LECTURE 3: DISCRETE EVENT SYSTEMS II

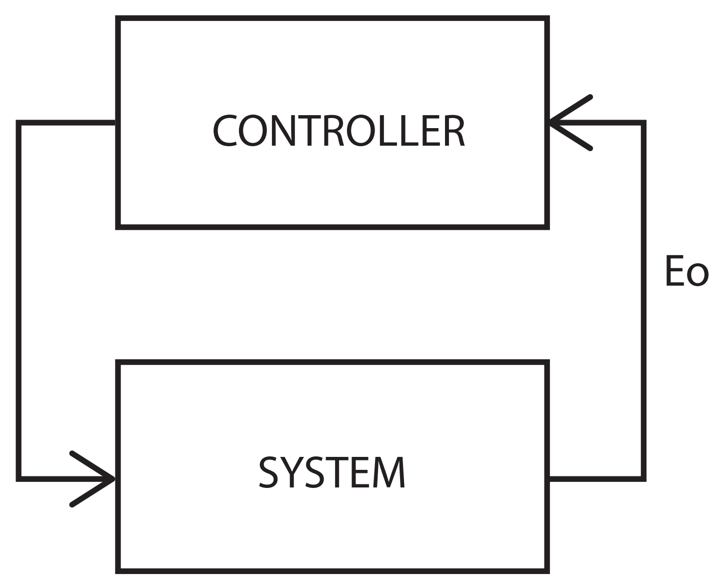
Modeling and Simulation 2

Daniel Georgiev

Winter 2014

Definition: (Observable events)
The event set can be generally divided into observable sets and unobservable sets.

$$E = E_o \cup E_{uo}$$



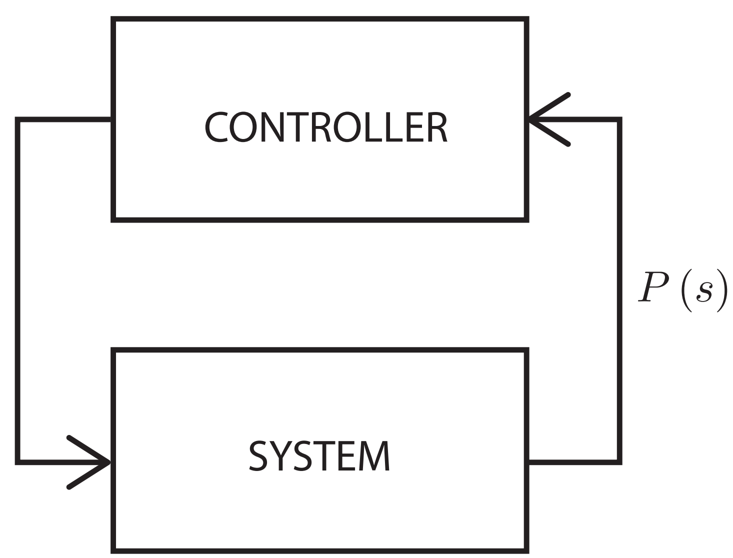
UNOBSERVABLE EVENTS

Definition: (Natural projection)
Natural projection $P : E^* \to E_o^*$ onto the set of observable events is defined by

$$P(\epsilon) = \epsilon$$

$$P(e) = \begin{cases} \epsilon, & \text{if } e \in E_{uo} \\ e, & \text{if } e \in E_o \end{cases}$$

