Quantitative Verification

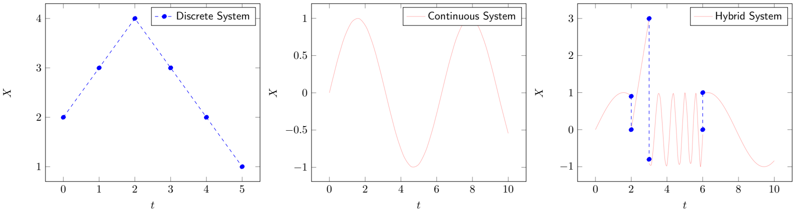
Chapter 8: Hybrid Automata

Jan Křetínský

Technical University of Munich

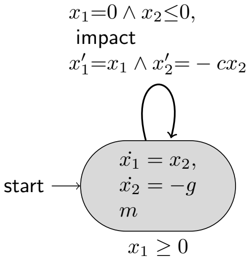
Winter 2016/17 19:14

Discrete, Continuous, and Hybrid Systems



Hybrid Automata

Hybrid Automata



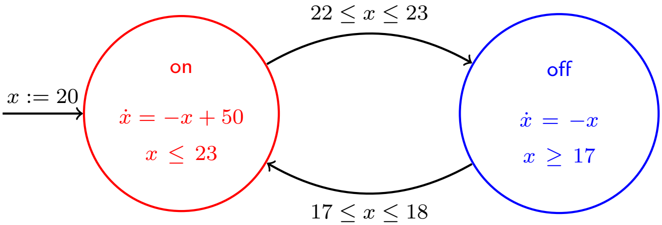
- Consider a bouncing ball system dropped from height ℓ and velocity 0.
- variables of interest : height of the ball x_1 and velocity of the ball x_2
- flow function: a system of first-order ODEs

$$\dot{x_1} = x_2 \text{ and } \dot{x_2} = -g$$

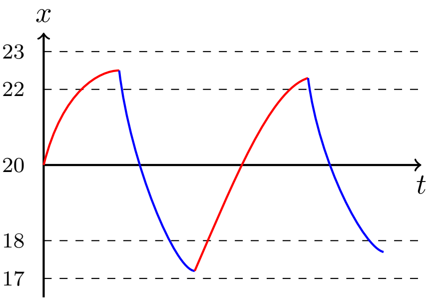
- Jump in the dynamics at impact!
- $x_1' = x_1$ and $x_2' = -cx_2$ where c is Restituition coefficient.

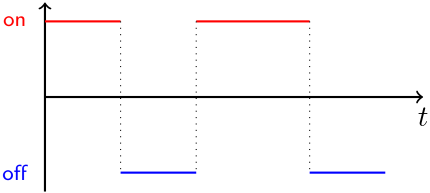
Modeling general hybrid systems: Hybrid automata

Let's take again the thermostat as an example.



23





Hybrid Automata: Syntax

Definition (HA: Syntax)

A hybrid automaton is a tuple $\mathcal{H} = (M, M_0, \Sigma, X, \Delta, I, F, V_0)$ where:

- M is a finite set of control modes including a distinguished initial set of control modes $M_0 \subseteq M$,
- Σ is a finite set of actions,
- X is a finite set of real-valued variable,
- $\Delta \subseteq M \times \mathrm{pred}(X) \times \Sigma \times \mathrm{pred}(X \cup X') \times M$ is the transition relation,
- $I : M \to \mathrm{pred}(X)$ is the mode-invariant function,
- $F : M \to \mathrm{pred}(X \cup \dot{X})$ is the mode-dependent flow function, and
- $V_0 \in \mathrm{pred}(X)$ is the set of initial valuations.

Hybrid Automata: Syntax

Definition (HA: Syntax)

A hybrid automaton is a tuple $\mathcal{H} = (M, M_0, \Sigma, X, \Delta, I, F, V_0)$ where:

- M is a finite set of control modes including a distinguished initial set of control modes $M_0 \subseteq M$,
- Σ is a finite set of actions,
- X is a finite set of real-valued variable,
- $\Delta \subseteq M \times \mathrm{pred}(X) \times \Sigma \times \mathrm{pred}(X \cup X') \times M$ is the transition relation,
- $I : M \to \mathrm{pred}(X)$ is the mode-invariant function,
- $F : M \to \mathrm{pred}(X \cup \dot{X})$ is the mode-dependent flow function, and
- $V_0 \in \mathrm{pred}(X)$ is the set of initial valuations.

