Regular Languages

- The language accepted by a state
- Arden’s Lemma
- NFA - DFA
\[(a|b)^*a\]

\[a(a|b)^*\]

\[(a|b)^*a(bc)^*\]

\[(a|b)^*ab(cb)^*\]

\[a(da|bc)^*\]
rules for regular languages

The following equations hold for any sets of strings $R, S, T$

- $\emptyset \mid S =$
- $\emptyset S =$
- $\varepsilon S =$
- $\varepsilon^* =$
- $\emptyset^* =$
- $R (S \mid T) =$
- $S R \mid T R =$
- $S^* S \mid \varepsilon =$
rules for regular languages

The following equations hold for any sets of strings $R, S, T$

- $\emptyset | S = \emptyset | S = S$
- $\emptyset S = S \emptyset = \emptyset$
- $\varepsilon S = S \varepsilon = S$
- $\varepsilon^* = \varepsilon$
- $\emptyset^* = \emptyset$
- $R (S | T) = R S | R T$
- $(S | T) R = S R | T R$
- $S^* = S^* S | \varepsilon = S S^* | \varepsilon$
Is there a regular expression for every FSM?

Let $L_i$ be the language accepted if $i$ is the accepting state.

$L_0 = \varepsilon$
$L_1 = L_0 a$
$L_2 = L_1 b \lor L_0 c$

$L_2 = L_0 a b \lor \varepsilon c$
$L_2 = \varepsilon a b \lor \varepsilon c$
$L_2 = a b \lor \varepsilon c$
Is there a regular expression for every FSM?

$L_1 = \epsilon$

$L_0 = \epsilon$

$L_2 = \epsilon$
Is there a regular expression for every FSM?
Is there a regular expression for every FSM?

- $a(bc)^*$
- $ab(cab)^*$
- $ab(cb \lor d)^*$
Is there a regular expression for every FSM? 

\[
L_1 = L_0 \ a \ | \ L_2 \ c \\
L_0 = \varepsilon \\
L_2 = L_1 \ b
\]
Is there a regular expression for every FSM?

\[ L_1 = L_0 a \mid L_2 c = a \mid L_1 bc \]

\[ L_0 = \varepsilon \]

\[ L_2 = L_1 b \]
Is there a regular expression for every FSM?

\[ L_1 = L_0 \ a \ | \ L_2 \ c \]
\[ = a \ | \ L_1 \ bc \]
\[ L_1 = a(bc)^* \]

\[ L_0 = \varepsilon \]

\[ L_2 = L_1 \ b \]
Arden’s Lemma

If $R$ and $S$ are regular expressions then the equation

$$X = R \mid X \cdot S$$

has a solution $X = R \cdot S^*$

If $\varepsilon \notin L(S)$ then this solution is unique.
Is there a regular expression for every FSM?

\[
L_1 = L_2 b \\
L_2 = L_3 b \mid L_1 a \\
L_3 = \varepsilon \mid L_1 b
\]
Is there a regular expression for every FSM?

\[ L_1 = L_2 b \]
\[ L_2 = L_3 b \mid L_1 a \]
\[ L_3 = \varepsilon \mid L_1 b \]
\[ = \varepsilon \mid L_2 b b \]

\[ L_2 = (\varepsilon \mid L_2 b b) b \mid L_2 b a \]
\[ = b \mid L_2 b b b \mid L_2 b a \]
\[ = b \mid L_2 (b b b \mid b a) \]
Arden’s Lemma

If $R$ and $S$ are regular expressions then the equation

$$X = R \mid X \ S$$

has a solution $X = R \ S^*$

If $\varepsilon \not\in L(S)$ then this solution is unique.

$L_2 = b \mid L_2 (b \ b \ b \ b \ l \ b \ a)$

$L_2 = b \ (b \ b \ b \ b \ l \ b \ a)^*$
Language

\( \Sigma \): a finite alphabet

A language \( L \) is a set of finite strings

\[ L \subseteq \Sigma^* \]

where the strings in \( \Sigma^* \) are of finite sequences of tokens from \( \Sigma \)

the string \(<x_0, \ldots, x_{n-1}>\) has length \( n \)

strings include the empty string

\( \varepsilon = <> \) of length 0
Finite Automata

finite alphabet $a, b \in \Sigma$; $\Sigma^+ = \Sigma \cup \{\varepsilon\}$

finite set of states $A, B \in Q$

start states $S \subseteq Q$ and final states $F \subseteq Q$

labelled transitions $A \xrightarrow{a} B \in \delta \subseteq Q \times \Sigma^+ \times Q$

A trace $q_0 \xrightarrow{s} q_n$ for $s \in \Sigma^*$ in $M$ is a

sequence $<q_0, \ldots, q_n> \in Q^*$ of states

such that $q_i \xrightarrow{x_i} q_{i+1} \in \delta$, for each $i < n$, and $s$ is the concatenation of the $x_i$ (with $\varepsilon = "\\"$)
Finite Automaton, $M$

finite alphabet $a, b \in \Sigma$; $\Sigma^+ = \Sigma \cup \{\varepsilon\}$
finite set of states $A, B \in Q$
start states $S \subseteq Q$ and final states $F \subseteq Q$
labelled transitions $A \xrightarrow{a} B \in Q \times \Sigma \cup \{\varepsilon\} \times Q$

The language accepted by $M$ is the set of strings $s$ for which there is a trace $q_0 \xrightarrow{s} q_n$

such that

for some $q_0 \in S$ and $q_n \in A$
Finite Automata

finite alphabet \( a, b \in \Sigma \); \( \Sigma^+ = \Sigma \cup \{\varepsilon}\)

finite set of states \( A, B \in Q \)

start states \( S \subseteq Q \) and final states \( F \subseteq Q \)

labelled transitions \( A \xrightarrow{a} B \in Q \times \Sigma \cup \{\varepsilon\} \times Q \)

**DFA**

- a single start state
- exactly one \( a \)-labelled transition from each state for each symbol \( a \in \Sigma \)
- no \( \varepsilon \)-transitions.

