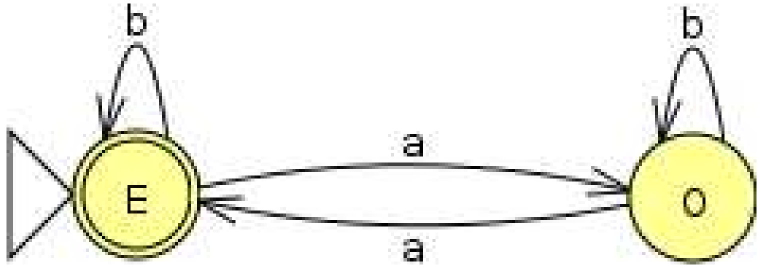
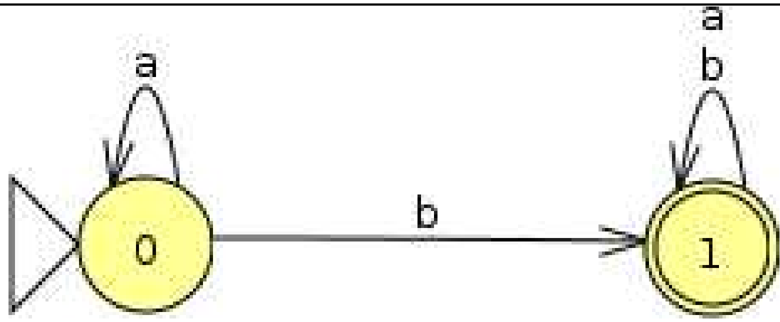
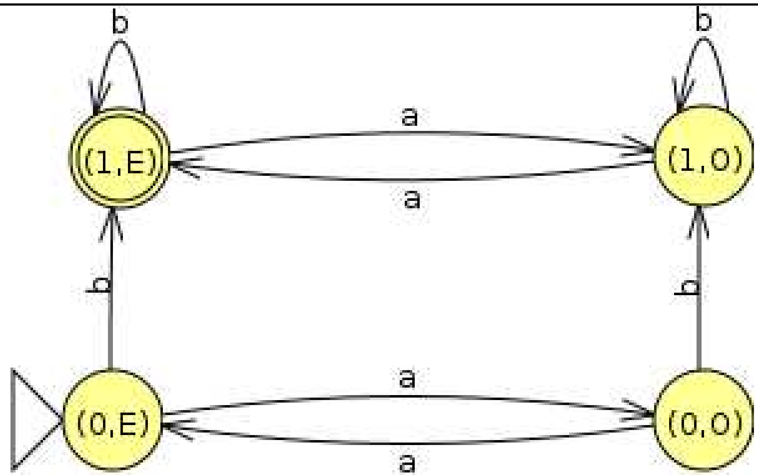
Regular Languages & Finite Automata

M	$L(M)$
	$L_1 = \{w \mid \#_a(w)$ is even$\}$
	$L_2 = \{w \mid \#_b(w) \geq 1\}$
	$L_1 \cap L_2 =$ $\{s \mid \#_a(w)$ is even and $\#_b(w) \geq 1\}$

Given two DFA

$$\begin{aligned} M_1 &= \langle Q_1, \Sigma, \delta_1, q_{01}, F_1 \rangle \\ M_2 &= \langle Q_2, \Sigma, \delta_2, q_{02}, F_2 \rangle \end{aligned}$$

Define

$$M_1 \times M_2 = \langle Q_1 \times Q_2, \Sigma, \delta, q_0, F \rangle$$

with

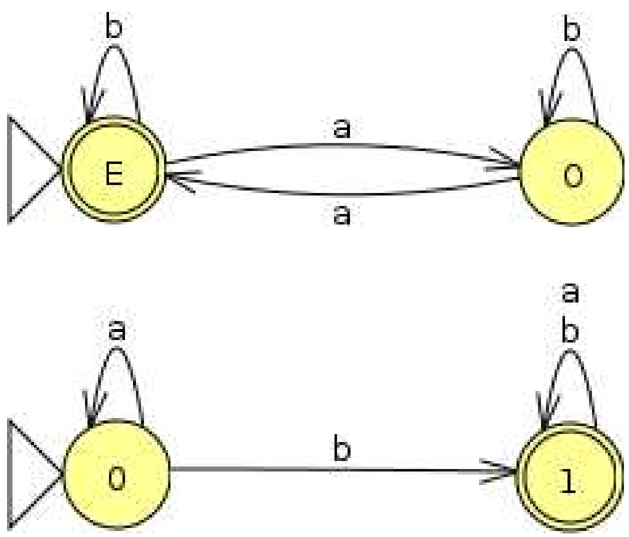
$$\begin{aligned} q_0 &\triangleq \langle q_{01}, q_{02} \rangle \\ F &\triangleq F_1 \times F_2 \triangleq \{\langle q_1, q_2 \rangle \mid q_1 \in F_1,\ q_2 \in F_2\} \\ \delta(\langle q_1, q_2 \rangle, a) &\triangleq \langle \delta_1(q_1, a), \delta_2(q_2, a) \rangle \end{aligned}$$

