Nick G. Toth
Oct. 19, 2020
CIS 420
Assignment II

Exercise 1 (Sipser 3e, Exercise 1.14)

$a)$ **Theorem₁:** If M is a DFA that recognizes language B, then swapping the accept and non-accept states in M yields a new DFA recognizing the complement of B. Conclude that the class of regular languages is closed under complement.

Proof:

Suppose $M := \left(Q, \Sigma, \delta, q_0, F\right)$ is a DFA that recognizes the regular language B, and let $N := \left(Q, \Sigma, \delta, q_0, Q \backslash F\right)$ be the result of swapping the accepting and non accepting states of M. We haven't mutated δ, so N will be also be deterministic.

Then each string $b \in B$ is accepted at a particular final state $f \in F$; if b could be accepted at some additional final state in $F \backslash \{f\}$, the clearly M would have to be nondeterministic. Consequently, b must end at a non-accepting state of N. And since N is also a DFA, we can conclude that b is rejected by N.

Now let $a \in B^c$. We know that a is rejected by M, so a ends on a non-accepting state of M. We also know that the non-accepting states of M are accepting states of N, so a ends at an accepting state of N, and we can conclude that N accepts a.

∎

Corollary₁: Regular languages are closed under complement.

Proof:

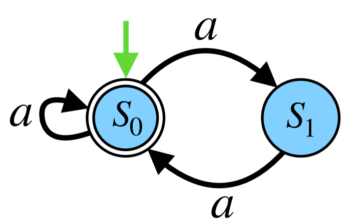
By Theorem₁, given a DFA M which recognizes the regular language B, we can invert the accepting and non-accepting states of M to obtain a new DFA N which recognizes the language B^c. And since the B^c is recognized by a DFA, B^c must also be regular. Therefore regular languages are closed under complement.

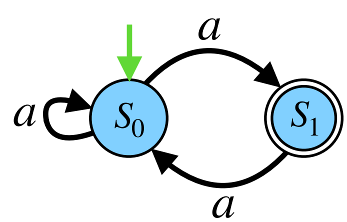
∎

$b)$ **Claim:** If M is an NFA that recognizes language C, then swapping the accept and non-accept states in M doesn't necessarily yield a new NFA that recognizes the complement of C.

Proof:

Observe the following NFAs.





Now observe that we can obtain either of these NFAs from the other, by swapping the accepting and non-accepting states. Additionally, each of these DFAs accept the string aa, so clearly they do not recognize complementary languages.

∎

Corollary₂: The class of languages recognized by NFA is closed under complement.

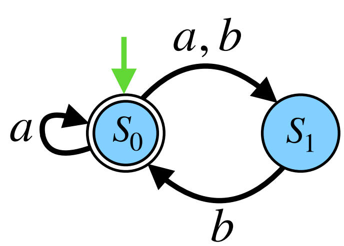
Proof:

The class of languages recognized by NFA is precisely the set of regular languages. Therefore, we can associate to each NFA, a DFA D which recognizes the same language. Now, by Theorem₁, we can obtain a new DFA D' such that $L(D') = L(D)^c$. Finally, since we have a DFA which recognizes the complementary language of the original NFA, and all DFAs are NFAs, we also have an NFA which recognizes the complementary language of the original NFA.

∎

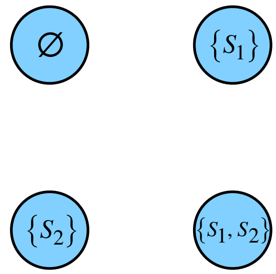
Exercise 2 (Sipser 3e, Exercise 1.16)

$a)$ We want to convert the following NFA into a DFA by the algorithm given in the proof of Theorem 1.39.



For convenience, let $M := \left(Q, \Sigma, \delta, q_0, F\right)$ be the formal description of the given NFA,

We begin by drawing four states, each corresponding to an element of $\mathscr{P}(Q)$.



Now we determine δ' for our new DFA.

