

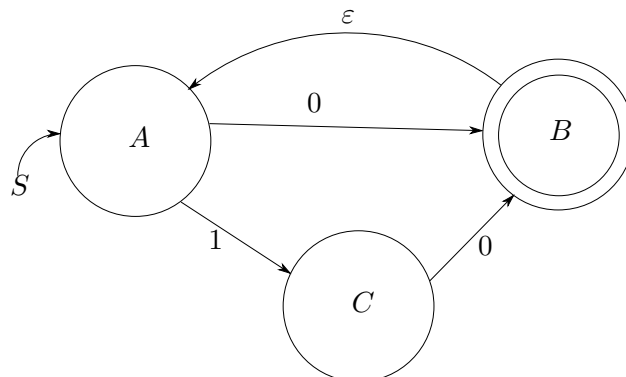
CSC236 Week 08: Machines, Expressions: Equivalence

Hisbaan Noorani

October 28 – November 3, 2021

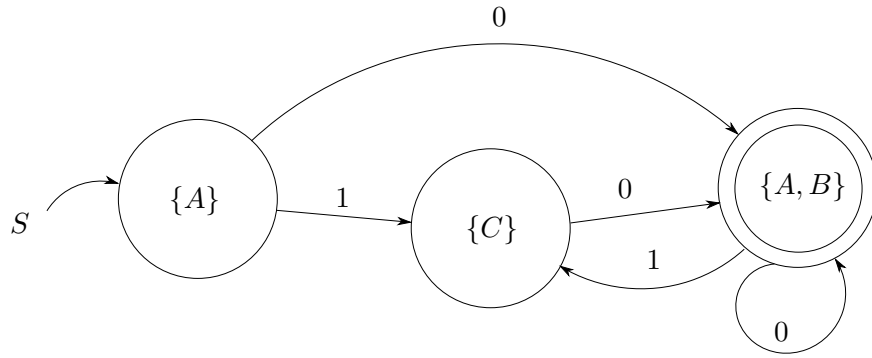
Contents

1	NFSA that accepts $L((0 + 10)(0 + 10)^*)$	1
2	NFSA that accepts $\text{Rev}(L((0 + 10)(0 + 10)^*))$	2
3	FSAs and regexes are equivalent.	2
3.1	Step 1.0: convert $L(R)$ to $L(M')$	3
3.2	Step 1.5: Convert $L(R)$ to $L(M')$	3
4	State elimination recipe for state q	5
5	FSAs and regexes are equivalent:	6
5.1	Step 3: convert $L(M)$ to $L(R)$ and eliminate states.	6
6	Regular languages closure	7
1	NFSA that accepts $L((0 + 10)(0 + 10)^*)$	

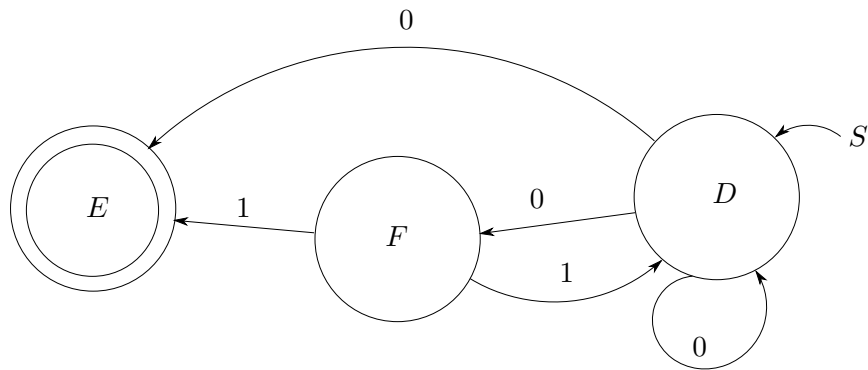


The ε transition makes it non deterministic. $A \xrightarrow{0} A \cup B$ and $C \xrightarrow{0} A \cup B$.