$$P(se) = P(s)\,P(e)\,, \forall s \in E^* \text{ and } e \in E$$

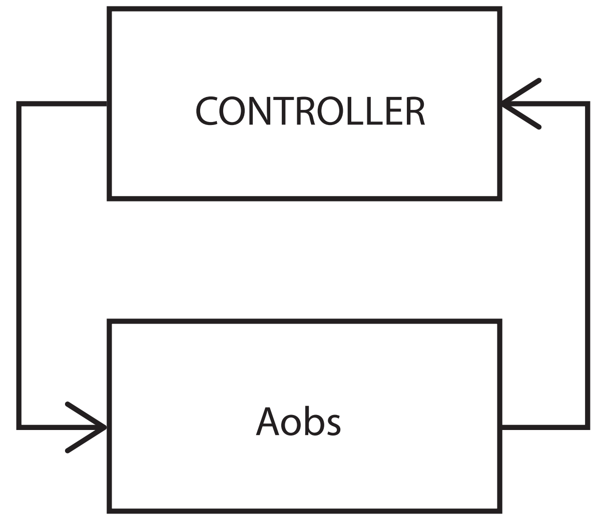


NATURAL PROJECTION

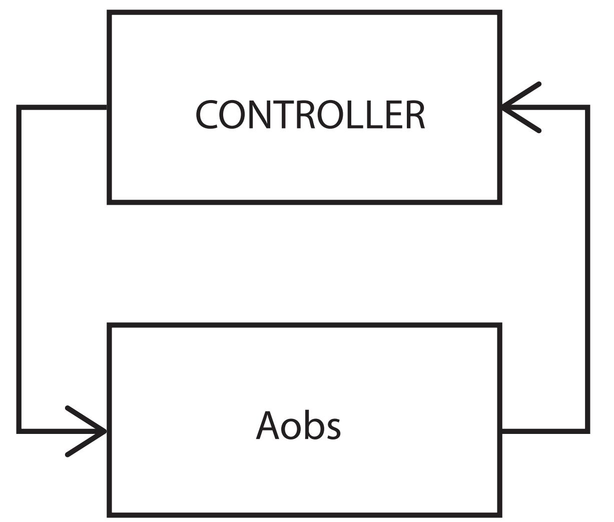
Definition: (Inverse projection)
Inverse projection $P^{-1} : E_o^* \to 2^{E^*}$ onto the set of all events is defined by

