编译原理

有限自动机的等价性

编译原理

有限自动机的等价性

——回顾

词法分析

- 词法分析器的设计
- 正规表达式与有限自动机
- 词法分析器的自动产生--LEX

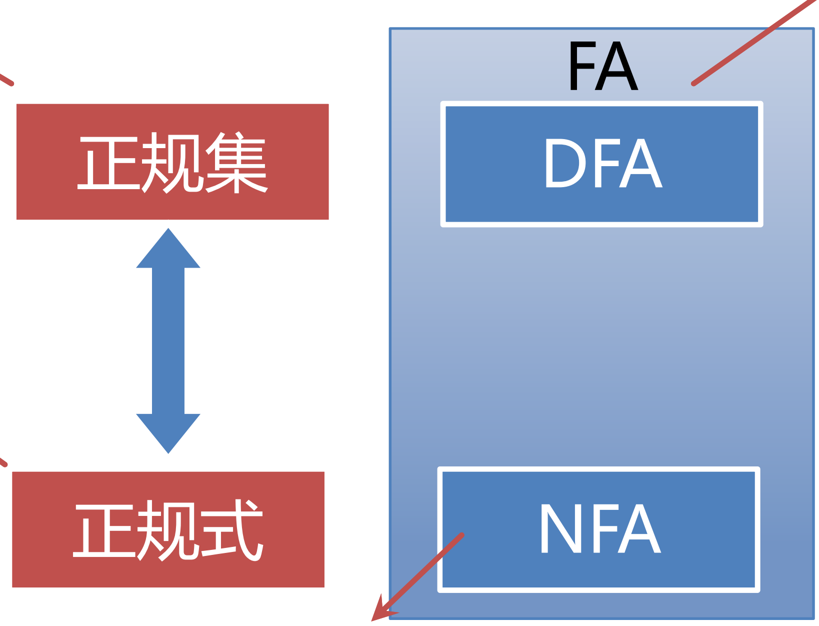
正规式、正

单词符号	种别编码	助忆符	内码值
DIM	1	$DIM	-
IF	2	$IF	-

```
curState = 初态
GetChar();
while( stateTrans[curState][ch]有定义){
    //存在后继状态，读入、拼接
    Concat();
    //转换入下一状态，读入下一字符
    curState= stateTrans[curState][ch];
    if curState是终态 then 返回strToken中的单词
    GetChar( );
}
```

DIM,IF, DO,STOP,E
number, name, ag
125, 2169
…

DIM
IF
DO
STOP
END
letter(letter|digit)*
digit(digit)*



易于人工设计

编译原理

有限自动机的等价性

——NFA转换成DFA

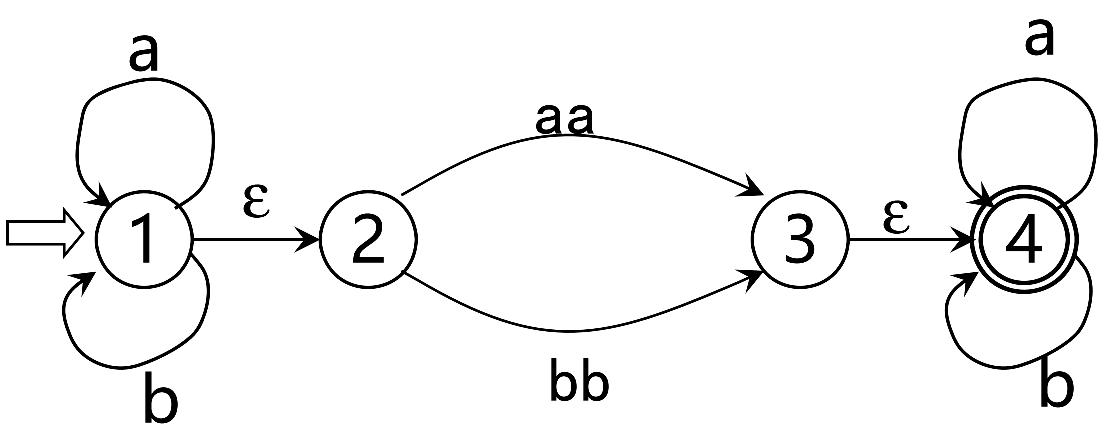
DFA与NFA的等价性

- 对于每个NFA M存在一个DFA M’，使得L(M)=L(M’)
  - 等价性证明
  - NFA的确定化
- 思路： NFA 和DFA的差别

	NFA	DFA
初始状态	不唯一	唯一
弧上的标记	字(单字符字、ε)	字符
转换关系	非确定	确定

DFA与NFA的等价性证明

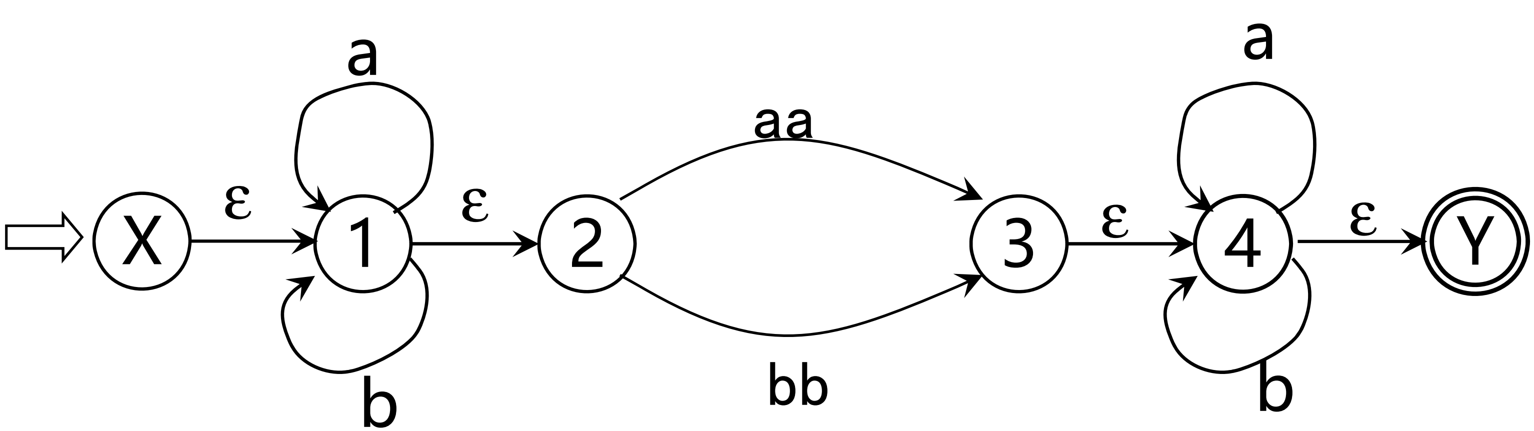
- 假定NFA M=<S, Σ, δ, S_0, F>，我们对M的状态转换图进行以下改造：
  - 引进新的初态结点X和终态结点Y，X,Y∉S，从X到S_0中任意状态结点连一条ε箭弧， 从F中任意状态结点连一条ε箭弧到Y。



