Trace analysis using an Event-driven Interval Temporal Logic

María del Mar Gallardo, and **Laura Panizo**
University of Málaga

LOPSTR'19

October, 2019

Motivation

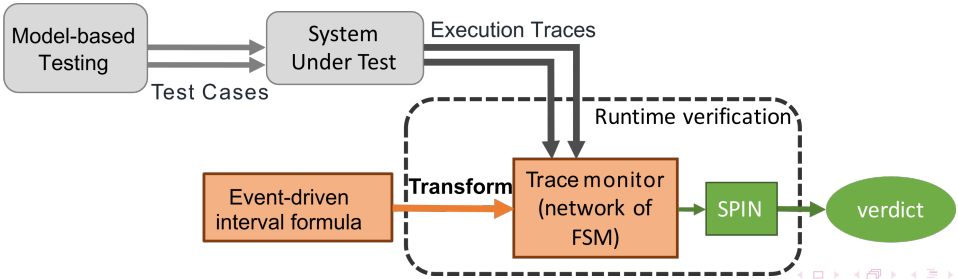
- Hybrid systems' analysis with exhaustive methods is difficult for most types of hybrid systems
- TRIANGLE project: testing of mobile apps in different network scenarios
- **Model-based testing + runtime verification of event-driven hybrid systems**

Motivation

- Hybrid systems' analysis with exhaustive methods is difficult for most types of hybrid systems
- TRIANGLE project: testing of mobile apps in different network scenarios
- **Model-based testing + runtime verification of event-driven hybrid systems**

Motivation

- Hybrid systems' analysis with exhaustive methods is difficult for most types of hybrid systems
- TRIANGLE project: testing of mobile apps in different network scenarios
- **Model-based testing + runtime verification of event-driven hybrid systems**



Objectives and approach

- Define an event-driven temporal logic suitable for runtime verification
  - Properties evaluated on time intervals determined by events that may occur during the execution trace
  - Transform the formula into a network of finite state machines to implement a trace monitoring system
  - work in progress: implementation using SPIN and SCALA

Some existing Interval logics

- Future Interval Logic (FIL) & Real-time FIL
- Duration Calculus (DC)
- Metric Interval Logic (MITL) & MITL$_{[a,b]}$
- Signal Temporal Logic (STL) & xSTL
- Differential Dynamic Logic (dL)
- HML

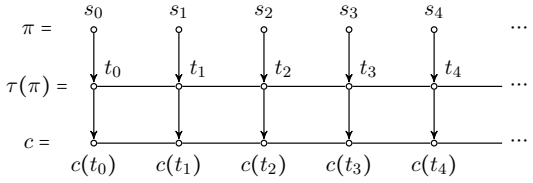
Event-driven hybrid system

The behaviour of an event-driven hybrid system is given by transition system

$$P = \langle \Sigma, \bar{\longmapsto}, L, s_0 \rangle \quad \rightarrow \quad \pi = s_0 \overset{l_0}{\longmapsto} s_1 \overset{l_1}{\longmapsto} \cdots \overset{l_{n-2}}{\longmapsto} s_{n-1}$$

Transitions take place when:

- an event arrives
- a discrete instruction is executed
- when time passes (continuous transition)



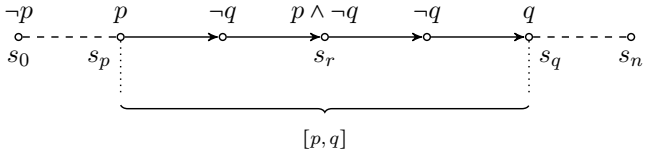
eLTL formulae

- State formula $p \in \mathcal{F}$ evaluated on single states. An event $e \in L$ is also a state formula
- Interval formula $\phi \in \Phi,\ \phi : \mathbb{I} \rightarrow \{true, false\}$ describe the expected behaviour of continuous variables
  - $[t_p, t_q] \in \mathbb{I}$ time intervals in which variables must be observed
  - Event intervals $[p, q],\ p, q \in \mathcal{F}$ determine intervals of states in the trace

