CSCI 3434 Theory of Computation Homework 2

Luke Meszar (Worked with Jeff Lipnick)

September 29, 2017

HW 3.1 Give regular expressions for each of the following subsets of $\{a, b\}^*$.

(a) $\{x \mid x \text{ contains an even numbers of } a\text{'s}\}$

The pattern $((aa)^*b^* + ab^*a)^*$ matches strings with an even number of a's.

The first part of the Or, $(aa)^*b^*$ will match any string that has an even number of a's followed by any number of b's. Since $(aa)^*$ can match the empty string, it is possible to start a string with a b. The second part of the Or, ab^*a will match any string that has two a's separated by any number of b's. The pattern matches either of these two possibilities any number of times. Thus, this patter will match strings if and only if they have an even number of a's.

(b) $\{x \mid x \text{ contains an odd numbers of } b\text{'s}\}$

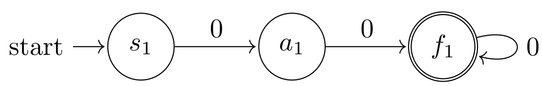
The pattern $(a^*ba^*)((bb)^*a^* + ba^*b)^*$ matches strings with an odd number of b's.

The first part of the concatenations, (a^*ba^*) guarantees there will always be at least one b which is necessary since every odd number has to form $2k + 1$ for $k \in \mathbf{Z}$. In the second part of the concatenation, we see the same pattern as in part (a) except the a's and b's are reversed. This gives the $2k$ part of the equation for an odd number. Thus, this pattern matches strings if and only if they have an odd number of b's.

HW 3.2 Give deterministic finite automata accepting the set of strings matching the following regular expressions.

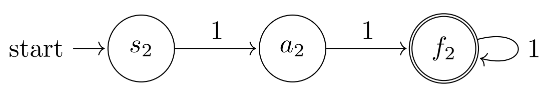
(a) $(000^* + 111^*)^*$.

We begin by constructing an NFA that accepts 000^*.

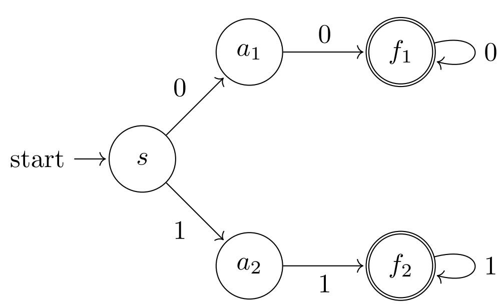


This NFA accepts strings that have at least 2 00's which is what the pattern 000^* matches.

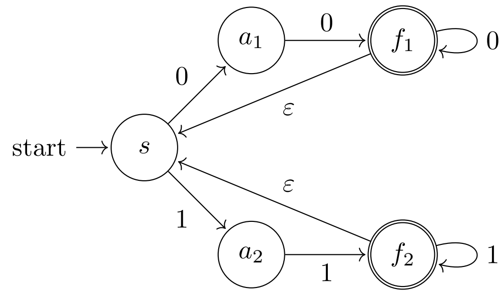
Likewise, we can construct an NFA that accepts 111^*.



Then we can combine them into an NFA that accepts $000^* + 111^*$ using ε-transitions.



Finally, we construct an NFA that accepts $(000^* + 111^*)^*$ using ε-transitions.



Next we use ε-closures $C_\varepsilon(A)$ as defined on page 318 of Kozen and the subset construction to construct an equivalent DFA. The ε-closures are as follows:

$$\begin{aligned} C_\varepsilon(s) &= \{s\} \\ C_\varepsilon(a_1) &= \{a_1\} \\ C_\varepsilon(a_2) &= \{a_2\} \\ C_\varepsilon(f_1) &= \{f_1, s\} \\ C_\varepsilon(f_2) &= \{f_2, s\} \end{aligned}$$

Then we use the subset construction to find an equivalent DFA[1]:

		0	1
→	$[s]$	$[a_1]$	$[a_2]$
	$[a_1]$	$[f_1]$	$\varnothing$
	$[a_2]$	$\varnothing$	$[f_2]$
	F$[f_1]$	$[f_1, s]$	$[a_2]$
	F$[f_2]$	$[a_1]$	$[f_2, s]$
	F$[f_1, s]$	$[f_1, s, a_1]$	$[a_2]$
	F$[f_2, s]$	$[a_1]$	$[f_2, s, a_2]$
	F$[f_1, s, a_1]$	$[f_1, s, a_1]$	$[a_2]$
	F$[f_2, s, a_2]$	$[a_1]$	$[f_2, s, a_2]$