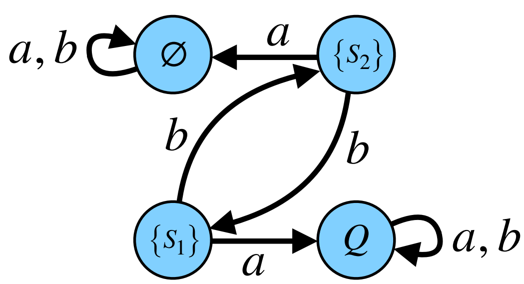
$$\delta'(\varnothing, a) = \varnothing$$
$$\delta'(\varnothing, b) = \varnothing$$

$$\delta'\left(\{S_2\}, a\right) = \varnothing$$
$$\delta'\left(\{S_2\}, b\right) = \{S_1\}$$

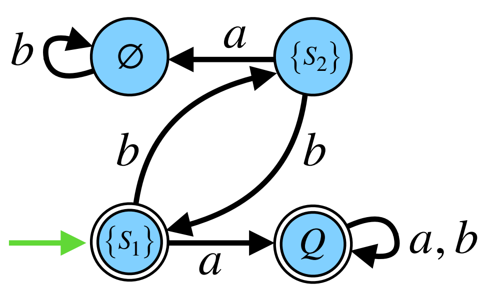
$$\delta'\left(\{S_1\}, a\right) = \{S_1, S_2\}$$
$$\delta'\left(\{S_1\}, b\right) = \{S_2\}$$

$$\delta'\left(Q, a\right) = \{S_1, S_2\}$$
$$\delta'\left(Q, b\right) = \{S_1, S_2\}$$

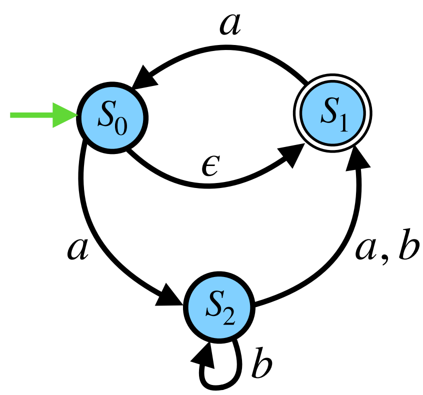
So we have



Finally, make $\{S_1\}$ the initial state of our new machine, and make $\{S_1\}$ and $\{S_1, S_2\}$ accepting states to obtain A DFA which recognizes the same language as M.

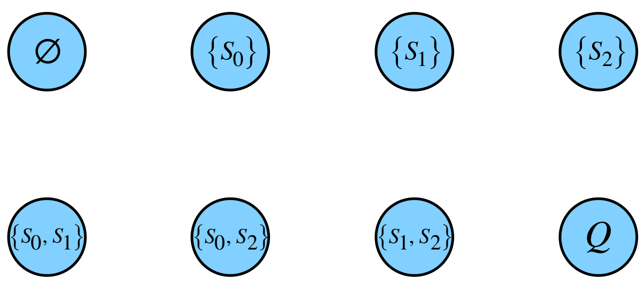


$b)$ We want to convert the following NFA into a DFA by the algorithm given in the proof of Theorem 1.39.



Once again, for convenience, let $M := (Q, \Sigma, \delta, q_0, F)$ be the formal description of the given NFA.

We begin by drawing four states, each corresponding to an element of $\mathscr{P}(Q)$.



Now we determine δ' for our new DFA.

$$\delta'(\varnothing, a) = \varnothing$$
$$\delta'(\varnothing, b) = \varnothing$$

