DD2552 - Seminars on Theoretical Computer Science, Programming Languages and Formal Methods, Seminar 10

Karl Palmskog (palmskog@kth.se)

2021-09-30

Last Seminar and Today

Last seminar:

- priced timed automata
- PWCTL and statistical verification

Today:

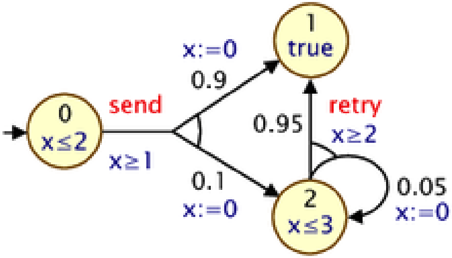
- note on tools for PTAs
- extension to more general hybrid systems

UPPAAL and timed automata

- base UPPAAL used for model checking timed systems
- extension for statistical verification (UPPAAL-SMC)
- models are XML with code declarations
- specifications are *queries* as below

```
Pr[time <= 200] (<> node1.s == 7 && node2.s == 8 )
```

PRISM and timed automata



PRISM language

```
pta
module M
    s : [0..2] init 0;
    x : clock;

    invariant
        (s=0 => x<=2) &
        (s=2 => x<=3)
    endinvariant

    [send] s=0 & x>=1 -> 0.9:(s'=1)&(x'=0) +
      0.1:(s'=2)&(x'=0);
    [retry] s=2 & x>=2 -> 0.95:(s'=1) +
      0.05:(s'=2)&(x'=0);
endmodule
```

PRISM timed automata engines

- PTAs (mostly) treated as extension of MDPs with clocks
- three engines:
  - stochastic games engine (default, no global variables)
  - digital clocks engine (only single-clock constraints)
  - backwards reachability engine (no global variables)
- PTAs must be well-formed (not checked)
- some restrictions on guards/invariants

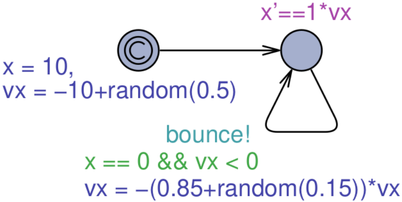
Clock rates depending on other clocks

- with regular PTAs, we can express clocks that evolve at different rates
- but clock rates can only depend on state and variables, not on clocks
- when clock rates can depend on clocks, we get ordinary differential equations (ODEs)
- we get (networks of) *stochastic hybrid automata*, SHAs

Statistical verification of SHAs

- with SMC, main problem is to generate **traces**
- for SHAs, need to solve ODEs (at least approximately)
- UPPAAL-SMC uses Euler integration method

SHA example



- `x`: x coordinate
- `vx`: uncertain derivative of x
- `bounce`: automaton output

SHA properties

- can introduce the usual probability operator on a HA logic
- example: Metric Temporal Logic (MTA)
- captures "quantitative timing properties"
- equip until operators with **intervals**, e.g., $\phi \, U_I \, \phi$
- define set of points R where we can evaluate formulas ϕ

Interval until semantics sketch

$$R, t \models \phi \, U_I \, \phi' \ \text{ iff}$$

- there exists $t' > t$ s.t. $t' - t \in I$
- $R, t' \models \phi'$
- for all t'' s.t. $t < t'' < t'$, $R, t'' \models \phi$

Statistical checking for networks of HAs

- define "closed networks" of HAs (no dangling outputs)
- define paths on closed networks
- define and validate measures on paths
- generate random traces
- analyze traces and estimate/decide

Bounding paths

can bound by:

- discrete system transitions
- clock value (Zeno issues)