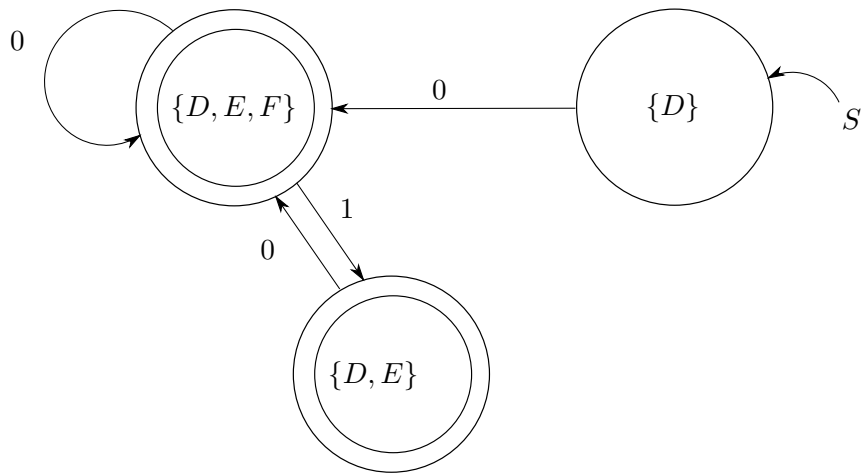
The corresponding DFSA is as follows:



2 NFSA that accepts $\text{Rev}(L((0 + 10)(0 + 10)^*))$



The corresponding DFSA is as follows:



3 FSAs and regexes are equivalent.

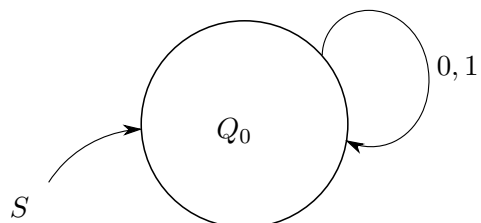
$L = L(M)$ for some DFSA $M \iff L = L(M')$ for some NFSA $M' \iff L = L(R)$ for some regular expression R .

3.1 Step 1.0: convert $L(R)$ to $L(M')$.

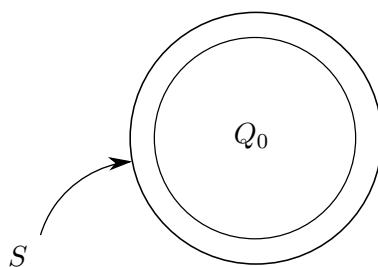
Start with $\emptyset, \varepsilon, a \in \Sigma$.

- Base case: Let s in $\{\emptyset, \varepsilon, a\}$ for some $a \in \Sigma$.

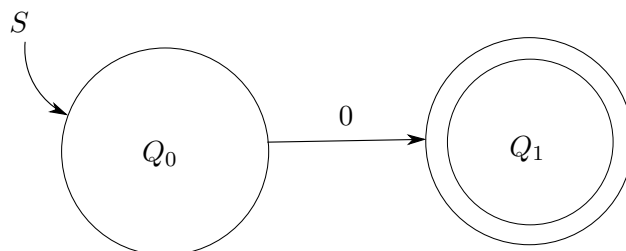
$L(\emptyset) = L(M)$, where M is:



$L(\varepsilon) = L(M)$, where M is:

