COMP424/524-06A Topics in Software Engineering

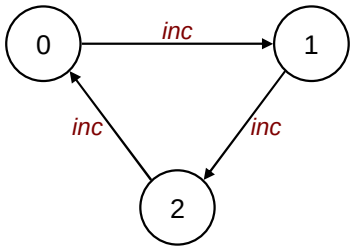
Part I – *Finite State Machines*

2. Automata and Languages

Robi Malik

THE UNIVERSITY OF WAIKATO
DEPARTMENT OF COMPUTER SCIENCE
TARI ROROHIKO

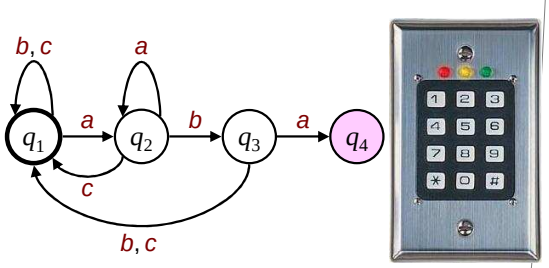
Model of a Modulo-3 Counter



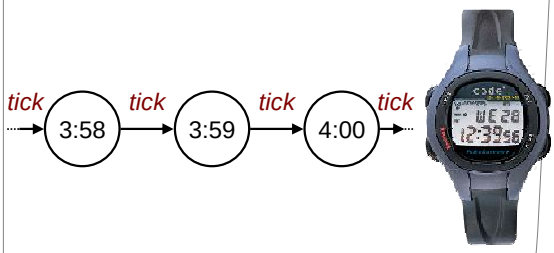
Finite State Machines

- Model checking techniques rely on finite-state representations of systems.
- A **finite-state machine** (**finite-state automaton**, **automaton**) is a machine evolving from one state to another under the action of transitions.

Model of a Digicode



Model of a Watch

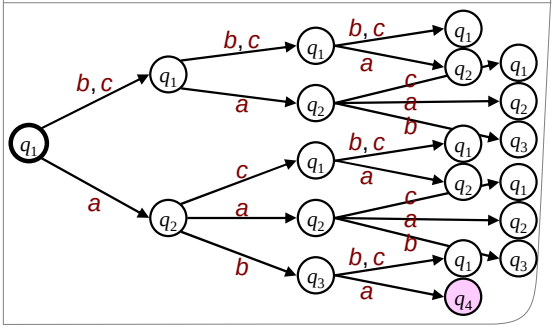


Possible Behaviours of Digicode

Sequences of actions that lead to the success state q_4:

aba,
aaba, *baba*, *caba*,
aaaba, *ababa*, *acaba*, *baaba*, *bbaba*, *bcaba*, *caaba*, *cbaba*, *ccaba*,
…

Computation Tree of Digicode



Definition of Automata

A finite-state automaton is a tuple

$$A = (Q, E, T, q_0, Q_m)$$

with

- finite set of **states** $Q = \{q_1, q_2, q_3, \ldots\}$
- finite set of **events** $E = \{a, b, c, \ldots\}$
- **transition** relation $T \subseteq Q \times E \times Q$
- **initial** state $q_0 \in Q$
- set of **marked** states $Q_m \subseteq Q$

Definition of a Path

Let $A = (Q, E, T, q_0, Q_m)$ be an automaton.

- A **path** in A is a sequence of transitions from T following each other.

$$q_1 \xrightarrow{e_1} q_2 \xrightarrow{e_2} q_3 \xrightarrow{e_3} q_4$$

- A **partial execution** of A is a path starting from the initial state q_0.

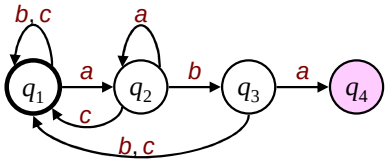
The Language of an Automaton

Let $A = (Q, E, T, q_0, Q_m)$ be an automaton.

- The **language** $\mathcal{L}(A)$ of A is the set of all strings $s \in E^*$ such that there exists a partial execution of A labeled with the events of s.
- The **marked language** $\mathcal{M}(A)$ of A is the set of all strings $s \in E^*$ such that there exists a partial execution of A labeled with the events of s and ending in a marked state $q_m \in Q_m$.

The Language of the Digicode

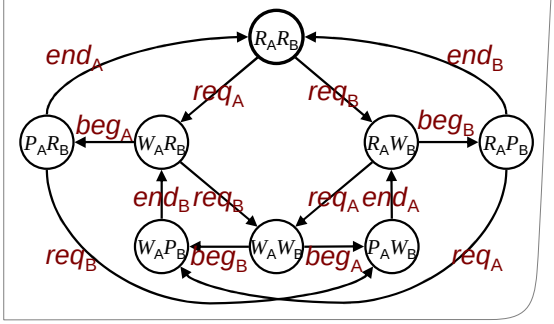
A:



$\mathcal{L}(A) = \{$

$\mathcal{M}(A) = \{$

A Printer Manager



Property 1 of Printer Manager

- Is it possible that both users A and B are printing at the same time?
- Can we reach a state marked $P_A P_B$?
- Can the printer manager execute a sequence of events containing a beg_A and a beg_B event without an end_A event between them?

Property 2 of Printer Manager

- Can a user start printing without having requested to do so?
- Can we reach a state marked P_A without passing through a state marked W_A?
- In any partial execution containing the event beg_A, is that event preceded by a req_A event?

Property 3 of Printer Manager

- If a user requests to print, will they eventually be able to print?
- Is every state marked W_A followed (possibly not immediately) by a state marked P_A?
- In any infinite partial execution containing the event req_A, is that event followed by a beg_A event?

Counterexample

- Property 3 is not satisfied for the printer manager.
- It can execute the event sequence

  $req_A req_B beg_B end_B req_B beg_B end_B \ldots$
- Model checkers can automatically compute such **counterexamples**.

Reading

Bérard et. al.: Chapter 1 – Automata