$$\delta'(\{S_0\}, a) = \{S_2\}$$
$$\delta'(\{S_0\}, b) = \varnothing$$

$$\delta'(\{S_1\}, a) = \{S_0, S_1\}$$
$$\delta'(\{S_1\}, b) = \varnothing$$

$$\delta'(\{S_2\}, a) = \{S_1\}$$
$$\delta'(\{S_2\}, b) = \{S_1, S_2\}$$

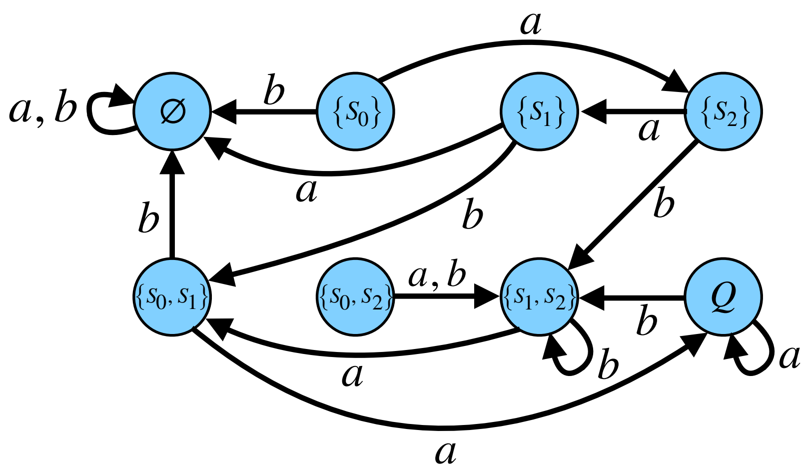
$$\delta'(\{S_0, S_1\}, a) = \{S_0, S_1, S_2\}$$
$$\delta'(\{S_0, S_1\}, b) = \varnothing$$

$$\delta'(\{S_0, S_2\}, a) = \{S_1, S_2\}$$
$$\delta'(\{S_0, S_2\}, b) = \{S_1, S_2\}$$

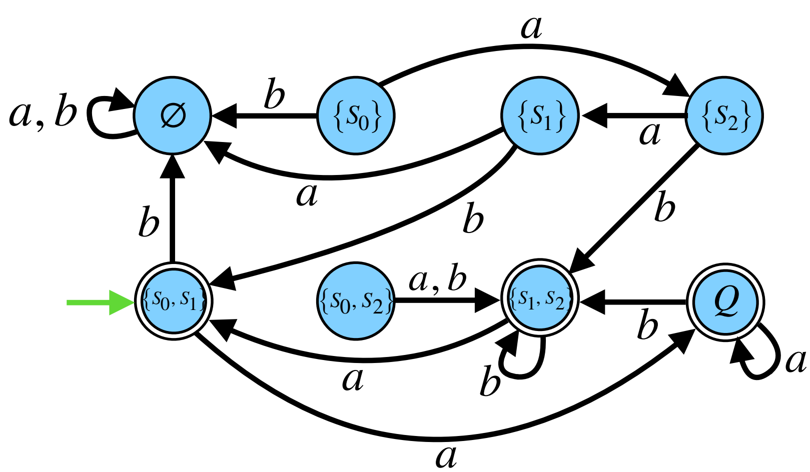
$$\delta'(\{S_1, S_2\}, a) = \{S_0, S_1\}$$
$$\delta'(\{S_1, S_2\}, b) = \{S_1, S_2\}$$

$$\delta'(Q, a) = Q$$
$$\delta'(Q, b) = \{S_1, S_2\}$$

So we have



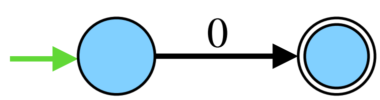
Finally, make $\{S_0, S_1\}$ the initial state of our new machine, and set each state which corresponds to a set containing S_1 into an accepting state; $\{S_0, S_1\}$, $\{S_1, S_2\}$, and Q. Our result is a DFA which recognizes the same language as M.



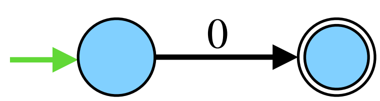
Exercise 3 (Sipser 3e, Exercise 1.19 a)

We want to convert the regular expression $(0 \cup 1)^* 000(0 \cup 1)^*$ into a nondeterministic finite automaton using the procedure described in lemma 1.55.

