABO

A → CA | CBC

B → 1C | ε

C → 1C | ε 去除ε.

可空变量集 {A、B、C}

S → ABS|AB0 |BS|AS|S|A0|B0|0

A → C|CC|B|A|CA|CBC|BC|CB

B → 1|1C

C → 1|1C

原来的

例7.5 消除下列文法中的 ε-产生式

S →AB

A →AaA | ε

B →BbB | ε

{A,B}

S → AB|A|B.

A → AaA | aA | Aa | a

B → BbB | bB | Bb | b

例7.5 消除下列文法的单一产生式

S → A | B | 0S1

A → 0A | 0

B → 1B | 1

S → 0A|0|1B|1|0S1.

A → 0A|0

B → 1B|1.

* 习题7.1.1 找到一个不含无用符号的等价于以下文法的文法：

$S \to AB \mid CA$

$A \to a$

$B \to BC \mid AB$

$C \to aB \mid b$

B：非产生 → B的自循环

$S \to CA$

$A \to a$

$C \to b$

* 习题7.1.2 从以下文法出发：

$S \to ASB \mid \varepsilon$

$A \to aAS \mid a$

$B \to SbS \mid A \mid bb$

a) 去除ε产生式。

b) 去除单位产生式。

c) 有没有无用符号？如果有去除它们。

d) 把该文法转化为乔姆斯基范式。

a) 初始：S

$S \to ASB \mid AB$

$A \to aA \mid aAS \mid a$

$B \to SbS \mid Sb \mid bS \mid b \mid A \mid bb$

b) $S \to ASB \mid AB$

$A \to aA \mid aAS \mid a$

$B \to SbS \mid Sb \mid bS \mid b \mid aA \mid aAS \mid a \mid bb$

c) 有 $A \to a$，$B \to b \mid a \mid bb$

但有 $S \to AB \to$（可产生）

则没有无用符号。

有用的：

能产生终结符的

+ 能产生终结符的状态组合。

d) $C \to a$，$D \to b$，$E \to AS$，$F \to DS$

$S \to EB \mid AB$

$A \to CA \mid CE \mid a$

$B \to SF \mid SD \mid DS \mid b \mid CA \mid CE \mid a \mid DD$

7 CNF 乔姆斯基范式

乔姆斯基范式文法(Chomsky normal form , CNF)简称为Chomsky文法，或 Chomsky 范式。
CFG G=(V, T, P, S) 中的产生式形式：
A→ BC
A→ *a*
其中，A, B, C∈V, *a*∈T。

① CNF 中，不允许有 ε-产生式、单一产生式；
② 利用CNF派生长度为n的串，需要2n-1步；
③ 存在算法判定字符串w是否属于CFL；
④ 利用CNF的多项式时间解析算法---CYK算法；

不让有：ε，单一，混合，多元(≥3)

形式固定
$A \to BC$
$A \to a$.

不是说替换完后所有的都要换，可以B，替换一部分，主要是满足形式

Chomsky Normal Form(乔姆斯基范式)
1. CFG = (V,Σ,R,S) is in CNF if every rule is of the form
   - A → BC 一分为二
   - A → x 或终极化
   - S → ε

其中，变元A∈V and B,C∈V \{S}, and x∈ Σ
2. Every context-free language can be described by a grammar in Chomsky normal form。
3. 符合乔姆斯基范式的CFG不存在二义性。

例7.6 试将下列文法转换成等价的 CNF。
S→*b*A | *a*B
A→*b*AA | *a*S | *a*
B→*a*BB | *b*S | *b*

① 引入 $B_a \to a$
$B_b \to b$
$S \to B_aA \mid B_aB$
$A \to B_bAA \mid B_aS \mid a$
$B \to B_aBB \mid B_bS \mid b$
$B_a \to a$
$B_b \to b$

② 引入 $B_1 \to AA$
$B_2 \to BB$
$S \to B_aA \mid B_aB$
$A \to B_bB_1 \mid B_aS \mid a$
$B \to B_aB_2 \mid B_bS \mid b$
$B_a \to a$
$B_b \to b$
$B_1 \to AA$
$B_2 \to BB$.

→ 所以5.

*7 格雷巴赫范式 GNF

如果CFG G=(V,T,P,S)中的所有产生式都具有形式：
A→aα
其中，A∈V，a∈T，α∈V*
则称G为格雷巴赫范式文法（Greibach Normal Form），简称格雷巴赫文法，或格雷巴赫范式，简记为GNF。

类似右线性文法。

例：将下列文法转化成GNF

G2:
S → AB
A → aB|bB|b
B → b

G2':
S → aBB|bBB|bB
A → aB|bB|b
B → b

…… 有点复杂，不考。

6. CFG 设计

定义 6.1 CFG(Context Free Grammar)上下文无关文法：G = (V, T, P, S)。其中：V 是变元集，变元也称为非终结符或语法范畴，T 是终结符集，P 是产生式规则，S 是开始字符。

$V \to (V \cup T)^*$

注意 ε

例 1: $L = \{0^n 1^n \mid n \in N\}$

$G = (\{S\}, \{0,1\}, P, S)$

P: $S \to 0S1 \mid \varepsilon$.

例 2 $L = \{0^n 1^{2n+1} \mid n \in N\}$

$G = (\{S\}, \{0,1\}, P, S)$

P: $S \to 0S11 \mid 1$

例 3. 开始和结尾的字符相同.

$G = (\{S, S'\}, \{0,1\}, P, S)$

$S \to 0S'0 \mid 1S'1$

$S' \to 1S' \mid 0S' \mid \varepsilon$

生成任意 0,1 组成的字符串.

例 4. $L = \{w \in (0,1)^* \mid w$ 至少包括 3 个 1$\}$

$G = (\{S, S'\}, \{0,1\}, P, S)$

P: $S \to S'1S'1S'1S'$

$S' \to 0S' \mid 1S' \mid \varepsilon$. → 0, 1 随便生成 string

例 5. $L = \{w \in (0,1)^* \mid w$ 中 0 和 1 个数相等$\}$.

$G = (\{S\}, \{0,1\}, P, S)$

P: $S \to 0S1 \mid 1S0 \mid SS \mid \varepsilon$ → 别忘空

SS → 00 11, 1100 这种.

例 6. $L = \{0^n 1^m \mid n, m \in N\}$

$G = (\{S, A, B\}, \{0,1\}, P, S)$

P: $S \to ABC$

$A \to 0A \mid \varepsilon$

$B \to 0B1 \mid \varepsilon$.

$C \to 1C \mid \varepsilon$.

0^p $0^k 1^k$ 1^q

A B C

小概念：语法分析树 二义性.

5. CFL 转化 PDA (上下文无关语言转 PDA)

NFA + Stack

PDA是 Nondeterministic 的 ~

PDA：

Formal Definition of PDA

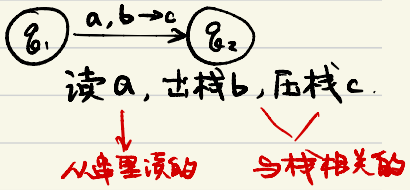
A Pushdown Automata M is defined by a six tuple $(Q,\Sigma,\Gamma,\delta,q_0,F)$, with

- Q finite set of states 有限个状态（寄存器）
- Σ finite input alphabet 字母表
- Γ finite stack alphabet 可压栈字符表
- q_0 start state $\in Q$
- F set of accepting states $\subseteq Q$ 接受态集合
- δ transition function 状态转移函数 ~ 相当于3种语句goto, push ,pop

$\delta: Q \times \Sigma\varepsilon \times \Gamma\varepsilon \to \mathcal{P}(Q \times \Gamma\varepsilon)$ 是一个超集

定义 6.8 Pushdown automaton(PDA) 下推自动机: $P=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$。其中 Q 是状态集，$\Sigma$ 是输入字符集，Γ 是栈字符集，$q_0 \in Q$ 是初始状态，$F \subset Q$ 是接受状态集合，Z_0 是初始栈底符号，$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$ 是转移函数。

- 转移函数的一般形式如下：
  $\delta(q, a, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots, (p_n, \gamma_n)\}$ ，表示当自动机处于状态 q，读到输入符 a，发现栈顶符号为 Z时，可以转移到状态 p_i，同时弹出栈顶的 Z，压入 γ_i
- 压入 γ 时，从右向左压入。即压入完成后 γ 左侧的元素对应栈顶。
- 符号约定：用字母表靠后的大写字母表示栈中的字符，如 X, Y ;用希腊字母表示栈中的字符串， 如 α,γ。



定义 6.9 (终态接受). 考虑 PDA $P=(Q,\Sigma,q_0,\delta,F,\Gamma,Z_0)$。定义语言 L(P) 如下:$w \in L(P)$ 当且仅当存在接受态 $q \in F$ 和栈字符串 $\alpha \in \Gamma^*$ ，满足 $(q_0,w,Z_0) \vdash^* (q,\varepsilon,\alpha)$。 //栈不一定为空

定义 6.10 (空栈接受). 考虑 PDA $P=(Q,\Sigma,q_0,\delta,F,\Gamma,Z_0)$。定义语言 N(P) 如下:$w \in N(P)$ 当且仅当存在状态 $q \in Q$，满足 $(q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon)$。 //栈一定为空

就是接受时栈的状态不一样了.

给定CFL，如何构造等价的PDA？

给定CFL L，如何构造一个等价的PDA P

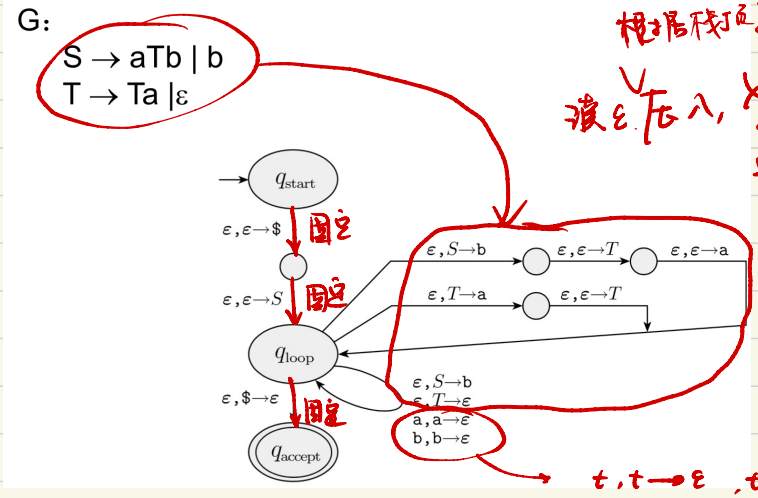
1. 把标记符号 $ 和起始变元放入栈中；
2. 重复下列步骤：
   ① 如果栈顶符号是变元 A，则非确定性第选择一个A的规则，并且把A替换成这条规则右边的字符串。
   ② 如果栈顶是终结符 a，则读取下一个输入符号，并且把它与a进行比较。如果它们匹配，则重复，如果它们不匹配，则拒绝这个非确定性分支。
   ③ 如果栈顶符号是 $，则进入接受状态，如果此刻输入已全部读完，则接受这个输入串。

直接看例子：

EXP：把CFG G转换成PDA P，其中G：

$S \to aTb \mid b$

$T \to Ta \mid \varepsilon$



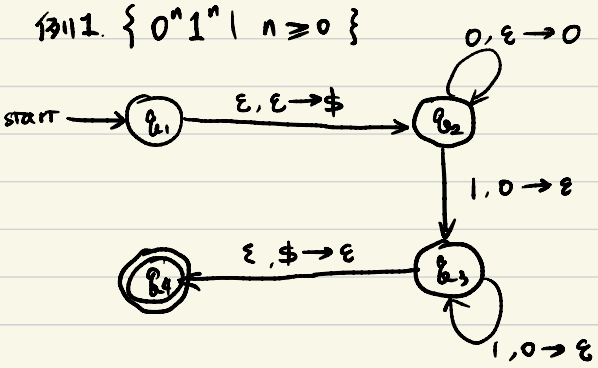
根据栈顶是什么

读ε压入，$X_n X_{n-1} \cdots X_1$

等价于 $\varepsilon, q \to X_{i-1}$ 的情况.

