Algorithmic Verification

Comp4151
Lecture 11-B
Ansgar Fehnker

Overview

Model Checking Approaches

- Explicit State Model Checking
- Symbolic Model Checking
- Bounded Model Checking
- Automatic Abstraction Refinement

- Correctness of software, hardware and protocols
- Correctness for finite state systems

Overview

Next two weeks

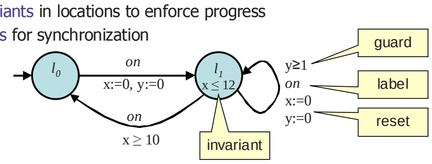
Model checking real-time systems

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

Today
- Decidability and region equivalence
- Symbolic model checking for the region automaton
- Other decidability results

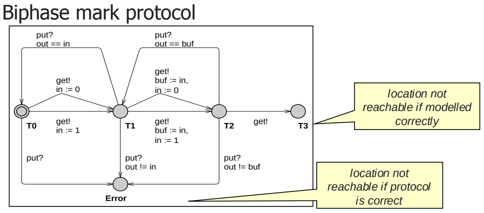
Recap

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



Recap

Biphase mark protocol



- Protocol and model correct if certain locations are not reachable
- Problems
  - The state space is infinite: $S = \{ (l,v) \mid l \in Loc,\ v \models Inv(l),\ v: C \to \mathbb{R}_{\geq 0} \}$
  - The transition relation is infinite: $R \subseteq S \times \Sigma \cup \mathbb{R}_{\geq 0} \times S$

Introduction

Region Automaton

- Proposed by Alur and Dill [AD94,AD91]
- Provides a finitie abstraction
- Used for many other decidability results

Reachability

- Check if a given location in a given TA is reachable from the initial state

Decidability

- Does there exist an algorithm that decides for any TA *A* and a location *l*, if *l* is reachable in *A* or not.

Preliminaries

Constraints

Given a set of Clocks C let $\Psi(C)$ be defined by

$$\varphi := \varphi \wedge \varphi \mid \neg \varphi \mid x \leq n \mid x < n \mid x - y \leq m \mid x - y < m$$

where $x, y \in C$, $n \in \mathbb{N}$, $m \in \mathbb{Z}$

comparison with integers

no diagonal constraints

Finite control graph

Timed automata $(Loc, l_0, \Sigma, E, Inv)$ has A finite set of locations *Loc* and a finite set of edges *E*.

Infinite transition system

The underlying timed transition system (S, s_0, R) has an infinite set of states and an infinite number of transitions

Approach

Given a TA $(Loc, l_0, \Sigma, E, Inv)$ with underlying TTS (S, s_0, R)

- Define an equivalence relation $\approx$ on clock valuations such that
  - Given states $(l,v), (l',v')$ of TTS with $(l,v) \approx (l',v')$ we have *location l_f is reachable from (l,v) iff l_f is reachable from (l',v')*
  - The number of equivalence classes $S/\approx$ is finite

Problems

- How to define such an equivalence relation? — main problem
- How to represent an equivalence class? — secondary problem

Region Equivalence

First observation

- All clocks are compared only to integer values

Equivalence

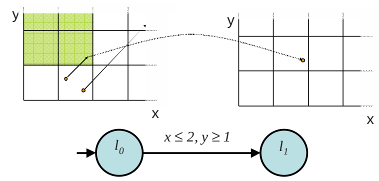
- Define a integer grid on clock valuations

$(l,v) \approx (l',v')$ iff $l = l$ and $\forall x \in C.\ \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ (?)

Comp4151 Ansgar Fehnker

Region Equivalence

Example



Requirement

$(l,v) \approx (l',v')$ iff

location l_f is reachable from $(l,v) \Leftrightarrow l_f$ is reachable from (l',v') ✗

Comp4151 Ansgar Fehnker

Region Equivalence

Second observation

- Need to distinguish between valuations above and below diagonals

Equivalence

- Define a integer grid on clock valuations
- Divide each cell along its diagonals
- Diagonals satisfy frac(v(x))=frac(v(y)))

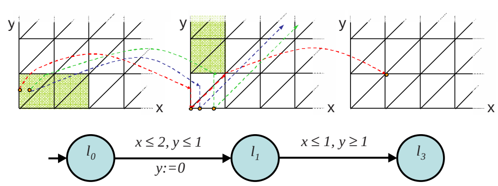
$(l,v) \approx (l',v')$ iff

- $l = l'$ and $\forall x \in C.\ \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$
- $\forall x,y \in C.$ frac(v(x)) ≤ frac(v(y)) ⇔ frac(v′(x)) ≤ frac(v′(y)) (?)