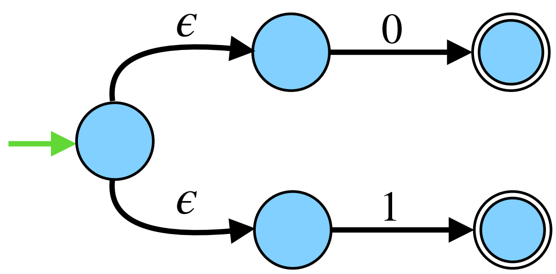
0



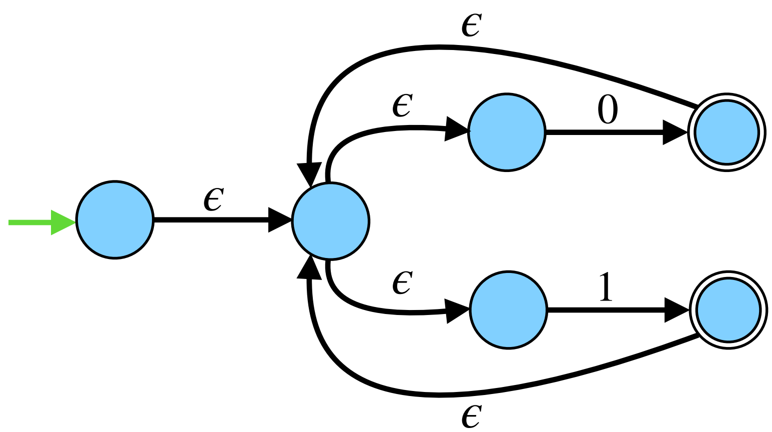
1



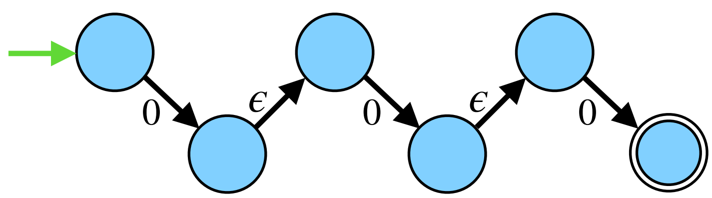
$0 \cup 1$



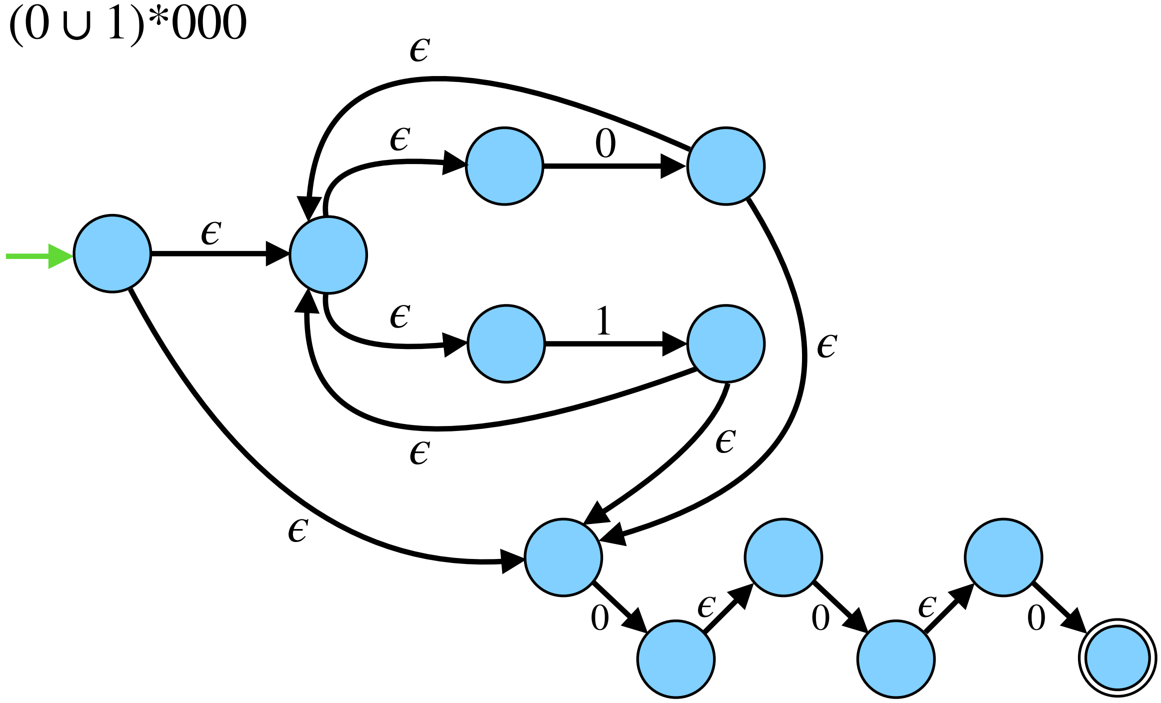
$(0 \cup 1)^*$



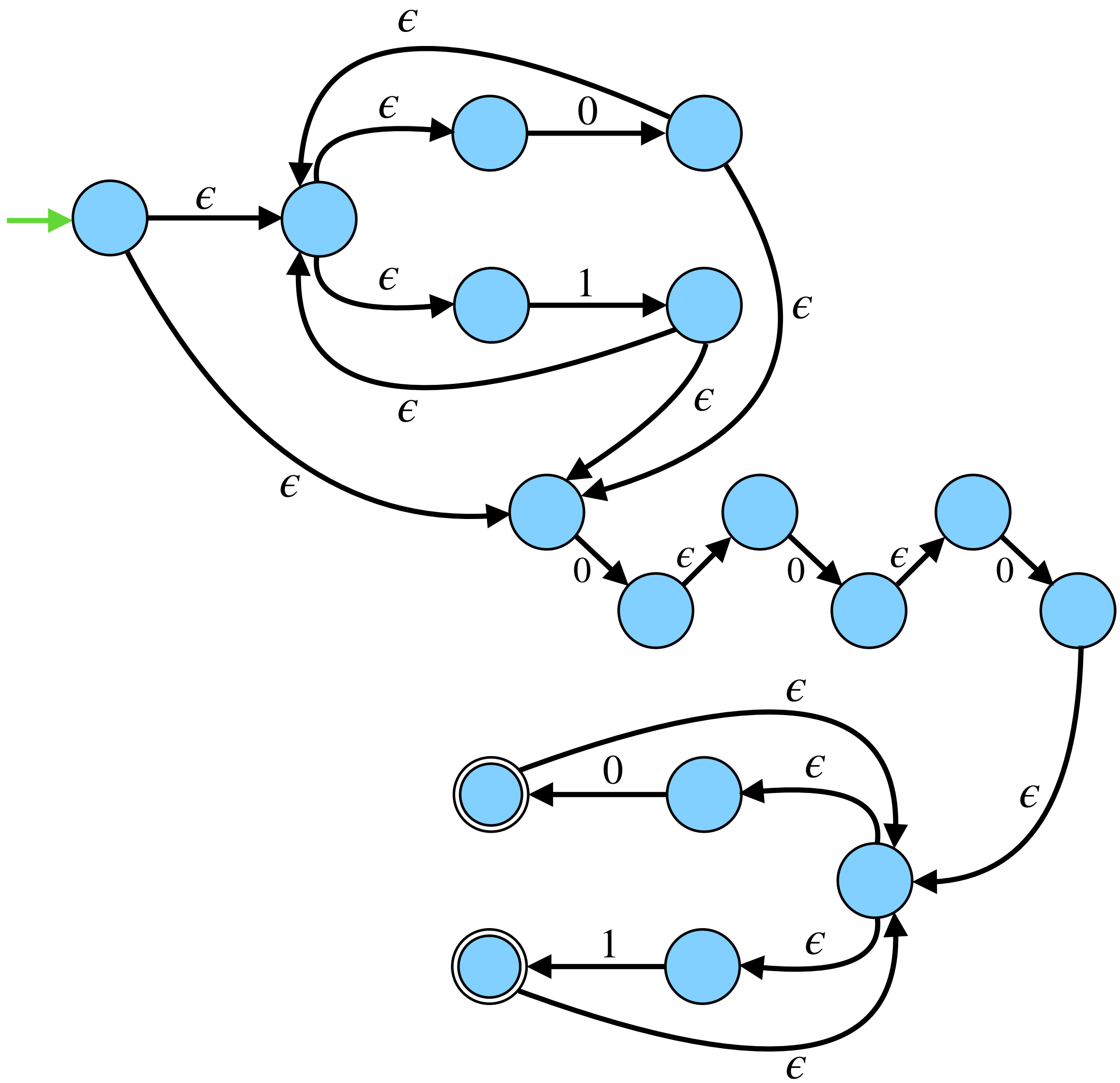
000



$(0 \cup 1)^*000$



$(0 \cup 1)^*000(0 \cup 1)^*$



Exercise 4

Here we view a string w over the alphabet {0,1,2} as representing an integer in base three. Give a DFA which accepts w iff (w mod 5) = 0.

Our DFA will have five states $S_0, \ldots, S_4$, corresponding to equivalence classes