- A configuration (m, ν) and a timed action (t, a)
- A transition $((m, \nu), (t, a), (m', \nu')$
  - solve flow ODE of mode m with ν as the starting state $\nu \oplus_{F(m)} t$.
  - invariant, guard, and jump conditions.
- A run or execution is a sequence of transitions

$$(m_0, \nu_0), (t_1, a_1), (m_1, \nu_1), (t_2, a_2) \ldots$$

Hybrid Automata: Semantics

Definition (HA: Semantics)

The semantics of a HA $\mathcal{H} = (M, M_0, \Sigma, X, \Delta, I, F, V_0)$ is given as a state transition graph $T^{\mathcal{H}} = (S^{\mathcal{H}}, S_0^{\mathcal{H}}, \Sigma^{\mathcal{H}}, \Delta^{\mathcal{H}})$ where

- $S^{\mathcal{H}} \subseteq (M \times \mathbb{R}^{|X|})$ is the set of configurations of $\mathcal{H}$ such that for all $(m, \nu) \in S^{\mathcal{H}}$ we have that $\nu \in [\![I(m)]\!]$;
- $S_0^{\mathcal{H}} \subseteq S^{\mathcal{H}}$ s.t. $(m, \nu) \in S_0^{\mathcal{H}}$ if $m \in M_0$ and $\nu \in V_0$;
- $\Sigma^{\mathcal{H}} = \mathbb{R}_{\geq 0} \times \Sigma$ is the set of labels;
- $\Delta^{\mathcal{H}} \subseteq S^{\mathcal{H}} \times \Sigma^{\mathcal{H}} \times S^{\mathcal{H}}$ is the set of transitions such that $((m, \nu), (t, a), (m', \nu')) \in \Delta^{\mathcal{H}}$ if there exists a transition $\delta = (m, g, a, j, m') \in \Delta$ such that
  - $(\nu \oplus_{F(m)} t) \in [\![g]\!]$;
  - $(\nu \oplus_{F(m)} \tau) \in [\![I(m)]\!]$ for all $\tau \in [0, t]$;
  - $\nu' \in (\nu \oplus_{F(m)} t)[j]$; and
  - $\nu' \in [\![I(m')]\!]$.

Hybrid systems

Continuous systems with a phased operation:

26

- bouncing ball
- walking robots
- biological cell growth and division

Continuous systems controlled by discrete logic:

- thermostat
- chemical plants with valves, pumps
- control modes for complex systems, e.g. intelligent cruise control in automobiles, aircraft autopilot modes

Coordinating processes:

- air and ground transportation systems, e.g. swarms of micro-air vehicles

HA – Reachability I

A timed automaton is a hybrid system where

- every variable is a clock,
- every jump condition is simple: comparison of variables to constants or the difference of two variables to a constant.

Reachability is decidable (PSPACE-complete) for TA.
(region construction)

HA – Reachability I

A timed automaton is a hybrid system where

- every variable is a clock,
- every jump condition is simple: comparison of variables to constants or the difference of two variables to a constant.

Reachability is decidable (PSPACE-complete) for TA.
(region construction)

A multirate timed system extends TA with variables with arbitrary constant slope.
Reachability is undecidable for 2-rate timed systems.
(counter value $n \iff$ accurate clock value $1/2^n$)

HA – Reachability II

A rectangular HA

- $x' \in [\min, \max]$
- Values of variables with different flows are never compared.
- Whenever the flow constraint of a variable changes, the variable is reset.

Reachability is decidable for rectangular HA.
Reachability is undecidable if either the second or the third constraint is violated.

HA – Reachability II

A rectangular HA

- $x' \in [\min, \max]$
- Values of variables with different flows are never compared.
- Whenever the flow constraint of a variable changes, the variable is reset.

Reachability is decidable for rectangular HA.
Reachability is undecidable if either the second or the third constraint is violated.

A linear HA

- all initial, jump and flow conditions are written using linear predicates such that variables from X and X′ never appear together in an atomic predicate, e.g., $x + 2y' \leq 7, x = x'$ not allowed, $x \leq 7 \wedge 3x' + 2y' = 8$ is ok.

Bounded reachability is decidable for linear HA.

HA – Reachability II

A rectangular HA

- $x' \in [\min, \max]$
- Values of variables with different flows are never compared.
- Whenever the flow constraint of a variable changes, the variable is reset.

Reachability is decidable for rectangular HA.
Reachability is undecidable if either the second or the third constraint is violated.