$$P^{-1}(t) = \{s \in E^* : P(s) = t\}$$

INVERSE PROJECTION



OBSERVER MODEL



$$A_{obs} = (Q_{obs}, E_o, g_{obs}, q_{0,obs}, Q_{m,obs})$$

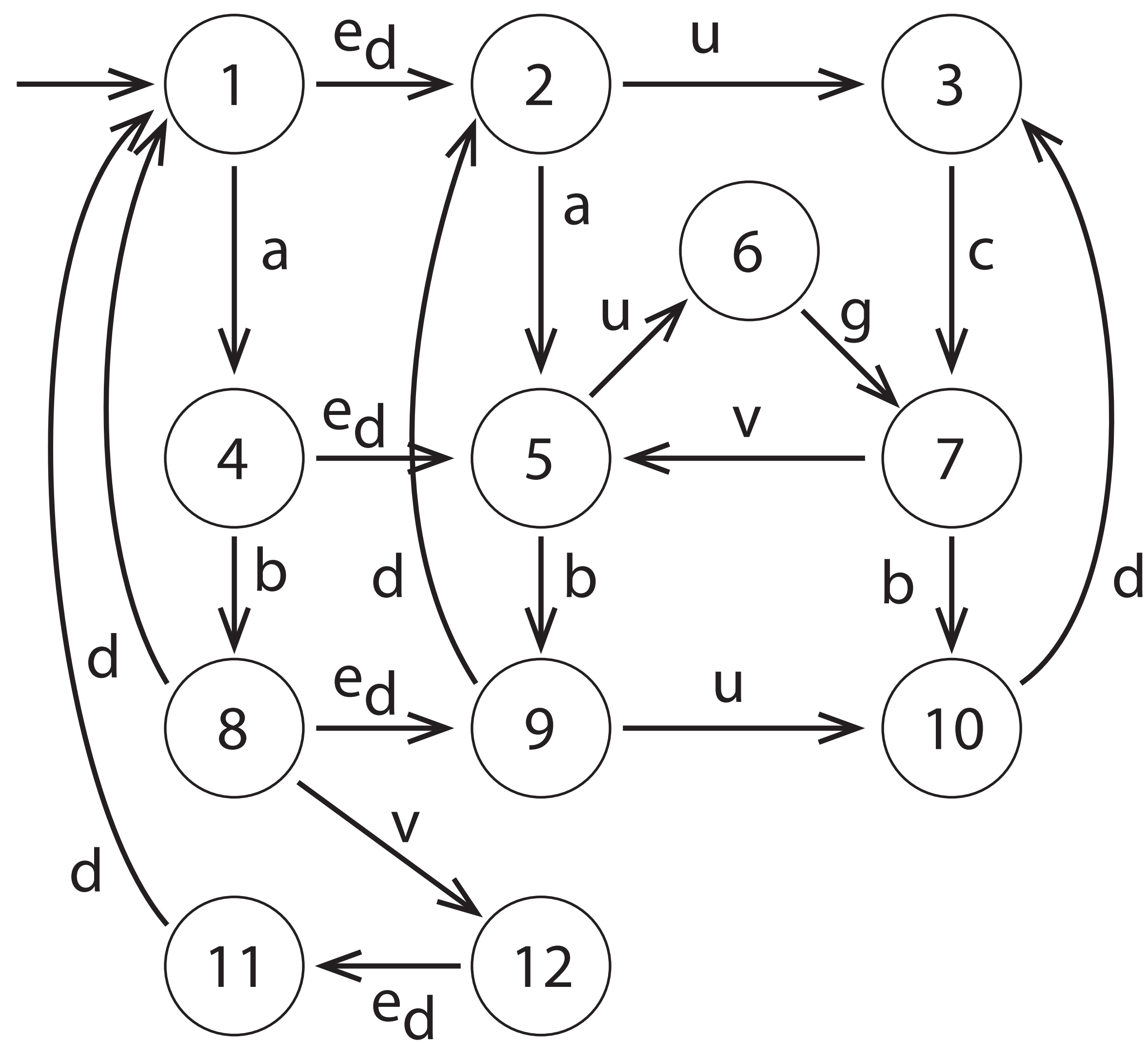
$$Q_{obs} = 2^Q \setminus \emptyset$$

$$q_{0,obs} = UR(q_0)$$

$$g_{obs}(S, e) = UR(\{q \in Q : \exists q_e \in S, q \in g(q_e, e)\})$$

$$Q_{m,obs} = \{S \subseteq Q : S \cap Q_m \neq \emptyset\}$$

OBSERVER MODEL



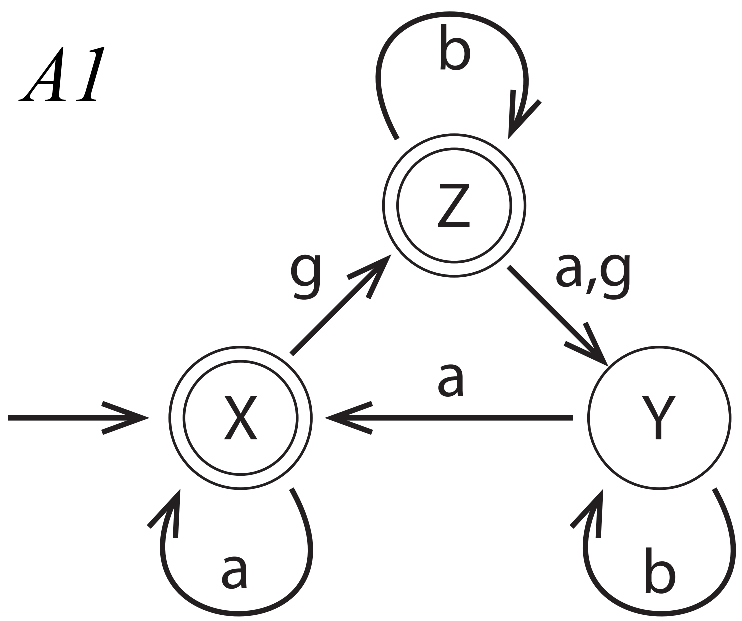
OBSERVER MODEL

Problem: (Diagnostics)
For an automata A and a set events $E = E_o \cup E_{uo}$, determine whether an unobservable event e occurred with certainty after observing a given string $s \in E_o^*$

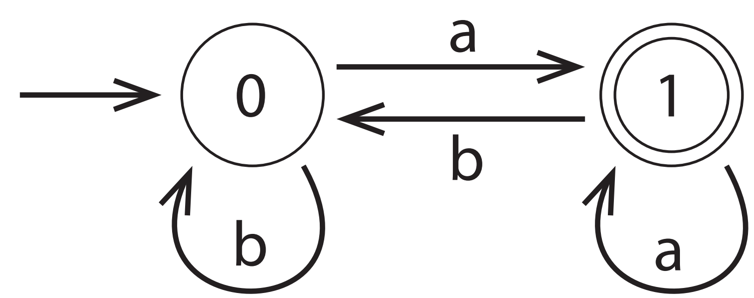
DIAGNOSER CONCEPT

Definition: (Composition operations)
For two automata *A1* and *A2*, one can define several compositions: concatenation, Kleene closure, union, parallel, and product.