of whole numbers mod 5. In general, strings end on S_k if and only if they represent ternary numbers m with the property that $m \equiv k \pmod 5$. We will set $q_0 = S_0$, since strings end on this state if and only if they represent ternary multiples of 5.

Let α be a string over Σ.

Suppose α is passed to S_0. If α starts with 0, then S_0 should clearly pass the tail of α to itself. Similarly, if α starts with 1, then S_0 passes the tail of α to S_1, and if α starts with 2, then S_0 passes the tail of α to S_1; making the assumption that α will be divisible by the corresponding number.

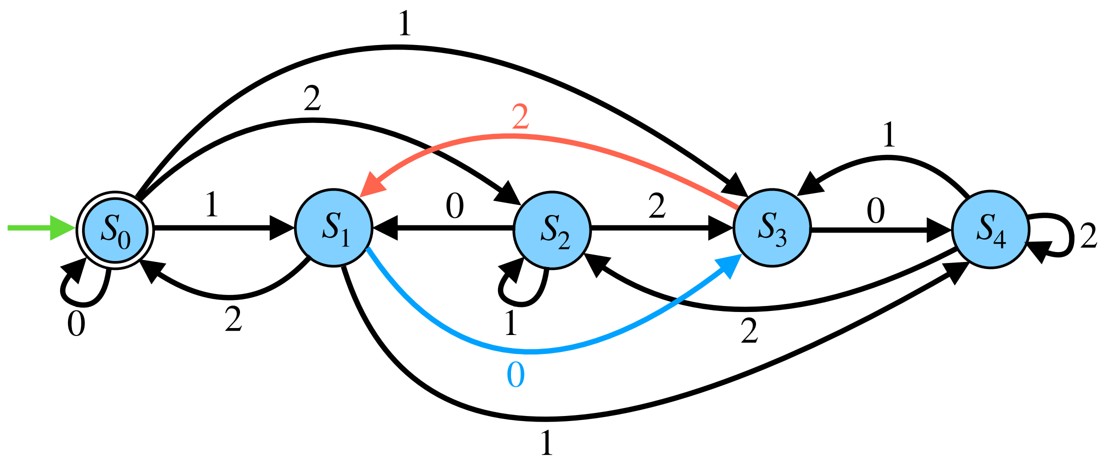
Now suppose α is passed to S_1, and assume α was passed from S_0. If α starts with 0, then S_1 should pass the tail of α to S_3, since $10_3 = 3_{10}$. If α starts with 1, then S_1 should pass the tail of α to S_4, since $11_3 = 4_{10}$. And if α starts with 2, then S_1 should pass the tail of α to S_0, since $12_3 = 5_{10} \equiv 0 \pmod 5$.

We continue in the same way for states S_2, S_3, and S_4, and record the information into a table.