$t,t\to\varepsilon$ ，$t\in T$.

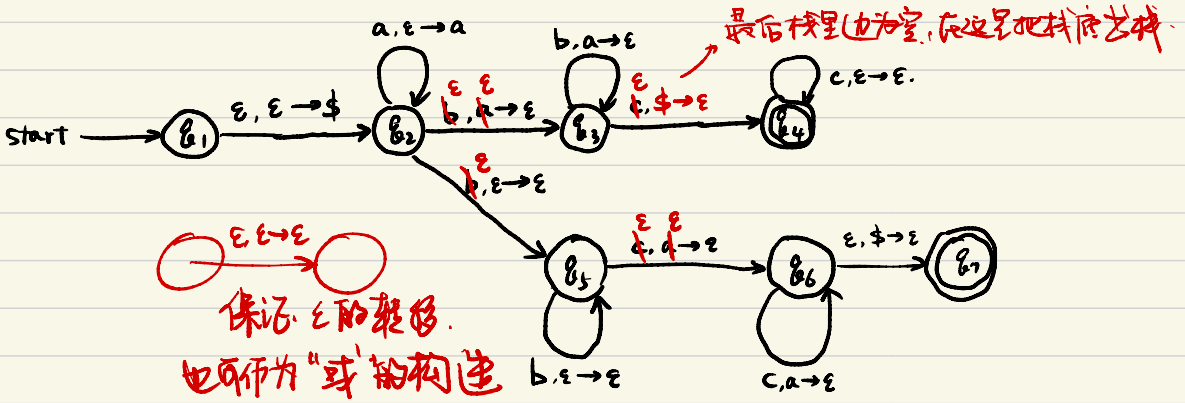
例1. $\{0^n1^n \mid n \geq 0\}$



（对消）

ε字符呢？

例2. $L = \{a^i b^j c^k \mid i, j, k \geq 0,\ i=j \text{ 或 } i=k\}$



最后栈里也为空，在这里把栈底去掉

保证ε的最多

也可作为"或"的构造

例10. 构造一个PDA按终结状态方式接受语言 $\{0^n1^n \mid n \geq 1\}$。

□ 形式化定义：

M = $(\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \{Z, A\}, \delta, q_0, Z, \{q_3\})$，其中，

$\delta(q_0, 0, Z) = \{(q_1, AZ)\}$ 在栈中加入A

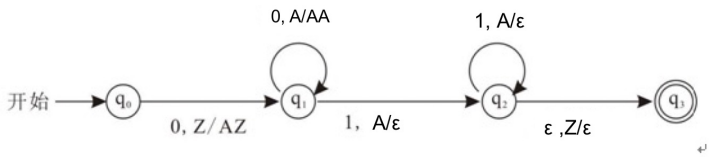
$\delta(q_1, 0, A) = \{(q_1, AA)\}$ 在栈中加入A

$\delta(q_1, 1, A) = \{(q_2, \varepsilon)\}$ 遇1消去栈顶符号

$\delta(q_2, 1, A) = \{(q_2, \varepsilon)\}$ 遇1消去栈顶符号

$\delta(q_2, \varepsilon, Z) = \{(q_3, Z)\}$ 接受

□状态转移图

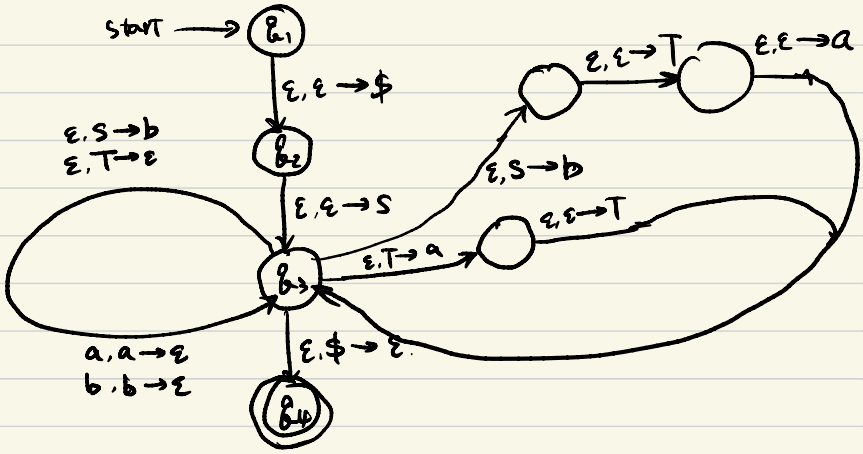


EXP：把CFG G转换成PDA P，其中

G：

S → aTb | b

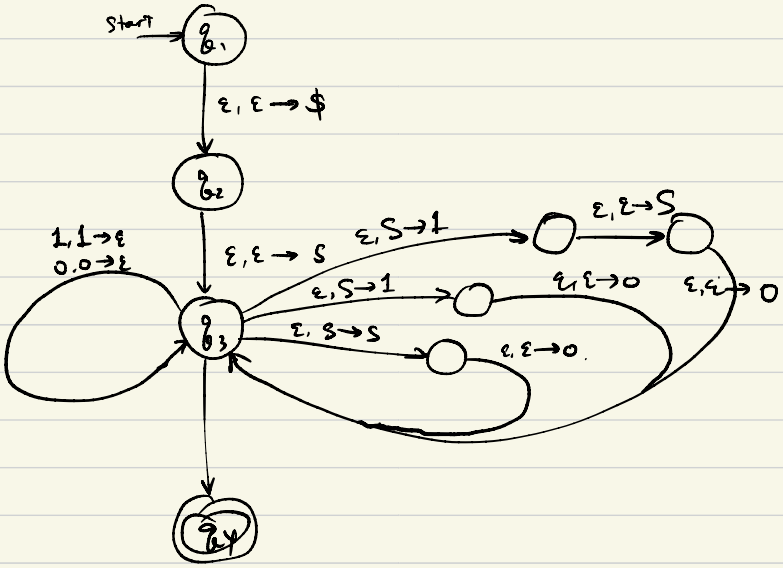
T → Ta |ε



例11. 设计一个PDA P，识别语言L={0^i1^j | i ≥ j ≥ 1}。

$0^i1^j \rightarrow 0^{i-j}0^j1^j$

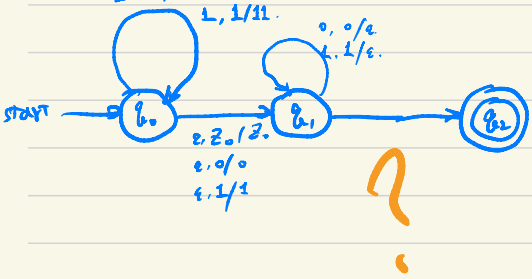
P: S → 0S1 | 01 | 0S.

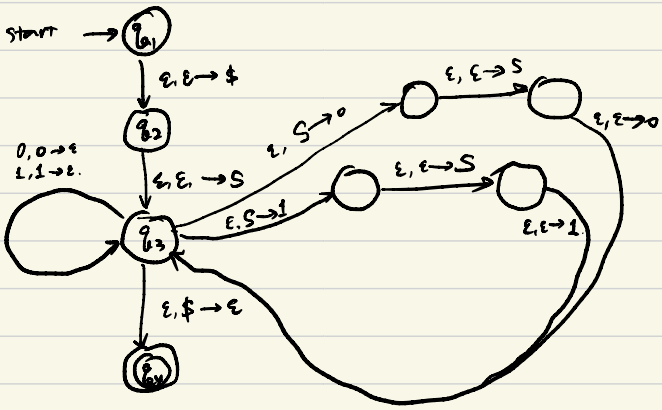


例12. 设计一个PDA，接受语言L_{wwr} = {ww^R | w∈ {0, 1}* }。

G:

P: S → 0S0 | 1S1 | ε



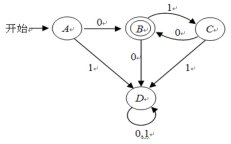


DPDA 好像不考.

4 正则语言的泵引理

DFA ⇌ 正则文法相互转化

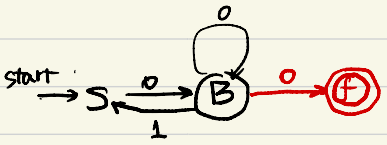
EXP5.1 将下列DFA转化为等价的正则文法。



A→0B | 0 | 1D
B→1C | 0D
C→0B | 0 | 1D
D→0D | 1D

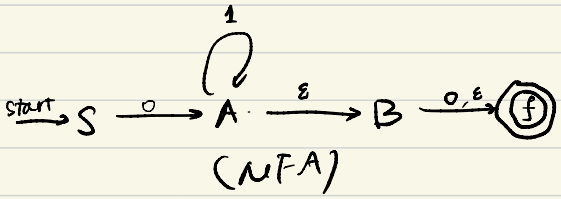
例5.2 给出正则文法G_1如下：

S→0B， B→0B，
B→1S， B→0 。



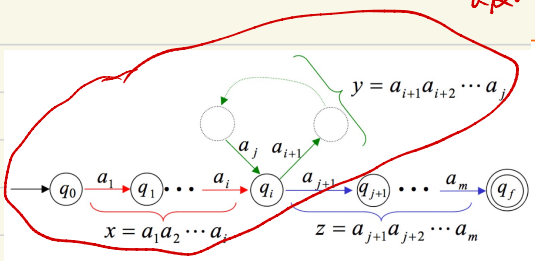
例5.3 给出正则文法G_2如下：

S→0A， A→1A，
A→B， B→0 , B→ε。



正则语言泵引理

长度小于n.(p)



$\Rightarrow w = xyz$ 满足：
- $|xy| \le n$
- $|y| \ge 1$ or $y \neq \varepsilon$
- $xy^kz \in L$, for any $k \ge 0$

Pumping lemma: For every regular language L, there is a *pumping length* **p**, such that for any string s∈L and |s|≥p, we can write s=xyz with
1) $x y^i z \in L$ for every i ∈ {0,1,2,…} 为什么打圈？
2) |y| >0
3) |xy| ≤ p 什么时候打圈？

向下泵 向上泵.

Note that
1) implies that xz ∈ L
2) says that y cannot be the empty string ε
3) is not always used

■ 经得起泵测试是**RL**的必要条件（不充分）。

EXP1:Prove B = {0^n1^n | n≥0} is not regular.

