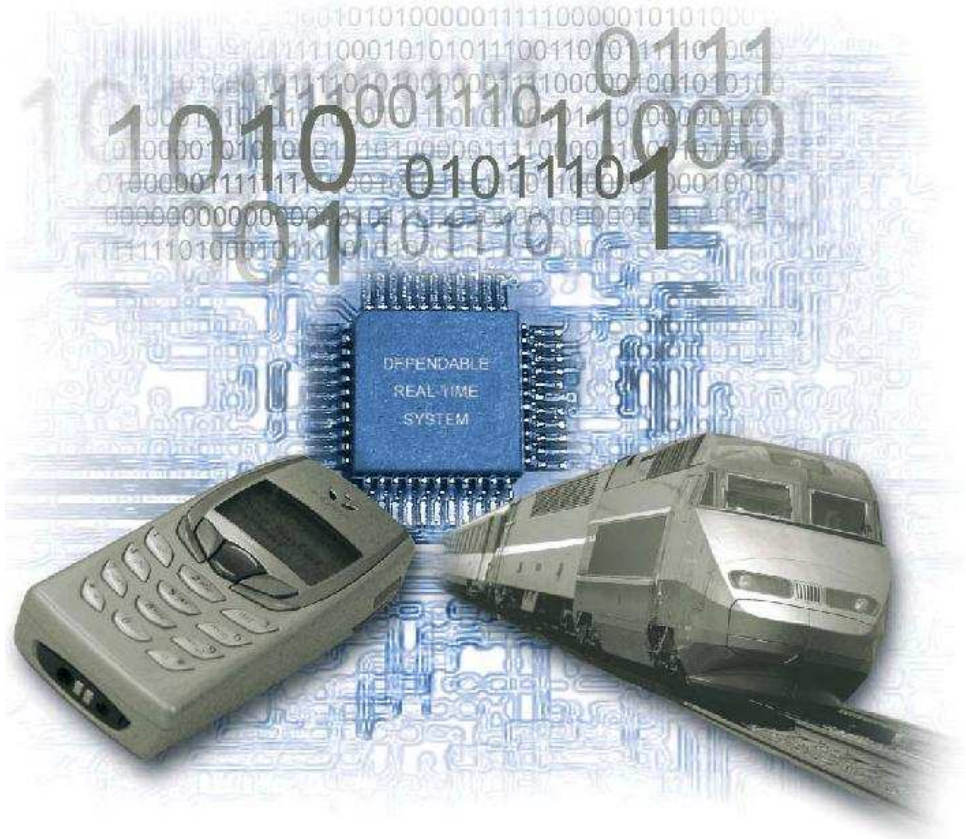
Embedded Systems 25



REVIEW: Automated Formal Methods

- Model Checking: automatically verify whether certain properties are guaranteed by the model; determine safe parameters
- Controller Synthesis: automatically construct control strategies that keep the system safe

Overview:

1. Intro: Analyzing FlexRay
2. Timed automata
3. Regions & zones
4. Model checking and controller synthesis
5. Hybrid automata

REVIEW: Timed Automata with Nondeterministic Delays [Alur/Dill]

A *timed automaton* is a tuple

$$TA = (Loc, Act, C, \rightsquigarrow, Loc_0, inv, AP, L) \quad \text{where:}$$

- Loc is a finite set of locations.
- $Loc_0 \subseteq Loc$ is a set of initial locations
- C is a finite set of clocks
- $L : Loc \to 2^{AP}$ is a labeling function for the locations
- $\rightsquigarrow \subseteq Loc \times CC(C) \times Act \times 2^C \times Loc$ is a transition relation, and
- $inv : Loc \to CC(C)$ is an invariant-assignment function

