Regular Expressions and FSMs

In this lecture we explore part of the relationship between language and state, by studying how finite state machines correspond to regular expressions.

In the process we will show how to design a FSM for a regular expression.

Kleene’s Theorem

A language, L, is a regular language (accepted by some FSM) if and only if there is some regular expression, R, such that L(R) = L

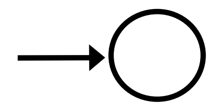
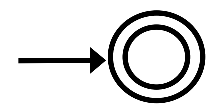
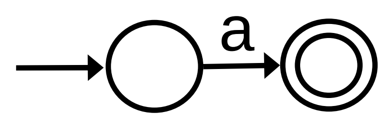
We look at a proof of half of this theorem:

For all regular expressions, R, there exists a FSM model, M, such that L(M) = L(R)

Structure of the Proof

The proof is inductive. It shows that the property holds for all the most primitive (atomic) forms of regular expression; then it shows that any extension to a larger, more complex, regular expression preserves the property.

Atomic Regular Expressions

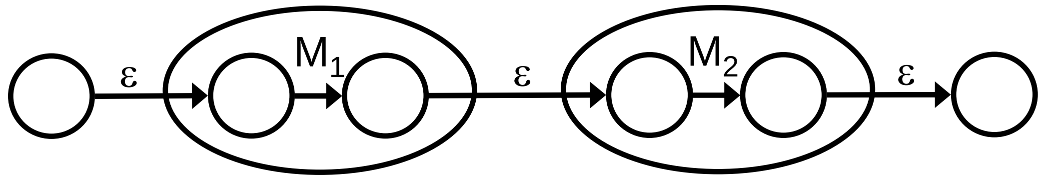
Regular expression	Intuitive meaning	Equivalent FSM
$\emptyset$	No string	
ε	Empty string	
a	Single symbol	

For every atomic regular expression, R, we can construct a FSM model, M, such that $L(\mathrm{M}) = L(\mathrm{R})$.

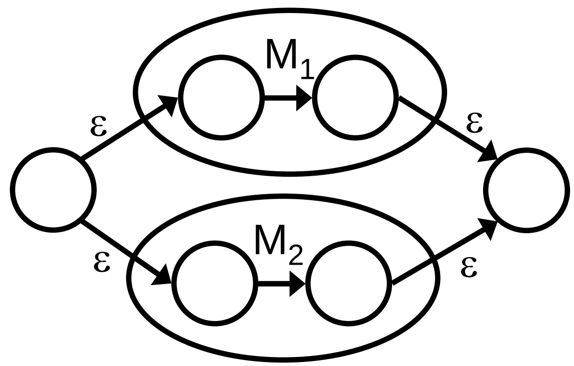
Larger Regular Expressions

Regular expressions that can be combined have FSMs that can be combined.

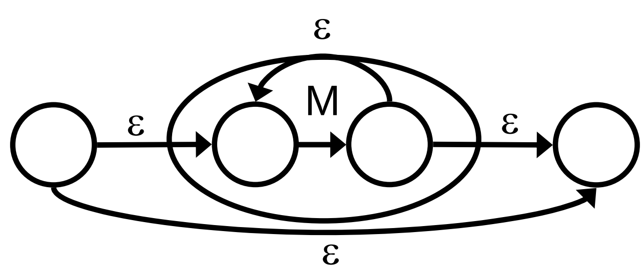
M_1M_2



$M_1|M_2$



M^*



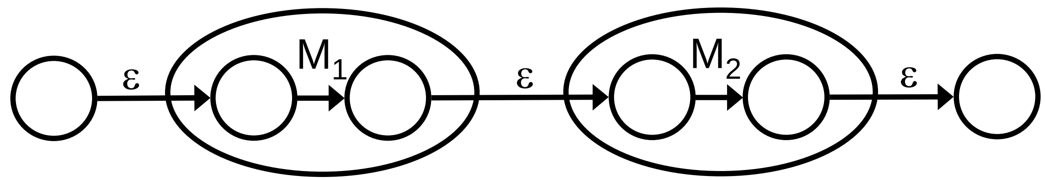
Now consider each of these three cases

Case1: Sequence

Suppose $R = R_1 R_2$

Assuming that:
- we have a machine M_1 such that $L(M_1) = L(R_1)$
- we have a machine M_2 such that $L(M_2) = L(R_2)$

we can construct



which accepts the language

$$\begin{aligned} L(M_1)L(M_2) &= \{ XY \mid X \in L(M_1) \text{ and } Y \in L(M_2) \} \\ &= L(R_1)L(R_2) \\ &= L(R_1 R_2) \end{aligned}$$

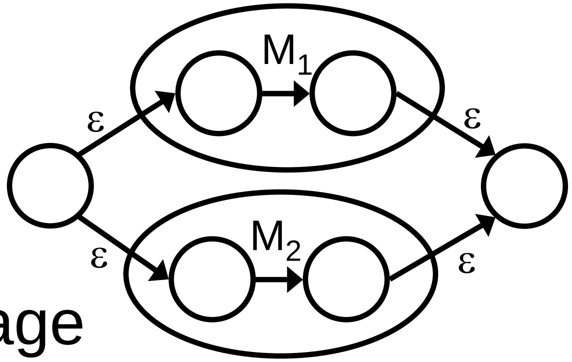
Case2: Choice

Suppose $R = R_1|R_2$

Assuming that: • we have a machine M_1 such that $L(M_1) = L(R_1)$

• we have a machine M_2 such that $L(M_2) = L(R_2)$

we can construct



which accepts the language

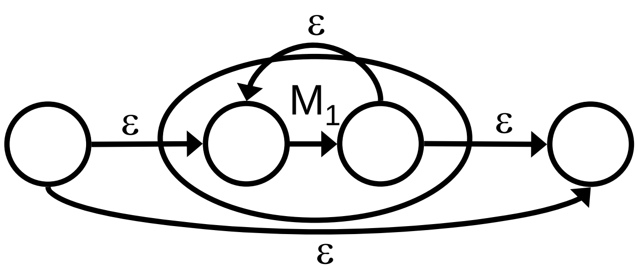
$$L(M_1) \cup L(M_2) = \{ X \mid X \in L(M_1) \text{ or } X \in L(M_2) \}$$
$$= L(R_1) \cup L(R_2)$$
$$= L(R_1|R_2)$$

Case3: Repeat

Suppose $R = R_1{}^*$

Assuming that: • we have a machine M_1 such that $L(M_1) = L(R_1)$

we can construct



which accepts the language

$$L(M_1)^* \;=\; \varepsilon \cup L(M_1) \cup L(M_1)^2 \cup L(M_1)^3$$
$$= \; L(R_1)^*$$
$$= \; L(R_1{}^*)$$

Concluding the Proof

1. Every well formed regular expression is composed via the application of sequence, choice or repeat operators to smaller well formed expressions, ending always in atomic expressions.
2. We have shown that there exists a FSM for every atomic regular expression.
3. We have shown that for any valid combination of regular expressions we can construct a FSM accepting the same language, assuming that we were able to construct FSMs for the sub-expressions.
4. This assumption always holds, given the structure at (1)

Example: Accept L((0|1)*11(0|1)*)