1. Assume that B is regular 反证法
2. Let p be the pumping length, and s = 0^p1^p ∈ B
   s = xyz = 0^p1^p, with xy^iz ∈ B for all i≥0
   Three options for y:
   1) $y=0^k$, hence xyyz = $0^{p+k}1^p \notin B$
   2) $y=1^k$, hence xyyz = $0^p1^{k+p} \notin B$
   3) $y=0^k1^l$, hence xyyz = $0^p1^l0^k1^p \notin B$
3. Conclusion: The pumping result does not hold, the language B is not regular.

p: pumping length

$S = 0^p1^p$

pump: $|xy| \le p$ 且 $\forall i > 1$ $xy^iz = 0^{p+(i-1)|y|}1^p \notin L$

EXP2, Show F = { ww | w∈{0,1}* } is not RL.

Let p be the pumping length, and take word s = 0^p10^p1 , w=0^p1

Let s = xyz = 0^p10^p1,

with condition 3) |xy|≤p

Only one option: x=0^{p-k} ,y=0^k, z=$10^{p-k}1$, (保证xz in L)

with xyyz = $0^{p+k}10^{p-k}1 \notin F$

p: pumping length.

s: $0^p1\,0^p1$

$|xy| \le p$ $y: 0^k$ $x: 0^{p-k}$

$xyyz = 0^{p+k}10^p1 \notin L$

1. L = { 0^n1^n | 0 ≤ n ≤ 100 } 是正则语言吗？

是，有限的均可以构造FA来识别。

EXP3, Show E={0^i1^j | i > j } is not RL.

Step 1：选择反证法；

Step 2：构造 string s = $0^{p+1}1^p$； 利用泵长度p

Step 3：发现矛盾

Pumping Down：The pumping lemma states that $xy^iz \in E$ even if when i=0, so lets consider the string $xy^0z=xz$.

结果怎么样呢？

$xz=0^{n}1^p$, ∵ $y=0^k$, k>0 , ∴ n≤p, 即0的个数不比1的个数多，显然，这与$s = 0^{p+1}1^p$ 相互矛盾 ∴ $xz \notin E$。

Step 4：得出结论。

p. $s: 0^p1^{p-1}$

$|xy| \le p$, $y = 0^k$, $k \ge 1$

$xy^0z = 0^{p-k}1^{p-1}$ $p-k \le p-1$ $\therefore xy^0z \notin L$

EXP4: 证明 L = {1^n | n 是素数 } 不是正则语言。

证明：假设 L 是正则语言，则存在 p 满足泵引理的性质。那么由于串 w = xyz = 1^{p_0}（其中 p_0 为大于 p 的素数）属于 L, 串 w′ = $xy^kz = 1^{p_0+(k-1)|y|}$ 也属于 L。取 k = p_0 + 1，w′ = $1^{p_0(1+|y|)}$ 显然不属 于 L，产生矛盾，因此 L 不是正则语言。

$s = 1^{p_0}$， p_0为大于p的素数

$xy^kz = 1^{p_0+|y|(k-1)}$

取 打圈次数 $k = p_0+1$ 则 $xy^kz = p_0(1+|y|)$ 成素数.

$xy^kz \notin L$.

课堂作业：试证 L = {1^n | n 是合数 } 不是正则语言。

狄利克雷定理 对于任意互质的正整数 a, d，形式如 a + nd 的素数有无限多个，其中 n 为正整数。

证明： 若 L 是正则语言，则存在 p 满足泵引理的性质。

那么由于串 w = xyz = $1^{p_0 \cdot p_0}$(其中 p_0 为大于 p 的素数)属于 L，串 w′ = $xy^kz = 1^{p_0 \cdot p_0+(k-1)|y|}$ 也属于 L。

由于 |y| ≤ p，p_0^2 和 |y| 互素，由狄利克雷定理，存在正整数 k 使得 $p_0^2 + (k-1)|y|$ 为素数，有 w′ 不属于 L，产生矛盾，因此 L不是正则语言。

$s = 1^{p_0^2}$ ，p_0素数，$p_0 > p$.

$xy^kz = 1^{p_0^2+(k-1)|y|}$

p_0^2假设跟 $|y|$互素.

则 存在正整数 $k-1$， $1^{p_0^2+(k-1)|y|}$为素数.

则 得证.

反证闭性.

自动机等价与最小化.

M - N 定理

定理5-1 (Myhill-Nerode定理)下列三个命题等价：

(1) L⊆Σ*是 RL ；

(2) L是Σ*上的某一个具有有穷指数的右不变等价关系R的某些等价类的并；

(3) R_L具有有穷指数。

■ 等价类 (equivalence class)——由等价关系R决定

S的满足如下要求的划分：S_1、S_2、S_3、…、S_n…称为S关于R的等价划分，S_i称为等价类。

(1) $S = S_1 \cup S_2 \cup S_3 \cup \cdots \cup S_n \cup \cdots$；

(2) 如果 $i \neq j$，则 $S_i \cap S_j = \Phi$；

(3) 对任意的i，S_i中的任意两个元素a、b，aRb恒成立；

(4) 对任意的i，j，$i \neq j$，S_i中的任意元素a和S_j中的任意元素b，aRb恒不成立 。

➢ 等价类：指的是该类中的元素之间存在等价关系。

➢ 等价关系R将S分成的等价类的个数称为R在S上的指数。

极小化算法

① 下三角网格图

② 可区分（接受，不接受）状态对进行标记（按照原DFA中的状态）

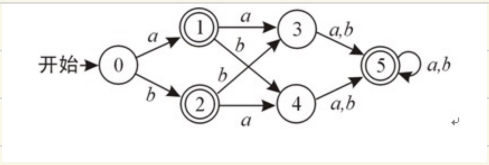
③ 找未标记位置，(A,B)，看横轴，

不断试 $(\delta(A,\Sigma),\ \delta(B,\Sigma))$，要是这个被标记了，那就把(A,B)也标记了。

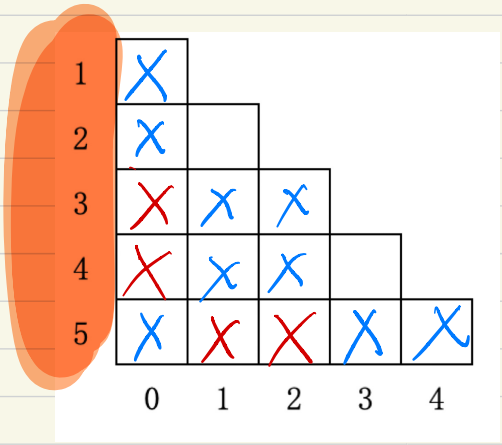
其中Σ为字母表中的字母。

④ 最后标不动了，剩下的没标的(Ai,Bi)则Ai,Bi状态在原DFA中等价

例：



注意排序。



看(0,3)

试 ((0,a),(3,a))

=(1,5) 不行

试 ((0,b),(3,b))

=(2,5) 不行。

则(0,3)标，同理(0,4)标。

看(1,5)

试 ((1,a),(5,a))

=(3,5) ✓

则(1,5)标，

看(2,5)

试 ((2,a),(5,a))

=(4,5) ✓

则(2,5)标

看(3,4)

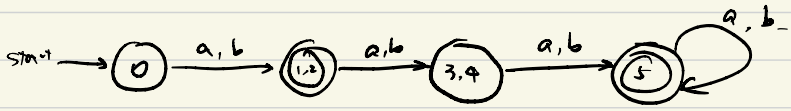
试 ((3,a),(4,a))

=(4,5) 不行。

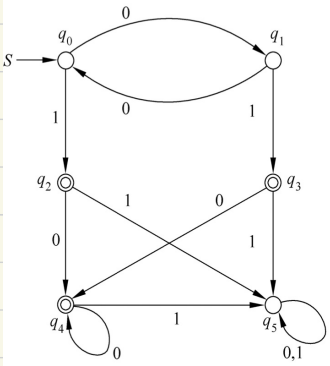
试 ((3,b),(4,b))

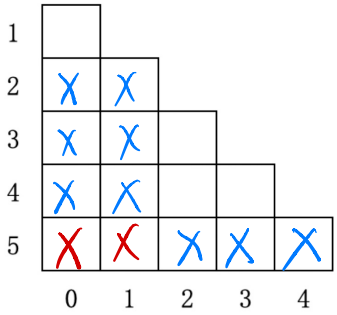
=(5,5) 不行

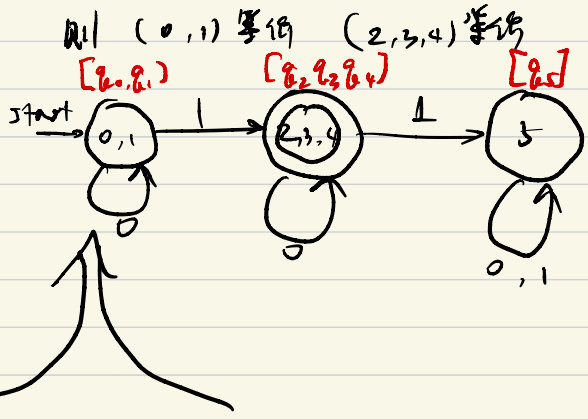
则原DFA中 1，2状态等价，3,4状态等价。



例5.16 对下图所示的DFA进行极小化。







① 标记 ✗

② 对比.

(0,1) —0→ (1,0) ✗
—1→ (2,3)

(1,5) —0→ (0,5) ✓

(2,4) —0→ (4,4) ✗
—1→ (5,5)

(0,5) —0→ (1,5)
—1→ (2,5) ✓

(2,3) —0→ (4,4) ✗
—1→ (5,5)

(3,4) —0→ (4,4)
—1→ (5,5) ✗

求 R_L 等价类

划分完的等价类并在一起是 Σ^* !!, 等价关系,最后不变的

$\forall x R_L y, \forall z \in \Sigma^*, xz R_L yz$.

方法一：间接 (L为正则的)

L为正则 ⇒ DFA M 识别 L.

⇒ 对此 DFA 极小化

⇒ 极小化合并后的状态 为等价类.

> 定理5-1 (Myhill-Nerode定理) 下列三个命题等价：
>
> (1) $L \subseteq \Sigma^*$是 RL ；
>
> (2) L是Σ^*上的某一个具有有穷指数的右不变等价关系R的某些等价类的并；
>
> (3) R_L具有有穷指数。

方法二：直接. (结构特征)

例17. L1={w∈{0,1}*|w包含的数字之和能被3整除}

特征. 整除3: $x \bmod 3 = 0$

划分: [0]: $x \bmod 3 = 0$.

[1]: $x \bmod 3 = 1$.

[2]: $x \bmod 3 = 2$.

[ε]

从级开始. [ε]

不能再合并了.

例18. L2 = $\{0w \mid w \in \Sigma^*, \Sigma=\{0,1\}\}$，求$R_L$的等价类。

特征：都以0开头.