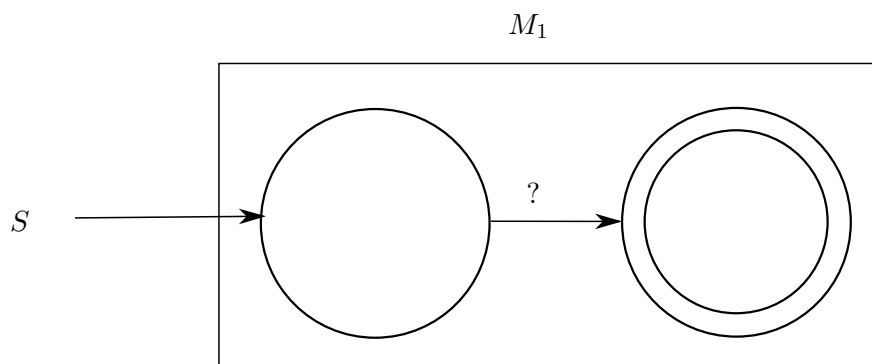


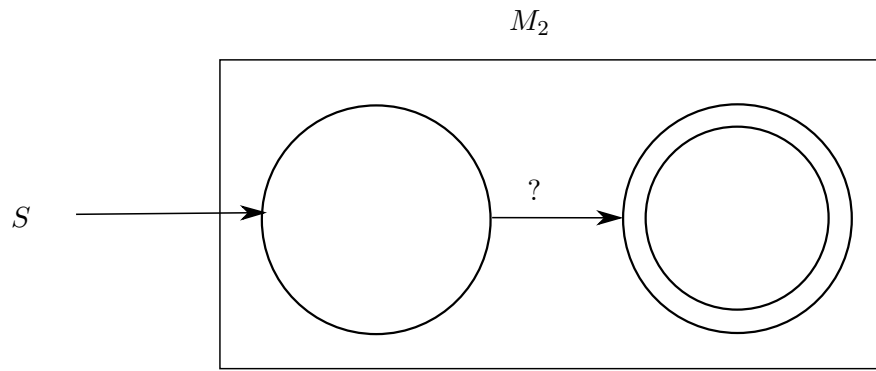
$L(0) = L(M)$, where M is:



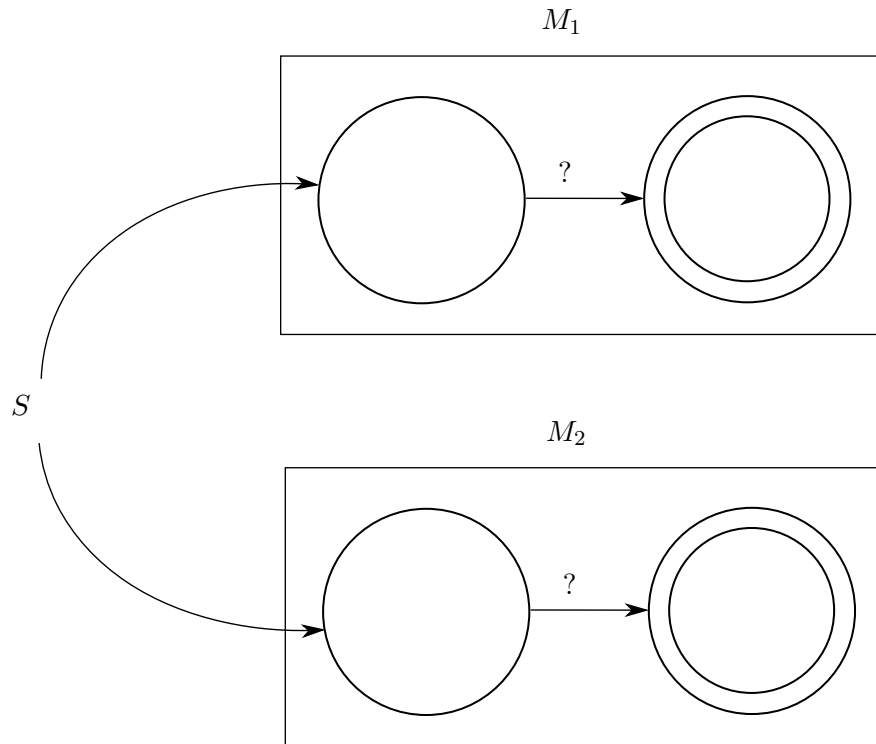
3.2 Step 1.5: Convert $L(R)$ to $L(M')$.

Suppose r_1 and r_2 denote languages accepted by M_1 and M_2 respectively.