A linear HA

- all initial, jump and flow conditions are written using linear predicates such that variables from X and X′ never appear together in an atomic predicate, e.g., $x + 2y' \leq 7, x = x'$ not allowed, $x \leq 7 \wedge 3x' + 2y' = 8$ is ok.

Bounded reachability is decidable for linear HA.

We want to approximate reachable sets for general HA.

Set-Based Reachability

Extending numerical simulation from numbers to sets

- account for nondeterminism
- exhaustive
- infinite time horizon

Downsides:

- only approximate for complex dynamics
- generally not scalable in # of variables
- trade-off between runtime and accuracy

Reachability Algorithm

One-step successors by time elapse from set of states S,

$$\mathrm{Post}_C(S) = \left\{(\ell, \xi(\delta)) \;\middle|\; \exists(\ell, x) \in S : (\ell, \mathbf{x}) \xrightarrow{\delta,\xi} (\ell, \xi(\delta))\right\}.$$

One-step successors by jump from set of states S,

$$\mathrm{Post}_D(S) = \Big\{(\ell', \mathbf{x}') \;\Big|\; \exists(\ell^{\cancel{\prime}}, \mathbf{x}^{\cancel{\prime}}) \in S, \exists\alpha \in \mathrm{Lab} \cup \{\tau\} : \\ (\ell, \mathbf{x}) \xrightarrow{\alpha} (\ell', \mathbf{x}')\Big\}.$$

Reachability Algorithm

Compute sequence

$$\begin{aligned} R_0 &= \mathrm{Post}_C(\mathrm{Init}), \\ R_{i+1} &= R_i \cup \mathrm{Post}_C(\mathrm{Post}_D(R_i)). \end{aligned}$$

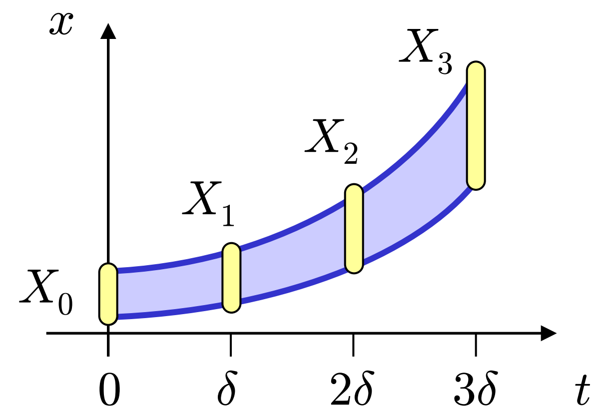
If $R_{i+1} = R_i$, then R_i = reachable states.

- may not terminate if states unbounded (counter)
- problem undecidable in general[6]

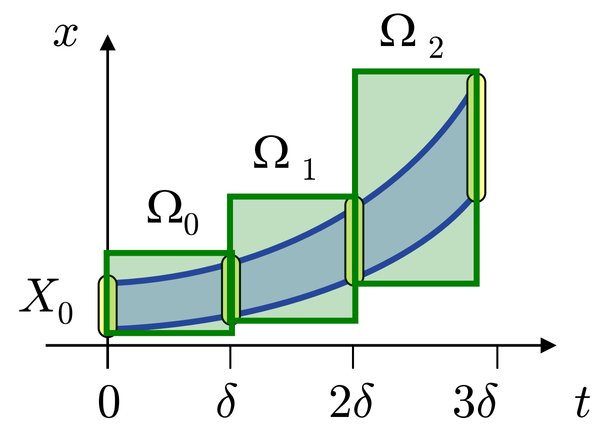
[6] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" *Journal of Computer and System Sciences*, vol. 57, pp. 94–124, 1998.

Reachability of Affine Continuous Dynamics

$$\underbrace{x(t) = e^{A\delta}x(0)}_{\text{autonomous dynamics}} + \underbrace{\int_0^{\tau} e^{A(\delta-\tau)}u(\tau)d\tau}_{\text{influence of inputs}}$$