REVIEW: Clock Constraints

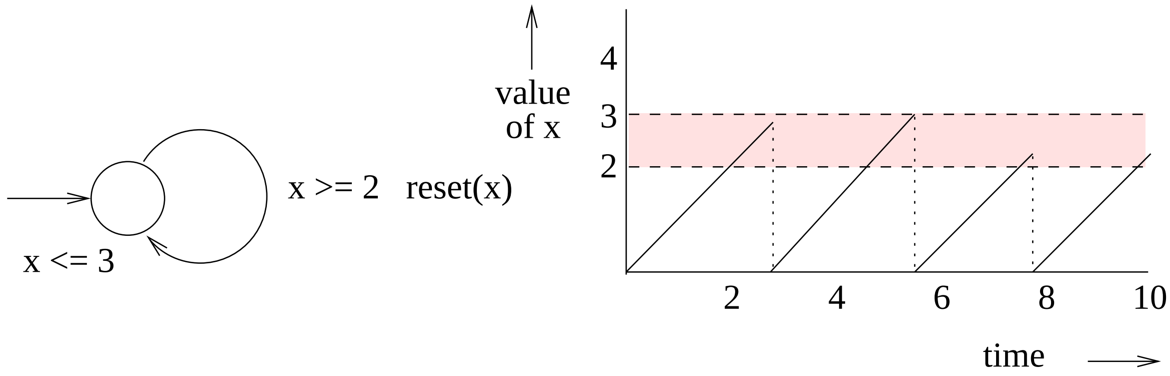
Clock constraints over set C of clocks are defined by:

$$g ::= True \mid x < c \mid x \leq c \mid \neg g \mid g \wedge g$$

- where $c \in \mathbb{N}$ and clocks $x, y \in C$
- rational constants would do; neither reals nor addition of clocks!
- let $CC(C)$ denote the set of clock constraints over C
- shorthands: $x \geq c$ denotes $\neg(x < c)$
  and $x \in [c_1, c_2)$ or $c_1 \leq x < c_2$ denotes $\neg(x < c_1) \wedge (x < c_2)$

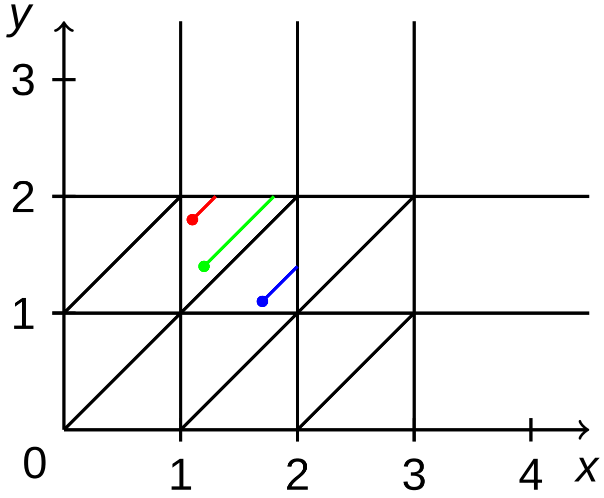
REVIEW: Guards vs. Location Invariants

The effect of a guard and an invariant:



REVIEW: Region Abstraction

- Consider a timed automaton with clocks x and y
- having maximal constants 3 and 2, respectively.

