Discrete and Continuous Dynamical Systems

Attila Magyar

University if Pannonia
Faculty of Information Technology
Department of Electrical Engineering and Information Systems

Discrete and continuous dynamical systems:
Introduction to discrete event systems

March 3, 2018

Overview

Alphabets and Languages

Definition (Language)

A language defined over an event set (or alphabet) E is a set of finite-length strings formed from events in E.

Notation

The empty sting is denoted by ε.
If $tuv = s$ with $t, u, v \in E^*$, then

t *is called prefix of* s

u *is caled substring of* s

v *is called suffix of* s

Operations on Languages

Let $L, L_a, L_b \subseteq E^*$ be languages

Concatenation

$$L_a L_b = \{s \in E^* : (s = s_a s_b) \text{ and } (s_a \in L_a) \text{ and } (s_b \in L_b)\}$$

Prefix-closure $\overline{L} = \{s \in E^* : (\forall t \in E^*)\ [st \in L]\}$

Kleene-closure $L^* = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$

Post-language $L/s = \{t \in E^* : st \in L\}$

Example (Operations on languages)

Let $E = \{a, b, g\}$ and consider the two languages $L_1 = \{\varepsilon, a, abb\}$ and $L_4 = \{g\}$. Neither L_1nor L_4 are prefix-closed, since $ab \notin L_1$ and $\varepsilon \notin L_4$

$$
\begin{aligned}
L_1 L_4 &= \{g, ag, abg\} \\
\overline{L_1} &= \{\varepsilon, a, ab, abb\} \\
\overline{L_4} &= \{\varepsilon, g\} \\
L_1 \overline{L_4} &= \{\varepsilon, a, abb, g, ag, abbg\} \\
L_4^* &= \{\varepsilon, g, gg, ggg, \dots\} \\
L_1^* &= \{\varepsilon, a, abb, aa, aabb, abba, abbabb, \dots\}
\end{aligned}
$$

Projections of Strings

Definition (Projection of strings)

Let $E_s \subset E_l$. Projection of strings $P : E_L^* \to E_s^*$ where

$$P(\varepsilon) = \varepsilon$$

$$P(e) = \begin{cases} e & \text{if } e \in E_s \\ \varepsilon & \text{if } e \in E_l \setminus E_s \end{cases}$$

$$P(se) = P(s)P(e) \text{ for } s \in E_l^*, e \in E_l$$

Inverse of a projection $P^{-1} : E_s^* \to 2^{E_l^*}$
$P^{-1}(t) = \{s \in E_l^* : P(s) = t\}$

Projections of languages

Definition (Projection of language)

Let $L \subseteq E_l^*$,

$$P(L) = \{t \in E_s^* : (\exists s \in L)\ [P(s) = t]\}$$

and for $L_s \subseteq E_s^*$

$$P^{-1}(L_s) = \{s \in E_l^* : (\exists t \in L_s)\ [P(s) = t]\}$$

Projections

Example (Projections)

Let $E_l = \{a, b, c\}$ and consider two proper subsets $E_1 = \{a, b\}$ and $E_2 = \{b, c\}$. Take

$$L = \{c, ccb, abc, cacb, cabcbbca\} \subset E_l^*$$

Consider the projections $P_i : E_l^* \to E_i^*,\ i = 1, 2.$

$$P_1(L) = \{\varepsilon, b, ab, abbba\}$$
$$P_2(L) = \{c, ccb, bc, cbcbbc\}$$
$$P_1^{-1}(\{\varepsilon\}) = \{c\}^*$$
$$P_1^{-1}(\{b\}) = \{c\}^*\{b\}\{c\}^*$$
$$P_1^{-1}(\{ab\}) = \{c\}^*\{a\}\{c\}^*\{b\}\{c\}^*$$

Overview

Automata

Definition (Deterministic Automaton)

A Deterministic Automaton G is a quintuple

$$G = (X, E, f, x_0, X_m)$$

where

X is the set of states

E is a finite set of events associated with G

f : $X \times E \to X$ (partial) transition function

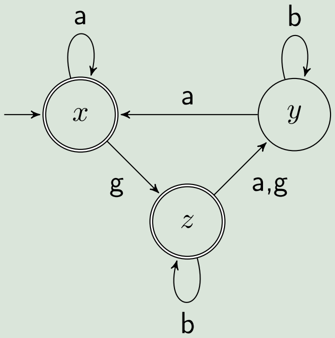
x_0 is the initial state

$X_m \subseteq X$ is the set of marked states (or accepting-, or final states)