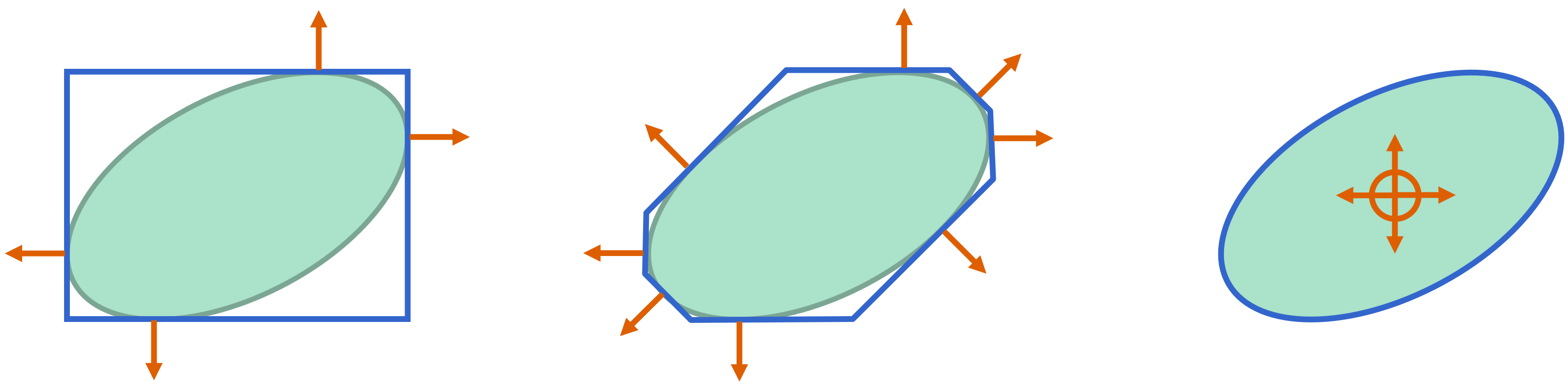


- **solution at discrete time steps**
- **cover flowpipe with convex sets Ω_i: approximation model**



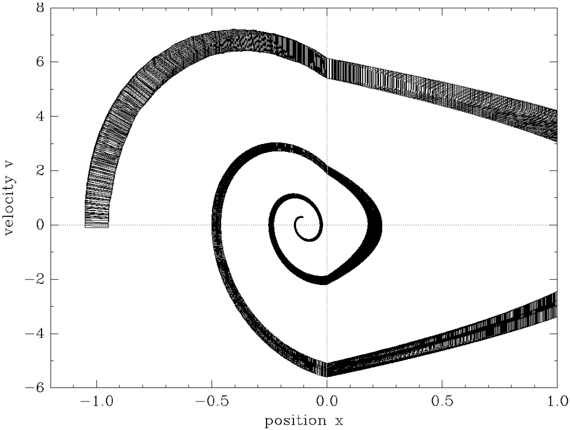
Representing of Convex Sets

- **Approximation with Supporting Halfspaces**
  - given template directions = **outer polyhedral approximation**



axis ($\pm x_i$)	octagonal ($\pm x_i \pm x_j$)	*all* directions
$\Downarrow$	$\Downarrow$	$\Downarrow$
bounding box	bounding polytope	exact set
2n facets	$2n^2$ facets	

Ball on String: Reachable States



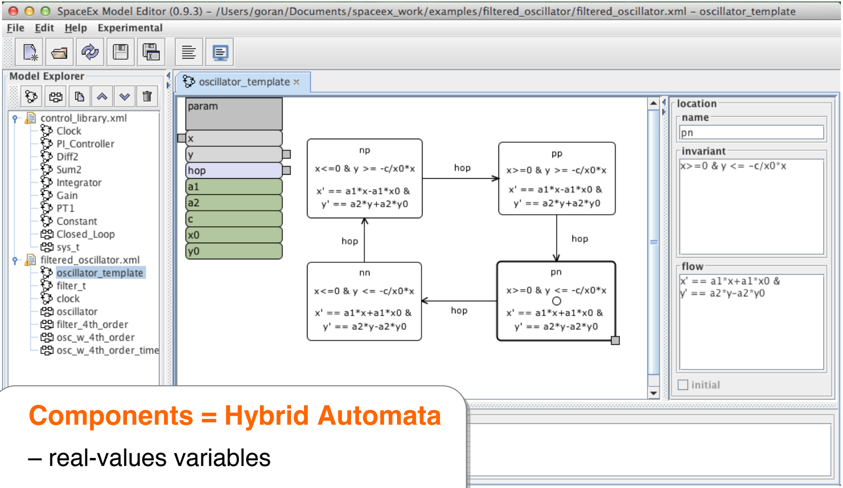
(clip from SpaceEx output)

Example: Controlled Helicopter

- **28-dim model of a Westland Lynx helicopter**
  - 8-dim model of flight dynamics
  - 20-dim continuous H∞ controller for disturbance rejection
  - stiff, highly coupled dynamics