识别所有含相继两个a或相继两个b的字的NFA

DFA与NFA的等价性证明

- 假定NFA M=<S, Σ, δ, S_0, F>，我们对M的状态转换图进行以下改造：
  - 引进新的初态结点X和终态结点Y，X,Y∉S，从X到S_0中任意状态结点连一条ε箭弧，从F中任意状态结点连一条ε箭弧到Y。 (解决初始状态唯一性)



识别所有含相继两个a或相继两个b的字的NFA

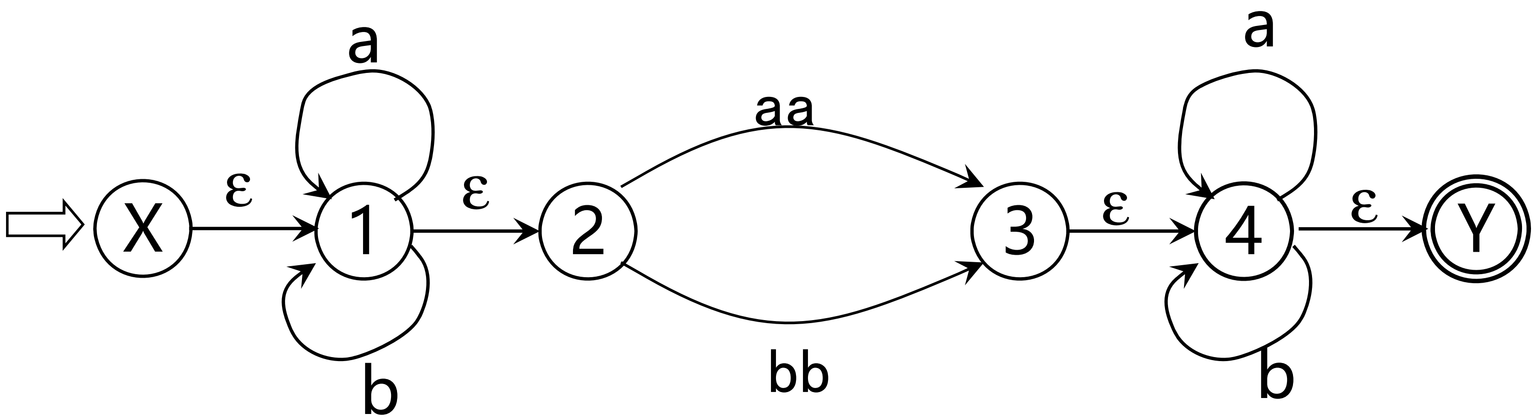
DFA与NFA的等价性证明

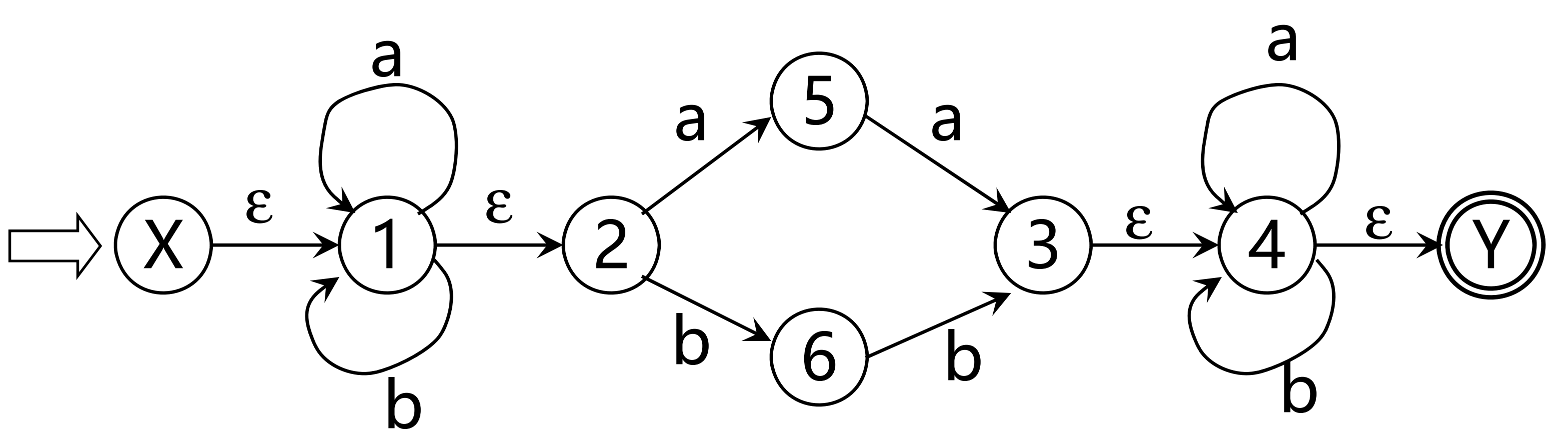
- 假定NFA M=<S, Σ, δ, S_0, F>，我们对M的状态转换图进行以下改造：
  - 引进新的初态结点X和终态结点Y，X,Y$\notin$S，从X到S_0中任意状态结点连一条ε箭弧，从F中任意状态结点连一条ε箭弧到Y。 (解决初始状态唯一性)
  - 对M的状态转换图进一步施行替换，其中k是新引入的状态。 (简化弧上的标记)