To avoid the issue of mapping to the empty set in a DFA, we add a new state Z to replace $\varnothing$

		0	1
→	$[s]$	$[a_1]$	$[a_2]$
	$[a_1]$	$[f_1]$	d
	$[a_2]$	d	$[f_2]$
	F$[f_1]$	$[f_1, s]$	$[a_2]$
	F$[f_2]$	$[a_1]$	$[f_2, s]$
	F$[f_1, s]$	$[f_1, s, a_1]$	$[a_2]$
	F$[f_2, s]$	$[a_1]$	$[f_2, s, a_2]$
	F$[f_1, s, a_1]$	$[f_1, s, a_1]$	$[a_2]$
	F$[f_2, s, a_2]$	$[a_1]$	$[f_2, s, a_2]$
	d	d	d

Then, using the DFA minimization algorithm, we can make this DFA simpler:

[1] Method taken from https://www.youtube.com/watch?v=oEraHUCwFVU

$[s]$									
	$[a_1]$								
		$[a_2]$							
x	x	x	$[f_1]$						
x	x	x		$[f_2]$					
x	x	x			$[f_1, s]$				
x	x	x				$[f_2, s]$			
x	x	x					$[f_1, s, a_1]$		
x	x	x						$[f_2, s, a_2]$	
			x	x	x	x	x	x	d

Table 1: First iteration

$[s]$									
x	$[a_1]$								
x	x	$[a_2]$							
x	x	x	$[f_1]$						
x	x	x	x	$[f_2]$					
x	x	x		x	$[f_1, s]$				
x	x	x	x		x	$[f_2, s]$			
x	x	x		x		x	$[f_1, s, a_1]$		
x	x	x	x		x		x	$[f_2, s, a_2]$	
x	x	x	x	x	x	x	x	x	d

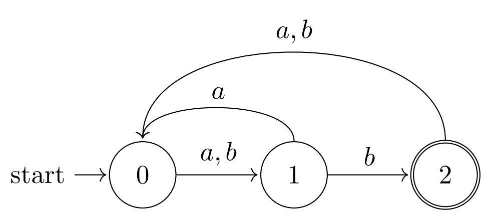
Table 2: Second iteration

From this, we see that $[f_1] \approx [f_1, s] \approx [f_1, s, a_1]$ and $[f_1 2] \approx [f_2, s] \approx [f_2, s, a_2]$. Thus, the final DFA's transition table is:

		0	1
$\rightarrow$	$[s]$	$[a_1]$	$[a_2]$
	$[a_1]$	$[F_1]$	d
	$[a_2]$	d	$[F_2]$
	F$[F_1]$	$[F_1]$	$[a_2]$
	F$[F_2]$	$[a_1]$	$[F_2]$
	d	d	d

Try to simplify as much as possible.

ME 15 Give a regular expression equivalent to the following automaton.



$\alpha_{02}^{\{0,1,2\}} = \alpha_{02}^{\{0,2\}} + \alpha_{01}^{\{0,2\}} \left(\alpha_{11}^{\{0,2\}}\right)^* \alpha_{12}^{\{0,2\}}$

We begin by expanding the first term

$\alpha_{02}^{\{0,2\}} = \alpha_{02}^{\{0\}} + \alpha_{02}^{\{0\}} \left(\alpha_{22}^{\{0\}}\right)^* \alpha_{22}^{\{0\}}$

$\alpha_{02}^{\{0\}} = \alpha_{02}^{\varnothing} + \alpha_{00}^{\varnothing} \left(\alpha_{00}^{\varnothing}\right)^* \alpha_{02}^{\varnothing}$

$\alpha_{02}^{\varnothing} = \varnothing$

$\alpha_{00}^{\varnothing} = \varepsilon$

$\alpha_{02}^{\{0\}} = \varnothing + \varepsilon \left(\varepsilon\right)^* \varnothing = \varnothing$ by 9.3 and 9.9

$\alpha_{02}^{\{0,2\}} = \varnothing + \varnothing \left(\alpha_{22}^{\{0\}}\right)^* \alpha_{22}^{\{0\}} = \varnothing + \varnothing = \varnothing$ by 9.3 and 9.9