Tools

SpaceEx Model Editor

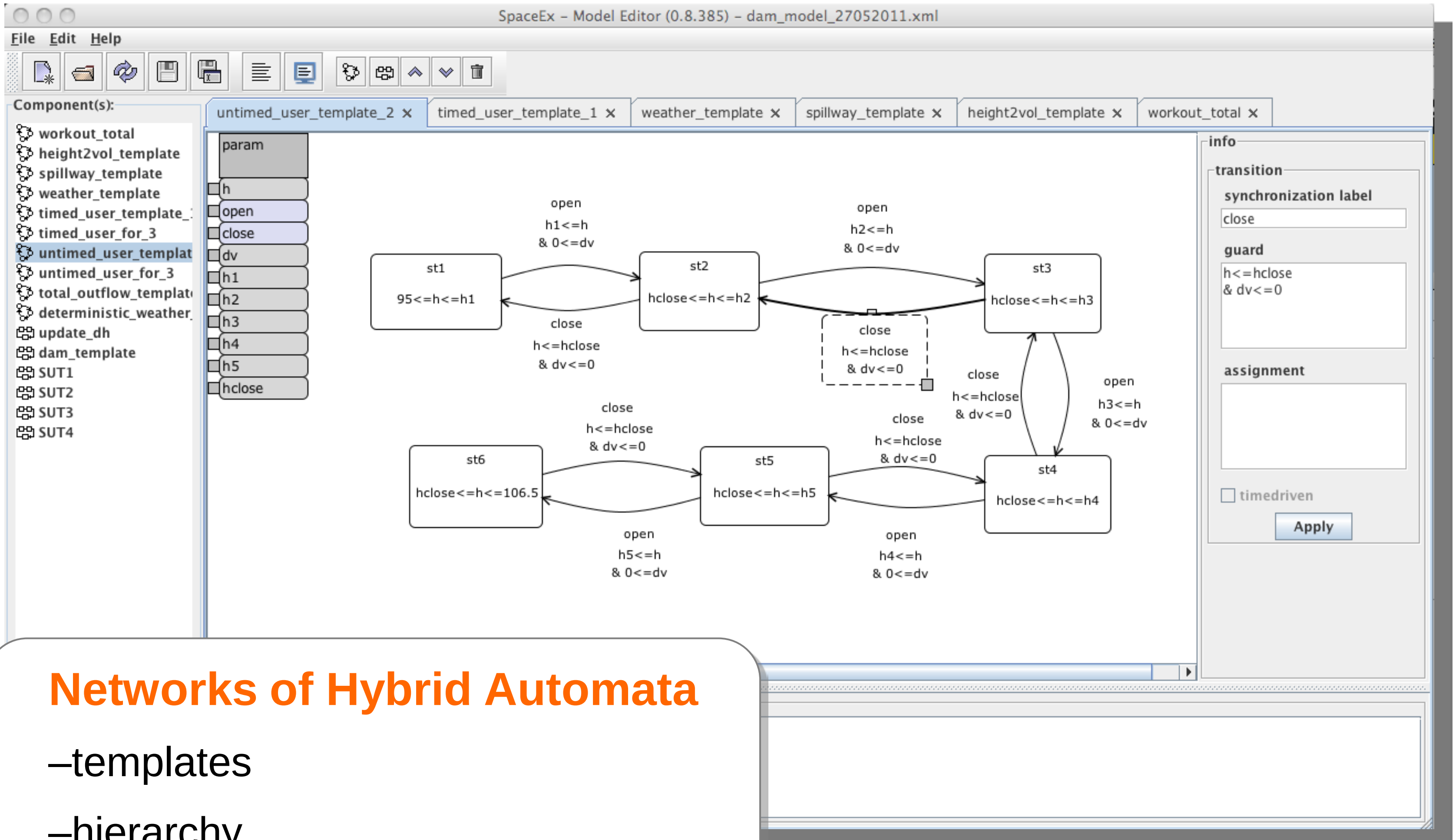


Components = Hybrid Automata

– real-values variables

– ODE, linear DAE

SpaceEx Model Editor

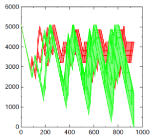


Networks of Hybrid Automata

–templates

–hierarchy

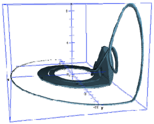
SpaceEx Reachability Algorithms



PHAVer

–constant dynamics (LHA)