$i \xrightarrow{AB} j$ 代之为 $i \xrightarrow{A} k \xrightarrow{B} j$

DFA与NFA的等价性证明

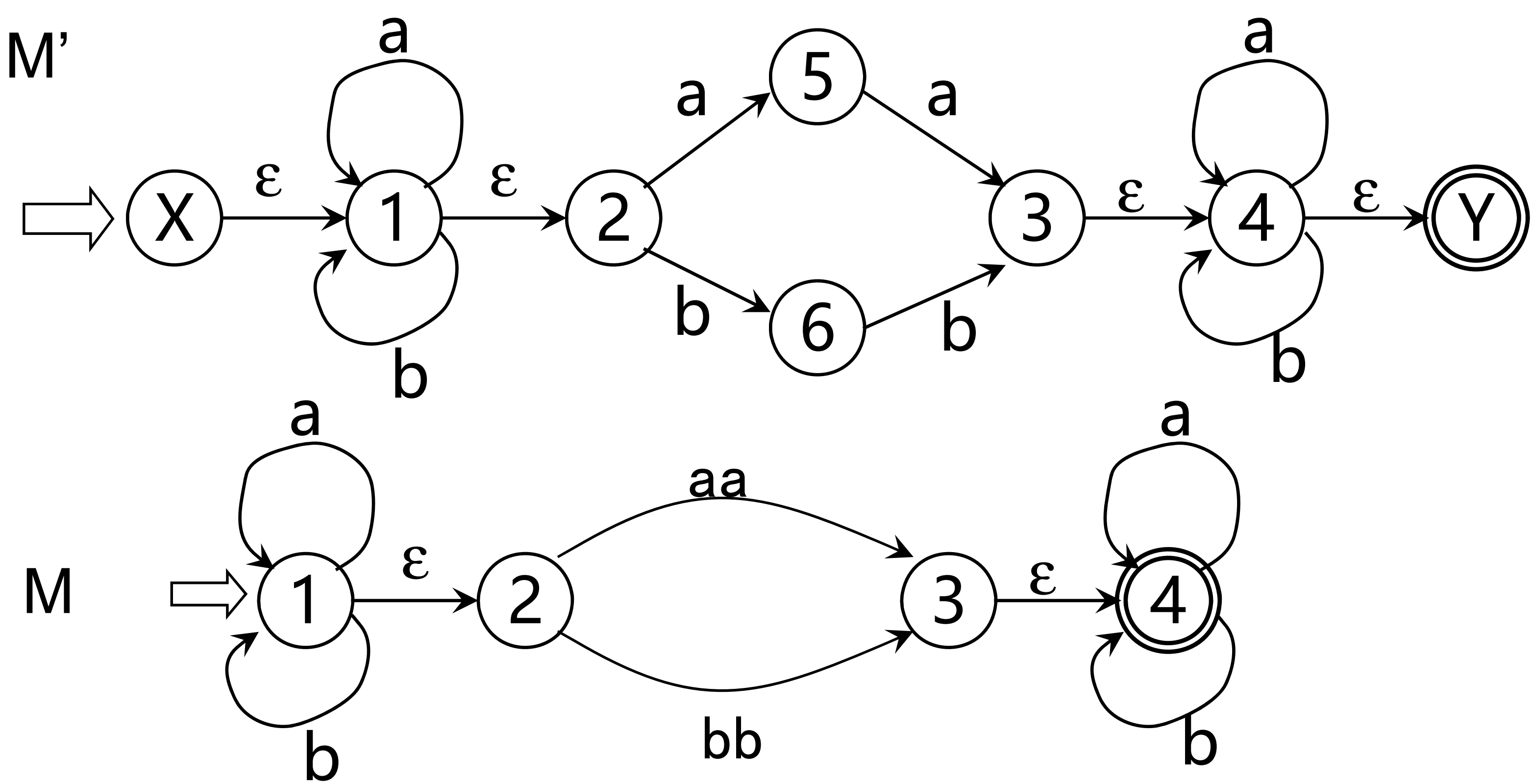
- 识别所有含相继两个a或相继两个b的字





DFA与NFA的等价性证明

- 逐步把这个图转变为每条弧只标记为Σ上的一个字符或ε，最后得到一个NFA M’，显然L(M’)=L(M)



DFA与NFA的等价性证明

- NFA确定化--子集法 (解决ε弧和转换关系)
- 设I是的状态集的一个子集，定义I的ε-闭包ε-closure(I)为:
  - 若s∈I，则s∈ε-closure(I)；
  - 若s∈I，则从s出发经过任意条ε弧而能到达的任何状态s'都属于ε-closure(I)

即，

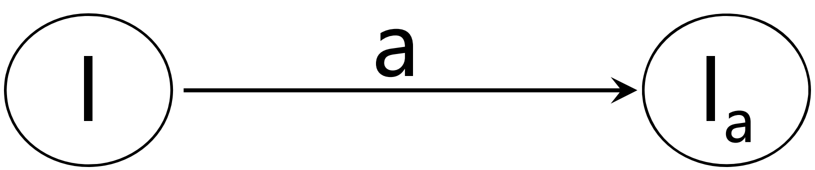
ε-closure(I)=I∪{s' |从某个s∈I出发经过任意条ε弧能到达s' }