Relation between time interval and event interval

A time interval $[t_p, t_q]$ satisfies an event interval $[p, q]$, denoted as $\pi \downarrow [t_p, t_q] \Vdash [p, q]$, iff:

1. $\langle \pi, t_p \rangle \vdash p$
2. $\forall t_j \in [t_p, t_q] \cap \tau(\pi), \langle \pi, t_j \rangle \nvdash q$
3. $\langle \pi, t_q \rangle \vdash q$
4. there exists no interval $[t'_p, t_q] \neq [t_p, t_q]$, verifying conditions 1–3 of this definition, such that $[t_p, t_q] \subset [t'_p, t_q]$



eLTL Syntax

Given $p, q \in \mathcal{F}$, and $\phi \in \Phi$, the formulae of eLTL logic are recursively constructed as follows:

$$\psi ::= \phi \mid \neg\psi \mid \psi_1 \vee \psi_2 \mid \psi_1 \mathcal{U}_{[p,q]} \psi_2 \mid \psi_1 \mathcal{U}_p \psi_2$$

The rest of the temporal operators are accordingly defined as:

$\Diamond_{[p,q]}\psi \equiv True\ \mathcal{U}_{[p,q]}\psi$ $\qquad$ $\Diamond_p\psi \equiv True\ \mathcal{U}_p\psi$

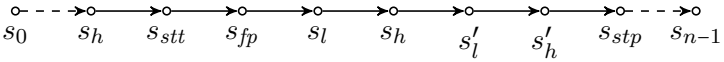
$\Box_{[p,q]}\psi \equiv \neg(\Diamond_{[p,q]}\neg\psi)$ $\qquad$ $\Box_p\psi \equiv \neg(\Diamond_p\neg\psi)$

eLTL semantics

Given $p, q \in \mathcal{F}$, $\phi \in \Phi$, and the eLTL formulae ψ, ψ_1, ψ_2, the satisfaction relation $\vDash$ is defined as follows:

$$
\begin{array}{lll}
(1)\ \langle \pi, t_i, t_f \rangle \vDash \phi & \textit{iff} & \phi([t_i, t_f]) \\
(2)\ \langle \pi, t_i, t_f \rangle \vDash \neg\psi & \textit{iff} & \langle \pi, t_i, t_f \rangle \nvDash \psi \\
(3)\ \langle \pi, t_i, t_f \rangle \vDash \psi_1 \vee \psi_2 & \textit{iff} & \langle \pi, t_i, t_f \rangle \vDash \psi_1 \ \textit{or}\ \langle \pi, t_i, t_f \rangle \vDash \psi_2 \\
(4)\ \langle \pi, t_i, t_f \rangle \vDash \psi_1 \mathcal{U}_{[p,q]} \psi_2 & \textit{iff} & \exists I = [t_p, t_q] \subseteq [t_i, t_f]\ \textit{such that} \\
 & & \pi \downarrow [t_p, t_q] \Vdash [p, q] \textit{and} \\
 & & \langle \pi, t_i, t_p \rangle \vDash \psi_1, \langle \pi, t_p, t_q \rangle \vDash \psi_2 \\
(5)\ \langle \pi, t_i, t_f \rangle \vDash \psi_1 \mathcal{U}_p \psi_2 & \textit{iff} & \exists t_p \in [t_i, t_f] \textit{and}\ \langle \pi, t_i, t_p \rangle \vDash \psi_1, \langle \pi, t_p, t_p \rangle \vDash \psi_2 \\
(6)\ \langle \pi, t_i, t_f \rangle \vDash \Diamond_{[p,q]} \psi & \textit{iff} & \exists I = [t_p, t_q] \subseteq [t_i, t_f],\ \textit{such that}\ \pi \downarrow [t_p, t_q] \Vdash [p, q] \\
 & & \textit{and} \langle \pi, t_p, t_q \rangle \vDash \psi \\
(7)\ \langle \pi, t_i, t_f \rangle \vDash \Diamond_p \psi & \textit{iff} & \exists t_p \in [t_i, t_f]\ \textit{such that}\ \langle \pi, t_p \rangle \vdash p\ \textit{and}\ \langle \pi, t_p, t_p \rangle \vDash \psi
\end{array}
$$