Next we expand the second term

$\alpha_{01}^{\{0,2\}} = \alpha_{01}^{\{0\}} + \alpha_{02}^{\{0\}} \left(\alpha_{22}^{\{0\}}\right)^* \alpha_{21}^{\{0\}}$

$\alpha_{01}^{\{0\}} = \alpha_{01}^{\varnothing} + \alpha_{00}^{\varnothing} \left(\alpha_{00}^{\varnothing}\right)^* \alpha_{01}^{\varnothing}$

$\alpha_{01}^{\varnothing} = a + b$

$\alpha_{01}^{\{0\}} = (a + b) + \varnothing \left(\varnothing\right)^* (a + b) = a + b$

$\alpha_{01}^{\{0,2\}} = (a + b) + \varnothing \left(\alpha_{22}^{\{0\}}\right)^* \alpha_{21}^{\{0\}} = a + b$

Then we expand the third term

$\alpha_{11}^{\{0,2\}} = \alpha_{11}^{\{2\}} + \alpha_{10}^{\{2\}} \left(\alpha_{00}^{\{2\}}\right)^* \alpha_{01}^{\{2\}}$

$\alpha_{11}^{\{2\}} = \alpha_{11}^{\varnothing} + \alpha_{12}^{\varnothing} \left(\alpha_{22}^{\varnothing}\right)^* \alpha_{21}^{\varnothing}$

$\alpha_{11}^{\varnothing} = \varepsilon$

$\alpha_{12}^{\varnothing} = b$

$\alpha_{22}^{\varnothing} = \varepsilon$

$\alpha_{21}^{\varnothing} = \varnothing$

$\alpha_{11}^{\{2\}} = \varepsilon + b \left(\varepsilon\right)^* \varnothing = \varepsilon$

$\alpha_{10}^{\{2\}} = \alpha_{10}^{\varnothing} + \alpha_{12}^{\varnothing} \left(\alpha_{22}^{\varnothing}\right)^* \alpha_{20}^{\varnothing}$

$\alpha_{10}^{\varnothing} = a$

$\alpha_{20}^{\varnothing} = a + b$

$\alpha_{10}^{\{2\}} = a + b \left(\varepsilon\right)^* a + b = a + b(a + b)$

$\alpha_{00}^{\{2\}} = \alpha_{00}^{\varnothing} + \alpha_{02}^{\varnothing} \left(\alpha_{22}^{\varnothing}\right)^* \alpha_{20}^{\varnothing}$

$\alpha_{00}^{\{2\}} = \varepsilon + \varnothing \left(\varepsilon\right)^* (a + b) = \varepsilon$

$\alpha_{01}^{\{2\}} = \alpha_{01}^{\varnothing} + \alpha_{02}^{\varnothing} \left(\alpha_{22}^{\varnothing}\right)^* \alpha_{21}^{\varnothing}$

$\alpha_{01}^{\{2\}} = a + b + \varnothing \left(\varepsilon\right)^* \varnothing = a + b$

$\alpha_{11}^{\{0,2\}} = \varepsilon + (a + b(a + b))(\varepsilon)^*(a + b) = \varepsilon + (a + b(a + b))(a + b)$

Finally we expand the fourth term

$\alpha_{12}^{\{0,2\}} = \alpha_{12}^{\{2\}} + \alpha_{10}^{\{2\}} \left(\alpha_{00}^{\{2\}}\right)^* \alpha_{02}^{\{2\}}$

$\alpha_{12}^{\{2\}} = \alpha_{12}^{\varnothing} + \alpha_{12}^{\varnothing} \left(\alpha_{22}^{\varnothing}\right)^* \alpha_{22}^{\varnothing}$

$\alpha_{12}^{\{2\}} = b + b(\varepsilon)^* \varepsilon = b + b = b$

$\alpha_{02}^{\{2\}} = \alpha_{02}^{\varnothing} + \alpha_{02}^{\varnothing} \left(\alpha_{22}^{\varnothing}\right)^* \alpha_{22}^{\varnothing}$

$\alpha_{02}^{\{2\}} = \varnothing + \varnothing(\varepsilon)^* \varepsilon = \varnothing$

$\alpha_{12}^{\{0,2\}} = b + (a + b(a + b))(\varepsilon)^* \varnothing = b$

Combining all four terms together, we get

$$\alpha_{02}^{\{0,1,2\}} = \varnothing + (a+b)(\varepsilon + (a+b(a+b))(a+b))^* b$$
$$= (a+b)(\varepsilon + (a+b(a+b))(a+b))^* b$$

1

HW 4.1 Show the following sets are not regular.