$[\varepsilon] = \{\varepsilon\}$

$[0] = \{x \mid x$ start with $0\}$

$[1] = \{x \mid x$ start with $1\}$

例19. L3=$\{\Sigma^*0\Sigma \mid \Sigma=\{0,1\}\}$，求$R_L$的等价类。

$A_1 = \Sigma^* 00$

$A_2 = \Sigma^* 01$

$A_3 = \Sigma^* 10 \cup 0$

$A_4 = \Sigma^* 11 \cup 1 \cup \varepsilon$

例20. 语言$L=\{xwx^R \mid x, w \in \{0,1\}^+\}$ 是正则语言吗？

解法1：RE：$0(0+1)^+0 + 1(0+1)^+1$

解2：RG：

$S \to 0A \mid 0B$

$A \to 0A \mid 1A \mid 00 \mid 10$

$B \to 0B \mid 1B \mid 01 \mid 11$

解3：M-N定理：

自动机≥3.

$[\varepsilon] = \{\varepsilon\}$

$[0] = \{0\}$

$[1] = \{1\}$

$[01] = \{x \mid 0$开头，1结尾，长度$\geq 2\} \cup \{00\}$

$[10] = \{x \mid 1, 0,$ 长度$\geq 2\} \cup \{11\}$

$[00] = \{x \mid 0, 0,$ 长度$\geq 3\}$

$[11] = \{x \mid 1, 1,$ 长度$\geq 3\}$.

保证和下边两个不重.

解：

1.29 使用泵引理证明下述语言不是正则的。

A a. $A_1=\{0^n1^n2^n \mid n\geqslant 0\}$

b. $A_2=\{www \mid w\in\{a,b\}^*\}$

A c. $A_3=\{a^{2^n} \mid n\geqslant 0\}$（这里 a^{2^n} 表示 2^n 个 a 构成的串）

a. 由pumping lema，存在常数p.

s.t. $s\in L$, $|s|\geq p$.

则 $s=xyz$, 且 $xy^iz\in L$, $i\in N$

其中 $|xy|\leq p$, $|y|>0$

则考察串 $0^p1^p2^p\in L$

由 $|xy|\leq p$ 得 y仅由0组成. 设 $y=0^k$

则 $xy^2z=0^{p+k}1^p2^p\notin L$

则原语言不是正则语言.

b. 由pumping lema，存在p

s.t. $s\in L$, $|s|\geq p$

则 $s=xyz$, $xy^iz\in L$, $i\in N$

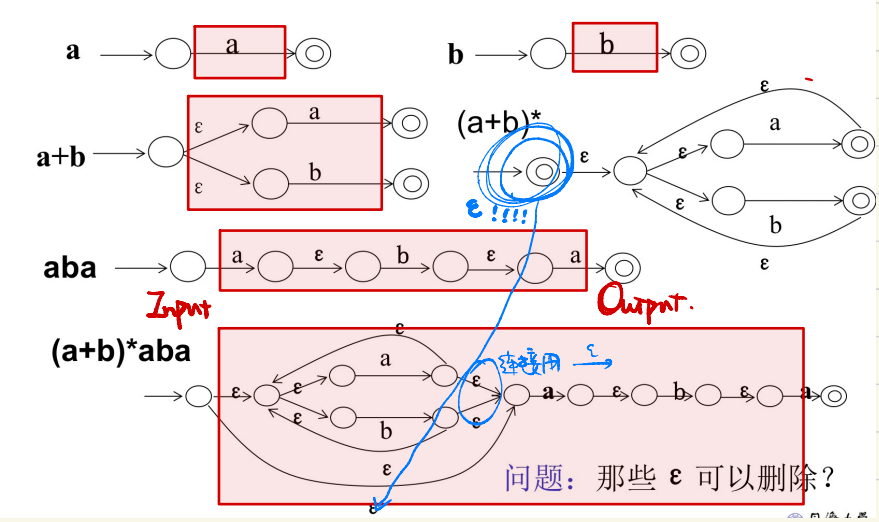
其中 $|xy|\leq p$, $|y|>0$.

