Model Checking Timed Systems using Clock Constraints (1)

Reachability Analysis and Constraint solving

Problem: reachability analysis
- Give an automaton and a location n, or a local property F
- Question: does it exist an execution of the automaton, that leads to n (or a state where F holds)?
- This is the so called reachability problem.

Formalizing requirements
- Reachability properties: E<> Q
- E<> P.stop
- E<> (y>200)
- Invariant properties: A[] Q (not E<> not Q)
- A[] not (P1.cs and P2.cs)
- A[] (i < 100)
- A[] (i>10 imply i>100)
- After 10:00AM, i should be larger than 100
- Bounded Liveness Properties: F1 → F2
  A[](F1 and i<>10 imply F2)

Other Verification Problems
- Timed Language Inclusion ☺
- Untimed Language Inclusion ☺
- (Un)Timed Bisimulation ☺
- Reachability Analysis ☺
- Optimal Reachability (synthesis problem) ☺
  - If a location is reachable, what is the minimal delay before reaching the location?

Infinite State Space!

Region: From infinite to finite

However, the reachability problem is decidable ☺ Alur&Dill 1991
Region equivalence (Intuition)

\[ u \approx v \iff \text{u and v satisfy exactly the same set of constraints in the form of } \quad \]
\[ x_i \sim m \quad \text{and} \quad x_i - x_j \sim n \quad \text{where } \sim \in \langle <, \leq, \geq \rangle \]
\[ \text{and } m, n < \text{MAX} \]

This is not quite correct; we need to consider the MAX more carefully.

Region equivalence:

Definition \cite{Alur and Dill 1990}

- \( u, v \) are clock assignments
- \( u \approx v \iff \)
  - For all clocks \( x_i \), either \( u(x_i) \leq C_x \) and \( v(x_i) \leq C_x \)
  - or \( \lceil u(x_i) \rceil = \lceil v(x_i) \rceil \) (the same integer part)
  - For all clocks \( x_i \), if \( u(x_i) \leq C_x \), \( u(x_i) = 0 \iff v(x_i) = 0 \)
  - For all clocks \( x, y \), if \( u(x_i) \leq C_x \) and \( u(y_i) \leq C_y \) \( (u(x_i) \leq (u(y_i)) \iff (v(x_i)) \leq (v(y_i))) \)

Regions

Finite partitioning of state space

OBS: there are only Finite many regions

An Important Theorem for Region Equivalence

- \( u \approx v \) implies
  - \( u(x=0) \approx v(x=0) \)
  - \( u+n \approx v+n \) for all natural number \( n \)
  - for all \( d<1 \):
    - \( u+d \approx v+d \) for some \( d'<1 \)
- that is, “region equivalence” is preserved by “addition” and reset
- in fact, it is also preserved by “subtraction” if clock values are “bounded”

Region graph of a simple timed automata

Fischers again

AQ(\( CS_1 \land CS_2 \))
Problems with Region Partitioning

- Too many ‘regions’
- Sensitive to the maximal constants
  - e.g. \( x > 1000000 \)
- The number of regions is highly exponential in the number of clocks and the maximal constants (used to compare with clocks)

The more efficient solution [1989 Dill ... 1994]

Symbolic Reachability
Using Clock Constraints

Zones: From infinite to finite

State: \((n, x=3.2, y=2.5)\)
Symbolic state (zone): \((n, 10x<4, 10y<3)\)

Zones: conjunction of

Fischer’s Protocol

analysis using zones

Initially \(Y=1\)

Critical Section

Fischers cont.

Untimed Case

Taking time into account
Fischers cont.

Untimed case

Taking time into account

Symbolic Transitions

delays to

\[ x > 3 \]

\[ y := 0 \]

conjuncts to

\[ 1 \leq x \leq 4 \]

\[ 1 \leq y \leq 3 \]

projects to

\[ 3 < x, 1 \leq y - 2 \leq x - y \leq 3 \]

\[ x = 0 \]

Thus: \( (n,1 \leq x \leq 4,1 \leq y \leq 3) \Rightarrow (m,3 < x, y = 0) \)

Zones = Conjunctive constraints

- A zone \( Z \) is a conjunctive formula:
  \[ g_1 \land g_2 \land \ldots \land g_n \]
  where \( g_i \) is a clock constraint:
  \[ x_i \sim b_i \text{ or } x_i - x_j \sim b_{ij} \]
- Use a zero clock \( x_0 \) (constant 0)
- A zone can be rewritten as a set:
  \[ \{x : x_i \sim b_i \mid i \leq n, (i,j) \in S\} \]
- This can be represented as a MATRIX, DBM
  (Difference Bound Matrices)
Solution set as semantics

Let $Z$ be a zone (a set of constraints)

Let $[Z] = \{ u \mid u \text{ is a solution of } Z \}$

The semantics

(We shall simply write $Z$ instead $[Z]$)

Operations on Zones

- Strongest post condition (Delay): $SP(Z) \text{ or } Z^\uparrow$
  
  $[Z]^\uparrow = \{ u + d \mid d \in R, u \in [Z] \}$

- Weakest pre condition: $WP(Z) \text{ or } Z^\downarrow$ (the dual of $Z^\uparrow$)
  
  $[Z]^\downarrow = \{ u \mid u + d \in [Z] \text{ for some } d \in R \}$

- Reset: $(x)Z \text{ or } Z(x := 0)$
  
  $[(x)Z] = \{ u[0/x] \mid u \in [Z] \}$

- Conjunction
  
  $[Z & g] = [Z] \cap [g]$

An important theorem on Zones

- The set of zones is closed under all constraint operations (including $x := x - c$ or $x := x + c$)
  
  That is, the result of the operations on a zone is a zone

  That is, there will be a zone (a finite object i.e a zone/constraints) to represent the sets: $[Z]^\uparrow$, $[Z]^\downarrow$, $[(x)Z]$

One-step reachability: $S_i \rightarrow S_j$

- Delay: $(n, Z) \rightarrow (n, Z') \text{ where } Z' = Z^\uparrow \land \text{inv}(n)$

- Action: $(n, Z) \rightarrow (m, Z') \text{ where } Z' = \{ x \} (Z \land g)$
  
  if $n \in g \land x := 0 \land m$

- Successors$(n, Z) = \{ (m, Z') \mid (n, Z) \rightarrow (m, Z'), Z' = \emptyset \}$
  
  Sometime we write: $(n, Z) \rightarrow (m, Z') \text{ if } (m, Z) \text{ is a successor of } (n, Z)$

Now, we have a search problem

Forward Reachability

- Initial: $\text{Passed} := \emptyset$; $\text{Waiting} := \{(n_0, Z_0)\}$

- Repeat
  
  - pick $(n, Z)$ in $\text{Waiting}$
  
  - if for some $Z' \supseteq Z$ in $\text{Passed}$ then STOP
  
  - else (explore) add successors $(n, Z)$ to $\text{Waiting}$; Add $(n, Z)$ to $\text{Passed}$

- Until $\text{Waiting} = \emptyset$ or Final is in $\text{Waiting}$

Init $\rightarrow$ Final ?

INITIAL Passed $\Rightarrow \emptyset$; Waiting $\Rightarrow \{(n_0, Z_0)\}$

REPEAT

- pick $(n, Z)$ in $\text{Waiting}$
  
  - if for some $Z' \supseteq Z$ in $\text{Passed}$ then STOP
  
  - else (explore) add successors $(n, Z)$ to $\text{Waiting}$; Add $(n, Z)$ to $\text{Passed}$

UNTIL $\text{Waiting} = \emptyset$ or Final is in $\text{Waiting}$
Forward Rechability

INITIAL Passed := Ø;
Waiting := \{(n_0, Z_0)\}

REPEAT
- pick (n, Z) in Waiting
  - if for some Z' \subseteq Z
    (n, Z') in Passed then STOP
  - else (explore) add successors(n, Z) to Waiting;
  Add (n, Z) to Passed
UNTIL Waiting = Ø or Final is in Waiting

Init -> Final?

Two more operations on Zones
- Inclusion checking: Z_1 \subseteq Z_2
- Emptiness checking: Z = Ø

All Operations on Zones
(needed for reachability analysis)
- Transformation
- Conjunction
- Post condition (delay)
- Reset
- Consistency Checking
  - Inclusion
  - Emptiness

Efficient Implementation
Canonical Datastructures for Zones
Difference Bounded Matrices
Bellman 1958, Dill 1989

Inclusion

Z1: \( x \leq 1 \)
\( y - x \leq 2 \)
\( z - y \leq 2 \)
\( z \leq 9 \)

Z2: \( x \leq 2 \)
\( y - x \leq 3 \)
\( y \leq 3 \)
\( z - y \leq 3 \)
\( z \leq 7 \)

Z1 \( \subseteq \) Z2

Inclusion

Emptiness

Negative Cycle iff empty solution set

Conjunction

Add new edge for \( g \)

Z \( \land \) \( g \)

Delay

Remove upper bounds on clocks

Reset

Removes all bounds involving \( y \) and set \( y \) to 0

Reset

\( \{y\}Z \)

\( y = 0, \ 1 \leq x \)
COMPLEXITY

- Computing the shortest path closure, the canonical form of a zone: $O(n^3)$ [Dijkstra’s alg.]
- Run-time complexity, mostly in $O(n)$ (when we keep all zones in canonical form)

How about termination?

We need the normalization operation according to the maximal constant

Example: is $\otimes$ reachable?

The new zones created (due to the red edge/transitions) is getting larger and larger ...

- Consider the same example and check the reachability of $(\otimes, x>y)$ (it is NOT reachable; then the search will try to construct the whole state space)
- We need a solution !! (using the Maximal constant)

Example: is $\otimes$ reachable?

The Red constraints should be "normalized" according to the maximal constant $10$
Example: is reachable?

\[ (m, x=0, y=0) \]

\[ (\emptyset, x>0, y=5) \]

This is the Normalized zone graph which is finite.

The normalization Operation
(For automata containing no constraints on clock differences, Only)

- First compute the shortest path closure of a zone
- Remove all constraints in the form:
  \[ Xi < (\leq) m \text{ or } Xi-Xj < (\leq) n \]
  where \( m, n > \text{MAX} \)
- Replace all constraints in the form:
  \[ Xi > (\geq) m \text{ or } Xi-Xj > (\geq) n \]
  where \( m, n > \text{MAX} \)
  with \( Xi > \text{MAX} \) or \( Xi-Xj > \text{MAX} \)

The number of “Normalized Zones” is bounded

By the number of regions!

NOW, YOU CAN GO HOME
to MAKE YOUR OWN
MODEL CHECKER