Synonyms: state machine, generator

Determinism: f is a function

Example (A simple automaton - state transition diagram)



- Event set $E = \{a, b, g\}$
- Nodes (states) $X = \{x, y, z\}$
- Transition function $f : X \times E \to X$

$$f(x,a) = x \quad f(x,g) = z$$
$$f(y,a) = x \quad f(y,b) = y$$
$$f(z,b) = z \quad f(z,a) = f(z,g) = y$$

Deterministic Automata

Extended transition function For sake of convenience f is always extended from domain $X \times E$ to $X \times E^*$ as follows

$$f(x, \varepsilon) = x$$
$$f(x, se) = f(f(x, s), e) \text{ for } s \in E^* \text{ and } e \in E$$

Active event set $\Gamma(x)$ is the set of all events e for which $f(x, e)$ is defined. Also known as feasible event set

Example (Ext. transition function)

$$\begin{aligned} f(y, \varepsilon) &= y \\ f(x, gba) &= y \\ f(x, aagb) &= z \\ f(z, b^n) &= z, \text{ for all } n \geq 0 \end{aligned}$$

Example (Active event set)

$$\begin{aligned} \Gamma(x) &= \{a, g\} \\ \Gamma(y) &= \{a, b\} \\ \Gamma(z) &= \{a, b, g\} \end{aligned}$$

Languages and automata

Definition (Languages generated and marked)

The language generated by $G = (X, E, f, x_0, X_m)$ is

$$\mathcal{L}(G) = \{s \in E^* : f(x_0, s) \text{ is defined}\}$$

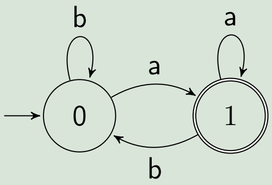
The language marked by G is

$$\mathcal{L}_m(G) = \{s \in \mathcal{L}(G) : f(x_0, s) \in X_m\}$$

f already means the extended transition function!

- Language $\mathcal{L}(G)$ represents all directed paths (i.e. strings) on the state transition digraph starting az x_0.
- Language $\mathcal{L}_m(G)$ represents all paths that end at a marked state
- $\mathcal{L}_m(G) \subseteq \mathcal{L}(G)$

Example (Marked language)



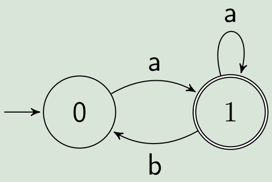
- Event set $E = \{a, b\}$
- Language marked

  $\mathcal{L}_m(G) = \{a, aa, ba, aaa, aba, baa, bba, \dots\}$

- Language generated

  $\mathcal{L}(G) = E^*$ (since f is a total function)

Example (Marked and generated language)



- Language generated

  $\mathcal{L}(G) =$ any b is the last or followed by a

- Language marked

  $\mathcal{L}_m(G) =$ strings end with event $a \subset \mathcal{L}(G)$

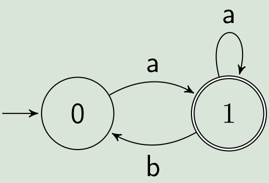
Language Equivalence

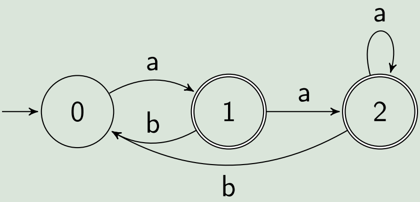
Definition (Language-equivalent automata)

Automata G_1 and G_2 are language-equivalent if

$$\mathcal{L}(G_1) = \mathcal{L}(G_2) \quad \text{and} \quad \mathcal{L}_m(G_1) = \mathcal{L}_m(G_2)$$

Example (Language-equivalent automata)





Blocking

Generally

$$\mathcal{L}_m(G) \subseteq \overline{\mathcal{L}_m(G)} \subseteq \mathcal{L}(G)$$

Definition (Blocking)

Automaton G is said to be blocking if

$$\overline{\mathcal{L}_m(G)} \subset \mathcal{L}(G)$$

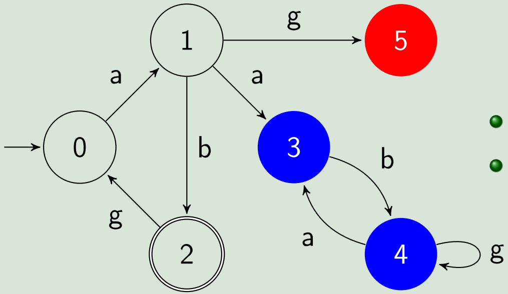
where the set inclusion is proper, and nonblocking if

$$\overline{\mathcal{L}_m(G)} = \mathcal{L}(G)$$

If an automaton is blocking, deadlock and livelock can happen.