考察串

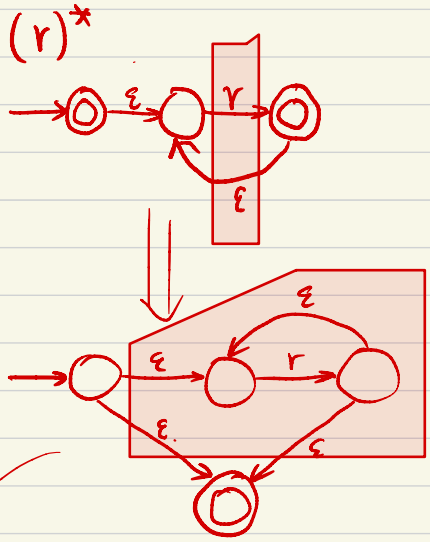
4. 正则表达式与 NFA

① 正则表达式转 NFA

例：$(a+b)^*aba$



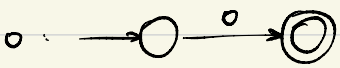
对应着 起点和终点，嵌入箭头。
Input Output



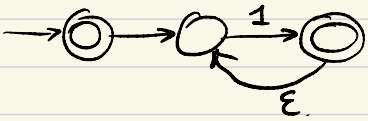
例：转 NFA：

a) 01^*

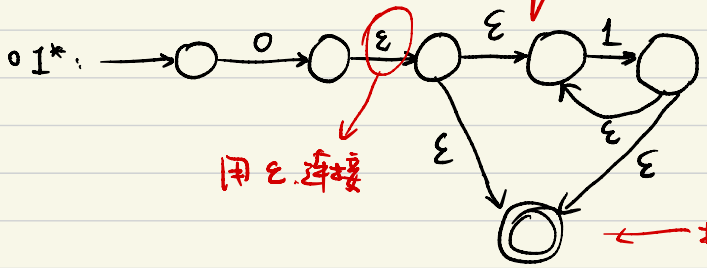
0：

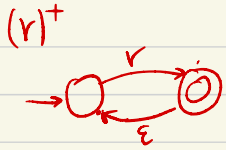


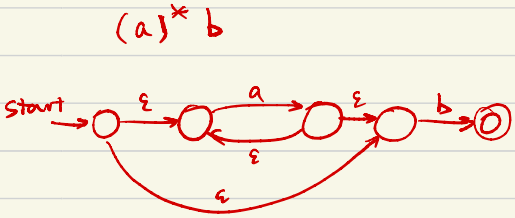
1^*：



01^*：

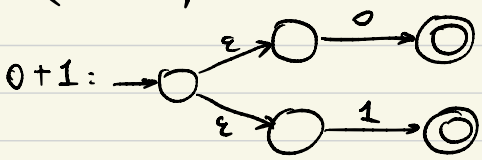




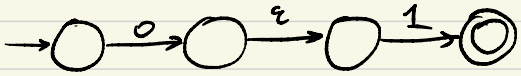


b) $(0+1)01$

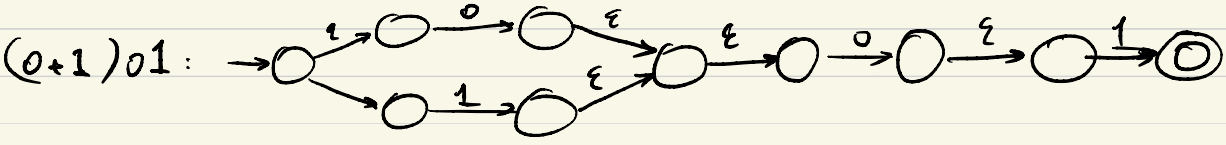
0+1：



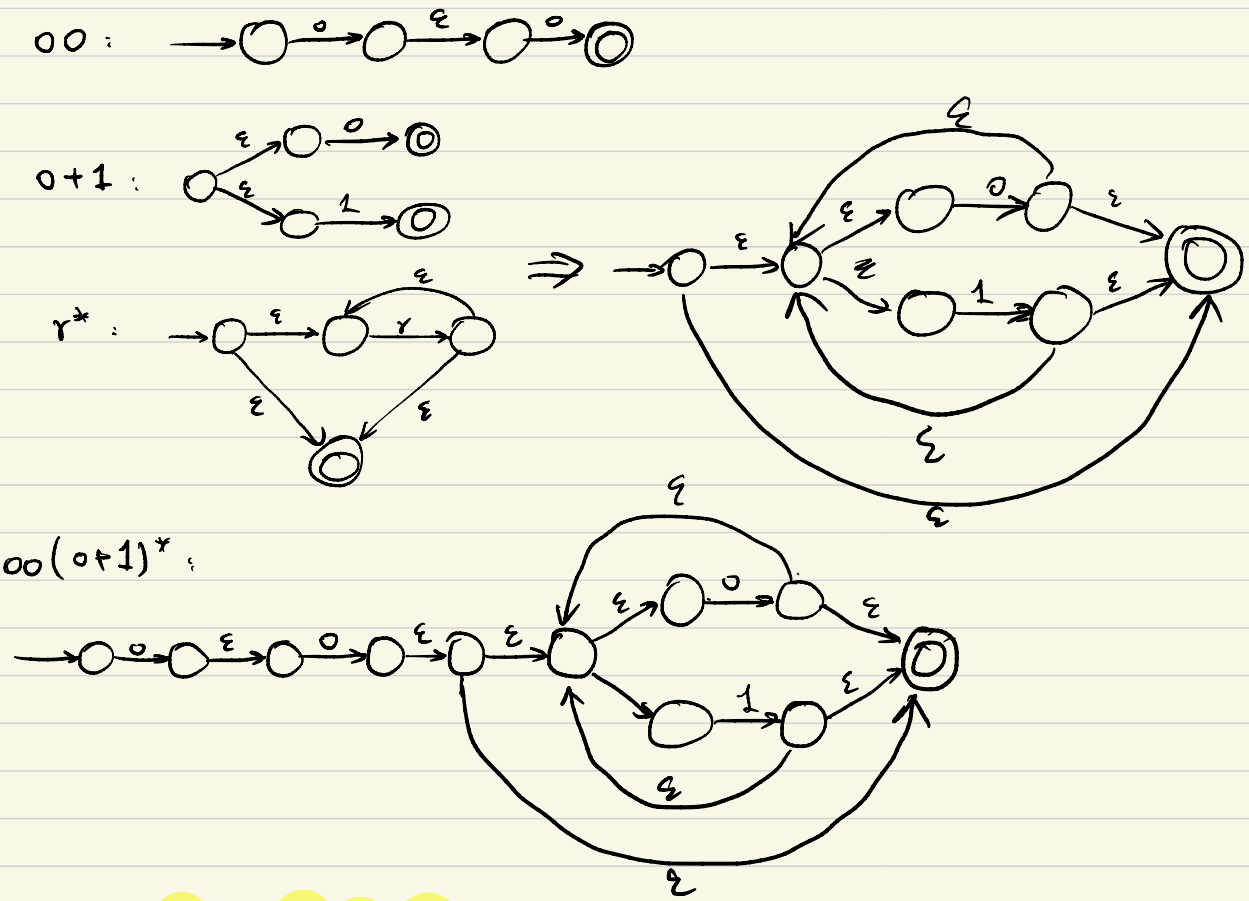
01：



$(0+1)01$：



c). $00(0+1)^*$

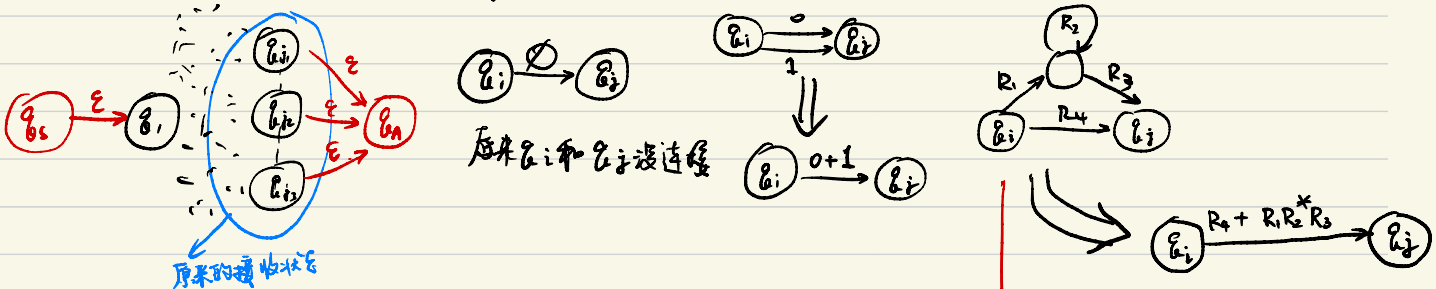


② NFA 转正则表达式

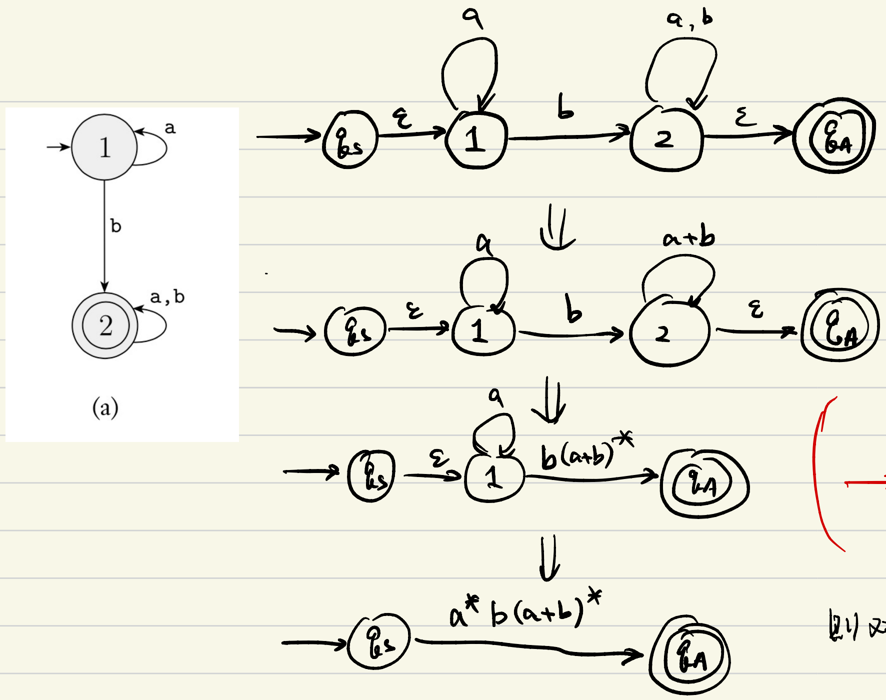
方法 1: GNFA.

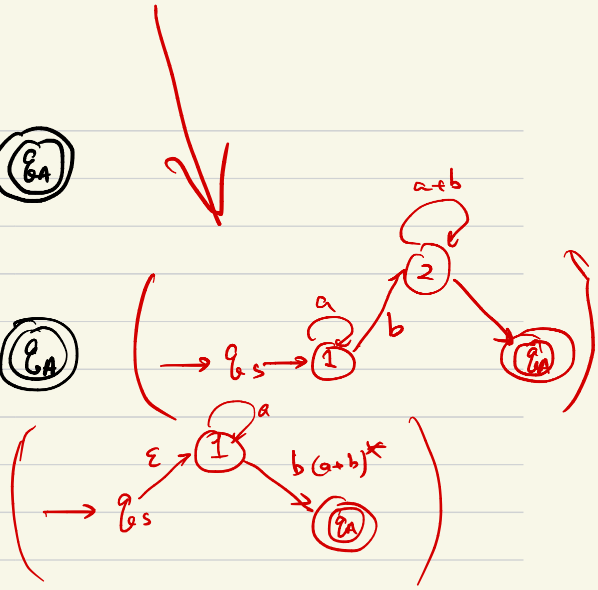
- q_{start} "入度"为 0, q_{end} "出度"为 0.
- $q_{accept} \neq q_{start}$.
- 一般省略不可能的转移 $(q_i, q_j) = \emptyset$

流程: 加首尾. 删空边. *省略. 并标号. 找内态.



例：



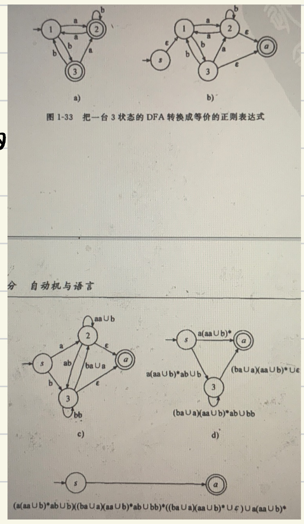


则对应的正则表达式为 $a^*b(a+b)^*$

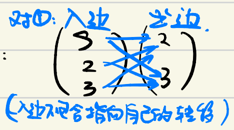
注：化简方法：按照标号顺序化简，

对每个标号有：确定入边，确定出边，一一对应，按照 o→o→o 化简

如对右图，我们想化简 ①，



则 r.p.1：



S→2：a

S→3：b

2→2：b+aa

2→3：ab

3→2：a+ba

3→3：bb

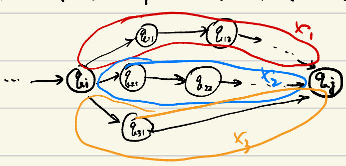
画到下一个图中即可。

方法二：R_{ij}^k 递归

i, j 无要求，可以大于 k.

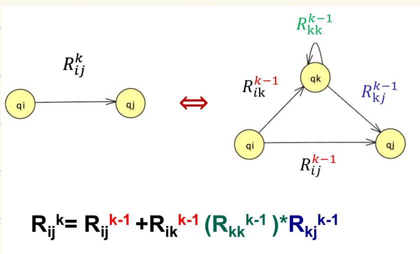
$$R_{ij}^k = \{x \mid \hat{\delta}(q_i, x) = q_j \text{ 且中间经过状态标号} \le k,\ x \in \Sigma^*\}$$

则 R_{ij}^k 这是字符串的集合，从 q_i 到 q_j 可以沿着 x 中的字符到达，而且这个旅途中经过的状态标号都小于 k.



q_{11} q_{12} …
q_{21} q_{22} … 中标号没有大于 k 的.
q_{31} …

基本思想：



从 i 到 j，中间标号没有大于 k 的.

||

从 i 到 j，中间标号没有大于 k-1 的 → 无 k

\+

从 i 到 k，中间标号没有大于 k-1 的

·

从 k 到 k，中间标号没有大于 k-1 的

·

从 k 到 j，中间标号没有大于 k-1 的 → 中间经过 k.

递归定义：

R_{ij}^k的递归定义如下：

① $R_{ij}^0 = \{a \mid a \in \sum, \text{且 } \delta(q_i,a) = q_j\}$ （i≠j）

② $R_{ij}^0 = \{a \mid a \in \sum, \text{且 } \delta(q_i,a) = q_j\} \cup \{\varepsilon\}$ （i = j）

③ $R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$ （k=1,2,…,n）

① : $q_i \xrightarrow{\Sigma} q_j$

② : Σ 或 ε 自环 q_i

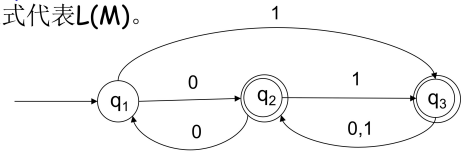
最后正则表达式就是 $r = \sum_{j \in A} r_{1j}^n$，A为接受状态的下标集合，n为状态数。

就是 从 q_1 到 $q_{accept1}$，中间状态标号不超 n.
+ 从 q_1 到 $q_{accept2}$，中间状态标号不超 n.
+ ....

注意开始状态不是接受状态，$1 \notin A$

例：

Example 4.8 给定一个DFA M，按照证明方法二 构造一个正则表达式代表L(M)。



（表格有 n^2 行，n+1 列（标号 0 ~ n））

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

	k=0	k=1	k=2	k=3
r_{11}^k	ε	ε	$\varepsilon + 0(\varepsilon+00)^*0 = \varepsilon + 00 + (00)^* = (00)^*$	
r_{12}^k	0	0	$0 + 0(\varepsilon+00)^*(\varepsilon+00) = 0(00)^*$	
r_{13}^k	1	1	0^*1	
r_{21}^k	0	0	$0(00)^*$	
r_{22}^k	ε	ε + 00	$(00)^*$	
r_{23}^k	1	1+01	0^*1	
r_{31}^k	∅	∅	$(0+1)(00)^*0$	
r_{32}^k	0+1	0 + 1	$(0+1)(00)^*$	
r_{33}^k	ε	ε	$\varepsilon + (0+1)0^*1$	

$0(00)^* + 0^*1\left(\varepsilon+(0+1)0^*1\right)^*(0+1)(00)^*$

$0^*1 + 0^*1\left(\varepsilon+(0+1)0^*1\right)^*\left(\varepsilon+(0+1)0^*1\right)$