	0	1	2
S_0	0	1	2
S_1	3	4	0
S_2	1	2	3
S_3	4	0	1
S_4	2	3	4

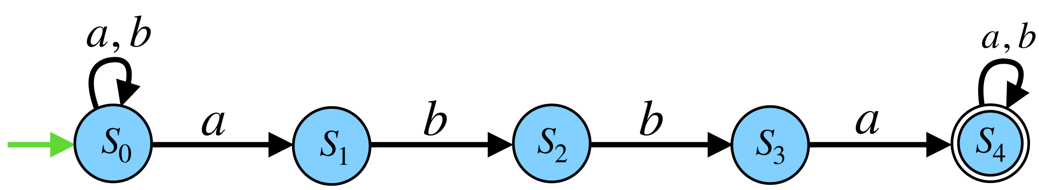
Table 1

Equivalently, we can represent our resultant DFA by the following graph. Note that some edges are colored to disambiguate crossed edges. Also, sorry for the crossed edges.



Exercise 5

Convert the NFA below to a DFA. It accepts the language over {a, b} of all strings that contain abba.



$\delta'(\varnothing, a) = \varnothing$
$\delta'(\varnothing, b) = \varnothing$

$\delta'(\{S_0\}, a) = \{S_0, S_1\}$
$\delta'(\{S_0\}, b) = \{S_0\}$

$\delta'(\{S_1\}, a) = \varnothing$
$\delta'(\{S_1\}, b) = \{S_2\}$

$\delta'(\{S_2\}, a) = \varnothing$
$\delta'(\{S_2\}, b) = \{S_3\}$

$\delta'(\{S_3\}, a) = \{S_4\}$
$\delta'(\{S_3\}, b) = \varnothing$

$\delta'(\{S_4\}, a) = \{S_4\}$
$\delta'(\{S_4\}, b) = \{S_4\}$

$\delta'(\{S_0, S_1\}, a) = \{S_0, S_1\}$
$\delta'(\{S_0, S_1\}, b) = \{S_0, S_2\}$

$\delta'(\{S_0, S_2\}, a) = \{S_0, S_1\}$
$\delta'(\{S_0, S_2\}, b) = \{S_0, S_3\}$

$\delta'(\{S_0, S_3\}, a) = \{S_0, S_1, S_4\}$
$\delta'(\{S_0, S_3\}, b) = \{S_0\}$

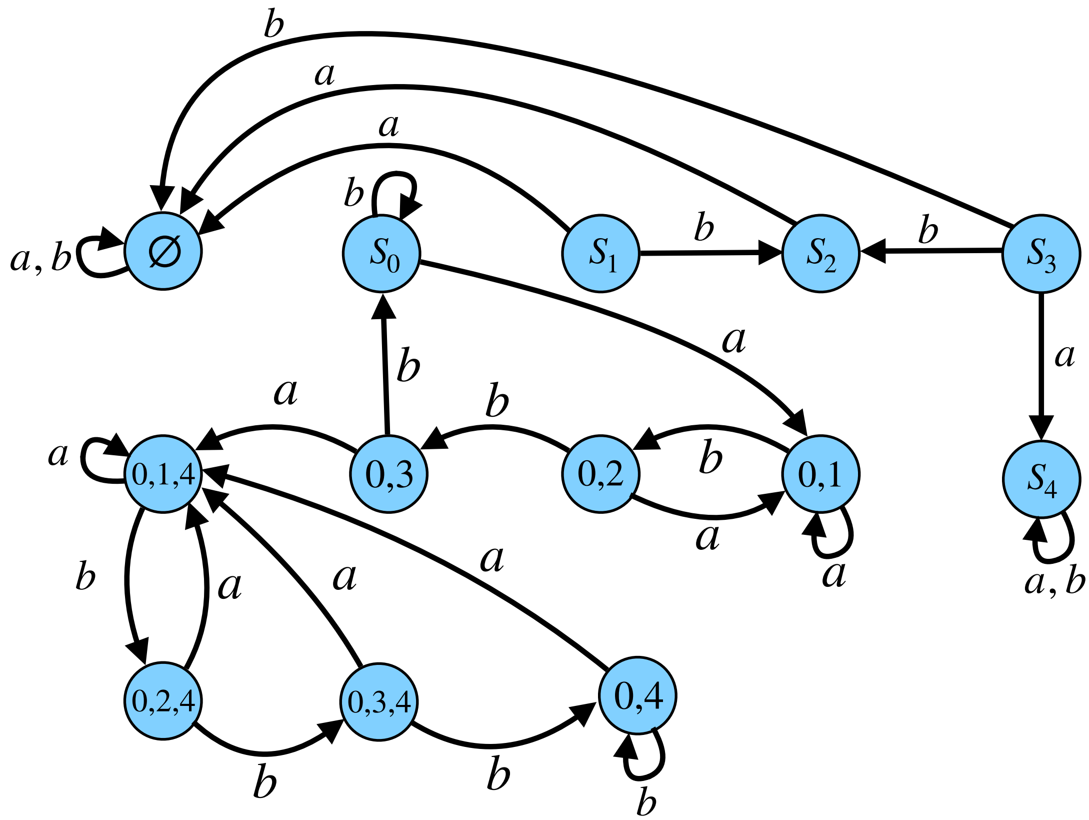
$\delta'(\{S_0, S_1, S_4\}, a) = \{S_0, S_1, S_4\}$
$\delta'(\{S_0, S_1, S_4\}, b) = \{S_0, S_2, S_4\}$