A1



A2



COMPOSITION

Definition: (Composition operations)
For two automata *A1* and *A2*, one can define several compositions: concatenation, Kleene closure, union, product, and parallel.

$$A_i = (Q_i, E_i, g_i, \Gamma_i, q_{0i}, Q_{mi})$$
$$A_1 \times A_2 = Ac\left(Q_1 \times Q_2, E_1 \cap E_2, g, \Gamma_{1\times 2}, \left(q_{01}, q_{02}\right), Q_{m1} \times Q_{m2}\right)$$

$$g\left(\left(q_1, q_2\right), e\right) = \begin{cases} \left(g_1\left(q_1, e\right), g_2\left(q_2, e\right)\right), & \text{if } e \in \Gamma_1\left(q_1\right) \cap \Gamma_2\left(q_2\right) \\ \text{undefined}, & \text{otherwise} \end{cases}$$

PRODUCT COMPOSITION

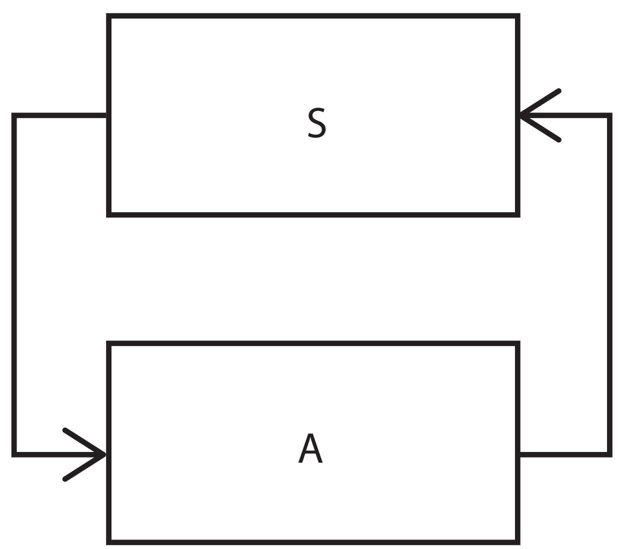
Definition: (Composition operations)
For two automata *A1* and *A2*, one can define several compositions: concatenation, Kleene closure, union, product, and parallel.

$$A_i = (Q_i, E_i, g_i, \Gamma_i, q_{0i}, Q_{mi})$$
$$A_1 || A_2 = Ac\left(Q_1 \times Q_2, E_1 \cup E_2, g, \Gamma_{1||2}, (q_{01}, q_{02}), Q_{m1} \times Q_{m2}\right)$$

$$g\left((q_1, q_2), e\right) = \begin{cases} \left(g_1\left(q_1, e\right), g_2\left(q_2, e\right)\right), & \text{if } e \in \Gamma_1\left(q_1\right) \cap \Gamma_2\left(q_2\right) \\ \left(g_1\left(q_1, e\right), q_2\right), & \text{if } e \in \Gamma_1\left(q_1\right) \setminus E_2 \\ \left(q_1, g_2\left(q_2, e\right)\right), & \text{if } e \in \Gamma_2\left(q_2\right) \setminus E_1 \\ \text{undefined}, & \text{otherwise} \end{cases}$$

PARALLEL COMPOSITION

Definition: (Required and Admissible languages)
For two automata *A1* and *A2*, one can define several compositions: concatenation, Kleene closure, union, product, and parallel.