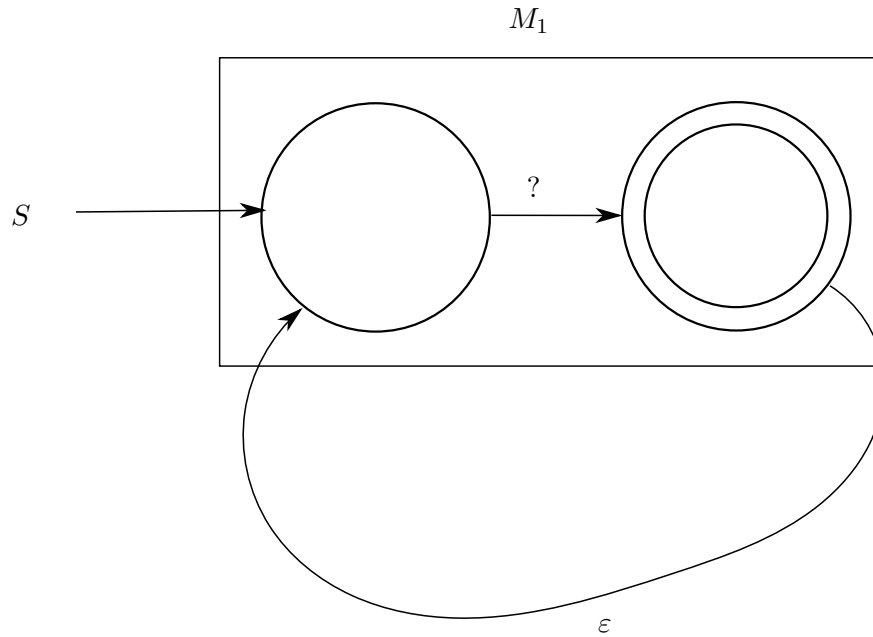




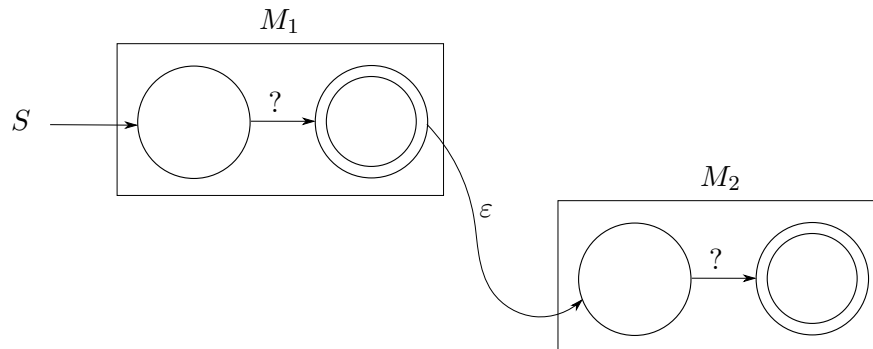
$L(r_1 + r_2) = L(r_1) \cup L(r_2)$. We could build the corresponding machine using product construction on M_1 and M_2 or we could use a NFSA.



$L(r_1^*) = L(r_1)^*$. We want to transform M_1 into a machine that accepts the Kleene star of the language accepted by M_1 .



$L(r_1r_2) = L(r_1)L(r_2)$. How do we combine M_1 and M_2 to accept this concatenated language? We concatenate the two machines!



Not that all three techniques use non-determinism but we can use subset construction to create an equivalent deterministic machine so these answers are no less valid.