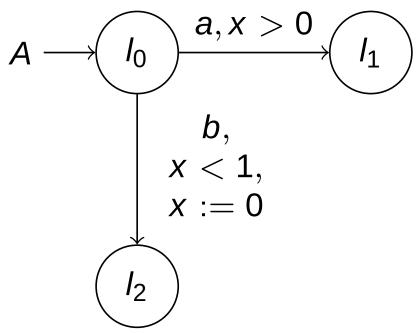


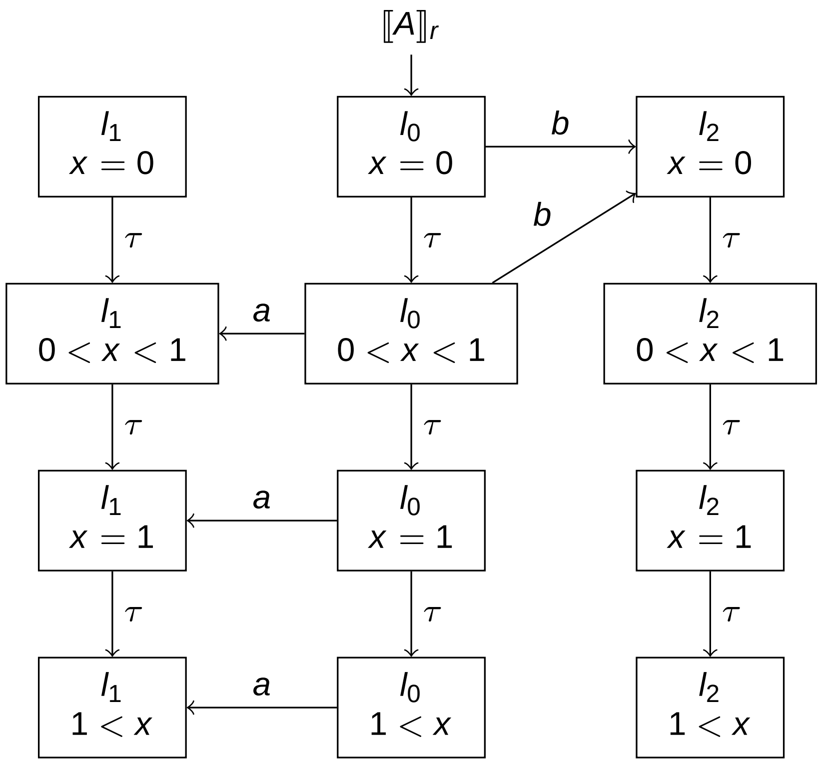
Equivalence relation $\simeq_R$

1. constraints
2. time elapsing
3. maximal constants

$\Longrightarrow$ finite index!

REVIEW: Region Automaton





REVIEW: Timed Analysis

Reachability is decidable

Theorem [Alur, 1994]:

$$\exists \text{ path } (l, \vec{t}) \longrightarrow (l', \vec{t}')$$

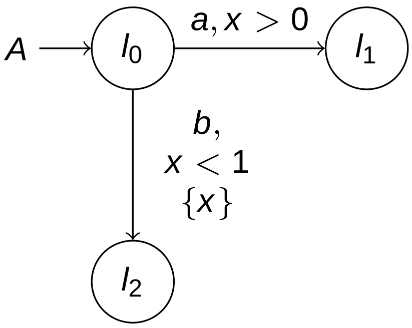
$$\text{iff}$$

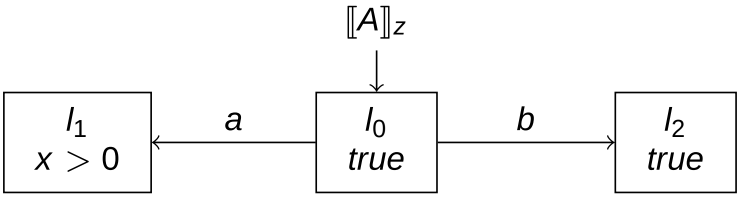
$$\exists \text{ path } (l, [\vec{t}]_R) \longrightarrow (l', [\vec{t}']_R)$$

Symbolic data structures

- Clock Region = Finest integral unit
- Clock Zone = Convex union of clock regions
- Federation = (Non-convex) union of clock zones

Zone graph

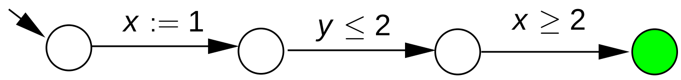


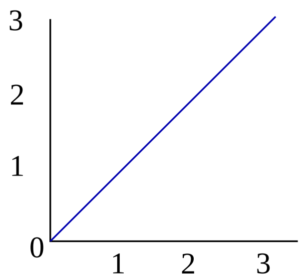


Zones

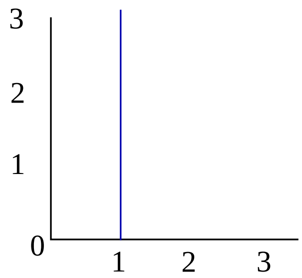
- Clock constraints are *conjunctions* of atomic constraints
  - $x \prec c$ and $x - y \prec c$ for $\prec \in \{<, \leq, =, \geq, >\}$
- A *clock zone* is the set of clock valuations that satisfy a clock constraint
  - a clock zone for g is the maximal set of clock valuations satisfying g
- Clock zone of g: $[\![g]\!] = \{\eta \in Eval(C) \mid \eta \models g\}$