Deadlock and livelock

Example



- State 5 is a deadlock state
- States 3 and 4 are involved in a livelock

Deadlock is a state x where $\Gamma(x) = \emptyset$ but $x \notin X_m$

Livelock is a set of unmarked states of G forming a strongly connected component (i.e. no transition is going out from the set)

Nondeterministic Automata

Definition (Nondeterministic automaton)

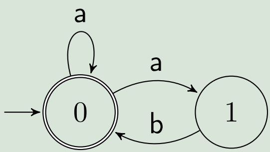
A nondeterministic automaton G_{nd} is a quintuple

$$G_{nd} = (X, E \cup \{\varepsilon\}, f_{nd}, x_0, X_m)$$

where all the objects have the same interpretation as in the definition of deterministic automaton except

1. f_{nd} is a function $f_{nd} : X \times E \cup \{\varepsilon\} \to 2^X$, i.e. $f_{nd}(x, e) \subseteq X$ whenever it is defined.
2. The initial state may itself be a set of states, $x_0 \subseteq X$

Example (A simple nondeterministic automaton)



Moore and Mealy automata

Moore
- An output function assigns an output to each state
- Generalizes the notion of marking
- Standard automata can be thought as having two outputs (marked, non-marked)

Mealy
- Input/output automata
- Transitions are labeled by events in the form input event / output event
- E_{out} may not be the same as E_{in}

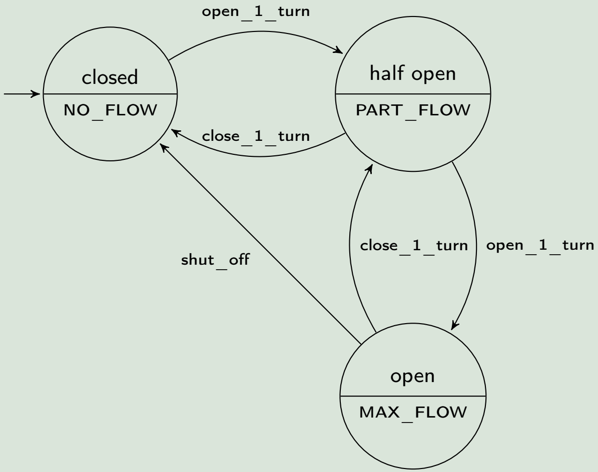
Interpretation (Mealy transitions)

When the system is in state x and the automaton receives an input event e_i it will make a transition to state y and will output the event e_o.

0

Moore automata

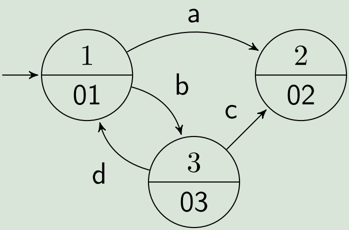
Example (Valve together with a flow sensor as a Moore automaton)



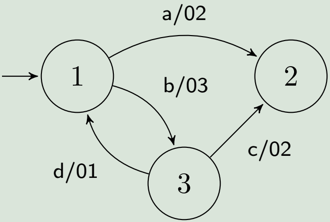
Mealy automata

Example

Moore:



Mealy:



Overview

Accessible Part

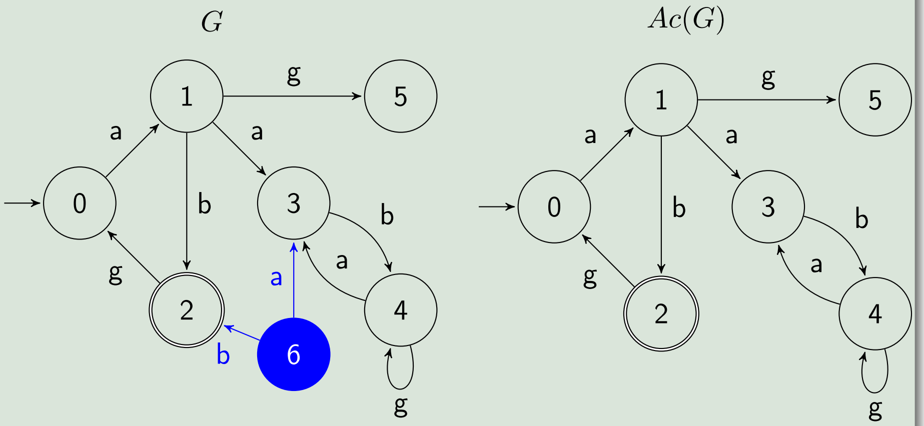
States of G not accessible from x_0 can be deleted without affecting $\mathcal{L}(G)$ and $\mathcal{L}_m(G)$.

$$\begin{aligned} Ac(G) &= (X_{ac}, E, f_{ac}, x_0, X_{ac,m}) \quad \text{where} \\ X_{ac} &= \{x \in X : (\exists s \in E^*)\ [f(x_0, s) = x]\} \\ X_{ac,m} &= X_m \cap X_{ac} \\ f_{ac} &= f|_{X_{ac} \times E \to X_{ac}} \end{aligned}$$

where $f|_{X_{ac} \times E \to X_{ac}}$ means restricting f to a smaller domain
Operation Ac has no effect on $\mathcal{L}(G)$ and $\mathcal{L}_m(G)$. From now on, $G = Ac(G)$ is assumed.

Accessible Part

Example (Accessible part)



Coaccessible Part

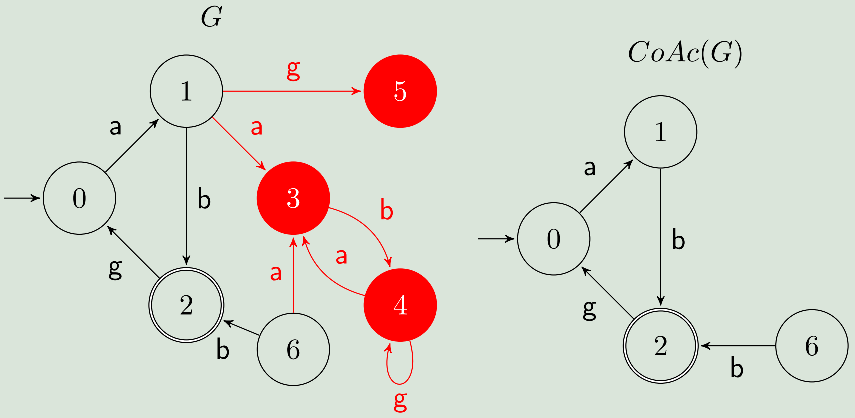
A state x of G is coaccessive (to X_m) if there is a path from state x to a marked state. The operation of deleting all the states of G not coaccessible is defined as follows

$$
\begin{aligned}
CoAc(G) &= (X_{coac}, E, f_{coac}, x_{0,coac}, X_m) \quad \text{where} \\
X_{coac} &= \{x \in X : (\exists s \in E^*)\ [f(x,s) \in X_m]\} \\
x_{0,coac} &= \begin{cases} x_0 & \text{if } x_0 \in X_{coac} \\ \text{undefined} & \text{otherwise} \end{cases} \\
f_{coac} &= f|_{X_{coac} \times E \to X_{coac}}
\end{aligned}
$$

Operation $CoAc$ may shrink $\mathcal{L}(G)$ but does not affect $\mathcal{L}_m(G)$.

Coaccessible Part

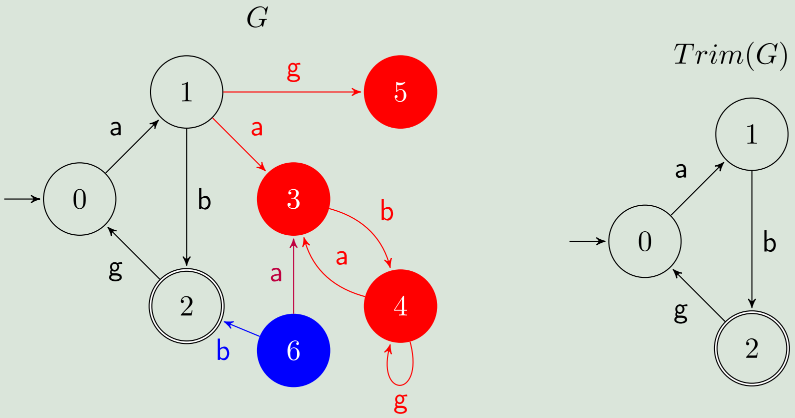
Example (Coaccessible part)



