Algorithmic Verification

Comp4151
Lecture 12-B
Ansgar Fehnker

Outline

Model checking real-time systems

Themes
- Decidability
- Efficient implementations and data structures
- Application examples

Today
- Hybrid Systems
- Hybrid automata
- Model checking hybrid system

Recap

Modelchecking real-time systems

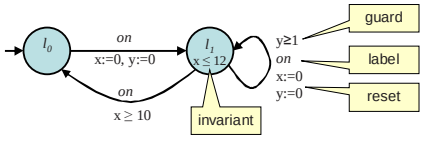
Tuesday, 24/05
- Modelling real continuous time can make sense.
- Timed Automata provide a frame work to do so

Thursday, 26/05
- Model checking timed automata is decidable
- Partitioning into region does the job

Tuesday, 31/05
- Zones do a better job
- DBMs are key

Recap

Timed automata
- A finite control graph with locations and edges
- Instantaneous transitions along edges, delays while in location
- Real-valued clocks, that increase at the same rate
- Constraints on clocks as guard on edges
- Clock resets to measure time between transitions
- Invariants in locations to enforce progress
- Labels for synchronization



Today

Towards hybrid automata

- A finite control graph with locations and edges
- Instantaneous transitions along edges, delays while in location

We keep this

- Real-valued clocks, that increase at the same rate
- Constraints on clocks as guard on edges
- Clock resets to measure time between transitions
- Invariants in locations to enforce progress

We generalize this

- Labels for synchronization

We keep this

Why?

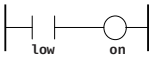
Hybrid Systems

Hybrid Systems are everywhere

- Hybrid systems are systems with a nontrivial interaction between discrete (digital) and continuous (analog) components.
- Application areas:
  - Automotive applications
  - Aerospace
  - Communication
  - Chemical engineering

Less than 1% of all microprocessors are in PCs

More than 40 of them are in your new car. (Drive-by-wire)

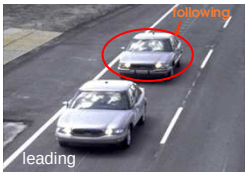


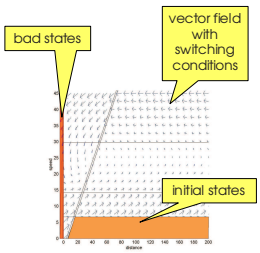
Switching Logic

Continuous System

Example: Vehicle-to-vehicle Cruise Control (V2V)

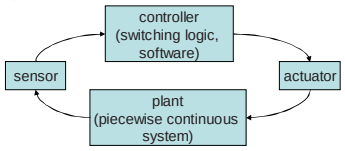
- Cruise Control Mode
  - maintain constant speed
- Following Cruise Control Mode
  - keep a safe distance
- Requirement: No collison!





Hybrid System Models

Modelling paradigm

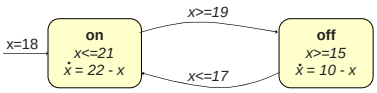


- Common modeling framework are *hybrid automata* [ACH+95]
- Combine a *finite state machine* for discrete dynamics, *flows* (differential equations) for the continuous dynamics and *jumps*.

Hybrid Automata

A set *Loc* of of locations
A set $\boldsymbol{X}$ of *n* continuous variables.
An initial location and an initial constraint on variables in $\boldsymbol{X}$
For each location an invariant *Inv(l)* and a flow $\dot{x} \in flow_l(x)$
A set of edges *E*. Each edge has

- source location and target location
- guard
- label
- update relation



Hybrid Automata

Note

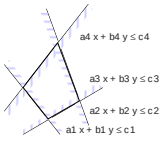
- The initial constraints, invariants and guards may be arbitrary constraints
- The update relation can be any kind of relation from $\boldsymbol{R}^n$ to $\boldsymbol{R}^n$

However

- All constraints a often assumed to be systems of linear inequalities
- The update relation is often assumed to be either affine linear

Polyhedra

- A solution set to a system of linear inequalities
- Alternatively: A intersection of half-planes

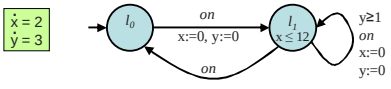


Hybrid Automata

Hybrid Automata are classified by the type of continuous flow.

Multirate Automata

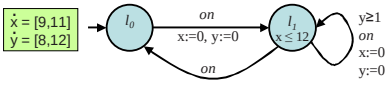
- Modest extension of timed automata
- Dynamics of the form ẋ = const (rate of a clock is same in all locations, but may be different for different clocks))
- Guards and invariants: x < const, x > const
- Resets: x := const
- Equivalent to a timed automaton. Reachability is decidable.