Zone automaton: intuition

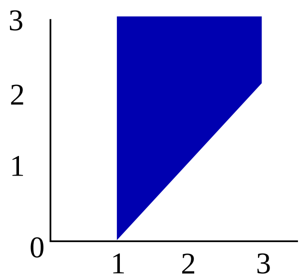




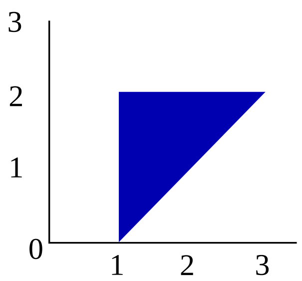
leaving initial



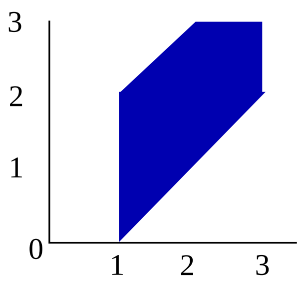
entering first



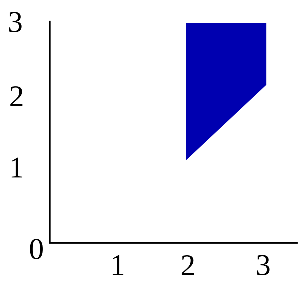
leaving first



entering second

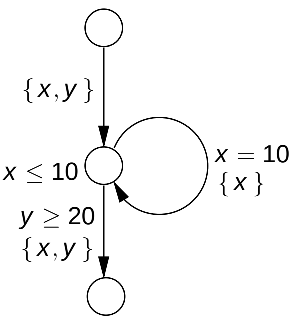


leaving second

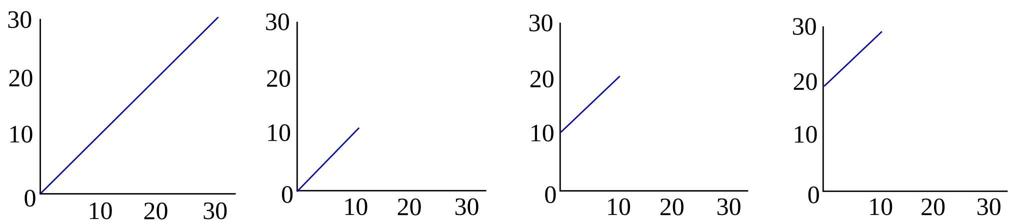


entering third

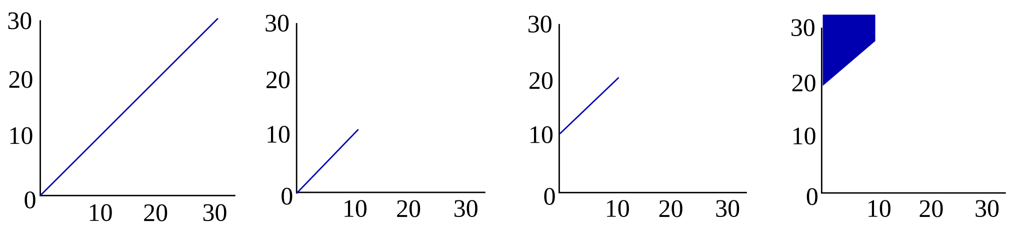
Normalization: intuition



symbolic semantics has infinitely many zones:



normalization yields a finite zone graph:



Successor and reset zones

- z' is the *successor* (clock) zone of z, denoted $z' = z^{\uparrow}$, if:
  - $z^{\uparrow} \;=\; \{\, \eta + d \mid \eta \in z, d \in \mathbb{R}_{>0} \,\}$
- z' is the zone obtained from z by *resetting* clocks D:
  - reset D in $z \;=\; \{\,\text{reset } D \text{ in } \eta \mid \eta \in z\,\}$

Representing zones

- Let $\mathbf{0}$ be a clock with constant value 0; let $C_0 = C \cup \{\,\mathbf{0}\,\}$
- Any zone $z \in Zone(C)$ can be written as:
  - conjunction of constraints $x - y < n$ or $x - y \leq n$ for $n \in \mathbb{Z}$, $x, y \in C_0$
  - when $x - y \preceq n$ and $x - y \preceq m$ take only $x - y \preceq \min(n, m)$

  $\Rightarrow$ this yields at most $|C_0| \cdot |C_0|$ constraints
- Example:

$$x - \mathbf{0} < 20 \;\wedge\; y - \mathbf{0} \leq 20 \;\wedge\; y - x \leq 10 \;\wedge\; x - y \leq -10 \;\wedge\; \mathbf{0} - z < 5$$

- Store each such constraint in a matrix
  - this yields a *difference bound matrix*

Difference bound matrices

- Zone z over C is represented by DBM $\mathbf{Z}$ of cardinality $|C{+}1|\cdot|C{+}1|$
  - for $C = x_1, \ldots, x_n$, let $C_0 = \{\, x_0, x_1, \ldots, x_n \,\}$ with $x_0 = \mathbf{0}$
  - $\mathbf{Z}(i,j) = (c, \prec)$ if and only if $x_i - x_j \prec c$
- Definition of $\mathbf{Z}$ for zone z:
  - for $x_i - x_j \prec c$ let $\mathbf{Z}(i,j) = (c, \prec)$
  - if $x_i - x_j$ is unbounded in z, set $\mathbf{Z}(i,j) = \infty$
  - $\mathbf{Z}(0,i) = (\leq, 0)$ and $\mathbf{Z}(i,i) = (\leq, 0)$
- Operations on bounds:
  - $(c, \preceq) < \infty$, $(c, <) < (c, \leq)$, and $(c, \preceq) < (c', \preceq')$ if $c < c'$
  - $c + \infty = \infty$, $(c, \leq) + (c', \leq) = (c{+}c', \leq)$ and $(c, <) + (c', \leq) = (c{+}c', <)$

Canonical DBMs

- A zone z is in *canonical form* if and only if:
  - no constraint in z can be strengthened without reducing $[\![\, z \,]\!] = \{\, \eta \mid \eta \in z \,\}$
- For each zone z: $\exists$ a *unique* and *equivalent* zone in canonical form
- Represent zone z by a *weighted digraph* $G = (V, E, w)$ where
  - $V = C_0$ is the set of vertices
  - $(x_i, x_j) \in E$ whenever $x_j - x_i \preceq c$ is a constraint in z
  - $w(x_i, x_j) = (\preceq, c)$ whenever $x_j - x_i \preceq c$ is a constraint in z
- Zone z is in *canonical form* if and only if DBM $\mathbf{Z}$ satisfies:
  - $\mathbf{Z}(i,j) \;\leq\; \mathbf{Z}(i,k) + \mathbf{Z}(k,j)$ for any $x_i, x_j, x_k \in C_0$
- Compute canonical zone?
  - use *Floyd-Warshall*'s all-pairs SP algorithm (time $\mathcal{O}(|C_0|^3)$)

Main operations on DBMs (1)