Comp4151 Ansgar Fehnker

Region Equivalence

Example



Requirement

$(l,v) \approx (l',v')$ iff

location l_f is reachable from $(l,v) \Leftrightarrow l_f$ is reachable from (l',v') ✗

Comp4151 Ansgar Fehnker

Region Equivalence

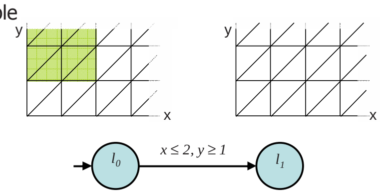
Third observation
- It matters whether the value of a clock is an integer

Equivalence
- Define a integer grid on clock valuations
- Divide each cell along its diagonals frac(v(x))=frac(v(y))
- Divide the cells into vertices, edges, diagonals, and open simplices

$(l,v) \approx (l',v')$ iff
- $l = l'$ and $\forall x \in C.\ \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$
- $\forall x,y \in C.$ frac(v(x)) ≤ frac(v(y)) ⇔ frac(v′(x)) ≤ frac(v′(y))
- $\forall x \in C.$ frac(v(x)) = 0 ⇔ frac(v′(x)) =0 (?)

Region Equivalence

Example



Requirement

(l,v) ≈ (l',v') iff

location l_f is reachable from (l,v) ⇔ l_f is reachable from (l',v')

There are countable but infinitely many equivalence classes.

Region Equivalence

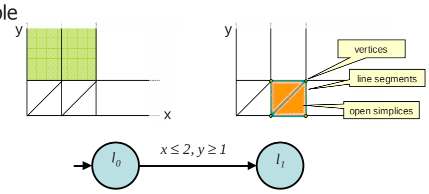
Forth observation
- The value of a clock is irrelevant once it exceeds the biggest constant

Equivalence
- Define a integer grid on clock valuations
- Divide each cell along its diagonals frac(v(x))=frac(v(y))
- Divide the cells into vertices, edges, diagonals, and open simplices
- Bound the partition using the biggest constant in guards and invariants

$(l,v) \approx (l',v')$ iff
- $l = l'$ and $\forall x \in C.$ $\mathbf{v(x) \le c_x} \Rightarrow \lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$
- $\forall x,y \in C.$ $\mathbf{v(x) \le c_x \wedge v(y) \le c_y} \Rightarrow$ (frac(v(x))≤frac(v(y)) ⇔ frac(v′(x))≤frac(v′(y)))
- $\forall x \in C.$ $\mathbf{v(x) \le c_x} \Rightarrow$ (frac(v(x)) = 0 ⇔ frac(v′(x)) =0)

Region Equivalence

Example



Requirement
- *(l,v) ≈ (l',v') iff l_f is reachable from (l,v) ⇔ l_f is reachable from (l',v')*
- *There are **finitely** many equivalence classes.*
- ***Reachability** for timed automata is **decidable**.*

Overview

Next two weeks

Model checking real-time systems

Themes

- Decidability
- Efficient implementations and data structures
- Application examples

Today

- Decidability and region equivalence
- Symbolic model checking for the region automaton
- Other decidability results

Symbolic Semantics

Use clock equivalence to define a finite region automaton.
1st step: represent regions symbolically:

Given a set of clocks C, with maximal constants c_x, we represent a region as a triple $H=(h,[C_0,\ldots,C_k], C_>)$ with

- $h: C \to$ Nat that assigns to each clock x a natural number $\leq c_x$,
- $C_0,\ldots,C_k$ and $C_>$ define a partition of the set of clocks.
- C_0 and $C_>$ may be empty.

Let $\mathcal{H}$ be the finite set of all possible H given the set of clocks and the maximal constants.

Symbolic Semantics

A clock valuation $v \in (h,[C_0,\ldots,C_k], C_>)$ if

- $\lfloor v(x) \rfloor = h(x)$ for $x \notin C_>$
- $\lfloor v(x) \rfloor > c_x$ for $x \in C_>$
- $frac(v(x)) = 0$ for $x \in C_0$
- $frac(v(x)) = frac(v(y))$ for $x,y \in C_i$
- $frac(v(x)) < frac(v(y))$ for $x \in C_i$, $y \in C_j$ $i<j$

Equivalent clock valuations

- $v,v' \in (h,[C_0,\ldots,C_k], C_>)$ implies $(l,v) \approx (l,v')$

Symbolic Semantics

Use clock equivalence to define a finite region automaton.
2st step: define symbolic operations on regions

Reset

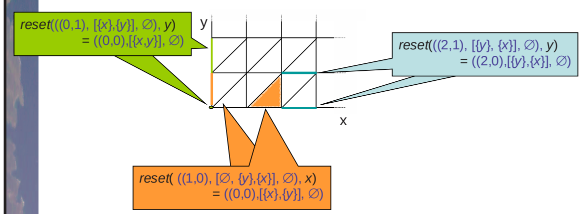
Given a region H $= (h,[C_0,\ldots,C_k], C_>)$ and $x \in C_i$

- if i=0 then $reset(H,x) = (h',[C_0,\ldots,C_k], C_>)$ with $h' = h[x:=0]$
- if $0 < i$ and $C_i = \{x\}$ then $reset(H,x) = (h',[C'_0,\ldots, C_{i-1}, C_{i+1},\ldots,C_k], C_>)$ with $h' = h[x:=0]$, $C_0' = C_0 \cup \{x\}$
- otherwise $reset(H,x) = (h',[C'_0,\ldots,C'_i,\ldots,C_k]\ C_>)$ with $h' = h[x:=0]$, $C_i' = C_i \mid \{x\}$, and $C_0' = C_0 \cup \{x\}$

Given a region H $= (h,[C_0,\ldots,C_k], C_>)$ and $x \in C_>$

- $reset(H,x) = (h',[C'_0,\ldots,C_k], C'_>)$ with $h' = h[x:=0]$, $C'_> = C_> \mid \{x\}$, and $C'_0 = C_0 \cup \{x\}$

Symbolic Sematics



Symbolic Semantics

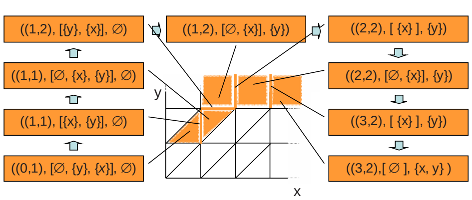
Use clock equivalence to define a finite region automaton.
2st step: define symbolic operations on regions

Delay

Given a region H $= (h, [C_0, \ldots, C_k], C_>)$

- if $C_0 \neq \emptyset$ then
  $delay(H) = (h, [\emptyset, C'_0, C_1, \ldots, C_k], C_>')$ with
  $C_0' = C_0 \setminus \{x \mid h(x) = c_x\}$ and $C_>' = C_> \cup \{x \mid h(x) = c_x\}$
- if $C_0 = \emptyset$ and k>0 then
  $delay(H) = (h', [C_k, C_0, \ldots, C_{k-1}], C_>)$ with
  $h'(x) = h(x)+1$ if $x \in C_k$ and $h'(x) = h(x)$ otherwise.
- otherwise (if all clocks in $C_>$)
  $delay(H) = H$

Symbolic Semantics



Symbolic Semantics

Reminder

The operational *semantics* of a timed automaton $A = (Loc, l_0, \Sigma, E, Inv)$ is given as a transition system $TS(A)$ with

- set of states $S = \{ (l,v) \mid l \in Loc,\ v \models Inv(l) \}$
- initial state $s_0 = (l_0, \mathbf{0})$
- transition relation $R \subseteq S \times \Sigma \cup \mathbf{R}_{\geq 0} \times S$ that contains the following

discrete transitions

$(l,v) \xrightarrow{\sigma} (l',v')$ if there exist $(l,g,\sigma,r,l') \in E$ s.t. $v \models g$, and $v[r:=0] = v'$

delay transitions

$(l,v) \xrightarrow{d} (l, v+d)$ for $d \in \mathbf{R}_{\geq 0}$ if for all $0 \leq d' \leq d$ holds $v + d \models Inv(l)$

Infinitely many states and transitions!

Symbolic Semantics

Definition

The *region semantics* of a timed automaton A= $(Loc, l_0, \Sigma, E, Inv)$ is given as a transition system *RA*(*A*) with

- set of states $S = \{ (l, H) \mid l \in Loc,\ H \in \mathcal{H} \}$
- initial state $s_0 = (l_0, (\mathbf{0}, [C], \emptyset))$
- transition relation $R \subseteq S \times \Sigma \cup \{\delta\} \times S$ that contains the following

discrete transitions

$(l,H) \xrightarrow{\sigma} (l',H')$ if $\exists\ (l,g,\sigma,r,l') \in E$ s.t. $H \models g$, $H' \models Inv(l')$ and $H'=reset(H,r)$

delay transitions

$(l,H) \xrightarrow{\delta} (l,H')$ if $H' \models Inv(l)$ and $H'=delay(H)$

Finitely many states and transitions!

Symbolic Semantics

Useful theorem

Given a location *l* of timed automaton *A*, it is reachable in *TS(A)* iff it is reachable in *RA(A).*

Sketch of proof

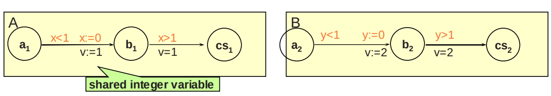
"=>"

Given an execution $(l_0,v_0)\ (\xrightarrow{d} \cup \xrightarrow{\sigma})^* (l,v)$ there exist a symbolic execution $(l_0,H_0)\ (\xrightarrow{\delta} \cup \xrightarrow{\sigma})^* (l,H)$ with $v \in H$

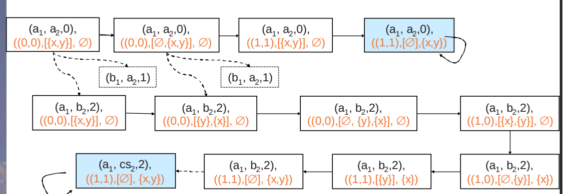
"<="

Given a symbolic execution $(l_0,H_0)\ (\xrightarrow{\varepsilon} \cup \xrightarrow{\sigma})^* (l,H)$ there exist execution $(l_0,v_0)\ (\xrightarrow{\delta} \cup \xrightarrow{\sigma})^* (l,v)$ with $v \in H$

Example



Show that A || B can not reach the critical section in location (cs_1,cs_2)



Model Checking

Forward reachability

- Start with the initial state $(l_0,(\mathbf{0},C,\emptyset))$ of the region automaton
- Explore the state space using the transition relation until either
  - A fix-point has been reached, or
  - The target location *l* has been reached.
- Search orders are DFS, BFS, random DFS, ….

Backward reachability

- Start with all regions in the target location
- Explore the state space using the inverse transition relation until either
  - A fix-point has been reached
  - The initial state $(l_0,(\mathbf{0},C,\emptyset))$ has been reached

Decidability for Timed Automata

Other positive results

- TCTL model checking for timed automata is decidable
  - $\phi ::= p \mid \alpha \mid \neg \phi \mid \phi \vee \phi \mid z \text{ in } \phi \mid \mathbf{A}[\phi\, \mathbf{U}\, \phi] \mid E[\phi\, \mathbf{U}\, \phi]$
- Emptiness of untimed language is decidable
  - Is the language accepted by an TA empty? (reachability, Buechi-like acceptance)
- Un-timed language inclusion
- Timed bisimulation is decidable
  - Two TAs are bismilar iff they perform the same actions in bisimilar states they reach bisimilar states.
- Untimed bisimulation is decidable

Decidability for Timed Automata

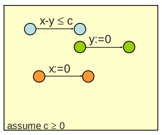
Negative Results

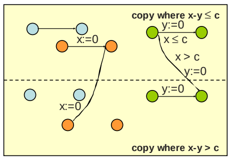
- The universality problem is undecidable.
  - Does an TA accept all timed words?
- Timed language inclusion is undecidable.
- Timed automata are not determinzable nor complementable
- The following leads to undecidability:
  - Decrementing clocks
  - Incrementing clocks
  - Linear expressions as guards
  - Guards that compare clocks with irrational constants
  - Stop-watches (i.e. clocks that can have rates 0 or 1)
- However there are subclasses of TA such that make of these problems decidable.

Diagonal Constraints

Another useful theorem (almost forgotten)

A timed automaton with diagonal constraints is timed bisimilar to an TA without diagonal constraints





[Berard,Diekert,Gastin,Petit 1998]

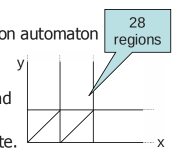
Summary

Results

- The reachability problem for timed automata is decidable
- Finite symbolic semantics for the region automaton
- The region-construction useful to prove decidability of other problems.

However

- Reachability is linear in the size of the region automaton
- The size of the region automaton is
  - linear in the number of locations,
  - exponential in the number of clocks, and
  - exponential in the maximal constants.
- The reachability problem is Pspace complete.



Next week: Efficient model checking of timed automata