DFA与NFA的等价性证明

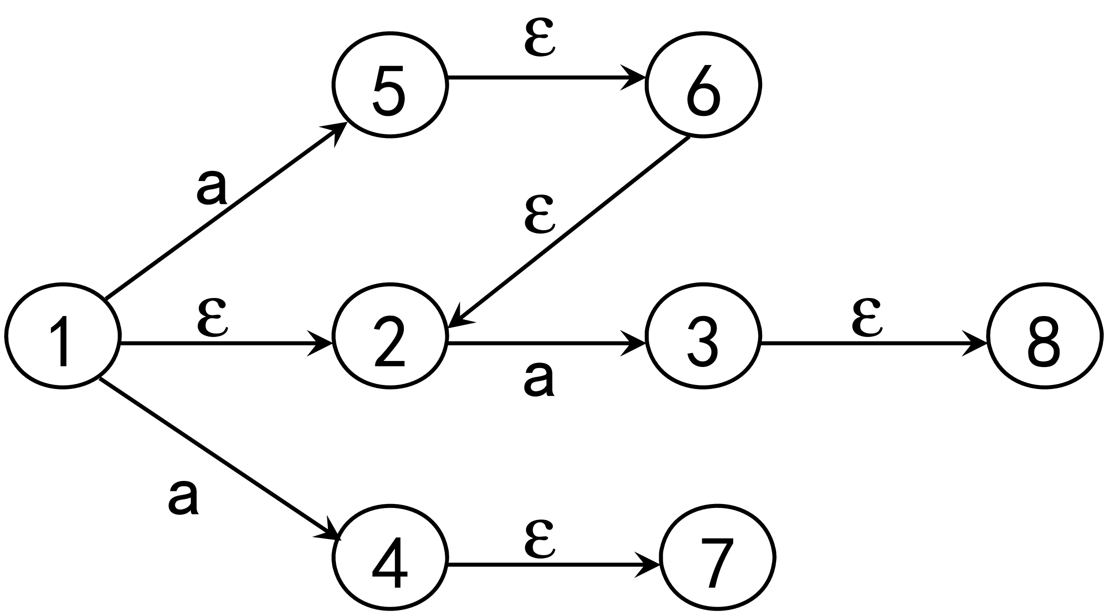
- 设a是Σ中的一个字符，定义

$$I_a = \varepsilon\text{-closure}(J)$$

其中，J为I中的某个状态出发经过一条a弧而到达的状态集合。



DFA与NFA的等价性证明



I_a= ε-closure(J)
其中，J为I中的某个状态出发经过一条a弧而到达的状态集合。

ε-closure({1}) = {1， 2} = I　I_a?

J={5， 4， 3}

I_a= ε-closure(J) = ε-closure({5， 4， 3})

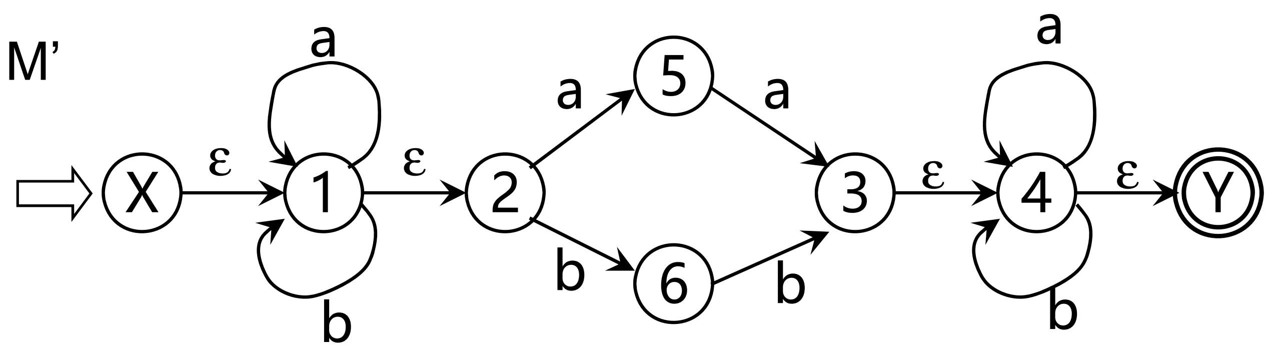
={5， 4， 3， 6， 2， 7， 8}

DFA与NFA的等价性证明

- 确定化：不失一般性，设字母表只包含两个 *a* 和*b*，我们构造一张计算状态集的转换表:
  - 首先，置第1行第1列为ε-closure({X})求出这一列的I_a，I_b；
  - 然后，检查这两个I_a，I_b，看它们是否已在表中的第一列中出现，把未曾出现的填入后面的空行的第1列上，求出每行第2，3列上的集合...
  - 重复上述过程，直到所有第2，3列子集全部出现在第一列为止

I	I_a	I_b
ε–Closure({X})	{…}	{…}
{…}	{…}	{…}
{…}	{…}	{…}

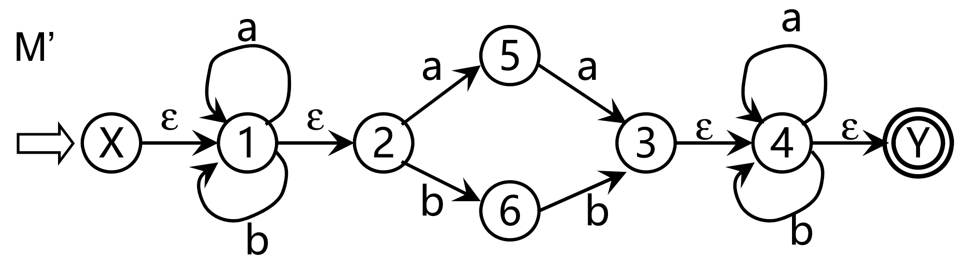
DFA与NFA的等价性证明



I	I_a	I_b
ε-closure({X})= {X, 1, 2}	{1, 5, 2}	{1, 6, 2}
{1, 5, 2}	{1, 3, 5, 2, 4, Y}	{1, 6, 2}
{1, 6, 2}	{1, 5, 2}	{1, 3, 6, 2, 4, Y}
{1, 3, 5, 2, 4, Y}	{1, 3, 5, 2, 4, Y}	{1, 6, 4, 2, Y}
{1, 3, 6, 2, 4, Y}	{1, 5, 4, 2, Y}	{1, 3, 6, 2, 4, Y}
{1, 6, 4, 2, Y}	{1, 5, 4, 2, Y}	{1, 3, 6, 2, 4, Y}
{1, 5, 4, 2, Y}	{1, 3, 5, 2, 4, Y}	{1, 6, 4, 2, Y}