Hybrid Automata

Rectangular Automata

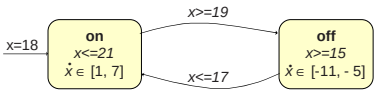
- Extension of timed automata for modelling clock drift an jitter
- Dynamics of the form x ∈ [a,b] (bound on clock-rate same in all locations)
- Guards/invariants/resets as before
- Translation to multi-rate automata that gives untimed language-equivalent system
- Reachability is decidable.



Hybrid Automata

Linear Hybrid Automata

- Linear guards and invariants (Ax <= b, matrix A , vectors x and b)
- Updates are linear transforms (x:= Ax)
- Dynamics: time-invariant, state-independent, specified by a convex polytope constraining rates
  - For example: $2 < \dot{x}$, $\dot{x} <= 3$, $\dot{x} = \dot{y}$
- Reachability is undecidable
- There exist decidable sub-classes
- Timed, multi-rate and rectangular automata are Linear HA
- All other Hybrid Automata are **Non-Linear Hybrid Automata**



Hybrid Automata

Non-linear hybrid automata

- An important sub-class class are those with
  - linear invariants and guards, and linear updates
  - dynamics specified as linear time-invariant differential equations

    $\dot{x} = Ax + B u$

    with u some (bounded) input
  - linear time-invariant systems are known in control theory as linear systems. Main source of confusion in hybrid systems.

Hybrid Automata

Non-linear hybrid automata

Decidability Results

- reachability is undecidable
- there exist some decidable sub-classes
- reachability for hybrid automata is assumed to be undecidable unless stated otherwise

People learned to live with it.

Verification

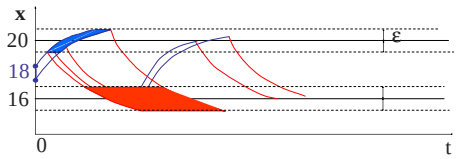
Hybrid systems verification is mostly about safety

- It easier
- And relevant

Verification

Hybrid systems verification is mostly about safety

- It easier
- And relevant

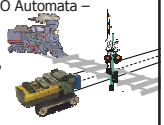


- Verification explores all behaviour
- Problem: Propagating sets of states given continuous dynamics

Verification

Mathematical Proof

- Verification of an Audio Control Protocol [BPV94]
- The Generalized Railroad Crossing - [HL95]
- Modeling and Verifying a Lego Car Using Hybrid I/O Automata – [FVZ03]



Hand proofs tend to be *tedious* and *error prone*

Computer aided verification

- Reachability Verification for Hybrid Automata - [HR98]
- Verification of Hybrid Systems: Formalization and Proof Rules in PVS – [MHS00]

However it is still tedious

Automatic Verification

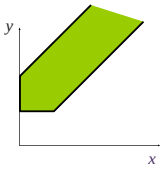
Model Checking

- Explore all possible behavior automatically
- An essential step will be:
  - Computing the successors due to continuous dynamics/flow
- We are looking for something similar to the delay in timed automata

$1 \le y \le 4$
$0 \le x \le 3$
$-2 \le x-y \le 0$

delay

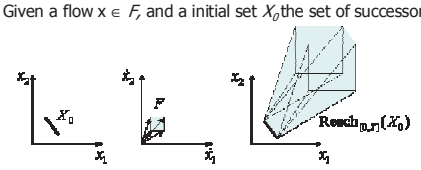
$1 \le y$
$0 \le x$
$-2 \le x-y \le 0$



Symbolic Successor

Linear Hybrid Automata

- Given a flow $x \in F$, and a initial set X_0 the set of successors

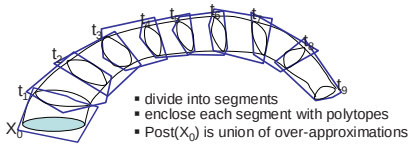


- The continuous successor can be computed exactly
- If a state is found to reachable, it is reachable
- If a state is found to be unreachable, it is unreachable
- Reachability analysis may not terminate (it's undecidable)

Symbolic Successor

Non-Linear Hybrid Automata

- For most continuous dynamics the successor can only be approximated.
- Use polyhedra to obtain an over-approximation



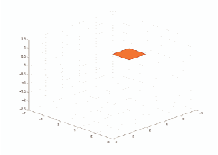
- divide into segments
- enclose each segment with polytopes
- Post(X_0) is union of over-approximations

