A (very) short introduction to timed automata

Prof. Gianmaria De Tommasi
Email: detommas@unina.it

May 2021

Università degli Studi di Napoli Federico II
Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione

Outline

- **Timed Discrete Event Systems (Timed DES)** are used to describe the behaviour of a DES by means of **timed sequences.**
- Note: the dynamic of the system is still event-driven!

Timed sequence

$$s = \left\{ (e^1, \tau_1), (e^2, \tau_2), \ldots, (e^n, \tau_n), \ldots \right\}$$

where

- e^i is the i-th event
- τ_i is the time instant associated to the occurrence of the i-th event

A **clock structure** is the *object* introduced to associate an occurrence time τ to each event e in a timed sequence s

- Given the event set E the clock structure is the set

$$\Theta = \{\Theta_e \ : \ e \in E\}$$

  where each element is a **lifetime** (or *enabling delay*) set

$$\Theta_e = \{\theta_{e,1}\,,\theta_{e,2}\,,\ldots\}$$

  where $\theta_{e,k} \in \mathbb{R}_+ \cup \{0\}$ with $e \in E\,, k \in \mathbb{N}$.

- $\theta_{e,k}$ is the k-th lifetime of e and **is the time interval between the activation (enabling) of** e **and its firing (occurrence)**
- The lifetimes can be either *a priori* known (**deterministic timed DES**) or defined as *probability density functions* (**stochastic timed DES**)

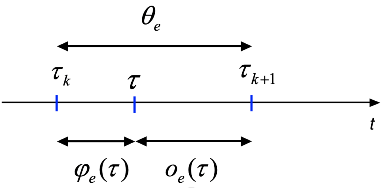
Let us consider an event $e \in E$ and let τ be a time instant such that

$$\tau_k < \tau < \tau_{k+1} ,$$

being τ_k the enabling time instant for e, and τ_{k+1} the firing time instant, i.e. the time when it will occur.

We use the following notation:

- $\varphi_e(\tau)$ is **activation time**, i.e. the time interval in which e has been enabled
- $o_e(\tau)$ is the **clock**, i.e. the residual lifetime until e will fire
- $\varphi(\tau) + o(\tau) = \theta_e \; \forall \, \tau \in (\tau_k \, , \tau_{k+1})$



Event scores (aka *activation counters*)

- Given an event $e \in E$ its **score** ν_e denotes the number of activations of e since the initial time τ_0.
- If a time τ' the state transition $x \to x'$ occurs due to the occurrence of e', then
  - $\nu'_e = \nu_e + 1$ if $[e \in \Gamma(x')] \wedge [(e = e') \vee (e \notin \Gamma(x))]$
  - $\nu'_e = \nu_e$ otherwise
- Given an event $e \in E$, the initial value of ν_e is
  - $\nu_e = 1$ if $e \in \Gamma(x_0)$
  - $\nu_e = 0$ otherwise

A timed automaton is an automaton *enriched* with a clock structure

$$G_d = (X, E, f, \Gamma, x_0, X_m, \Theta)$$

- G_d *generates* a timed sequence s according
  - to the usual transition function

    $$x' = f(x, e')$$

    where x' is the state *reached* after the occurrence of $e' \in \Gamma(x)$
  - to the current values of the clocks o_e, following the rule

    $$e' = \arg \min_{e \in \Gamma(x)} \{o_e\}$$

Clock update

Given an event $e \in E$, let o'_e be the value of the correspondent clock o_e when the state x' is reached, then

- $o'_e = o_e - o^*$ if $[e \in \Gamma(x')] \wedge [e \neq e'] \wedge [e \in \Gamma(x)]$
- $o'_e = \theta_{e,\nu_e+1}$ if $[e \in \Gamma(x')] \wedge [(e = e') \vee (e \notin \Gamma(x))]$

where

$$o^* = \min_{e \in \Gamma(x)} \{o_e\}$$

i.e. o^* denotes the elapsed time to perform the transition from x to x', and

$$e' = \arg \min_{e \in \Gamma(x)} \{o_e\}$$

- Given an event $e \in E$, the initial value for the correspondent clock o_e is
  - $o_e = \theta_{e,1}$ if $e \in \Gamma(x_0)$
  - o_e not defined, otherwise

Algorithm to compute the timed sequence generated by a timed automaton

Let us assume that G_d is in the state x at time τ

1. Compute
$$o^* = \min_{e \in \Gamma(x)} \{o_e\}$$
2. Compute the *next* event as
$$e' = \arg \min_{e \in \Gamma(x)} \{o_e\}$$
3. Compute the *next* state as
$$x' = f\left(x\,,e'\right)$$
4. Compute the time of occurrence τ' of e' as
$$\tau' = \tau + o^*$$
5. Update o'_e and ν'_e for all $e \in E$
6. **goto** Step 1

Enabling memory and total memory

- The **enabling memory** assumption implies that a clock o_e is reset to a new lifetime θ_{e,ν_e+1} every time is newly enabled (*standard* assumption)
- The **total memory** assumption implies that o_e is update until is not expired, i.e. it is NOT reset to a new lifetime θ_{e,ν_e+1} every time is newly enabled
- Update of the score (total memory case)
  - $\nu'_e = \nu_e + 1$ if $e = e'$
  - $\nu'_e = \nu_e$ otherwise (ν_e is initialized to 1 for all $e \in E$)
- Update of the clock (total memory case)
  - $o'_e = o_e - o^*$ if $[e \neq e'] \wedge [e \in \Gamma(x)]$
  - $o'_e = \theta_{e,\nu_e+1}$ if $e = e'$
  - $o'_e = o_e$ otherwise (o_e is initialized to $\theta_{e,1}$ for all $e \in E$)

If the clock structure is made of probability density functions, than the timed automaton becomes *stochastic*

Let us denote with Ψ the stochastic clock structure

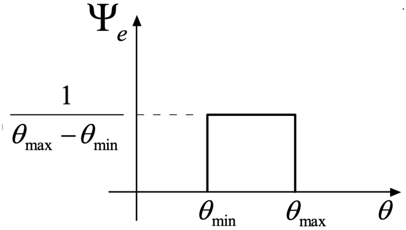
$$\Psi = \{\Psi_e \ : \ e \in E\}$$

with Ψ_e is the *pdf* defined in $\mathbb{R}_+ \cup \{0\}$ associated to $e \in E$.