DFA与NFA的等价性证明

- 把表看成状态转换矩阵，子集视为状态
- 转换表唯一刻划了一个确定的有限自动机M
  - 初态是ε-closure({X})
  - 终态是含有原终态Y的子集
- 不难看出，这个DFA M与M'等价
- 对于每个NFA M存在一个DFA M'，使得 L(M)=L(M')
- NFA和DFA等价

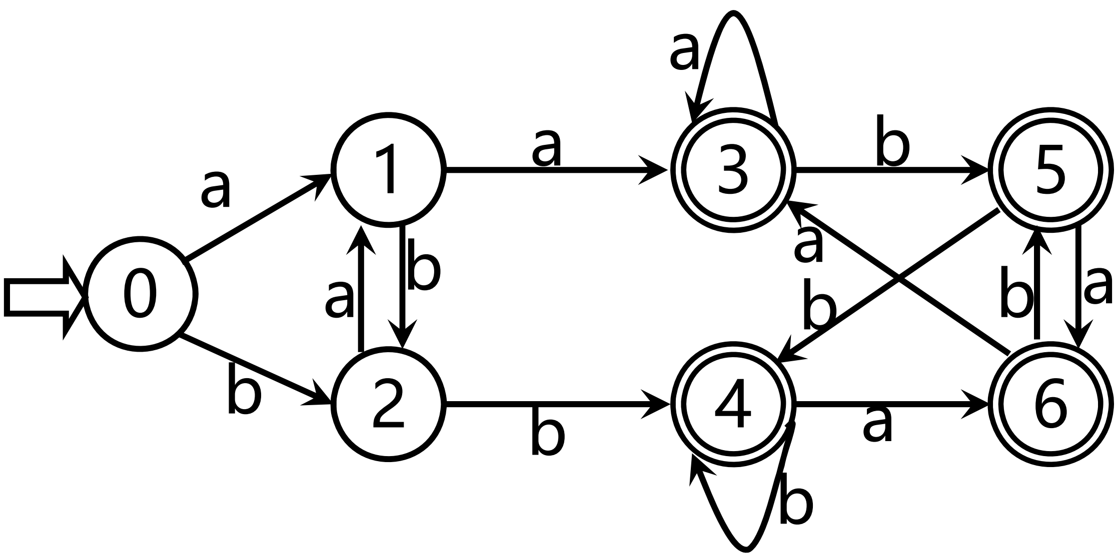


I	I_a	I_b
{X, 1, 2}	{1, 5, 2}	{1, 6, 2}
{1, 5, 2}	{1, 3, 5, 2, 4, Y}	{1, 6, 2}
{1, 6, 2}	{1, 5, 2}	{1, 3, 6, 2, 4, Y}
{1, 3, 5, 2, 4, Y}	{1, 3, 5, 2, 4, Y}	{1, 6, 4, 2, Y}
{1, 3, 6, 2, 4, Y}	{1, 5, 4, 2, Y}	{1, 3, 6, 2, 4, Y}
{1, 6, 4, 2, Y}	{1, 5, 4, 2, Y}	{1, 3, 6, 2, 4, Y}
{1, 5, 4, 2, Y}	{1, 3, 5, 2, 4, Y}	{1, 6, 4, 2, Y}

DFA与NFA的等价性证明

I	I_a	I_b
{X, 1, 2}	{1, 5, 2}	{1, 6, 2}
{1, 5, 2}	{1, 3, 5, 2, 4, Y}	{1, 6, 2}
{1, 6, 2}	{1, 5, 2}	{1, 3, 6, 2, 4, Y}
{1, 3, 5, 2, 4, Y}	{1, 3, 5, 2, 4, Y}	{1, 6, 4, 2, Y}
{1, 3, 6, 2, 4, Y}	{1, 5, 4, 2, Y}	{1, 3, 6, 2, 4, Y}
{1, 6, 4, 2, Y}	{1, 5, 4, 2, Y}	{1, 3, 6, 2, 4, Y}
{1, 5, 4, 2, Y}	{1, 3, 5, 2, 4, Y}	{1, 6, 4, 2, Y}

I	a	b
0	1	2
1	3	2
2	1	4
3	3	5
4	6	4
5	6	4
6	3	5



单词符号	种别编码	助忆符	内码值
DIM	1	$DIM	-
IF	2	$IF	-

正规式、正

```
curState = 初态
GetChar();
while( stateTrans[curState][ch]有定义){
    //存在后继状态，读入、拼接
    Concat();
    //转换入下一状态，读入下一字符
    curState= stateTrans[curState][ch];
    if curState是终态 then 返回strToken中的单词
    GetChar( );
}
```