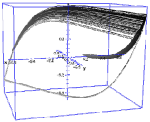
–formally sound and exact



Support Function Algo

–many continuous variables

–low discrete complexity



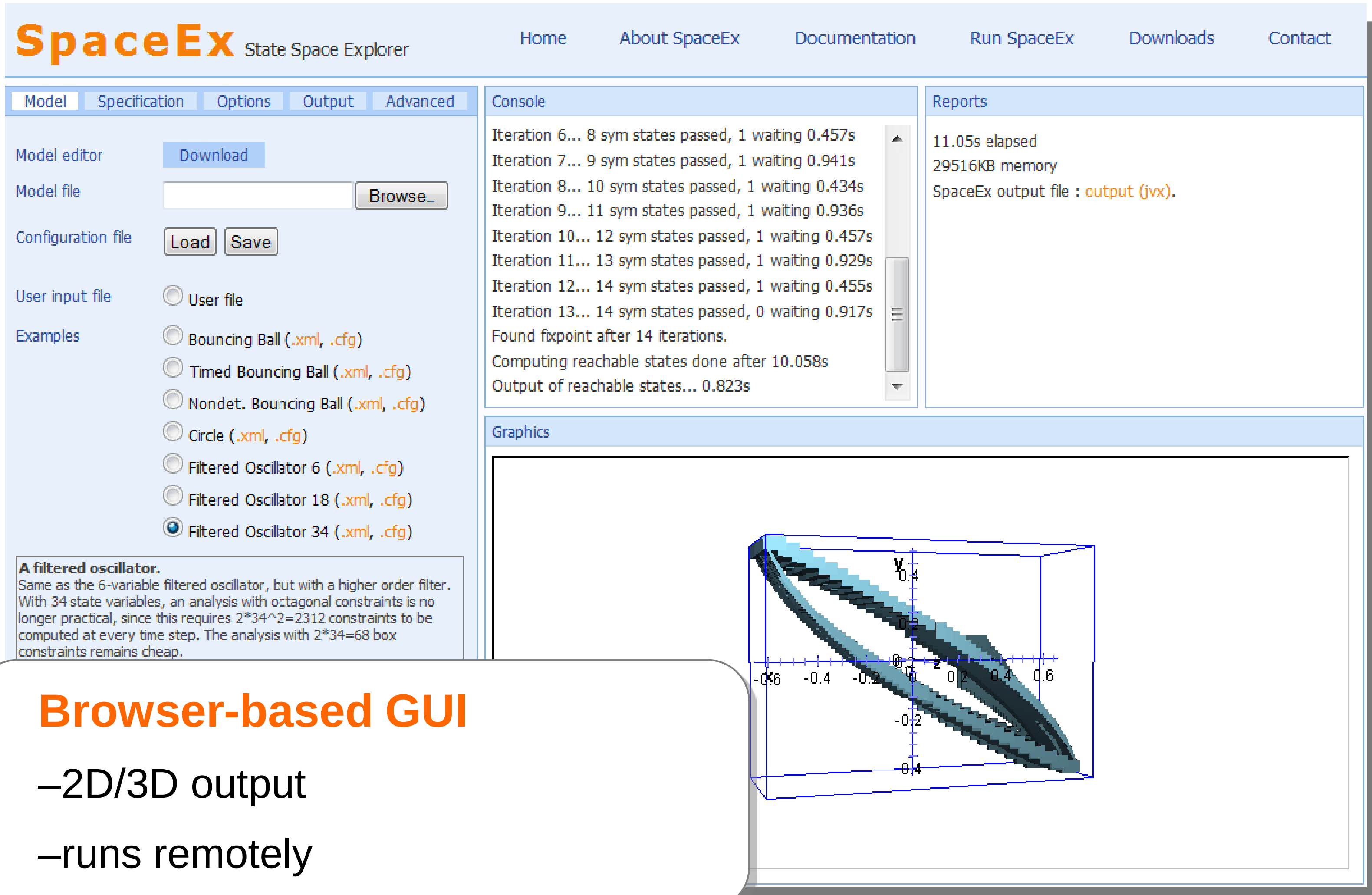
Simulation

–nonlinear dynamics

–based on CVODE

spaceex.imag.fr

SpaceEx Web Interface



Browser-based GUI

–2D/3D output

–runs remotely

Conclusions

- Hybrid systems are easy to model with **hybrid automata** but difficult to analyze.
- **Numerical simulation** scales, but is not exhaustive and critical behavior may be missed.
- **Set-based reachability** covers all runs, sufficient for safety and bounded liveness.
  - computational cost,
  - scalable for piecewise affine dynamics
- Remaining challenges: trade-off between approximation accuracy and computational cost, scalable extension to nonlinear dynamics