(b) $A = \{x \in \{a,b,c\}^* \mid x \text{ is a palindrome; i.e.,} x = \text{rev } x\}$. (use the pumping lemma thoroughly) Let $k \geq 0$. Choose $x = \varepsilon$, $y = a^k$ and $z = bcb(a)^k$. This is a palindrome since there are an equal number of a on the left and right sides of the string and the bcb in the middle is a palindrome. We can also see that $|y| \geq k$ since $|y| = k$. Now, for all uvw such that $y = uvw$ and $v \neq \varepsilon$ we will show there exists an $i \geq 0$ such that $uv^iw \notin A$. Any uvw can be written as $a^p a^q a^r$ where $p + q + r = k$ and $q > 0$. Then, choose $i \geq 2$. Then $uv^iw = a^p a^{iq} a^r$. Thus $|uv^iw| = p + iq + r > k$. This means that xuv^iwz is of the form $a^l bcba^k$ where $l > k$. This string is no longer a palindrome since there are more as on the left than on the right so the reverse string will not be the same. Thus, palindromes are not regular languages.

(d) The set PAREN of balanced parentheses ( ). For example, the string ((( ) ( )) ( )) is in PAREN, but the string) (() is not.

For $k > 0$, let $x = (^k, y =)^k$, and $z = \varepsilon$. This string is in PAREN since the number of opening parenthesis is the same as the number of closing parenthesis. We can also see that $|y| \geq k$ since $|y| = k$. Then, for all $uvw = y$ such that $|v| > 0$, we know that $v =)^m$ where $m \leq k$. Write uvw as $)^l)^m)^n$ where $l + m + n = k$ and $m > 0$. Then, choose $i \geq 2$. Thus, $uv^iw =)^l[)^m]^i)^n =)^l)^im)^n$. Thus $|uv^iw| = l + im + n > k$. This means that there are now more closing parenthesis in uv^iw than opening parenthesis in x so $xuv^iwz \notin$ PAREN.

ME 37 Which of the following sets are regular and which are not? Give justification.

(a) $A = \{a^n b^{2m} \mid n \geq 0 \text{ and } m \geq 0\}$

Assume that A is regular. Then rev A is also regular. Define $h : \{a,b\}^* \to \{a,b\}^*$ by $h(a) = b$ and $h(b) = a$. Then, consider $A' = h(\text{rev } A)$. This language should also be regular since h is a homomorphism. Finally, the language $A \cap A'$ should also be regular. There are two cases to consider with this intersection. If $n < 2m$, then

$$A \cap A' = \{a^n b^n \mid n \geq 0\}$$

which we know is not regular.

If, $n > 2m$, then

$$A \cap A' = \{a^{2m} b^{2m} \mid m \geq 0\}.$$

The set $\{a^{2m} b^{2m} \mid m \geq 0\}$ can be easily be seen to be non regular via the pumping lemma. For $k \geq 0$, let $x = a^2k, y = b^2k$, and $z = \varepsilon$. Then for any u, v, w such that $uvw = y$, choose $i \geq 2$. Then $|uv^iw| \geq 2k$ so $xuv^iwz \notin \{a^{2m} b^{2m} \mid m \geq 0\}$.

Since $A \cap A'$ is not regular, than our assumption that A is regular must be wrong. Therefore, A is not regular.

(b) $B = \{a^n b^m \mid n = 2m\}$

This is the same as the set $\{a^{2m} b^m \mid m \geq 0\}$. Then, let $x = \varepsilon, y = a^{2k}$, and $z = b^k$. This string is clearly in B and $|y| = 2k$. Then, choose $i \geq 2$. Thus $|uv^iw| > 2k$ so $xuv^iwz \notin B$ since it is not of the form $a^{2m} b^m$. Thus, B is not regular.