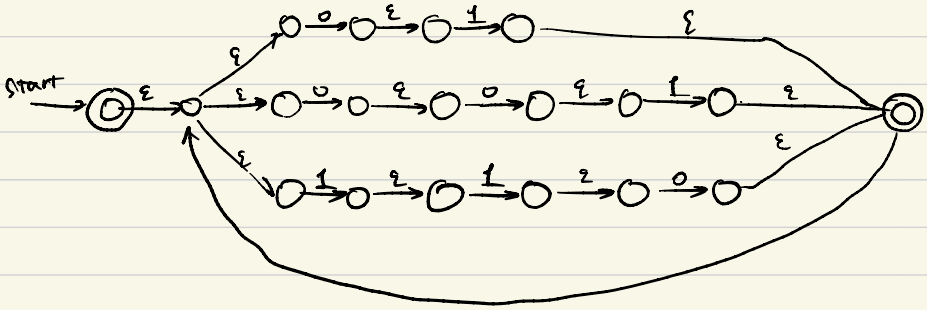
只有 r_{kk} 才能有 ε，否则 ∅

$r = r_{12}^3 + r_{13}^3$

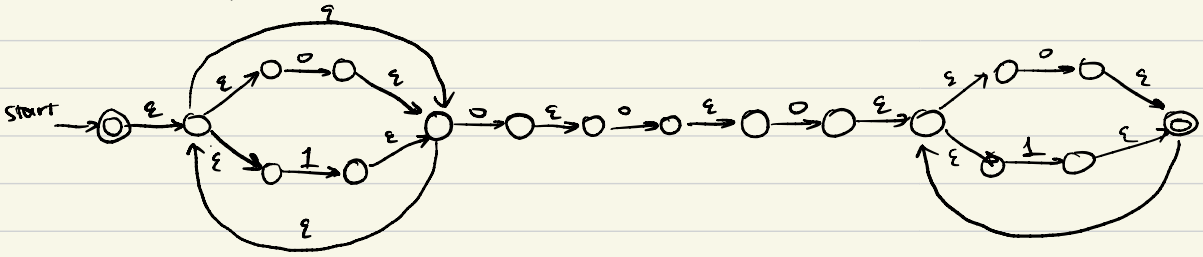
$= \ldots$

评价：不如画 GNFA 这么麻烦！

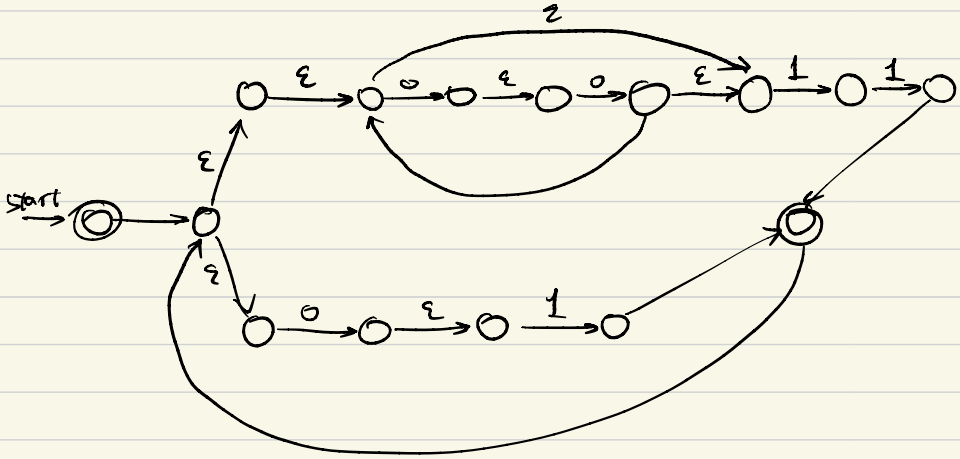
例：给出 $(01 + 001 + 010)^*$



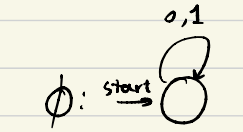
例：$(0+1)^*000(0+1)^*$

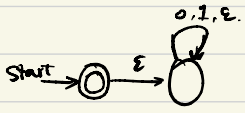


例：$\left(\left((00)^*(11)\right) + 01\right)^*$



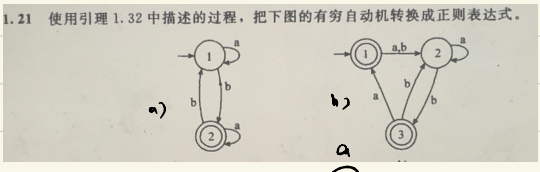
例：ϕ^*



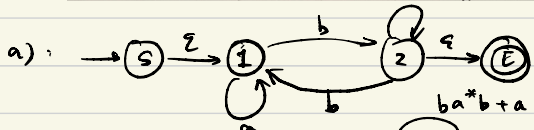


例：

1.21 使用引理 1.32 中描述的过程，把下图的有穷自动机转换成正则表达式。

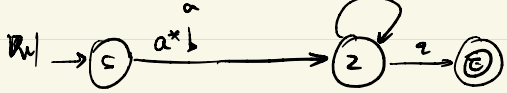


a)：



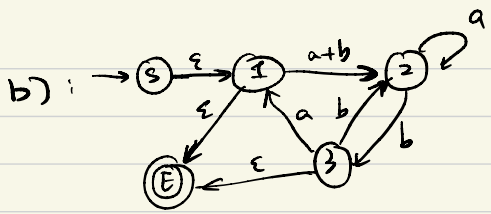
$\begin{pmatrix} S \\ 2 \end{pmatrix} \quad (2)$

例：



则：既为 $a^*b(ba^*b+a)^*$

b)：



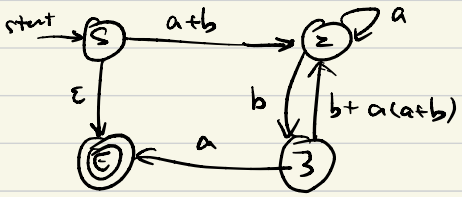
① : $\begin{pmatrix} S \\ 3 \end{pmatrix}\begin{pmatrix} E \\ 2 \end{pmatrix}$

$S \to E : \varepsilon$

$S \to 2 : a+b$

$3 \to E : a$

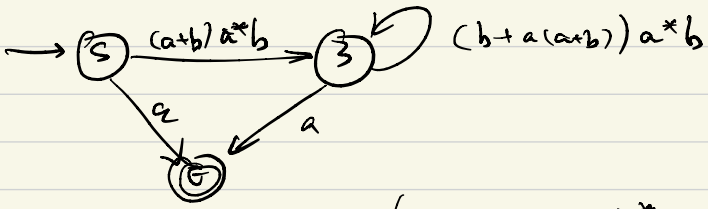
$3 \to 2 : b + a(a+b)$



② : $\begin{pmatrix} S \\ 3 \end{pmatrix}\begin{pmatrix} 3 \end{pmatrix}$

$S \to 3 : (a+b)a^*b$

$3 \to 3 : (b+a(a+b))a^*b$



最终RE : $\varepsilon + (a+b)a^*b\left((b+a(a+b))a^*b\right)^*a$

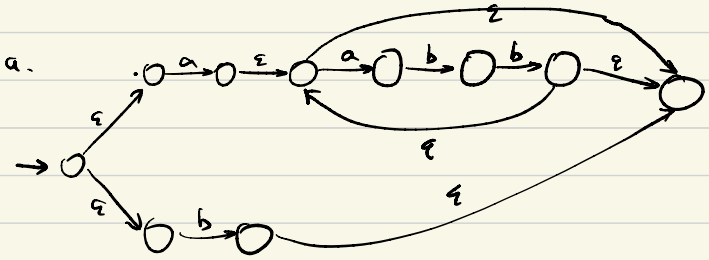
1.28 使用定理 1.28 给出的过程将下述正则表达式转换成 NFA。在所有问题中 $\Sigma=\{a, b\}$。

a. $a(abb)^* \cup b$

b. $a^+ \cup (ab)^+$

c. $(a \cup b^+)a^+b^+$

a.



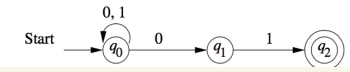
b、c 略

3、NFA转DFA

方法一：Th 3.1 法

不断对新产生的"状态"作用 Σ，直到不产生新状态为止。

EXP3-11 将下列NFA转化为等价的DFA。



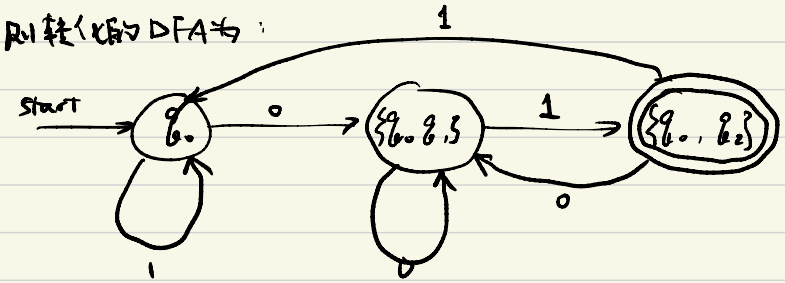
	0	1
q_0	$\{q_0, q_1\}$	q_0
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1\}$	q_0

所有的

带原来终止状态的在新DFA中就都是终止状态。

省略

则转化的DFA为：



方法二：子集构造法

第一步：构建 DFA 的状态（子集构造法）

1、如果q_1是NFA的初始状态，则$\{q_1\}$是DFA的一个状态；

2、如果p是NFA的一个状态，从初始状态，沿着带$a_1a_2\ldots a_m$符号的路径可达该p，则有一个DFA状态是由下列NFA状态组成的集合：

（a）q_1；

（b）p；

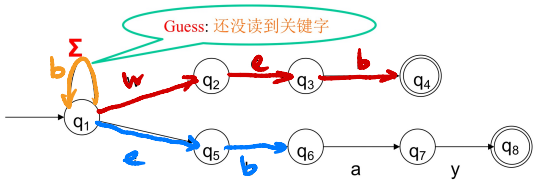
（c）每一个从q_1出发，沿着$a_1a_2\ldots a_m$的后缀可到达的NFA状态。

第二步：对构造好的DFA状态，计算DFA的状态转移

1、对于任一状态集合 $Q(q_1, p_1, \cdots, p_n)$，考察每个可能的输入x，如果在NFA中存在转移$q_i=\delta(p_i, x)$，则，在DFA中，构造转移$\{q_i\}=\delta(Q, x)$；

2、如果不存在从任何p_i出发的带x的转移，则，在DFA中，构造转移$\{q_1, \delta(q_1, x)\}=\delta(Q, x)$；

EXP 3-10：设计一台识别单词web和ebay的NFA。



举例说明：

1、按照上述的子集构造法，上例NFA中的起始状态q_1，必将出现在对应DFA的每个状态中；

2、从起始状态q_1出发，沿着带符号web的路径（q_1、q_2、q_3、q_4）可到达q_4；web的后缀为eb及b，其中eb满足2(c)的要求，即从q_1出发，沿着带eb标记的路径可达q_6。所以，q_1、q_4、q_6构成DFA的一个状态。

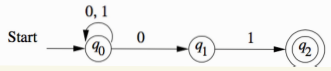
web: to q_4
eb: to q_6
b: to q_1
状态

ebay: to q_8
y: to q_1
状态

we, e: q_{135}　　w: q_{12}　　eba, a: q_{17}　　eb, b: q_{16}　　e: q_{15}

Σ: q_1

EXP3-11 将下列NFA转化为等价的DFA。



看后缀，打圈加留3.

$q_0 \to q_2$: 01.
$q_0 \to q_0$: 1.

(q_2, q_0)

$q_0 \to q_0$: 1

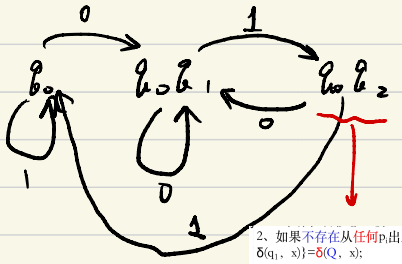
(q_0)

$q_0 \to q_1$: 0
$q_0 \to q_0$: 0

(q_0, q_1)

对所有状态作用Σ看状态

从q_0到每一个节点都试过了，则结束.