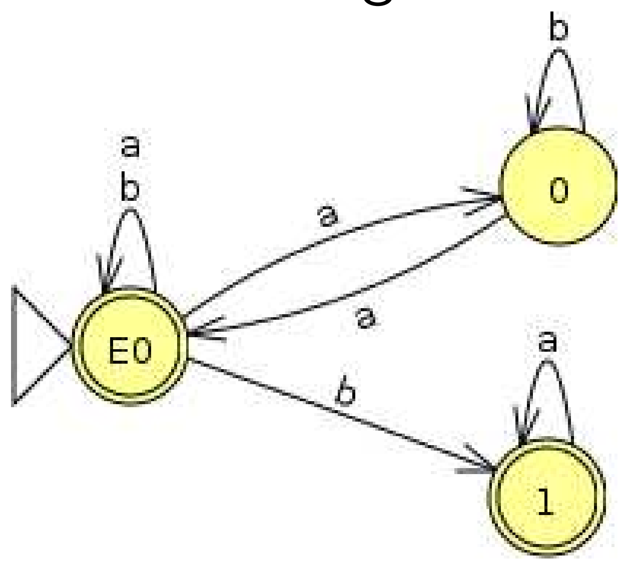
Then

$$L(M_1 \times M_2) = L(M_1) \cap L(M_2)$$

Suppose we want $\{w \mid \#_a \text{ even or } \#_b \geq 1\} = L_1 \cup L_2$

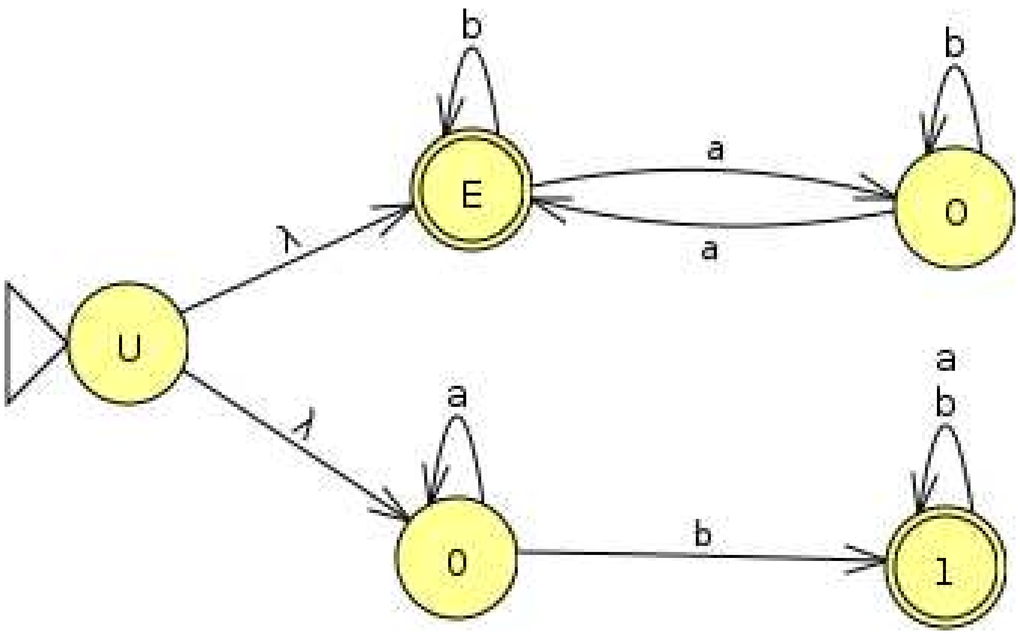
Last week we suggested to put two machines together





