# Finite-State Machines (Automata) lecture 14

cl

- a simple form of computation
- used widely
- one way to find patterns
- one way to describe reactive systems

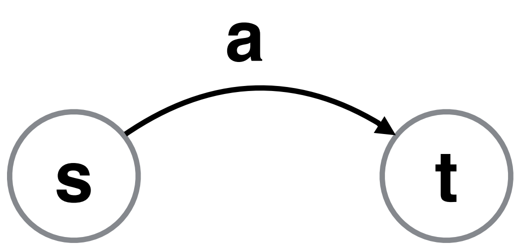
# Formal Definition

Non-deterministic FSM model,
$M = (Q, s_0, F, \Sigma, \Delta)$

- Set of states, $Q$
- Initial state $s_0 \in Q$
- Accepting states $F \subseteq Q$
- Alphabet $\Sigma$
- Transition relation, $\Delta(s, a, t)$

  where $s, t \in Q$

  and $a \in \Sigma \cup \{\varepsilon\}$.



# Reactive Systems

- Wait to receive an input
- Respond with:

  an output (changing to a new state) or

  change to new state without output
- Response depends on (finite) history

# Finite State Machines

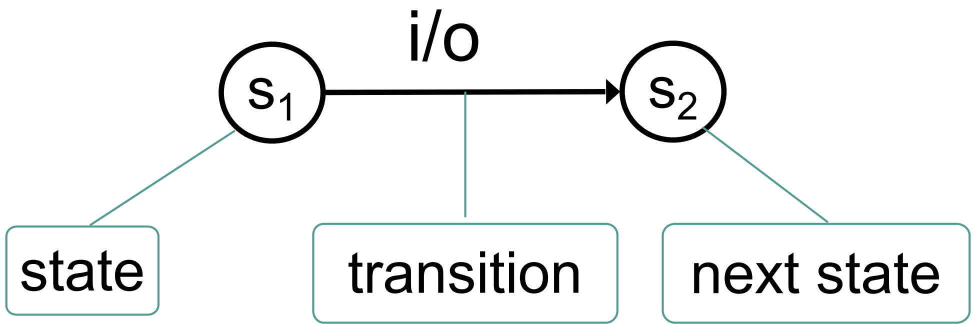
- A conceptual tool for modelling reactive systems.
- Not limited to software systems.
- Pre-dates computing science.
- Used to specify required system behaviour in a precise way.
- Then implement as software/hardware (and perhaps verify against FSM).

# Formal Definition

FSM transducer model, M, consists of:

- Set of states, Q
- Initial state $s_0 \in Q$
- Alphabets of input and output symbols i/o
- Transition relation, $\Delta(s, a, t)$ where $s, t \in Q$

  and $a \in (\mathrm{In} \cup \{\varepsilon\}) \times (\mathrm{Out} \cup \{\varepsilon\})$.



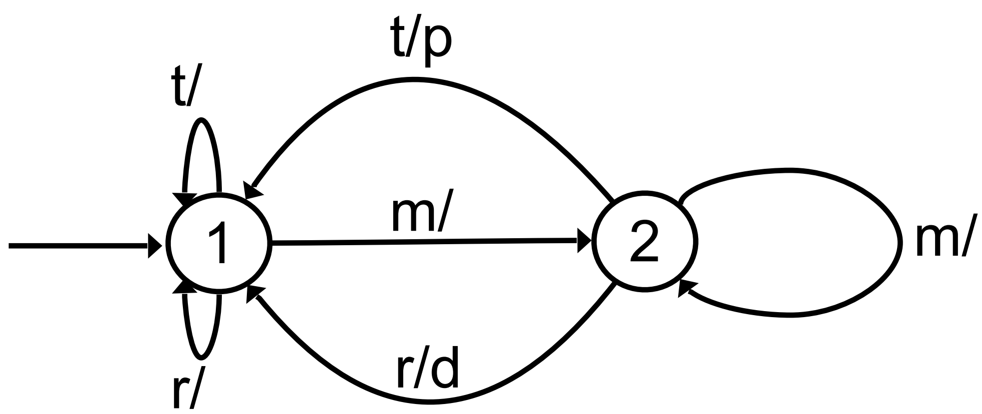
# Parking Meter Example

$\Sigma = \{m,t,r\}$ money, ticket request, refund request

$\Lambda = \{p,d\}$ print ticket, deliver refund

$Q = \{1,2\}$

$T = \{(1,t/\varepsilon,1), (1,r/\varepsilon,1), (1,m/\varepsilon,2), (2,t/p,1), (2,r/d,1), (2,m/\varepsilon,2)\}$

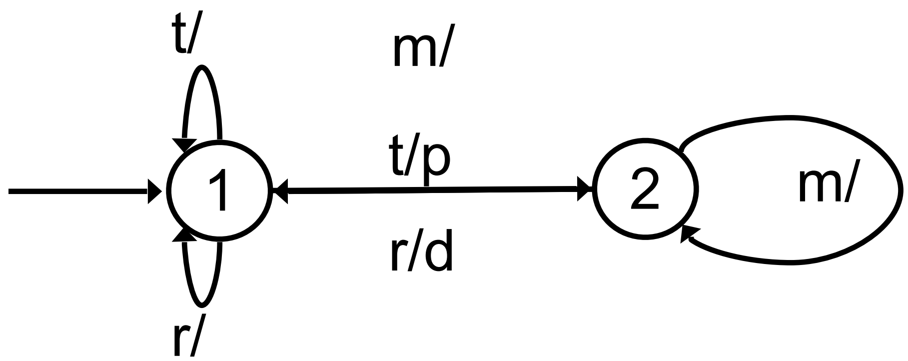



This is a **transducer** FSM because it has some outputs.