- *Nonemptiness*: is $[\![\mathbf{Z}]\!] \neq \emptyset$?
  - search for negative cycles in the graph representation of **Z**, or
  - mark **Z** when upper bound of some clock is set to value $<$ its lower bound
- *Inclusion test*: is $[\![\mathbf{Z}]\!] \subseteq [\![\mathbf{Z}']\!]$?
  - for DBMs in canonical form, test whether $\mathbf{Z}(i,j) \leq \mathbf{Z}'(i,j)$, for all $i,j \in C_0$
- *Delay*: determine $\mathbf{Z}^{\uparrow}$
  - remove the upper bounds on any clock, i.e.,
  - $\mathbf{Z}^{\uparrow}(i,0) = \infty$ and $\mathbf{Z}^{\uparrow}(i,j) = \mathbf{Z}(i,j)$ for $j \neq 0$

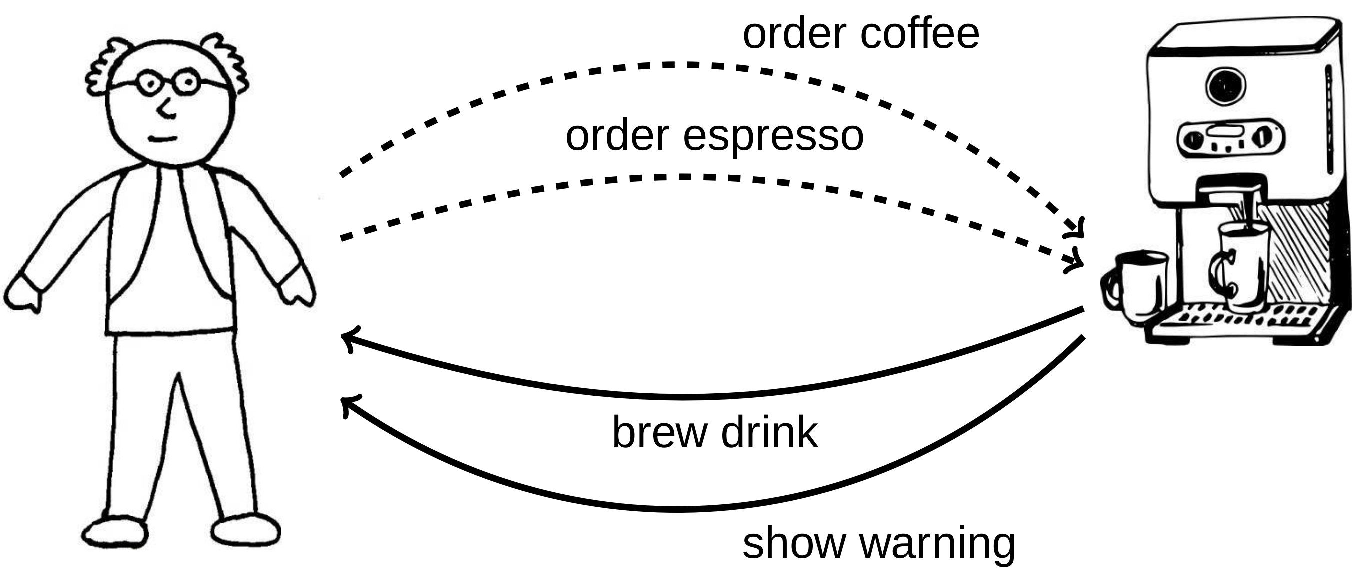
Main operations on DBMs (2)

- *Conjunction*: $z \wedge (x_i - x_j \preceq n)$
  - if $(n, \preceq) < \mathbf{Z}(i,j)$ then $\mathbf{Z}(i,j) := (n, \preceq)$ else do nothing
  - put **Z** back into canonical form (in time $\mathcal{O}(|C_0|^2)$ using that only $\mathbf{Z}(i,j)$ changed)
- *Clock reset*: $x_i := 0$
  - $\mathbf{Z}(i,j) := \mathbf{Z}(0,j)$ and $\mathbf{Z}(j,i) := \mathbf{Z}(j,0)$
- *Normalization*
  - remove all bounds $x - y \preceq m$ for which $(m, \preceq) > (c_x, \leq)$, and
  - set all bounds $x - y \preceq m$ with $(m, \preceq) < (-c_y, <)$ to $(-c_y, <)$
  - put the DBM back into canonical form (Floyd-Warshall)