Trim Operation

An automaton both accessible and coaccessible is said to be trim:
$Trim(G) = CoAc(Ac(G)) = Ac(CoAc(G))$

Example (Trim Operation)



Projection and Inverse Projection

Projection

- Let G have event sef E. Furthermore, let $E_s \subset E$
- The projections of $\mathcal{L}\{G\}$ $\mathcal{L}_m\{G\}$ from E^* to E_s^* can be implemented on G by replacing all labels in $E \setminus E_s$ by ε.
- The result is a nondeterministic automaton.

Inverse projection

- Let $K_s = \mathcal{L}(G) \subset E_s^*$ and $K_{m,s} = \mathcal{L}_m(G)$. Furthermore, let $E_s \subset E_l$ and P_s is the projection from E_l^* to E_s^*
- The automaton that generates $P_s^{-1}(K_s)$ and marks $P_s^{-1}(K_{m,s})$ can be obtained by adding self-loops for all the events in $E_l \setminus E_s$ at all the states of G

Complement

Given an automaton $G = (X, E, f, x_0, X_m)$ with $\mathcal{L}_m(G) \subseteq E^*$. Thus, $\mathcal{L}(G) = \overline{\mathcal{L}_m(G)}$.

Let's build G^{comp} for which $\mathcal{L}_m(G^{comp}) = E^* \setminus \mathcal{L}_m(G)$

Step 1 Add a dump state x_d and all undefined $f(x, e)$ will be assigned to x_d

$$f_{tot}(x, e) = \begin{cases} f(x, e) & \text{if } e \in \Gamma(x) \\ x_d & \text{otherwise} \end{cases}$$

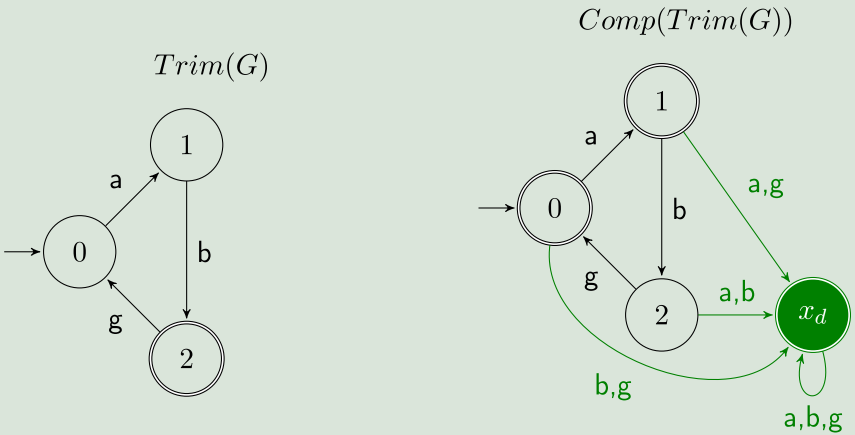
$$f_{tot}(x_d, e) = x_d, \quad \forall e \in E$$

Step 2 Mark all unmarked states (and x_d) and unmark all marked states

$$Comp(G) = (X \cup \{x_d\}, E, f_{tot}, x_0, (X \cup \{x_d\}) \setminus X_m)$$

Complement

Example (Complement)



Product of automata

Definition (Product)

The product of G_1 and G_2 is the automaton

$$G_1 \times G_2 = Ac\,(X_1 \times X_2, E_1 \cup E_2, f, (x_{01}, x_{02}), X_{m1} \times X_{m2})$$