The DFA on the right accepts ‘aaa’ which is the wrong intention

Now we add 'silent steps' (Bas: 'diodes') to NFAs



In a NFA$_\lambda$ we allow

$$\delta(q, \lambda) = q'$$

for $q \neq q'$. That means

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \to Q$$

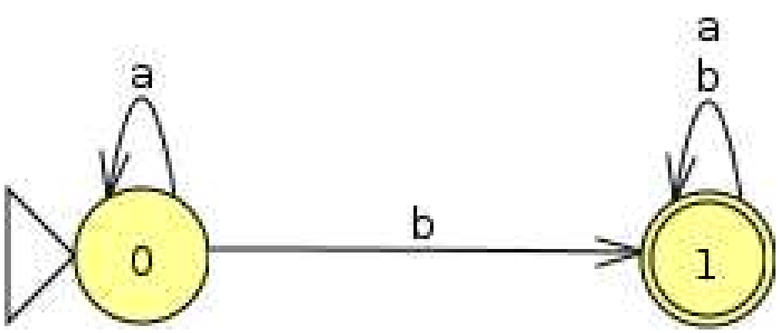
A finite automaton M is called *insulated*

(i) if q_0 in M has no in-going arrows
(ii) there is only one final state which has no out-going arrows

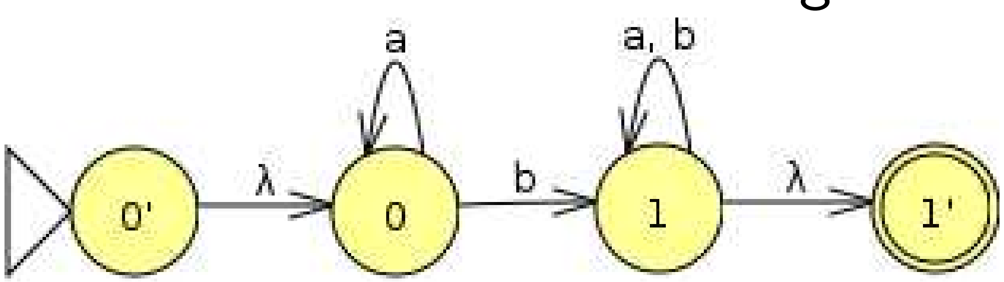
Proposition. One can insulate any machine M such that

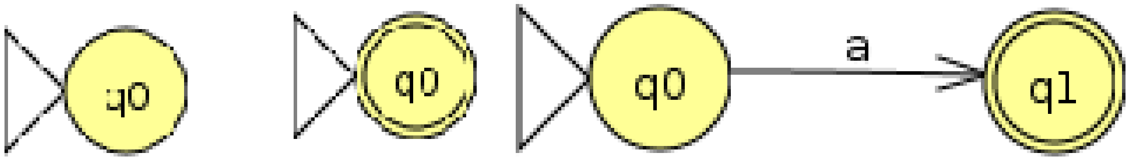
the result M' accepts the same language

Proof. Use diodes, for example



gives

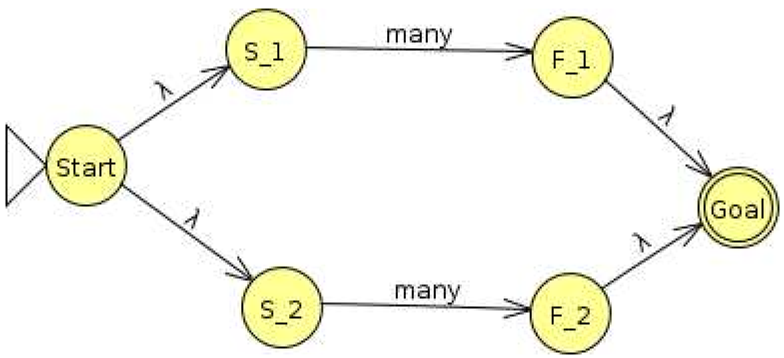




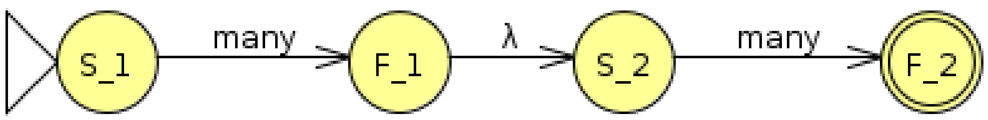
$\emptyset$ $\{\lambda\}$ $\{a\}$

$L_1 = L$

L_2



$L_1 \cup L_2$ insulated



$L_1 L_2$ insulated

L^* insulated

Proposition 1. For every regular expression e there is an NFA$_\lambda$ M_e such that

$$L(M_e) = L(e).$$

Proof. Apply the toolkit. M_e can be found 'by induction on the structure of e': first do this for the simplest regular expressions; then for a composed regular expression compose the automata. ■

Proposition 2. For every regular language L there is an NFA$_\lambda$ M such that

$$L(M) = L.$$

Proof. By definition of regular languages there is a regular expression e such that $L = L(e)$. Then $L(M_e) = L(e) = L$. ■

This proof one finds somtimes explained as on the following slide

Proposition. Every regular language is the language of a NFA$_\lambda$

Proof. Let $L = L(e)$ for some regular expression e.