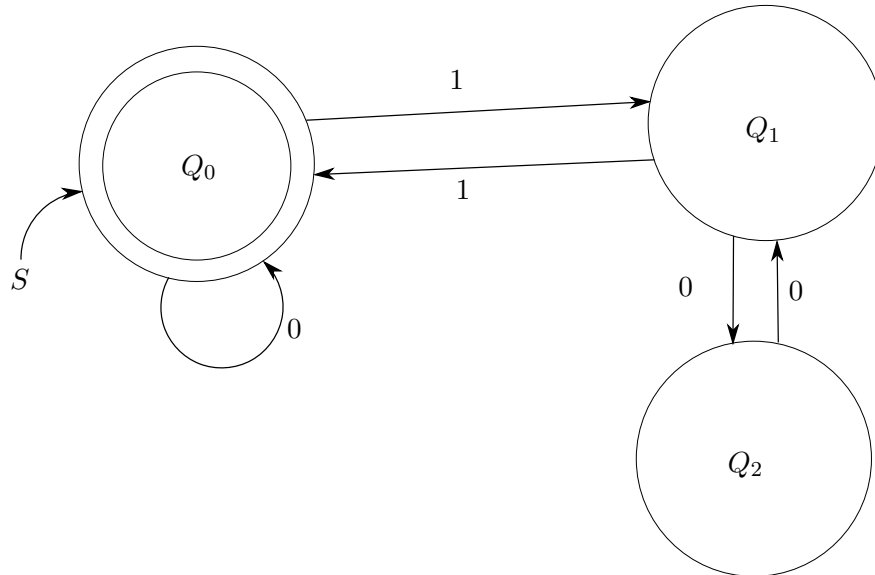
4 State elimination recipe for state q

1. $s_1 \dots s_n$ are states with transition to q , with labels $S_1 \dots S_n$.
2. $t_1 \dots t_n$ are states with transition from q , with labels $T_1 \dots T_n$.
3. Q is any self-loop state on q .
4. Eliminate q , and add (union) transition label $S_i Q^* T_j$ from s_i to t_j .

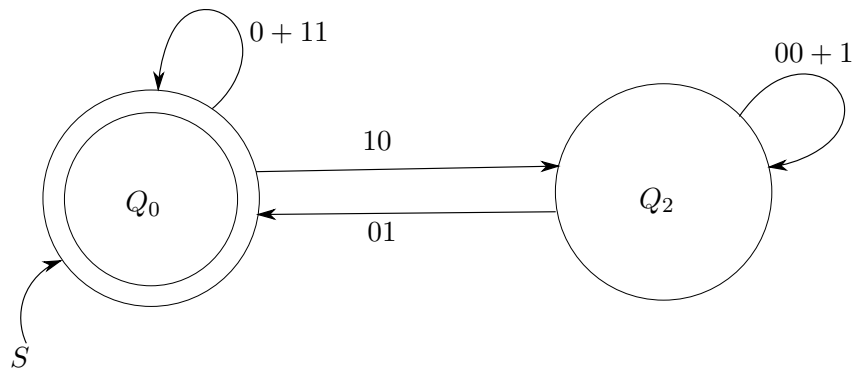
5 FSAs and regexes are equivalent:

$L = L(M)$ for some DFSA $M \iff L = L(M')$ for some NFSA $M' \iff L = L(R)$ for some regular expression R .

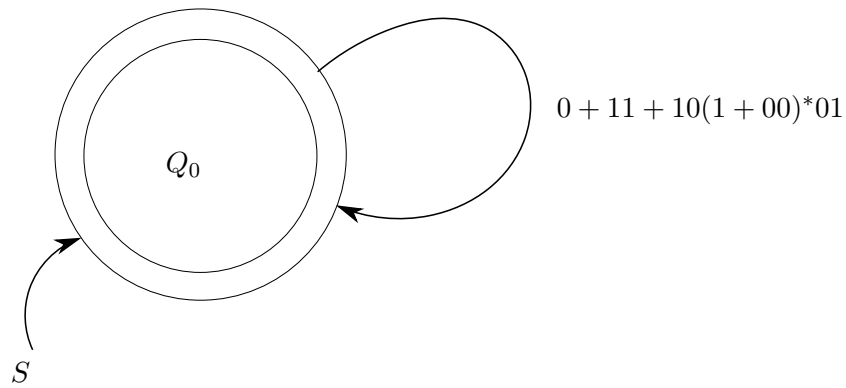
5.1 Step 3: convert $L(M)$ to $L(R)$ and eliminate states.



Then we eliminate Q_1 :



Then we eliminate Q_2 !



So our regular expression is $(0 + 11 + 10(1 + 00)^*01)^*$.

6 Regular languages closure

Regular languages are those that can be