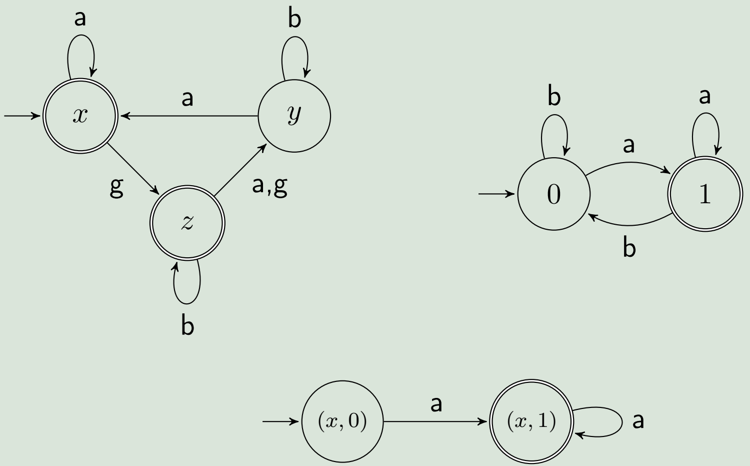
where

$$f((x_1, x_2), e) = \begin{cases} (f_1(x_1, e), f_2(x_2, e)) & \text{if } e \in \Gamma(x_1) \cap \Gamma(x_2) \\ \text{undefined} & \text{otherwise} \end{cases}$$

- $\Gamma_{1\times 2}(x_1, x_2) = \Gamma_1(x_1) \cap \Gamma_2(x_2)$
- $\mathcal{L}(G_1 \times G_2) = \mathcal{L}(G_1) \cap \mathcal{L}(G_2)$
- $\mathcal{L}_m(G_1 \times G_2) = \mathcal{L}_m(G_1) \cap \mathcal{L}_m(G_2)$
- $G_1 \times G_2 \times G_3 = (G_1 \times G_2) \times G_3 = G_1 \times (G_2 \times G_3)$

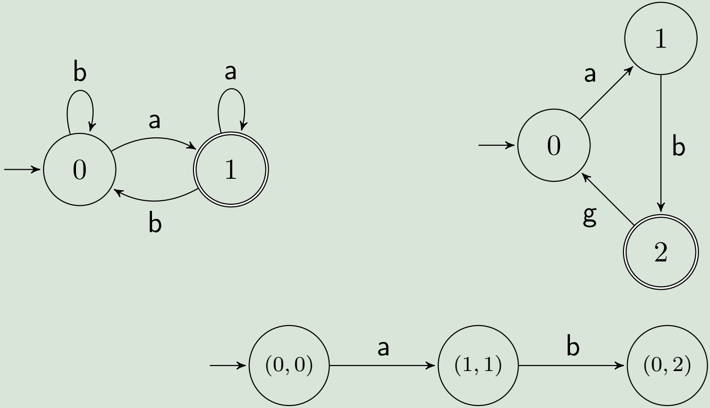
Product of automata

Example (Product)



Product of automata

Example (Product)



Parallel Composition of Automata

Definition (Parallel composition)

The parallel composition of G_1 and G_2 is the automaton

$$G_1 || G_2 = Ac\left(X_1 \times X_2, E_1 \cup E_2, f, (x_{01}, x_{02}), X_{m1} \times X_{m2}\right)$$

where

$$f((x_1, x_2), e) = \begin{cases} (f_1(x_1, e), f_2(x_2, e)) & \text{if } e \in \Gamma(x_1) \cap \Gamma(x_2) \\ (f_1(x_1, e), x_2) & \text{if } e \in \Gamma_1(x_1) \setminus E_2 \\ (x_1, f_2(x_2, e)) & \text{if } e \in \Gamma_2(x_2) \setminus E_1 \\ \text{undefined} & \text{otherwise} \end{cases}$$

- $\Gamma_{1\times 2}(x_1, x_2) = [\Gamma_1(x_1) \cap \Gamma_2(x_2)] \cup [\Gamma_2(x_2) \setminus E_1] \cup [\Gamma_1(x_1) \setminus E_2]$

Parallel composition

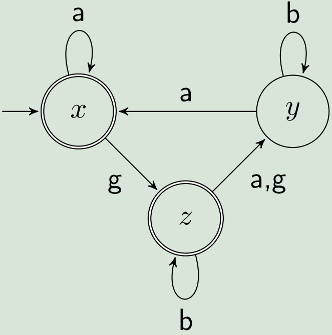
- The two automata are synchronized on the common events $e \in E_1 \cap E_2$ (can be executed simultaneously)
- Private events $e \in E_2 \setminus E_1$ or $e \in E_1 \setminus E_2$ can be executed whenever its possible (concurrently)
- If $E_1 = E_2$, then $G_1 || G_2 = G_1 \times G_2$
- $\mathcal{L}(G_1 || G_2) = P_1^{-1}[\mathcal{L}(G1)] \cap P_2^{-1}[\mathcal{L}(G2)$
- $\mathcal{L}_m(G_1 || G_2) = P_1^{-1}[\mathcal{L}_m(G1)] \cap P_2^{-1}[\mathcal{L}_m(G2)$