DIM,IF, DO,STOP,E
number, name, ag
125, 2169
…

DIM
IF
DO
STOP
END
letter(letter|digit)*
digit(digit)*

正规集

正规式

FA

DFA

NFA

易于人工设计

编译原理

有限自动机等价性
——DFA的化简

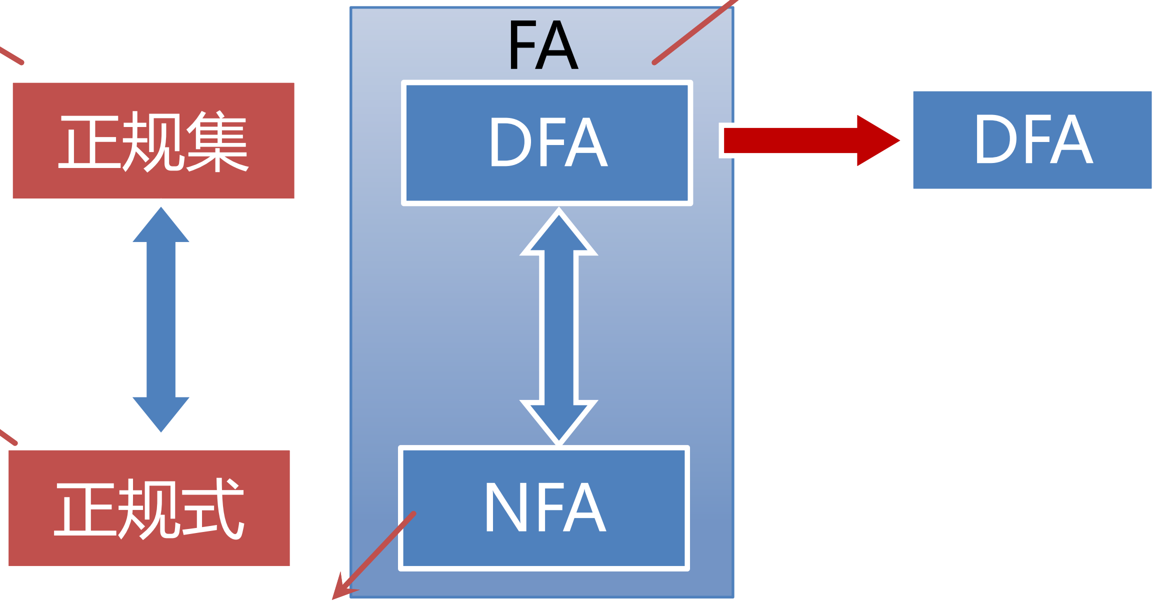
正规式、正

单词符号	种别编码	助忆符	内码值
DIM	1	$DIM	-
IF	2	$IF	-

```
curState = 初态
GetChar();
while( stateTrans[curState][ch]有定义){
    //存在后继状态，读入、拼接
    Concat();
    //转换入下一状态，读入下一字符
    curState= stateTrans[curState][ch];
    if curState是终态 then 返回strToken中的单词
    GetChar( );
}
```

DIM,IF, DO,STOP,END
number, name, age
125, 2169
…

DIM
IF
DO
STOP
END
letter(letter|digit)*
digit(digit)*



易于人工设计

确定有限自动机的化简

- DFA的化简(最小化)
  - 对于给定的DFA M，寻找一个状态数比M少的DFA M’，使得L(M)=L(M’)
- 状态的等价性
  - 假设s和t为M的两个状态，称s和t等价：如果从状态s出发能读出某个字α而停止于终态，那么同样，从t出发也能读出α而停止于终态；反之亦然
  - 两个状态不等价，则称它们是可区别的

测试：状态的可区分性

- 两个状态s和t是可区分的，是指( )

A. 对于任意字α，要么s读出α停止于终态而t读出α停止于非终态，要么t读出α停止于终态而s读出α停止于非终态

B. 存在一个字α，要么s读出α停止于终态而t读出α停止于非终态，要么t读出α停止于终态而s读出α停止于非终态

确定有限自动机的化简

- 基本思想
  - 把M的状态集划分为一些不相交的子集，使得任何两个不同子集的状态是可区别的，而同一子集的任何两个状态是等价的。

确定有限自动机的化简

- 基本思想
  - 把M的状态集划分为一些不相交的子集，使得任何两个不同子集的状态是可区别的，而同一子集的任何两个状态是等价的。
  - 最后，让每个子集选出一个代表，同时消去其他状态。

确定有限自动机的化简

- 基本思想
  - 把M的状态集划分为一些不相交的子集，使得任何两个不同子集的状态是可区别的，而同一子集的任何两个状态是等价的。
  - 最后，让每个子集选出一个代表，同时消去其他状态。

确定有限自动机的化简

- 基本思想
  - 把M的状态集划分为一些不相交的子集，使得任何两个不同子集的状态是可区别的，而同一子集的任何两个状态是等价的。
  - 最后，让每个子集选出一个代表，同时消去其他状态。

测试：初始划分

- 按照上述原则对DFA的状态集合S进行第一次划分，正确的分法是( )

A. 初态和非初态

B. 终态和非终态

C. 初态、终态、其他状态

把状态集划分为一些不相交的子集，使得任何两个不同子集的状态是可区别的，而同一子集的任何两个状态是等价的。

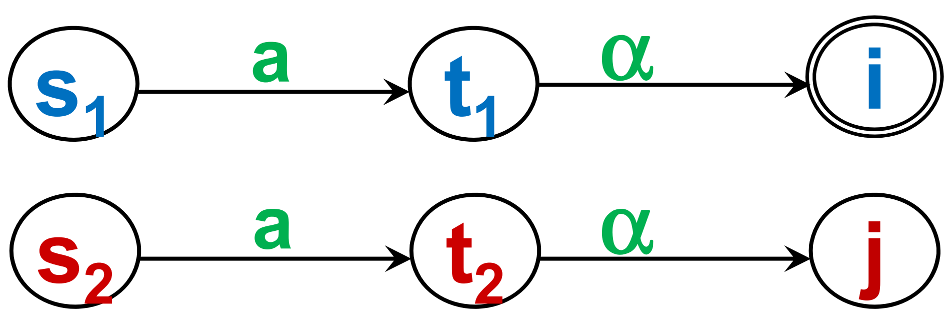
可区别：存在一个字α，要么s读出α停止于终态而t读出α停止于非终态，要么t读出α停止于终态而s读出α停止于非终态

确定有限自动机的化简

- 首先，把S划分为终态和非终态两个子集，形成基本划分Π。
- 假定到某个时候，Π已含m个子集，记为$\Pi=\{I^{(1)}, I^{(2)}, \ldots, I^{(m)}\}$，检查Π中的每个子集看是否能进一步划分：
  - 对某个$I^{(i)}$，令$I^{(i)}=\{s_1,s_2, \ldots,s_k\}$，若存在一个输入字符a使得$I_a^{(i)}$不会包含在现行Π的某个子集$I^{(j)}$中，则至少应把$I^{(i)}$分为两个部分。

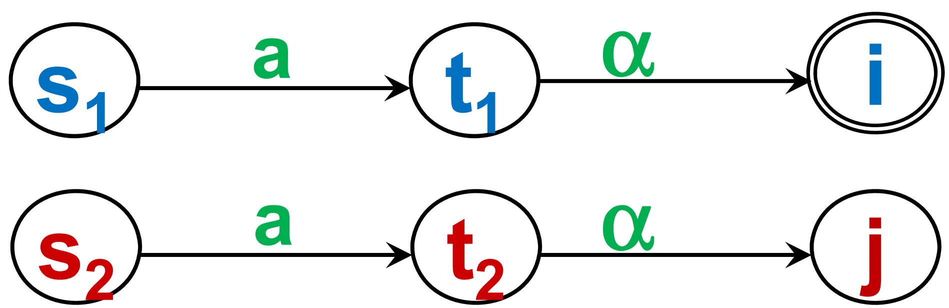
确定有限自动机的化简

- 假定状态s_1和s_2是$I^{(i)}=\{s_1,s_2, \ldots,s_k\}$中的两个状态，它们经a弧分别到达$t_1$和$t_2$，而$t_1$和$t_2$属于现行$\Pi$中的两个不同子集
  - 说明有一个字α，t_1读出α后到达终态，而t_2读出α后不能到达终态，或者反之
  - 那么对于字$a\alpha$，s_1读出$a\alpha$后到达终态，而s_2读出$a\alpha$不能到达终态，或者反之
  - 所以s_1和s_2不等价



确定有限自动机的化简

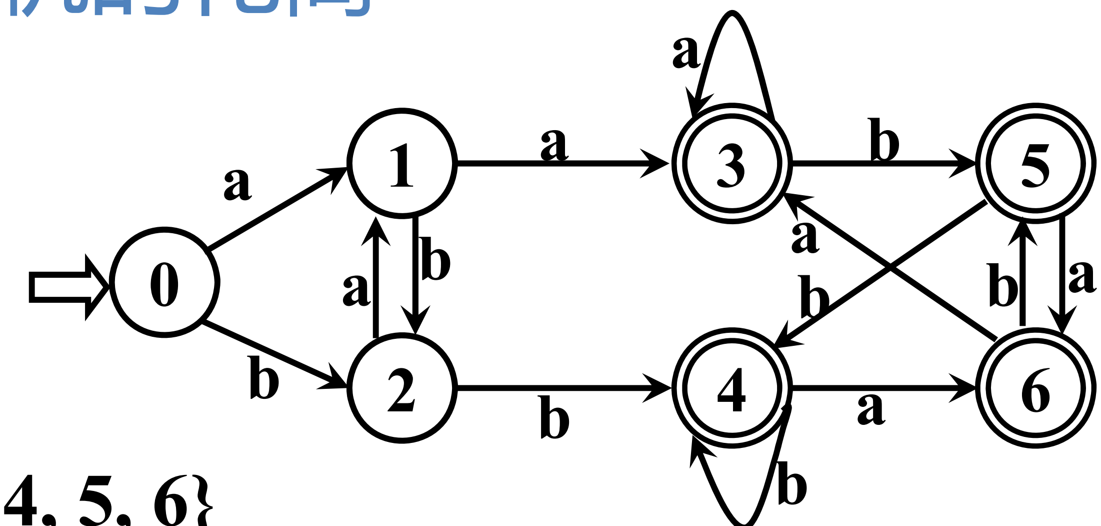
- 将$I^{(i)}$分成两半，一半含有s_1，一半含有s_2
  - $I^{(i1)}$含有s_1: $I^{(i1)} = \{s|s\in I^{(i)}$且s经a弧到达t, 且t与$t_1$属于现行$\Pi$中的同一子集$\}$
  - $I^{(i2)}$含有s_2 : $I^{(i2)} = I^{(i)} - I^{(i1)}$



确定有限自动机的化简

- 一般地，对某个a和$I^{(i)}$，若$I_a^{(i)}$落入现行Π中 N 个不同子集，则应把$I^{(i)}$划分成N个不相交的组，使得每个组J的J_a都落入的Π同一子集。
- 重复上述过程，直到Π所含子集数不再增长。
- 对于上述最后划分Π中的每个子集，我们选取每个子集I中的一个状态代表其他状态，则可得到化简后的DFA M’。
- 若I含有原来的初态，则其代表为新的初态，若I含有原来的终态，则其代表为新的终态。