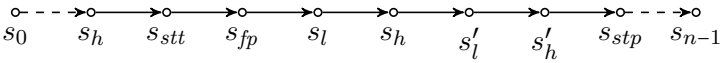
Examples: Execution trace of a video streaming app



- "During video playback the first picture must be loaded at least once"

$$\Box_{[stt,stp]} \Diamond_{fp} \mathit{True}$$

Examples: Execution trace of a video streaming app

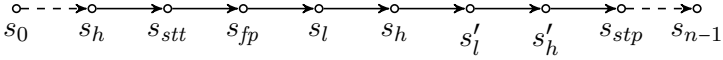


- "During video playback, if the video resolution changes from high to low, the peak signal strength is less than -45dBm"

$$\Box_{[stt,stp]}(\Box_{[h,l]}\phi)$$

$$where\ \phi([t_i,t_f]) = \begin{cases} true & if\ \exists t \in [t_i,t_f], \\ & maxRSSI(t) \leq -45dBm \\ false & otherwise \end{cases}$$

Examples: Execution trace of a video streaming app



- "The time elapsed until the video playback is less than 10 seg and the playback last more than 1 minute"

$$\phi_1 \mathcal{U}_{[stt,stp]} \phi_2$$

$$where\ \phi_1([t_i, t_f]) = \begin{cases} true & if\ \ (tf - ti) < 10seg \\ false & otherwise \end{cases}$$

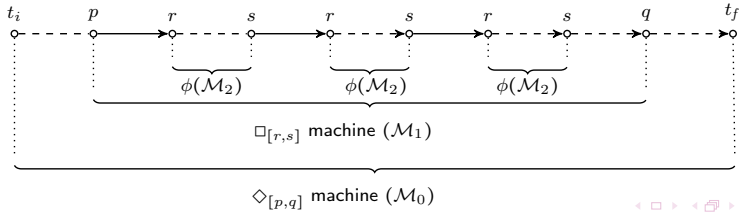
$$\phi_2([t_i, t_f]) = \begin{cases} true & if\ \ (tf - ti) > 60seg \\ false & otherwise \end{cases}$$

Network of Finite State Machines (FSMs)

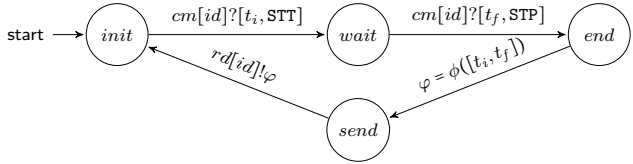
The objective is to implement a trace monitoring system. A formula/sub-formula is transformed into a FSM:

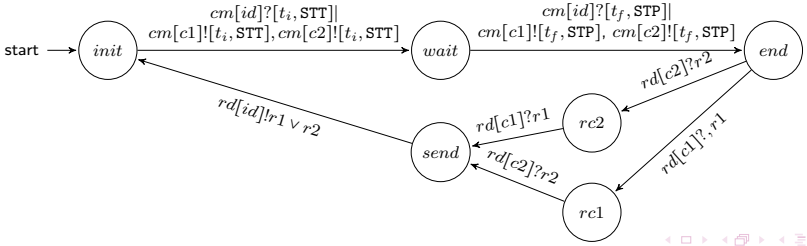
- that identify valid intervals of events in a bounded trace
- pass the time interval to its nested sub-formulas
- determines whether the formula is satisfied