$\delta'(\{S_0, S_2, S_4\}, a) = \{S_0, S_1, S_4\}$
$\delta'(\{S_0, S_2, S_4\}, b) = \{S_0, S_3, S_4\}$

$\delta'(\{S_0, S_3, S_4\}, a) = \{S_0, S_1, S_4\}$
$\delta'(\{S_0, S_3, S_4\}, b) = \{S_0, S_4\}$

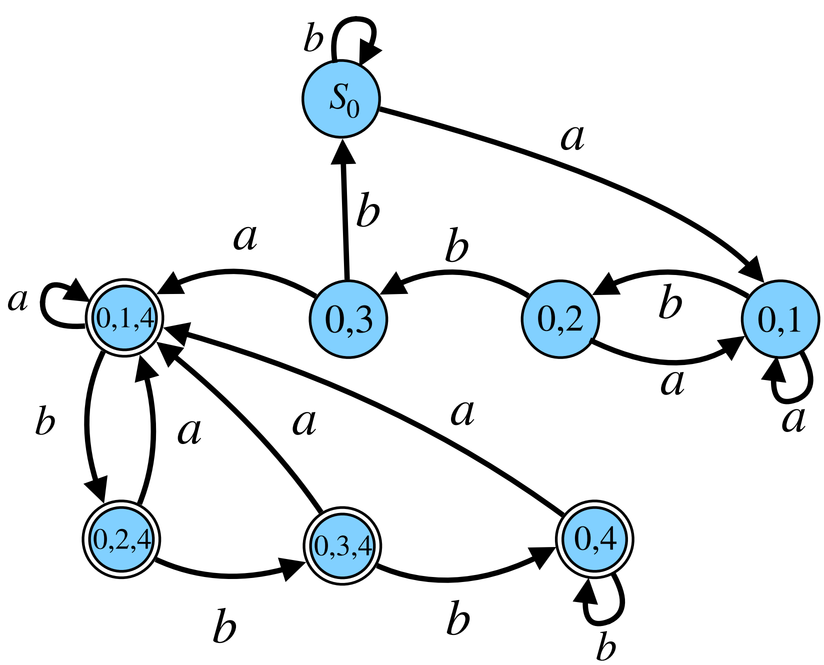
$\delta'(\{S_0, S_4\}, a) = \{S_0, S_1, S_4\}$
$\delta'(\{S_0, S_4\}, b) = \{S_0, S_4\}$

So we have



,

There's clearly no way to reach states $\varnothing$, S_1, S_2, S_3, or S_4, so we can get rid of them. Finally, set $q_0 = S_0$, and let each state with a four into an accepting state.



Lastly, we can remove states $(0,2,4)$, $(0,3,4)$, and $(0,4)$, add a loop edge at state $(0,1,4)$ for b, and clean up the presentation to obtain our final DFA.