**NFA**

- no restrictions

Are there any languages recognised by some NFA, but by no DFA?
Try a simple example binary strings that end in 1

Each state $X$ lights up when we’ve seen a string with a trace from some start set to $X$
Each state $X$ lights up when we’ve seen a string with a trace from some start set to $X$.

The states of the DFA are sets of states of the NFA. *subset construction*
Each state $X$ lights up when we’ve seen a string with a trace from some start set to $X$.

The states of the DFA $M$ are sets of states of the NFA $\mathcal{P}(M)$. 

*subset construction*

The start state of $\mathcal{P}(M)$ is the set of start states of $M$.

$X \subseteq M$ is an **accepting state** of $\mathcal{P}(M)$ iff there is an accepting state $A$ of $M$ with $A \in X$.

In general not all subsets of $M$ are reachable from the start state we can ignore those any are not reachable.
In this example, we have no \( \varepsilon \)-transitions.

The **start state** of \( \mathcal{P}(M) \) is the set \( S \) of start states of \( M \).

For any reachable \( X \subseteq M \), for each \( a \in \Sigma \), the \( a \)-labelled transition from \( X \) leads to the set \( Y \) of states reachable in \( M \) from a state \( X \) in \( X \) by an \( a \)-transition.

\[
Y = \{ Y \mid \text{for some } X \in X, \ X \xrightarrow{a} Y \}
\]

\( X \xrightarrow{a} Y \) in \( \mathcal{P}(M) \) and \( Y \) is reachable.
What if we do have an \(\varepsilon\)-transition?

We have two traces for “a”.

\[ r \xrightarrow{a} q \text{ and } r \xrightarrow{a} q \xrightarrow{\varepsilon} s \]

transition table

\[
\begin{array}{c|ccc}
 M & a & b & \varepsilon \\
\hline
 r & q & & \\
 q & & s & \\
 s & q & & \\
\end{array}
\]

\[P(M)\]

\[
\begin{array}{c|cc}
 P(M) & a & b \\
\hline
 \{r\} & \{q,s\} & \{} \\
 \{q,s\} & \{} & \{q,s\} \\
\end{array}
\]
What if we do have some $\varepsilon$-transitions? The **start state** of $\mathcal{P}(M)$ is the set of states $t$ such that $s \xrightarrow{\varepsilon} t$ where $s \in S$ is a start state of $M$.

For any reachable $X \subseteq M$, for each $a \in \Sigma$, the $a$-labelled transition from $X$ leads to the set $Y$ of states reachable in $M$ from a state $X$ in $X$ by an $a$-trace.

$$Y = \{ Y \mid \text{for some } X \in X, \ X \xrightarrow{a} Y \}$$

$X \xrightarrow{a} Y$ in $\mathcal{P}(M)$ and $Y$ is reachable.
A machine with **at most one** transition with a given label from a given state is not a DFA but it is equivalent to a DFA.

What is the DFA given by the subset construction?

- Number of 1’s is one larger than number of 0’s.
- Number of 0’s is one larger than number of 1’s.

### Diagram

- States: 1, 2, 3
- Transitions:
  - From 1 to 2 on 1
  - From 1 to 3 on 0
  - From 2 to 1 on 0
  - From 2 to 3 on 1
  - From 3 to 1 on 1
  - From 3 to 2 on 0
Are these two machines equivalent?

The subset construction adds each singleton, plus the empty set which acts as a new black-hole state, which is not an accepting state. Any missing transitions go there.
Are these two machines equivalent?

The two machines accept the same strings.
A machine with at most one transition with a given label from a given state

If a machine has at most one transition with a given label from any state then it has at most one trace for any input string

The equivalent DFA produced by the subset construction has one new state, a black-hole state, which is not an accepting state. All missing transitions go there.
Deterministic FSMs

Many authors give an informal definition of deterministic

• each state has at most one transition leaving the state for each input symbol.

  Formal definition says, exactly one state …

• We consider the informal presentation to include an implicit “black hole”, or “sink” state, from which there is no escape.

• Where there is no explicit transition for a symbol, it takes us to the black hole.
Determinism

If we have a machine with at most one transition for each (q,s) pair, we can always convert to an equivalent DFA for which every state has exactly one transition leaving the state for each input symbol.

- Proof

  Add a new “black hole” state, ●

  For every pair (q, s) for which there is no state r with a transition T(q, s, r), add a transition T(q, s, ●).

  This includes a transition T(●, a, ●) for each a ∈ Σ. You cannot escape from the black hole.

  The black hole ● is not an accepting state.

This machine accepts the same language as the original.
NFA are easy to define and easy to combine.
DFA are easy to implement.

searching for cucumber