$$\Diamond_{[p,q]}(\Box_{[r,s]}\phi)$$

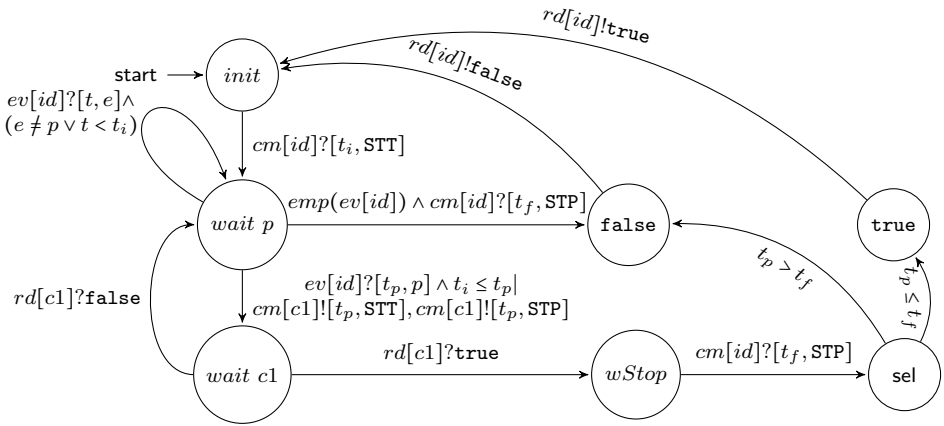


eLTL Monitor Templates: ϕ and $\vee$

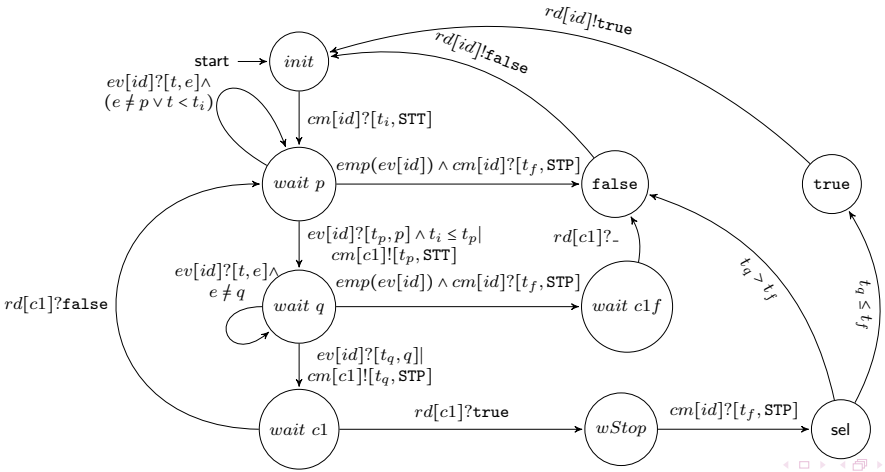




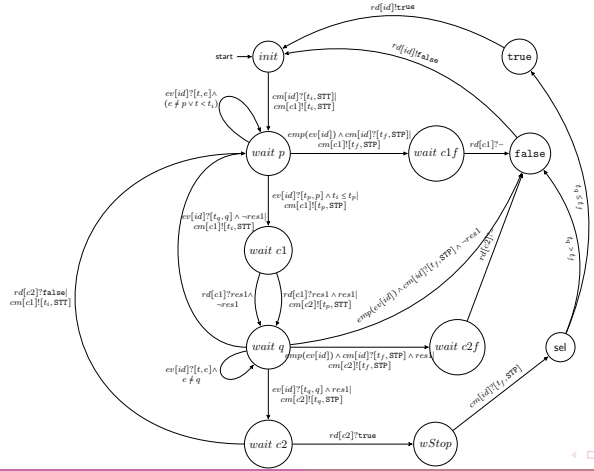
eLTL Monitor Template: $\Diamond_p$



eLTL Monitor Template: $\Diamond_{[p,q]}$



eLTL Monitor Template: $\psi_1 \mathcal{U}_{[p,q]} \psi_2$



Conclusions

- eLTL is suitable for describing properties in terms of time intervals determined by events
- eLTL can be transformed into a network of FSM to analyze execution trace

Work in progress

- Implementation with Spin + C code
- Implementation with SCALA

Future Work

- Evaluate both implementations
- Apply the logic and the trace monitoring system to other domains (e.g. network slice monitoring)

Questions?


EuWireless
European Mobile Network Operator for Research