(c) $C = \{a^n b^m \mid n \neq m\}$

Assume that C is regular. Then $\sim C$ is also regular. Now, take the following intersection

$$\sim C \cap L(a^* b^*)$$

We know that $L(a^* b^*)$ is regular since it is the language of a valid regular expression. The result of this intersection is $\{a^n b^n \mid n \geq 0\}$. To see that this is the result of the intersection, we note that $L(a^* b^*)$ are any string of the form, some number a's followed by some number of b's. If the number of a's and b's were different, it would mean this string would not be in $\sim C$. Thus, the strings have to have the same number of a's and b's. Since $\{a^n b^n \mid n \geq 0\}$ is not regular, then C cannot be regular either.

(d) $D = \{a^{p-1} \mid p \text{ is prime}\}$

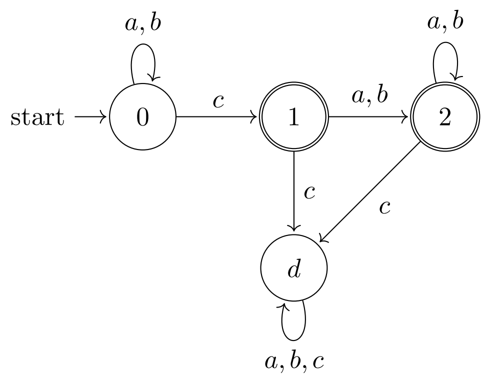
If D where regular, then $\{a\}D$ should be regular due to regular languages being closed under concatenation. However, $\{a\}D = \{aa^{p-1} \mid p \text{ is prime}\} = \{a^p \mid p \text{ is prime}\}$. But, we know that this set is not regular.

(e) $E = \{xcx \mid x \in \{a,b\}^*\}$

Given $k \geq 0$, consider $x = \varepsilon, y = a^k$ and $z = ca^k$. Then $xyz \in E$ since $a^k \in \{a,b\}^*$ and thus xyz has the form xcx and we can see that $|y| = k$. Let uvw be such that $y = uvw$ and $v \neq \varepsilon$. Let $i \geq 2$. Then $|uv^iw| > k$ (same logic as 4.1b). This means that $xuv^iwz = a^lca^k$ where $l > k$ so which means this string is not of the form xcx. Thus, E is not regular.

(f) $F = \{xcy \mid x, y \in \{a,b\}^*\}$

Consider the following DFA:



This DFA works by either accepting as soon as it has seen one has seen one c so strings of the form $xc\varepsilon$ are accepted or accepting if any more a's or b's are seen. However, if another c is seen, then it fails to accept. Thus, F is regular.

(g) $G = \{a^nb^{n+481} \mid n \geq 0\}$

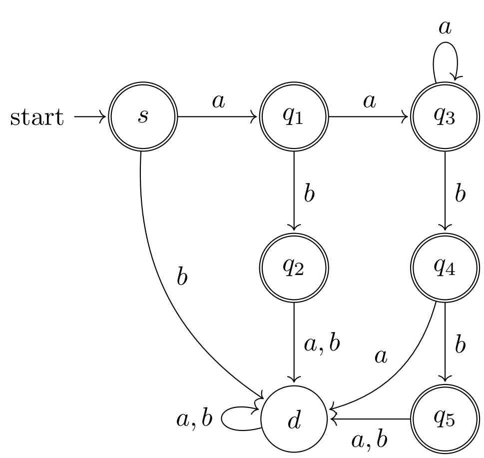
Given $k \geq 0$, then let $x = a^k, y = b^{k+481}$, and $z = \varepsilon$. We have $xyz \in G$ and $|y| > k$. For all $uvw = y$, let $i \geq 2$. Then $|uv^iw| > k + 481$, say $l + 481$. Then $xuv^iwz \notin G$ because it is not of the form a^nb^{n+481} since $l \neq k$. Thus G is not regular.

(h) $H = \{a^nb^m \mid n - m \leq 481\}$

Given $k \geq 0$, let $x = \varepsilon, y = a^k, z = b^k$. Then $xyz \in H$ since $k - k = 0 \leq 481$. Now for any u, v, w such that $y = uvw$ and $v \neq \varepsilon$, then choose $i \geq 482$. The smallest v can be is if it just the symbol a. In this case, $|uv^iw| = k + 482$. Thus, $n - m = k + 481 - k = 482 \nleq 481$. For any other choice of v, we will have $|uv^iw| > k + 482$ so it will still fail. Thus $xuv^iwz \notin H$ and so H is not regular.