# FSM Traces

- Finite sequence of alternating state and transition labels, starting and ending with a state: $[s_0, i_1/o_1, s_1, i_2/o_2, s_2, \ldots s_{n-1}, i_n/o_n, s_n]$
- $s_0$ is the initial state.
- Each $[s_{i-1}, i_i/o_i, s_i]$ subsequence must appear as a transition in $T$

# Parking Meter Trace Example




Traces include:

[1, m/, 2, t/p, 1]
[1, m/, 2, m/, 2, r/d, 1]
[1, m/, 2, t/p, 1, m/, 2, m/, 2]
[1, t/, 1, t/, 1, m/, 2]
… *etc*

Behaviour of FSM is the set of all possible traces.
This is not necessarily a finite set.

# Transducer FSM in Logic

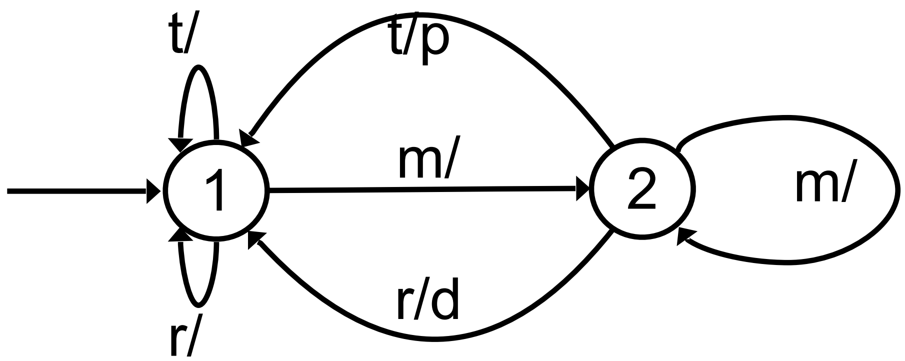
$$\text{trace}(S_i, S_f, T, []) \leftarrow S_i = S_f$$

$$\text{trace}(S_i, S_f, T, [I/O|R]) \leftarrow$$
$$\exists S_n . \Delta(S_i, I/O, S_n) \wedge$$
$$\text{trace}(S_n, S_f, T, R)$$

[] is the empty sequence
[X|R] separates first element, X,
from rest of sequence, R.

$T = \{(1,t/\varepsilon,1), (1,r/\varepsilon,1), (1,m/\varepsilon,2), (2,t/p,1), (2,r/d,1), (2,m/\varepsilon,2)\}$

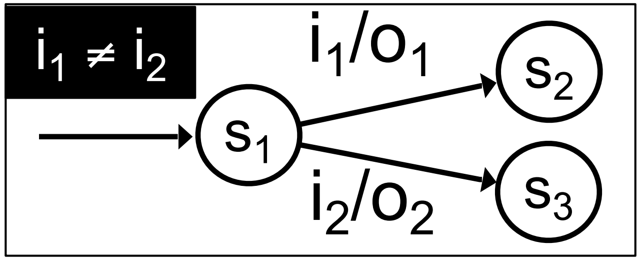
trace(1, 1, T, [m/ε, t/p])
trace(1, 1, T, [m/ε, m/ε, r/d])
trace(1, 2, T, [m/ε, t/p, m/ε, m/ε])
trace(1, 2, T, [t/ε, t/ε, m/ε])
*… etc*

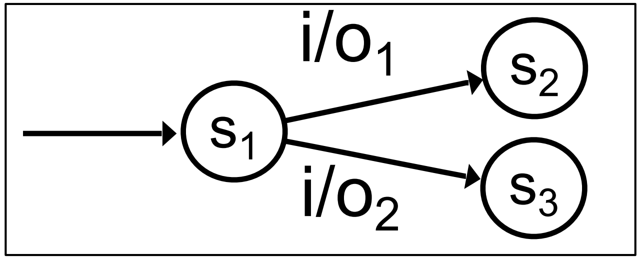


# Deterministic FSMs

A specific class of finite state system:
• deterministic FSMs
• that are acceptors

# Determinism



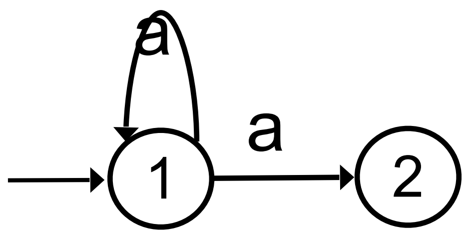




- In a deterministic FSM, all states have no more than one transition leaving the state for each input symbol.
- In a non-deterministic FSM, some states have more than one transition leaving to different successor states for the same input symbol.
- Sometimes non-deterministic FSMs are easier to define.
- Can always convert from a non-deterministic to a deterministic FSM.

# Determinism and Traces

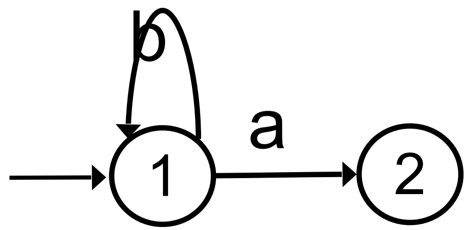
A FSM, $M$, is deterministic if for every string $x \in \Sigma^*$
there is at most one trace for $x$ in $M$
(where $\Sigma^*$ is the set of all strings in alphabet of $M$)

non-deterministic (choice)




| Sequence | Trace |
|---|---|
| aa | [1,a,1,a,2] |
| aa | [1,a,1,a,1] |

deterministic (no choice)




| Sequence | Trace |
|---|---|
| ba | [1,b,1,a,2] |
| bb | [1,b,1,b,1] |