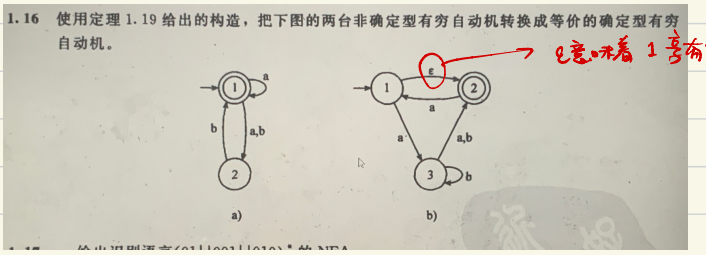


2、如果不存在从任何p_i出发的带x的转移，则，在DFA中，构造转移$\{q_i, \delta(q_i, x)\}=\delta(Q, x)$;

就是先让q_i接受x走一步

例1

1.16 使用定理 1.19 给出的构造，把下图的两台非确定型有穷自动机转换成等价的确定型有穷自动机。



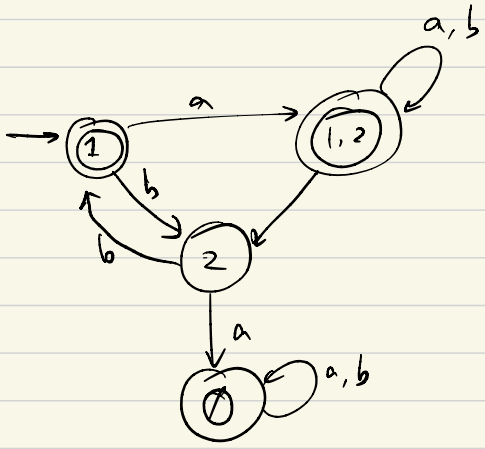
ε意味着1拿有和2一样的转移权利！

到达时2自己转移

也为停在1内部就行了。

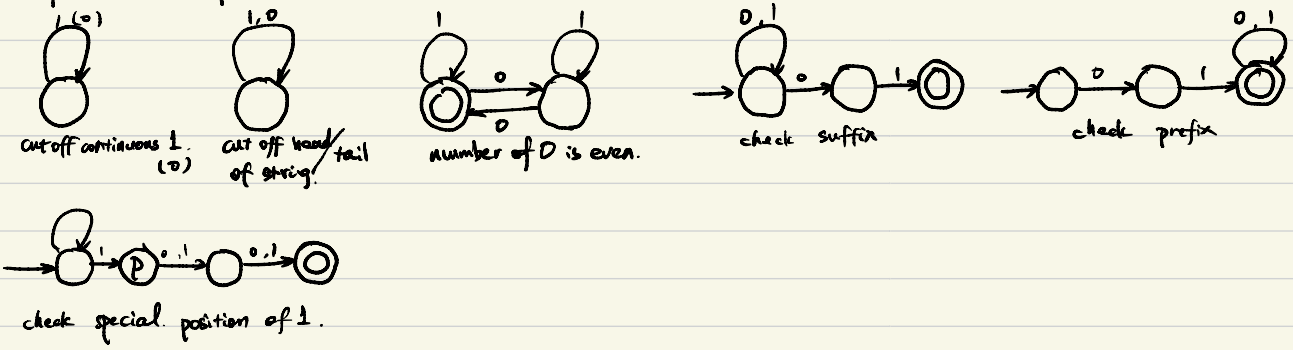
a):

	a	b
*→ 1	1, 2	2
2	∅	1.
* 1, 2.	1, 2	1, 2
∅	∅	∅

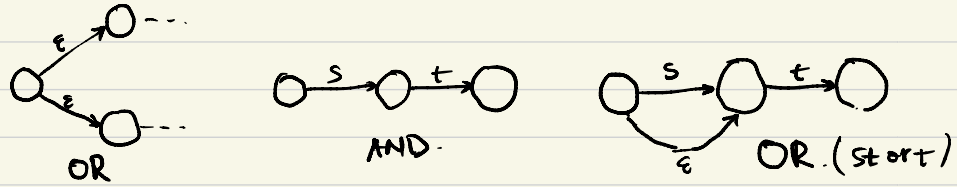


1. DFA 构造

① Experience (Inspiration)



② Combination

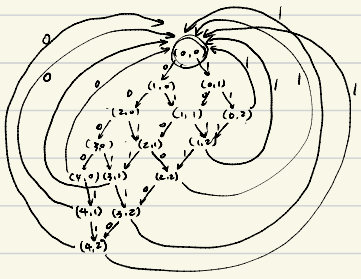


③ limited multiple status.

$\begin{cases} \text{number of 0}: 5n \\ \text{number of 1}: 3n. \end{cases}$

(n_1, n_2) : $\begin{cases} \text{number of 0}: 5k+n_1 \\ \text{number of 1}: 3k+n_2 \end{cases}$

(0,0)



"TREE"

④ deduction.

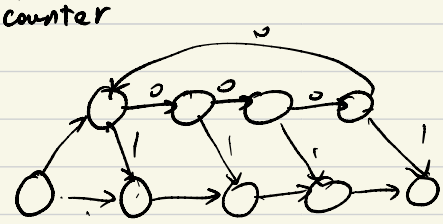
binary string : multiple of 5

101 : accept 1010 : accept.

$a_n = (2a_{n-1} + X) \% 5 \quad (X \in \{0,1\})$

0 1 2 3 4

⑤ counter



⑥ Cycle: math + mod

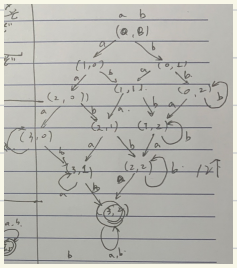
例：

"和"类型

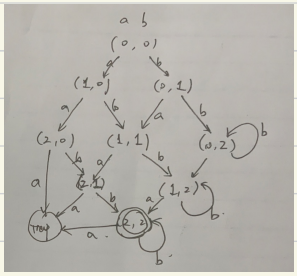
a. {w | w 含有至少 3 个 a 和至少 2 个 b}
A b. {w | w 含有正好 2 个 a 和至少 2 个 b}
c. {w | w 含有偶数个 a 和 1 个或 2 个 b}
A d. {w | w 含有偶数个 a 并且每个 a 后都跟有至少一个 b}
e. {w | w 从 a 开始并且最多有 1 个 b}
f. {w | w 含有奇数个 a 并且以 b 结束}
g. {w | w 的长度为偶数并且有奇数个 a}

还是没找到通式.

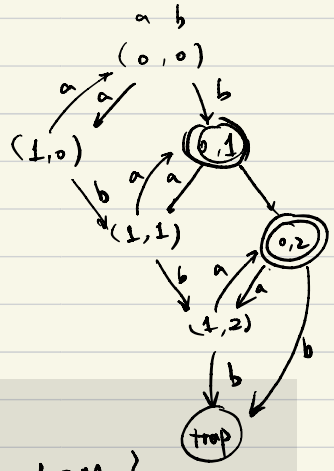
a.



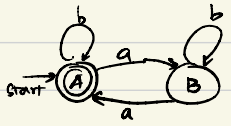
b.



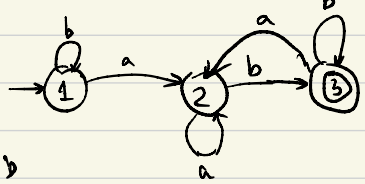
c.



d. 含偶数个a：



首个a后过渡到一个B：



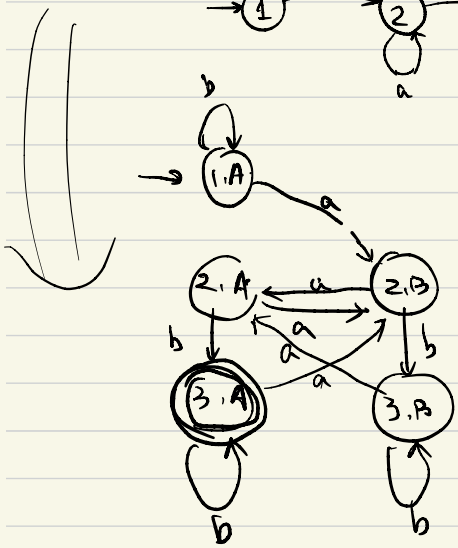
两台一起：

$Q: \{ \langle a,b \rangle \mid a \in M_1, b \in M_2 \}$

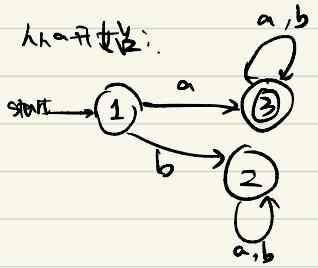
$q_{00} = \langle q_{01}, q_{02} \rangle$

$\delta(\langle a,b \rangle, x) = \delta(\delta_1(a,x), \delta_2(b,x))$

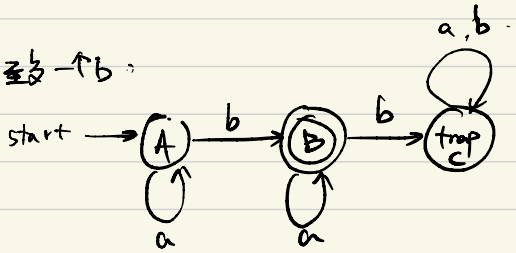
$F: \{ \langle a,b \rangle \mid a \in F_1 \text{ and } b \in F_2 \}$

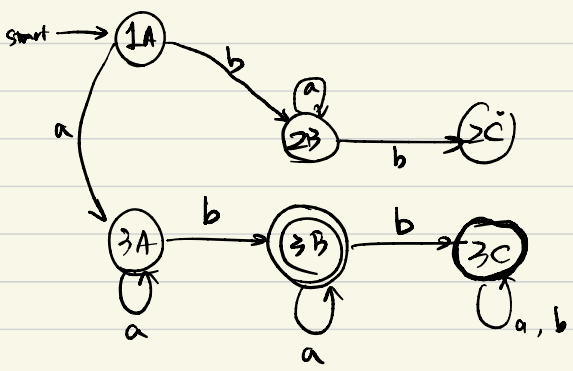


e. 从a开始：



至多一个b：





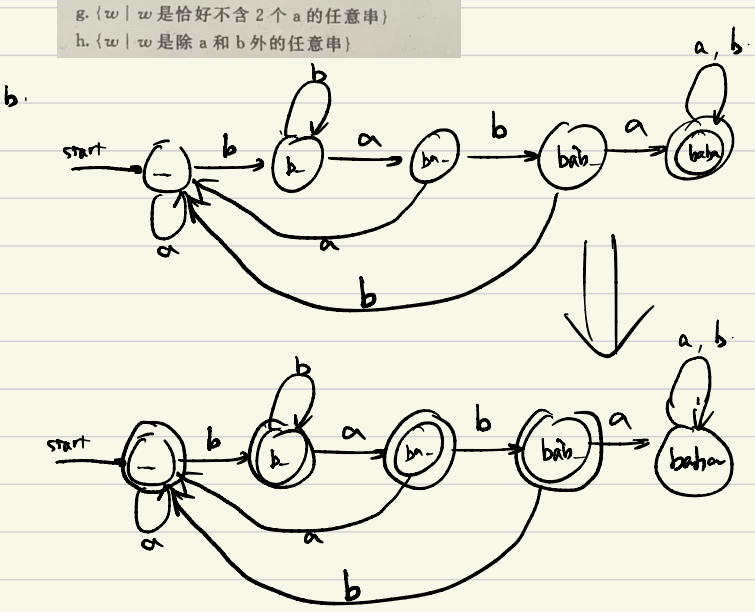
"补"类型

例：

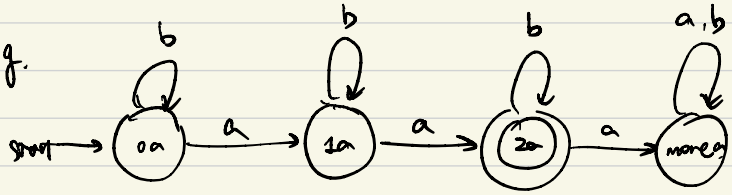
b. {w | w 中不含子串 baba}
c. {w | w 中既不含子串 ab 也不包含子串 ba}
d. {w | w 是不在 a^*b^* 中的任意串}
e. {w | w 是不在 $(ab^+)^*$ 中的任意串}
f. {w | w 是不在 $a^* \cup b^*$ 中的任意串}
g. {w | w 是恰好不含 2 个 a 的任意串}
h. {w | w 是除 a 和 b 外的任意串}

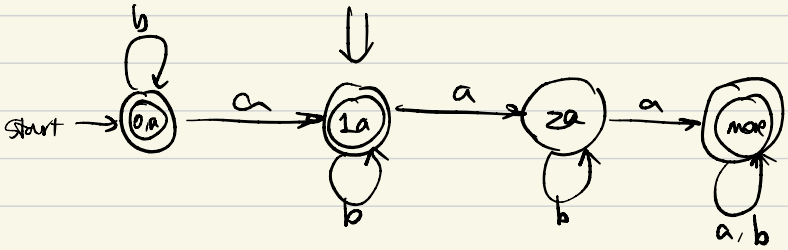
即，正着写，添把接受和非接受态，掉换过来即可。

b.