(i) $I = \{a^nb^m \mid n \geq m \text{ and } m \leq 481\}$

This language is regular. Consider the language $I' = \{a^nb^m \mid n \geq m \text{ and } m \leq 2\}$. This is essentially the same language as I since it just differs by a constant. We will only have to change the DFA necessary to accept I by a finite number of states. The following DFA accepts I':



We can see that this DFA does accept I' by considering a few different cases. First, it accepts ε. Next, if there is only one a, then there can only be one b as denoted by the path from state q_1 to q_2. If there are 2 or more a's in the string, then we can accept up to the maximum number of b's which is two. This can be seen by the path q_1 to q_3 to q_4 to (potentially) q_5. We can also accept any number of a's followed by no b's. If we see too many b's or we see an a after a b we go to the state d. If we see a b before any a's, then we also go to d.

A machine of this form can be extended to actually accept I. Thus, I is regular.

(j) $J = \{a^n b^m \mid n \geq m \text{ and } m \geq 481\}$

Given $k \geq 0$, let $z = a^l, y = b^l$ and $z = \varepsilon$ such that $l \geq k$ and $l \geq 481$. Then, $xyz \in J$ since $n = l = m \geq 481$. Then for all u, v, w such that $y = uvw$ and $v \neq \varepsilon$, choose $i \geq 2$. Then $|uv^i w| > l$. This means that $xuv^i wz \notin J$ since it is no longer true that $n \geq m$.

(k) $K = L((a^*b)^*a^*)$

The pattern $(a^*b)^*a^*$ is a valid regular expression which means that it is a regular language.

(l) $L = \{a^n b^n c^n \mid n \geq 0\}$

Given $k \geq 0$, let $x = a^k, y = b^k$ and $z = c^k$. Then for all u, v, w such that $y = uvw$ and $v \neq \varepsilon$, choose $i \geq 2$. Then $|uv^i w| > k$ so $xuv^i wz \notin L$ since not all of the exponents are the same anymore. Thus, L is not regular.

(m) M {syntactically correct Python programs}

This language is not regular. An example of a syntatically correct Python program is

```
arr = ['a','a',...,'a'] //where there are p a's, p prime
```

This is essentially the language $A = \{a^p \mid p \text{ prime}\}$. In fact, it is the language

$$A = \{\text{arr} = [('a',)^p] \mid p \text{ prime}\}.$$

There are simple DFAs that can accept arr $=$ [ ] since it is just a constant string. However, accepting $\{('a',)^p \mid p \text{ prime}\}$ is impossible since this language is isomorphic to $\{a^p \mid p \text{ prime}\}$.

HW 4.3 For the following automata

		a	b
→	1	1	4
	2	3	1
	3F	4	2
	4F	3	5
	5	4	6
	6	6	3
	7	2	4
	8	3	1

(a) say which states are accessible and which are not;

The inaccessible states are 7 and 8. The DFA with only accessible states is as follows:

		a	b
→	1	1	4
	2	3	1
	3F	4	2
	4F	3	5
	5	4	6
	6	6	3.

(b) list the equivalence classes of the collapsing relation $\approx$ defined in Lecture 13:

$$p \approx q \stackrel{\text{def}}{\iff} \forall x \in \Sigma^*, (\hat{\delta}(p, x) \in F \iff \hat{\delta}(q, x) \in F);$$

We begin by marking pairs of states, $\{p, q\}$ if $p \in F$ and $q \notin F$ or vice versa.

Table 3

1					
x	2				
x	x	3			
x	x		4		
		x	x	5	
		x	x		6

After going through the states again and marking states if $\{\delta(p,a), \delta(q,a)\}$ is already marked, we get.

Table 4

1					
x	2				
x	x	3			
x	x		4		
x		x	x	5	
	x	x	x	x	6

Recursing further will not change anything. Thus, the equivalent states are

$$1 \approx 6$$
$$2 \approx 5$$
$$3 \approx 4.$$

Denote $1 \approx 6$ as p, $2 \approx 5$ as q and $3 \approx 4$ as r.

(c) give the automaton obtained by collapsing equivalent states and removing inaccessible states. The automaton with obtained by collapsing equivalent states and removing inaccessible states is:

		a	b
→	p	p	r
	q	r	p
	rF	r	q