By induction on e we show there is a machine M_e such that $L = L(M_e)$

Basis. Automata accepting $\emptyset$, $\{\lambda\}$ and $\{a\}$ are easy

Induction step. Now we show that if L_1, L_2 are accepted by NFA$_\lambda$-s then so are $L_1 \cup L_2,\ L_1 L_2, L_1^*$

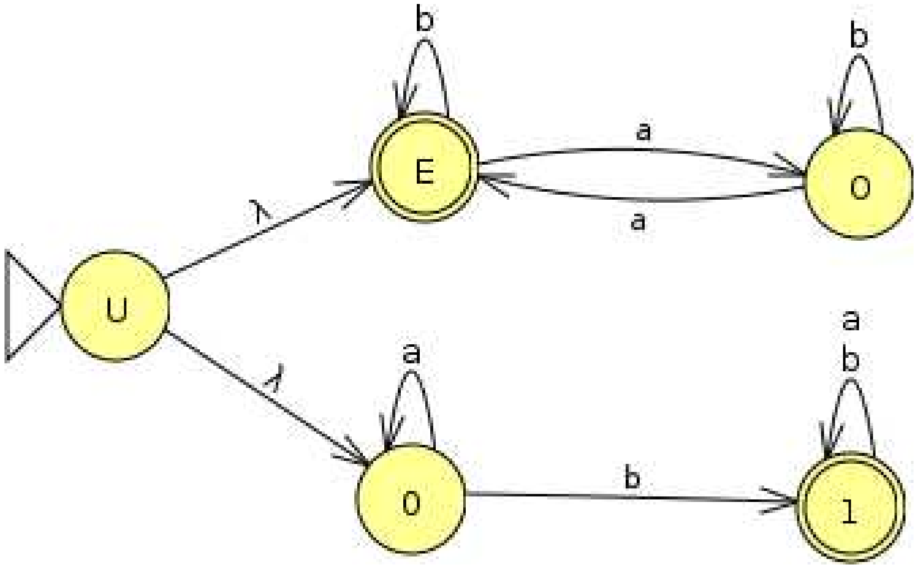
By the induction hypothesis there are NFA$_\lambda$-s such that $L_i = L(M_i)$

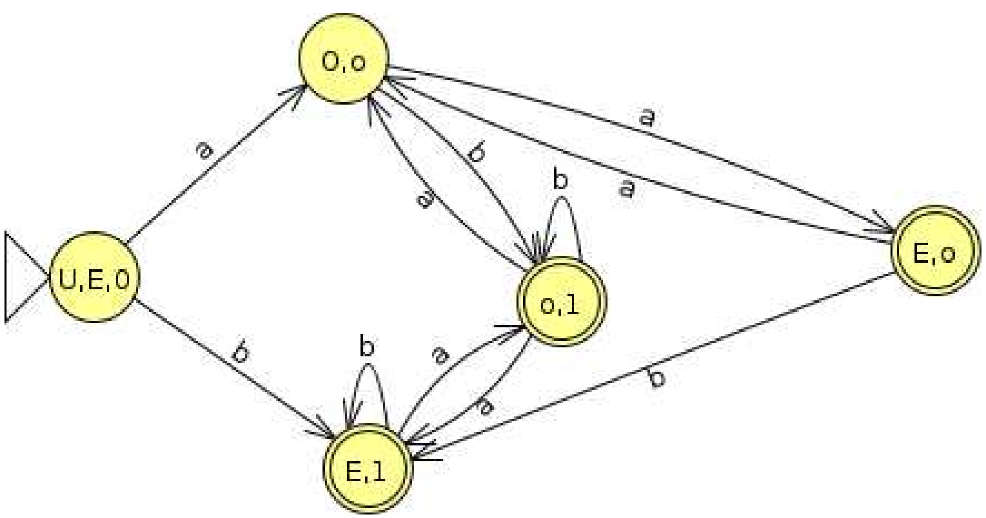
Take care that these machines become insulated, obtaining M_1', M_2'

•For concatenation of languages: use serial composition of automata

•For union of langugages: use parallel composition of automata

•For the *-operator use a feedbackloop and a λ arrow from q_0 to the end state ■





Keep track of where you can go!
A combination is final if one of the members is final.

Avoiding the non-determinism preserved the accepted language

Theorem Every regular language L there is a DFA M such that

$$L = L(M).$$

Proof. First find a NFA$_\lambda$ M such that $L(M) = L$ and then change it into a DFA preserving the language that is accepted. ■