Controller Synthesis

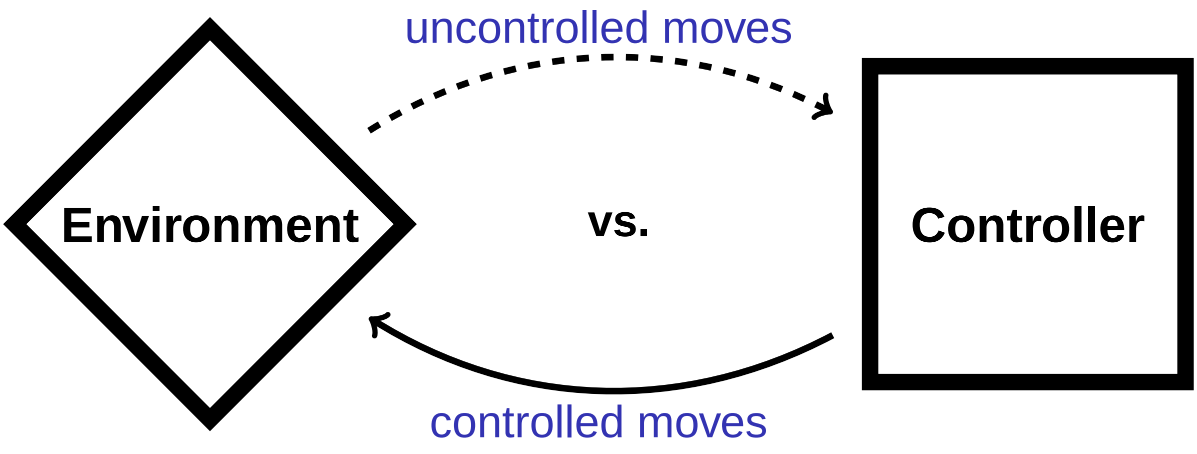
Controller Synthesis

We distinguish between external (uncontrolled) and internal (controlled) nondeterminism



Games

Game between two players

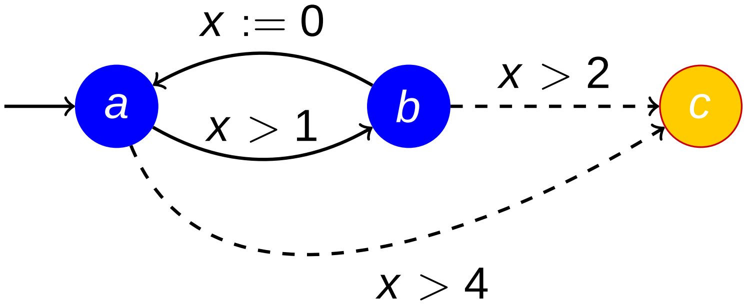


"wants to violate the spec."

"wants to satisfy the spec."

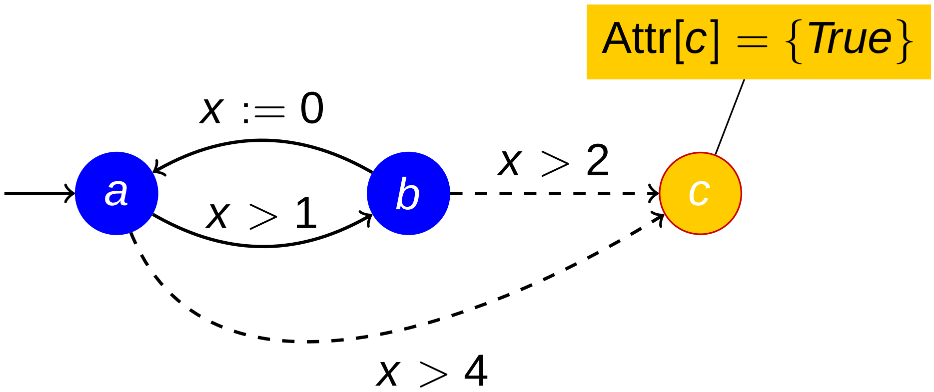
Reachability games played on timed automata

From where can $\dashrightarrow$ enforce a run to c ?