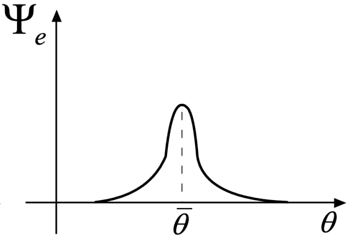
By using Ψ_e the lifetimes of e become random variables.

Hence a **stochastic automaton** is the tuple

$$G_s = (X\,, E\,, f\,, \Gamma\,, x_0\,, X_m\,, \Psi)$$



(a) Uniform pdf.



(b) Normal pdf.

Figure: Examples of pdf for Ψ_e.

The evolution of a stochastic automaton depends on the *next* event as for (deterministic) timed automata

$$e = \arg \min_{e \in \Gamma(x)} \{o_e\}$$

and both the enabling or total memory approaches can be adopted, taking into account that the lifetimes for each event $e \in E$ are generated using Ψ_e

For stochastic automata, the state trajectory $x(\cdot)$ is a random process $\{x(\tau)\}$, while, for a given time instant τ, $x(\tau)$ is a random variable

Markov processes

- A **Markov process** is a random process such that

$$\Pr\left\{x(\tau + d\tau) = x_j \mid x_{[\tau_0, \tau]}\right\} = \Pr\left\{x(\tau + d\tau) = x_j \mid x(\tau) = x_i\right\}$$

- In a Markov process
  - All past state information is irrelevant (*no state memory*)
  - How long the process has been in the current state is irrelevant (*no state age memory*)
- For a *stationary* Markov process it is

$$\Pr\left\{x(\tau + d\tau) = x_j \mid x(\tau) = x_i\right\} = \lambda_{ij} d\tau$$

- It can be proved that, for a stochastic automaton $\{x(\tau)\}$ is a *Generalized Semi-Markov Process* (GSMP).
- A **GSMP** is a random process such that

$$\Pr\left\{x(\tau + d\tau) = x_j \mid x(\tau) = x_i\right\} = \lambda_{ij}(\varphi_{e_1}, \varphi_{e_2}, \dots, \varphi_{e_n})d\tau$$

where φ_{e_k} are the lifetimes of the enabled events $e \in \Gamma(x_i)$

- If

$$\Psi_e(\theta) = \lambda_{ij} e^{-\lambda_{ij}\theta}$$

i.e., if the pdfs for the lifetimes are exponential, then a GSMP becomes a Markov process

Since an automaton as a *discrete* state space X, then Markov process induced by a stochastic automata with exponential lifetimes becomes a (discrete) **Markov chain**, in which

$$\Pr\left\{x(k+1) = x_j \mid x(k) = x_i\right\} = p_{ij}(k)\,, \quad \forall\, x_i\,, x_j \in X \text{ and } k \in \mathbb{N}$$

and where

$$\sum_{x_j \in X} p_{ij}(k) = 1\,, \forall\, x_i \in X \text{ and } k \in \mathbb{N}$$

A discrete Markov chain is defined as

$$\mathcal{MC} = (X\,, P(k)\,, \pi(0))$$

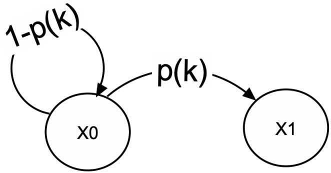
where

- X is the state space
- $P(k) = \left[p_{ij}(k)\right]\,, k \in \mathbb{N}\,, x_i\,, x_j \in X$ is the transition matrix
- $\pi(0) = \Pr\left\{x(0) = x_i\right\}\,, x_i \in X$ is the initial probability

and it is

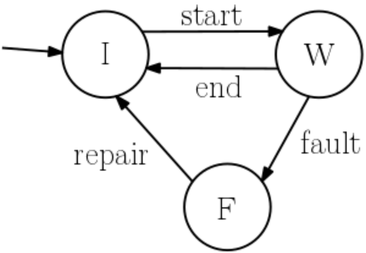
$$\pi(k+1) = \pi(k)P(k)\,, k \in \mathbb{N}$$

Similarly to automata, Markov chains can be graphically represented by a graph



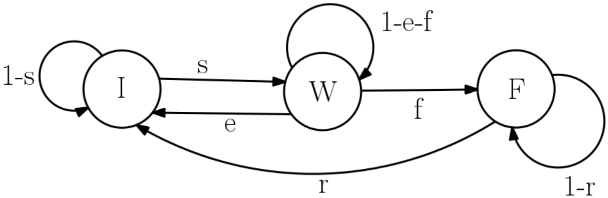
Simple manufacturing process

- $E = \{start, end, fault, repair\}$
- $X = \{I, W, F\}$
- If $\Psi = \{\Psi_{start}, \Psi_{end}, \Psi_{fault}, \Psi_{repair}\}$ are all exponential then the stochastic automata can be seen as a Markov chain



Let

$$P = \begin{bmatrix} 1-s & s & 0 \\ e & 1-e-f & f \\ r & 0 & 1-r \end{bmatrix}$$



A (very) short introduction to timed automata

Prof. Gianmaria De Tommasi
Email: detommas@unina.it

May 2021

UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE