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: \( E<> Q \)
  - \( E<> P \text{stop} \)
  - \( E<> (y>200) \)
- Invariant properties: \( A[] Q \) (not \( E<> Q \))
  - \( A[] \text{not} (P1.CS \text{and} P2.cs) \)
  - \( A[] (i < 100) \)
  - \( A[] (x>10 \implies i>100) \)
- Bounded Liveness Properties: \( F1 \implies F2 \)
  - \( A[](f1 \text{and} x>10 \implies f2) \)

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!
- However, the reachability problem is decidable ☺ Alur&Bill 1991

Region: From infinite to finite
- Concrete State
  \( (n, x=2.2, y=1.5) \)
- Symbolic state (region)
  \( (n, \quad) \)
- An equivalence class (i.e., a region) There are only finite many states.
Region equivalence: Definition [Alur and Dill 1990]

- $u,v$ are clock assignments
- $u \equiv v$ iff
  - For all clocks $x$, either $u(x) \leq C_x$ and $v(x) \leq C_x$ or $\lfloor u(x) \rfloor = \lfloor v(x) \rfloor$ (the same integer part)
  - For all clocks $x$, if $u(x) \leq C_x$, $(u(x)) \equiv 0$ iff $(v(x)) \equiv 0$
  - For all clocks $x,y$, if $u(x) \leq C_x$ and $u(y) \leq C_y$ $(u(x)) \equiv (u(y))$ iff $(v(x)) \equiv (v(y))$

Regions

Finite partitioning of state space

Region graph of a simple timed automata

An Important Theorem for Region Equivalence

- $u \equiv v$ implies
  - $u(x^0) = v(x^0)$
  - $u(x^n) = v(x^n)$ for all natural number $n$
  - For all $d < 1$: $u+d' \equiv v+d'$ 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'

Fischers again

Region Graph

An Important Theorem for Region Equivalence

- $u \equiv v$ implies
  - $u(x^0) = v(x^0)$
  - $u(x^n) = v(x^n)$ for all natural number $n$
  - For all $d < 1$: $u+d' \equiv v+d'$ 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'
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

Fischer’s Protocol
Analysis using zones

Fischer’s cont.

Fischer’s cont.

Taking time into account
Fischers cont.

Untimed case

Taking time into account

Symbolic Transitions

Zones = Conjective 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 re-written as a set:

\( (x_i, x_j) \sim b_i \mid \sim \in \{=, <, \leq \} \)

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 $[Z] = \{u \mid u$ is a solution of $Z\}$
  - The semantics

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

Operations on Zones

- Strongest post-condition (Delay): $SP(Z)$ or $Z^\uparrow$
  - $[Z^\uparrow] = \{u+d \mid d \in R, u \in [Z]\}$
- Weakest pre-condition: $WP(Z)$ or $Z^\downarrow$ (the dual of $Z^\uparrow$)
  - $[Z^\downarrow] = \{u\mid u+d \in [Z]$ for some $d \in R\}$
- Reset: $(x)Z$ or $Z(x:=0)$
  - $[(x)Z] = \{u[0/x] \mid u \in [Z]\}$
- Conjunction
  - $[Z \land g] = [Z] \cap [g]$

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: $(n,Z) \rightarrow (n,Z')$ where $Z' = Z^\uparrow \land inv(n)$
- Action: $(n,Z) \rightarrow (m,Z')$ where $Z' = (\{x\}Z \land g)$

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

Now, we have a search problem

Forward Reachability

- Initial $\rightarrow$ Final ?

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

- REPEAT
  - if $(n,Z)$ in Waiting then PRINT $n$, $Z$
  - if for some $Z'$ $\supseteq Z$ $(n,Z')$ in Passed then STOP
  - else (explore) add $Successors(n,Z)$ to Waiting; Add $(n,Z)$ to Passed

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

Init -> Final?

INITIAL Passed := Ø;
Waiting := \{(n0,Z0)\}

REPEAT
- pick (n,Z) in Waiting
- if for some Z' Z (n,Z') in Passed then STOP
- else (explore) add successors(n,Z) to Waiting;

UNTIL Waiting = Ø or Final is in Waiting

Two more operations on Zones
- Inclusion checking: Z₁ ⊆ Z₂
- solution sets
- Emptiness checking: Z = Ø
- no solution

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

x \leq 1
y - x \leq 2
z - y \leq 2
z \leq 9

x \leq 2
y - x \leq 3
y \leq 3
z - y \leq 3
z \leq 7

Z_1 \subseteq Z_2

Inclusion

Shortest Path Closure

Emptiness

Negative Cycle
iff empty solution set

Conjunction

Remove all bounds involving y and set y to 0

Reset

Delay

Add new edge for g

Conjunct
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 $\bigcirc$ reachable?

The search process may never terminate

- The new zones created (due to the red edge/transition) is getting larger and larger ...
- Note that in this example, Breadth-first may terminate the search with result: yes (i.e. $\bigcirc$ is reachable)
- But in general, there is no guarantee for termination even using Breadth-first e.g.
  - Consider the same example and check the reachability of $(\bigcirc, x>y)$ (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 reachable?

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:
  \[ X_i < (\leq) m \text{ or } X_i - X_j < (\leq) n \]
  where \( m, n > \text{MAX} \)
- Replace all constraints in the form:
  \[ X_i > (\geq) m \text{ or } X_i - X_j > (\geq) n \]
  where \( m, n > \text{MAX} \)
  with \( X_i > \text{MAX} \text{ or } X_i - X_j > \text{MAX} \)

The number of “Normalized Zones” is bounded

By the number of regions!

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