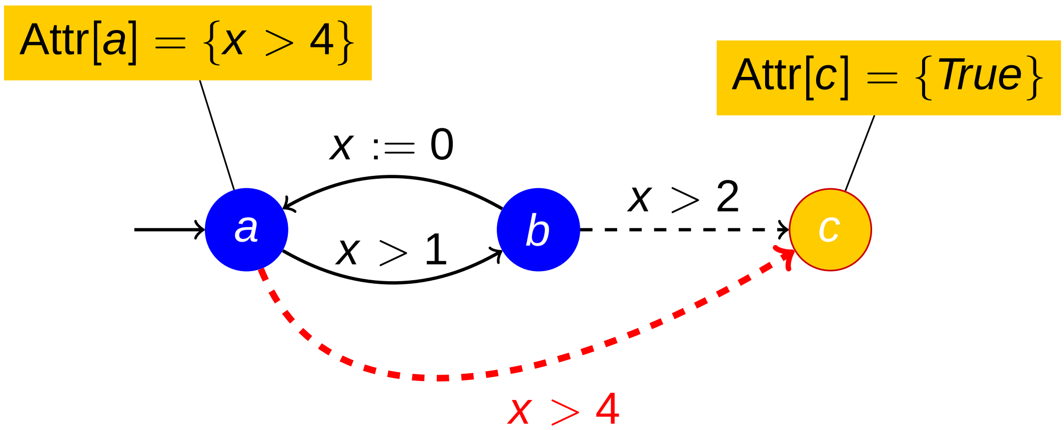
Zone-based timed game solving

From where can ⇢ enforce a run to c ?



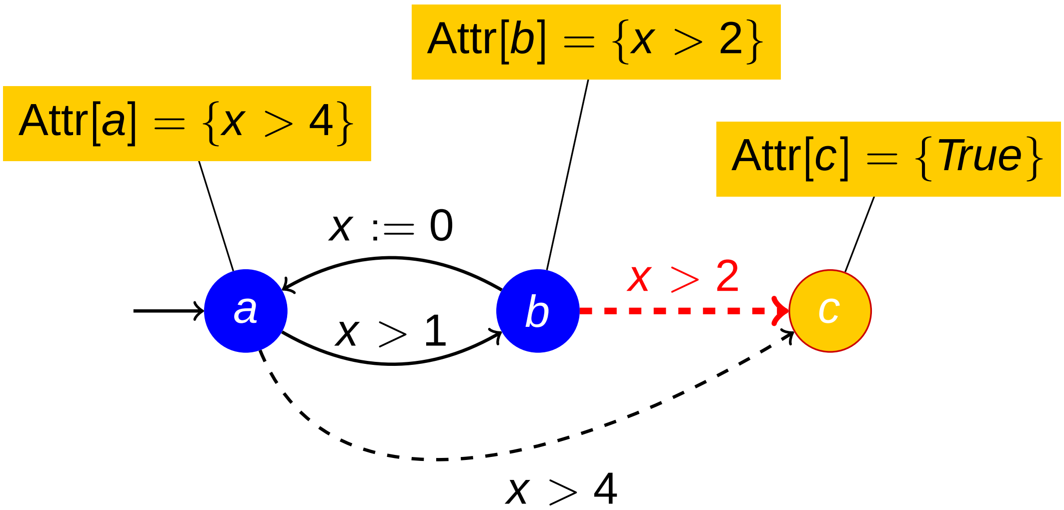
Zone-based timed game solving

From where can ⇢ enforce a run to c ?



Zone-based timed game solving

From where can $\dashrightarrow$ enforce a run to c ?



Zone-based timed game solving

From where can $\dashrightarrow$ enforce a run to c ?