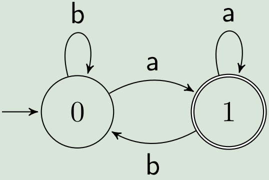
where $P_i : (E_1 \cup E_2)^* \rightarrow E_i^*$ for $i = 1, 2$

Parallel Composition of Automata

Example (Parallel Composition)

Give the parallel composition of the following two automata!

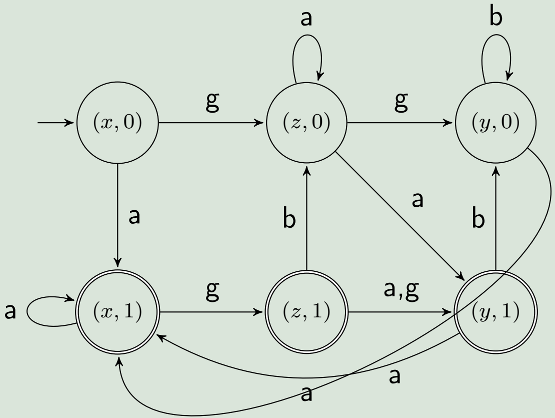




Parallel Composition of Automata

Example (Parallel Composition)

Solution



Overview

Nondeterminism

Possible sources of nonterminism

- Stochastic transitions (model is not detailed enough)
- Unobservable events

Problem: The actual state of the automaton is unknown by knowing the sequence of observable events

Nondeterministic Automata

Definition (Nondeterministic automaton)

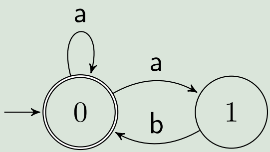
A nondeterministic automaton G_{nd} is a quintuple

$$G_{nd} = (X, E \cup \{\varepsilon\}, f_{nd}, x_0, X_m)$$

where all the objects have the same interpretation as in the definition of deterministic automaton except

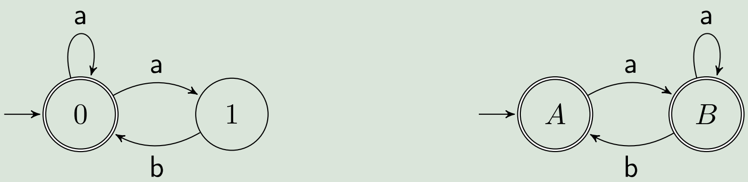
1. f_{nd} is a function $f_{nd} : X \times E \cup \{\varepsilon\} \to 2^X$, i.e. $f_{nd}(x, e) \subseteq X$ whenever it is defined.
2. The initial state may itself be a set of states, $x_0 \subseteq X$

Example (A simple nondeterministic automaton)



Motivating example

Example (Nondeterministic and deterministic automata)



Reachability function

ε-reachability function

$$\varepsilon R(x) = \{p \in X : p \text{ is reachable from } x \text{ by } \varepsilon\}$$
$$\varepsilon R(B) = \cup_{x \in B} \varepsilon R(x)$$

Extended transition mapping

$$f_{nd}^{ext}(x, \varepsilon) = \varepsilon R(x)$$
$$f_{nd}^{ext}(x, ue) = \varepsilon R[\{z : z \in f_{nd}(y, e) \text{ for some state } y \in f_{nd}^{ext}(x, u)\}]$$

Observer automata

Procedure of building an observer $Obs(G_{nd})$

Step 1: Define $x_{0,obs} = \varepsilon R(x_0)$. Set $X_{obs} = \{x_{0,obs}\}$.