- Used for example in the tool CheckMate

Symbolic Successor

Non-Linear Hybrid Automata

- For most continuous dynamics the successor can only be approximated.
- Use polyhedra to obtain an over-approximation

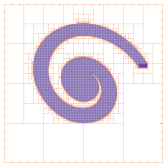


- Used for example in the tool CheckMate

Symbolic Successor

Non-Linear Hybrid Automata

- For most continuous dynamics the successor can only be approximated.
- Use orthogonal polyhedra or variable grid to obtain an over-approximation



- Used for example in the tool d/dt and Phaver

Symbolic Successor

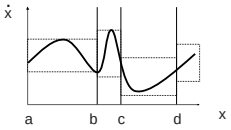
Non-Linear Hybrid Automata

- For most continuous dynamics the successor can only be approximated.
  - Bounded polyhedra
  - Orthogonal polyhedra
  - Ellipsoids
  - Oriented hyper-rectangles
  - Zonotopes
- It a conservative over approximation
- If a state is found to reachable, it may be not reachable
- If a state is found to be unreachable, it is unreachable
- Typically, despite progress, computationally very expensive
- 10 continuous variables is considered a lot

Symbolic Successors

Linear Phase-Portrait Approximation

- If a system cannot be verified due to its nasty dynamics, one may use its linear phase-portrait approximation instead
- Approximate the non-linear hybrid automaton with a linear hybrid automaton



Replace

$\dot{x} = f(x)$

By

$\dot{x} \in \begin{cases} [l_1,u_1] \text{ if } x \in [a,b] \\ [l_2,u_2] \text{ if } x \in [b,c] \\ [l_3,u_3] \text{ if } x \in [c,d] \end{cases}$

Model Checking

We have

- Procedures to compute the set of continuous successor.

We want

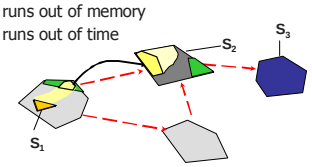
- A procedure for model checking or reachability analysis.

Model Checking

Forward Reachability

Start with the initial set of states, and compute the successor, until either

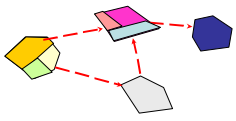
- the procedure reaches a fix point (system is safe)
- the procedure finds that a unsafe location is reachable (system may be unsafe)
- it runs out of memory
- it runs out of time



Model Checking

Finite bisimulation

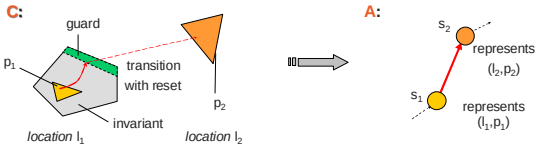
- Compute/find a finite partition of the state space
- States in the same partition element exhibit the same behavior (like region automaton)
- Compute symbolic successors to check if it is a bisimulation
- Refine if necessary
- Problems
  - Might be expensive to compute
  - Bisimulation might not exist



Model Checking

Finite abstraction

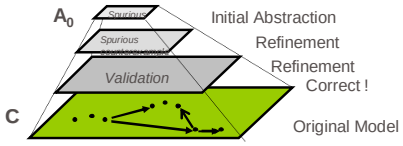
- Find/compute a finite abstraction A of hybrid systems C
- Each partition element is represented by an abstract state
- There exists a transition in A if there exists a corresponding transition in C
- May contain spurious behavior, i.e. transitions in A with no corresponding transition in C



Model Checking

Finite abstraction

- An abstraction does exist
- Abstractions may be too coarse, too much spurious behaviour.
- Use abstraction refinement to find a suitable abstraction
- Example: Counterexample guided abstraction refinement



- It may not terminate (It is undecidable after all)

Misc

Things that should be mentioned

- Recently bounded model checking for linear hybrid automata using SMT (Satisfiability modulo theory) made its entry
- Common tools are
  - HyTech
  - Checkmate
  - Charon
  - d/dt
  - Verishift
  - Phaver
  - Hysdel
- There is a hybrid systems Wiki
  http://wiki.grasp.upenn.edu/~graspdoc/hst/index.php

Recap

Model checking real-time systems

Tuesday, 23/05
- Modelling real continuous time can make sense.
- Timed Automata provide a frame work to do so

Thursday, 25/05
- Model checking timed automata is decidable
- Partitioning into region does the job

Tuesday, 30/05
- Zones do a better job
- DBMs are key

Today
- Hybrid systems verification is not a sinecure
- Drive carefully, keep distance