确定有限自动机的化简



$I^{(1)}=\{0, 1, 2\}$ $I^{(2)}=\{3, 4, 5, 6\}$

$I_a^{(1)}=\{1, 3\}$

$I^{(11)}=\{0, 2\}$ $I^{(12)}=\{1\}$ $I^{(2)}=\{3, 4, 5, 6\}$

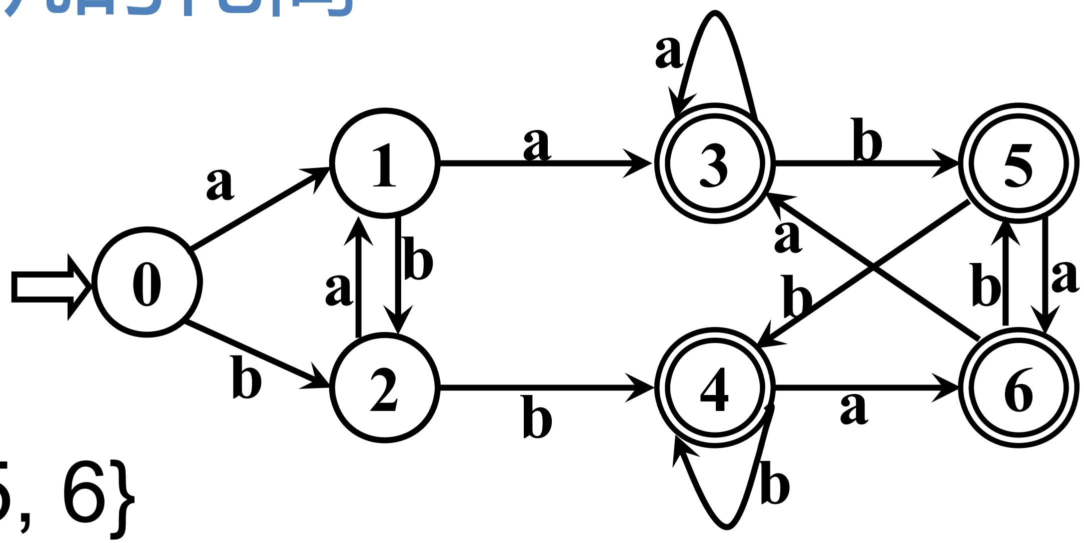
$I^{(11)}=\{0, 2\}$

$I_a^{(11)}=\{1\}$ $I_b^{(11)}=\{2, 4\}$

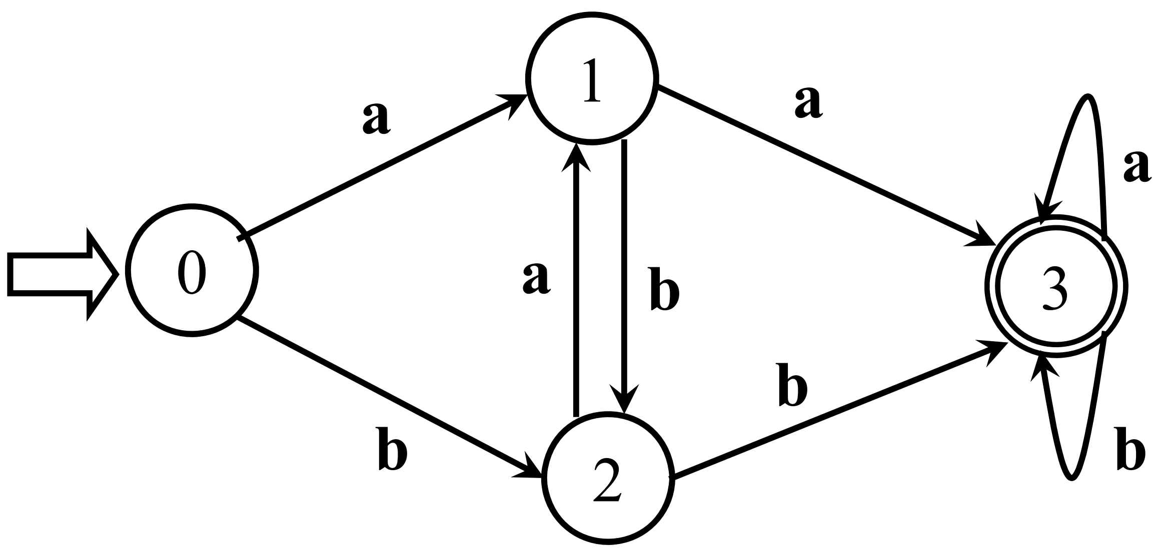
$I^{(111)}=\{0\}$ $I^{(112)}=\{2\}$ $I^{(12)}=\{1\}$ $I^{(2)}=\{3, 4, 5, 6\}$

$I_a^{(2)}=\{3, 6\}$ $I_b^{(2)}=\{4, 5\}$

确定有限自动机的化简



{0} {1} {2} {3, 4, 5, 6}



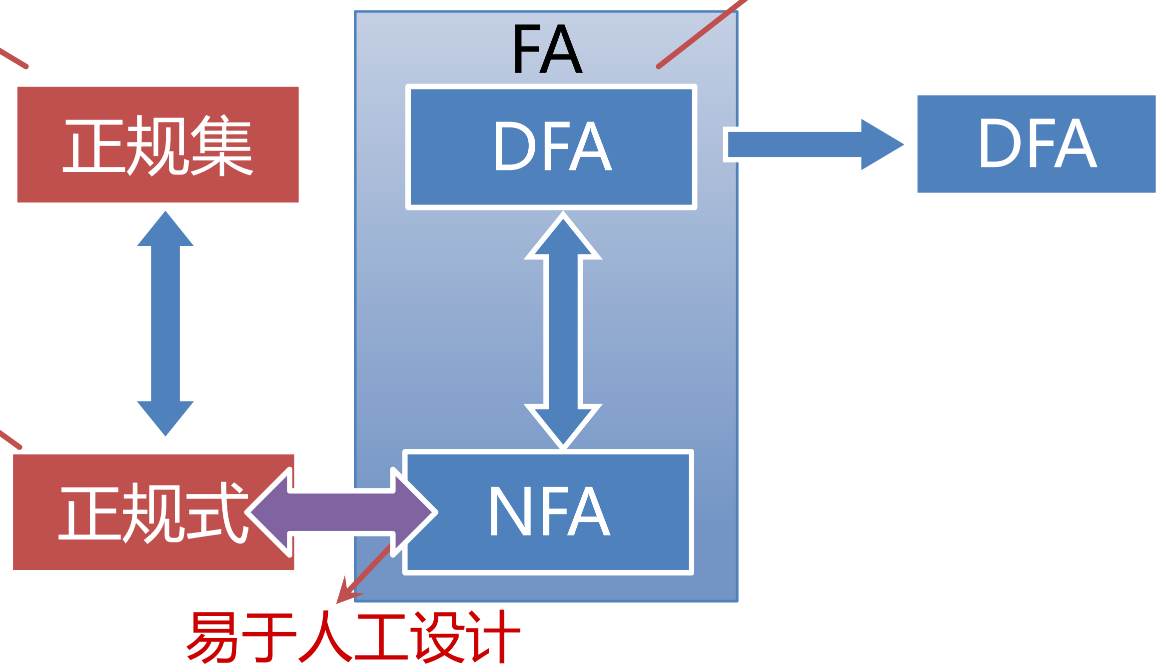
单词符号	种别编码	助忆符	内码值
DIM	1	$DIM	-
IF	2	$IF	-

正规式、正

```
curState = 初态
GetChar();
while( stateTrans[curState][ch]有定义){
    //存在后继状态，读入、拼接
    Concat();
    //转换入下一状态，读入下一字符
    curState= stateTrans[curState][ch];
    if curState是终态 then 返回strToken中的单词
    GetChar( );
}
```

DIM,IF, DO,STOP,END
number, name, age
125, 2169
…

DIM
IF
DO
STOP
END
letter(letter|digit)*
digit(digit)*



小结

- DFA与NFA的等价性
- NFA转换成DFA
  - 解决初始状态唯一性
  - 简化弧上的标记
  - 解决ε弧和转换关系(子集法)
- DFA的化简
  - 状态的等价和可区别
  - 最小化算法