Step 2: for each $B \in X_{obs}$ and $e \in E$

$$f_{obs}(B, e) = \varepsilon R(\{x \in X : (\exists x_e \in B)\ [x \in f_{nd}(x_e, e)]\})$$

Step 3: Repeat Step 2 until the accessible part of $Obs(G_{nd})$ has been constructed

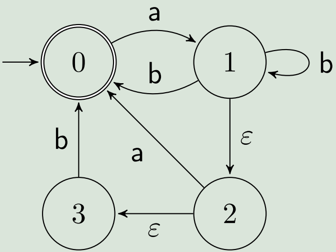
Step 4: $X_{m,obs} = \{B \in X_{obs} : B \cup X_m \neq \emptyset\}$

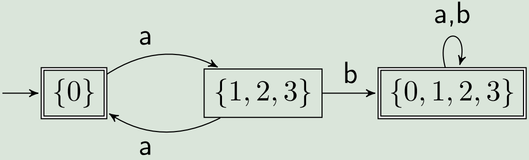
- $Obs(G_{nd})$ is a deterministic automaton
- $\mathcal{L}(Obs(G_{nd})) = \mathcal{L}(G_{nd})$
- $\mathcal{L}_m(Obs(G_{nd})) = \mathcal{L}_m(G_{nd})$

Important in studying partially observed DES

Example

Example (Another example)





Partially observed DES

- ε-transitions were defined to describe unobservable events
- Let us define genuine events for this phenomenon: unobservable events $E = E_{uo} \cup E_o$ where $E_{uo} \cap E_O = \emptyset$

Definition (Unobservable reach)

The unobservable reach of state $x \in X$ denoted by $UR(x)$ is

$$UR(x) = \{y \in X : \ (\exists t \in E_{uo}^*)[f(x,t) = y]\}$$

The definition can be extended to sets of states $B \subseteq X$ by

$$UR(B) = \cup_{x \in B} UR(x)$$

Observer for automaton G with unobservable events

Let $G = (X, E, f, x_0, X_m)$ be a deterministic automaton and let $E = E_{uo} \cup E_o$. Then $Obs(G) = (X_{obs}, E_o, f_{obs}, x_{0,obs}, X_{m,obs})$ can be built as follows

Step 1: Define $x_{0,obs} = UR(x_0)$
set $X_{m,obs} = \{x_{0,obs}\}$

Step 2: For each $B \in X_{obs}$ and $e \in E_o$ define

$$f_{obs}(B, e) = UR(\{x \in X : (\exists x_e \in B)[x \in f(x_e, e)]\})$$

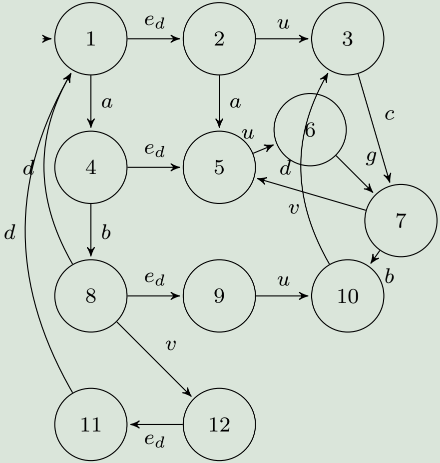
whenever $f(x_e, e)$ is defined for some $x_e \in B$

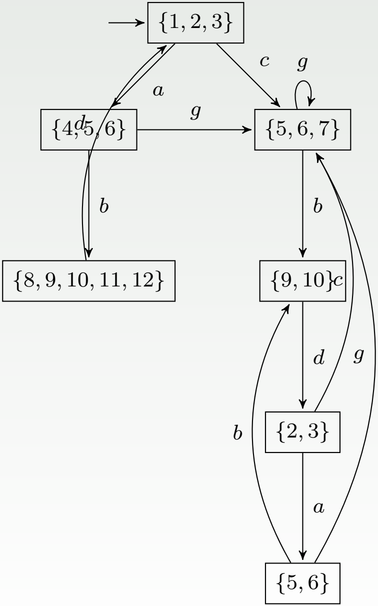
Step 3: Repeat Step 2 until the entire accessible part of $Obs(G)$ has been constrcted

Step 4: $X_{m,obs} = \{B \in X_{obs} : \; B \cap X_m \neq \emptyset\}$

Observer with unobservable events

Example (Automaton with unobservable events)




{1, 2, 3}
c
g
a
g
{4, 5, 6}
d
{5, 6, 7}
b
b
{8, 9, 10, 11, 12}
{9, 10}
c
d
g
b
{2, 3}
a
{5, 6}