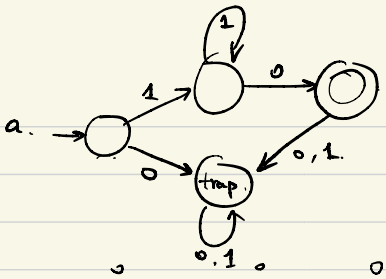
g.

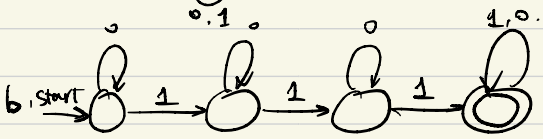




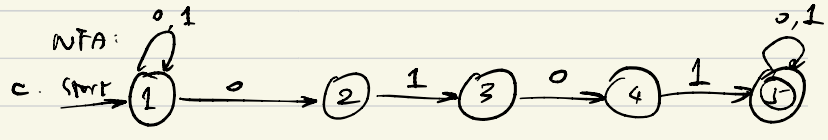
1.6 画出识别下述语言的 DFA 状态图。在所有问题中字母表均为{0，1}

a. {w | w 从 1 开始且以 0 结束}
b. {w | w 含有至少 3 个 1}
c. {w | w 含有子串 0101，即对某个 x 和 y，$w=x0101y$}
d. {w | w 的长度不小于 3，并且第 3 个符号为 0}
e. {w | w 从 0 开始且长度为奇数，或者从 1 开始且长度为偶数}
f. {w | w 不含子串 110}
g. {w | w 的长度不超过 5}
h. {w | w 是除 11 和 111 外的任意串}
i. {w | w 的奇数位置均为 1}
j. {w | w 含有至少 2 个 0，并且至多含 1 个 1}
k. {ε，0}
l. {w | w 含有偶数个 0 或恰好 2 个 1}
m. 空集
n. 除空串外的所有字符串





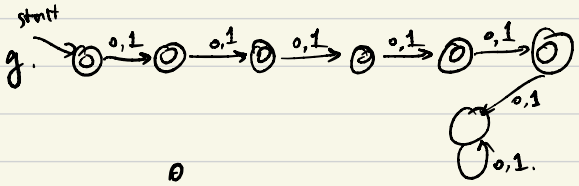
含有子串 ⟶ 可以变为 NFA再转DFA.

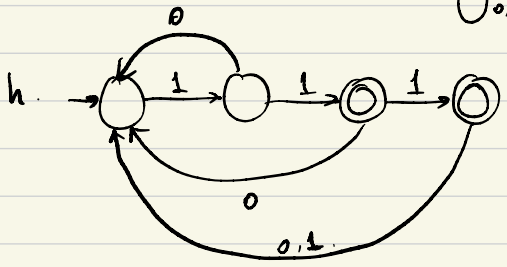


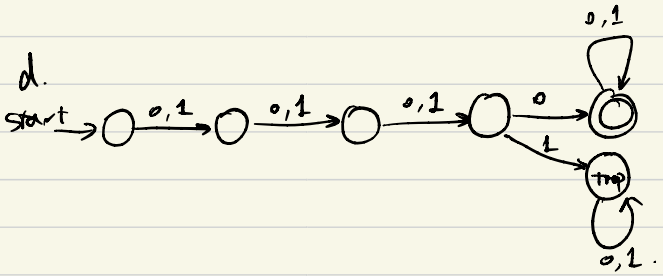
DFA:

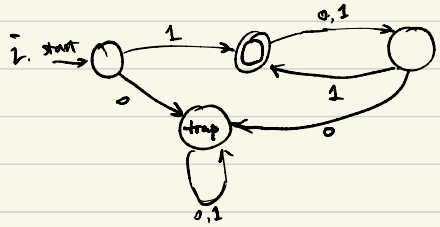
	0	1
1	12	1
12	12	13
13	124	1
124	12	135
135	1245	15
1245	125	135
125	125	135

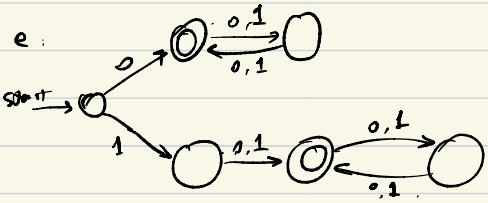
⇒ ✓

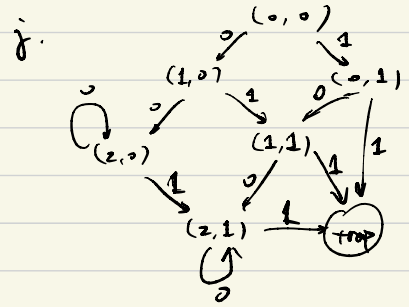


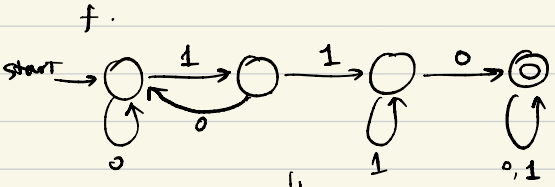




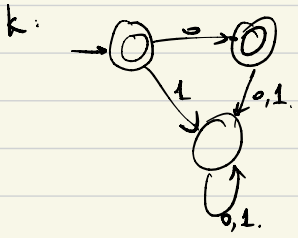


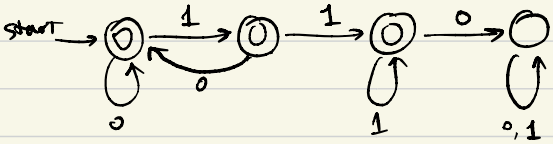




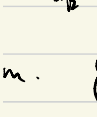


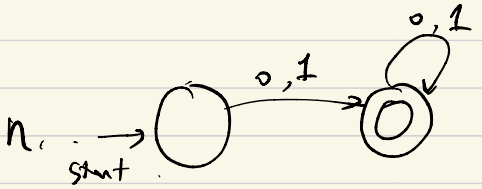
Reverse!



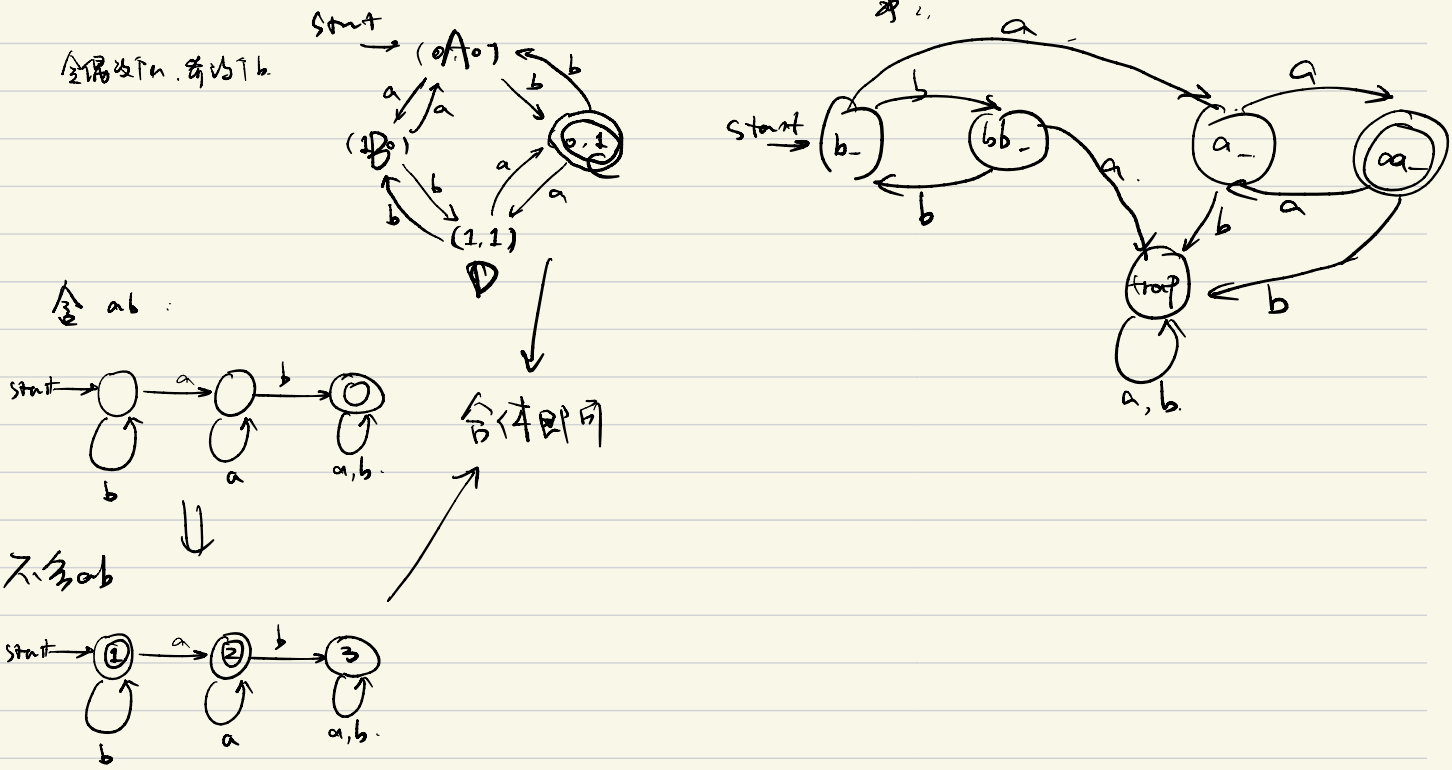


l. 略





1.12 令 $D=\{w \mid w$ 含有偶数个 a 和奇数个 b 并且不包含子串 ab$\}$。画出识别 D 的 5 状态的 DFA 图和生成 D 的正则表达式。（建议尽量简单地描述 D。）



1.13 令 F 是$\{0, 1\}$上所有串构成的语言，并且 F 中任意两个 1 之间间隔的符号数都不是奇数。画出识别 F 的 5 状态的 DFA 图。（先找一个 4 状态的 NFA 作 F 的补集可能有助于求解问题。）