$$L_r \subseteq \mathcal{L}\left(S/A\right) \subseteq L_a$$

AUTOMATA SPECIFICATION

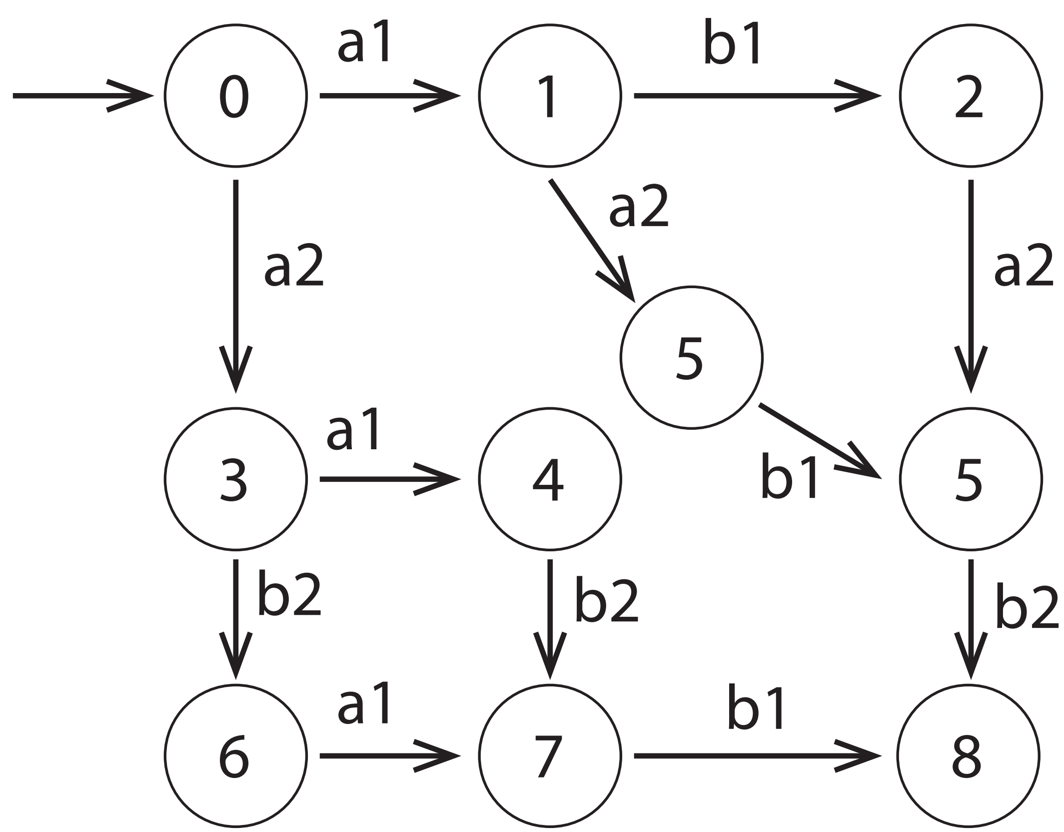
Definition: (Automata specifications)
Automata can be used to formulate specifications.

$$L_a = \mathcal{L}\left(A_{spec}||A\right), \text{ or}$$
$$L_a = \mathcal{L}\left(A_{spec} \times A\right)$$

Example: (Illegal states)
Aspec is constructed from *A* by deleting illegal states.
Example: (State splitting)
Aspec requiring knowledge of how a state was reached is constructed from *A* with additional state splitting.
Example: (Event alternance)
Aspec is a two state automata with only the two alternating events.
Example: (Illegal substring)
Aspec forbidding some substring is constructed by building a memory automata.

EXAMPLE SPECIFICATIONS

Claim: (Formal methods)
Formal methods are required to design specification automata in the presence of uncontrollable and unobservable events.



Example 1: $E_{uc} = \{a_2, b_2\}$

Example 2: $E_{uo} = \{a_2\}$

Example 2: $E_{uo} = \{a_2\}, E_{uc} = \{a_1\}$

UNCONTROLLABILITY AND UNOBSERVABILITY

Theorem: (Controllability Theorem)
Consider an automaton A with uncontrollable event sets Euc. Consider also the nonempty language $K \subseteq \mathcal{L}(A)$. There exists a nonblocking supervisor S such that

$$\mathcal{L}_m(S/A) = K \text{ and } \mathcal{L}(S/A) = \overline{K}$$

if and only if

1) $\overline{K}E_{uc} \cap \mathcal{L}(A) \subseteq \overline{K}$
2) K is $\mathcal{L}_m(A) - \text{closed}$

CONTROLLABILITY

Definition: (Observability)
Let K and $M = \overline{M}$ be languages over event set E. Let Ec be a designated subset of E. Let Eo be another designated subset of E with P as the corresponding natural projection. K is said to be observable with respect to M, P, and Ec if for all $s \in \overline{K}$ and for all $\sigma \in E_c$,

$$\left(s\sigma \notin \overline{K}\right) \text{ and } (s\sigma \in M) \Rightarrow P^{-1}\left[P\left(s\right)\right]\sigma \cap \overline{K} = \emptyset$$

OBSERVABILITY

Theorem: (Controllability and Observability Theorem) Consider an automaton A with uncontrollable and unobservable event sets *Euc* and *Euo*. Let P be the natural projection onto the observable events. Consider also the nonempty language $K \subseteq \mathcal{L}(A)$. There exists a nonblocking supervisor S such that

$$\mathcal{L}_m(S/A) = K \text{ and } \mathcal{L}(S/A) = \overline{K}$$

if and only if

1) K is controllable with respect to $\mathcal{L}(A)$ and E_{uc}
2) K is observable with respect to $\mathcal{L}(A)$, P, and E_c
3) K is $\mathcal{L}_m(A)$ – closed

CONTROLLABILITY AND OBSERVABILITY THM