Using non-convex Approximations for Efficient Analysis of Timed Automata

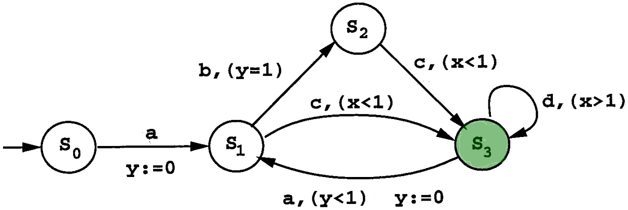
F. Herbreteau[1], D. Kini[2], B. Srivathsan[1] and I. Walukiewicz[1]

LaBRI, Université de Bordeaux 1

Indian Institute of Technology Bombay, India

ACTS III, January 2011 - Chennai, India

Timed Automata [AD94]



Run: finite **sequence** of transitions,

$$(s_0, \overbrace{0}^{x}, \overbrace{0}^{y}) \xrightarrow{0.4,a} (s_1, 0.4, 0) \xrightarrow{0.5,c} (s_3, 0.9, 0.5)$$

- A run is **accepting** if it ends in a green state.

The problem we are interested in ...

Given a TA, does there **exist** an **accepting run**?

The problem we are interested in ...

Given a TA, does there **exist** an **accepting run**?

Theorem [AD94]

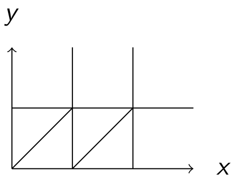
This problem is **PSPACE-complete**

First solution to this problem

Key idea: Partition the space of valuations into a **finite** number of **regions**

First solution to this problem

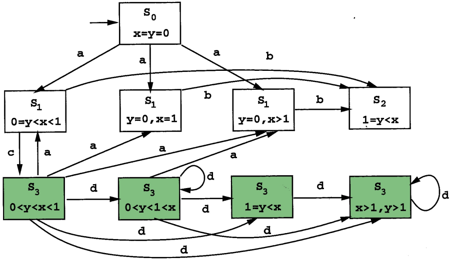
Key idea: Partition the space of valuations into a **finite** number of **regions**



- Region: set of valuations satisfying the **same guards** w.r.t. time
- Finiteness: Parametrized by **maximal constant**

First solution to this problem

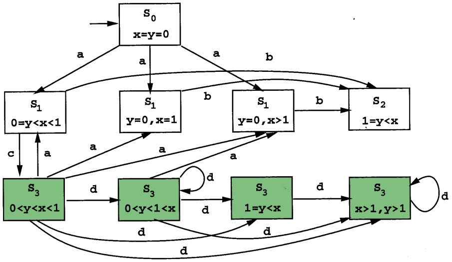
Key idea: Partition the space of valuations into a **finite** number of **regions**



- Region: set of valuations satisfying the **same guards** w.r.t. time
- Finiteness: Parametrized by **maximal constant**

First solution to this problem

Key idea: Partition the space of valuations into a **finite** number of **regions**



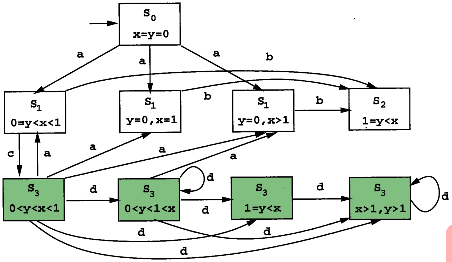
- Region: set of valuations satisfying the **same guards** w.r.t. time
- Finiteness: Parametrized by **maximal constant**

Sound and complete

Region graph preserves state **reachability**

First solution to this problem

Key idea: Partition the space of valuations into a **finite** number of **regions**



- Region: set of valuations satisfying the **same guards** w.r.t. time
- Finiteness: Parametrized by **maximal constant**

$\mathcal{O}(|X|!.M^{|X|})$ many regions!

Sound and complete

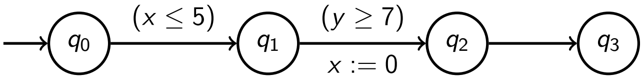
Region graph preserves state **reachability**

A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path

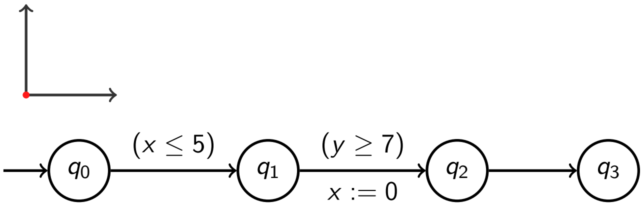
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



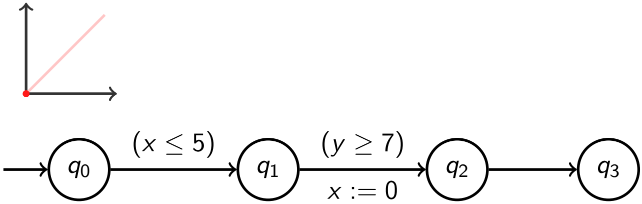
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



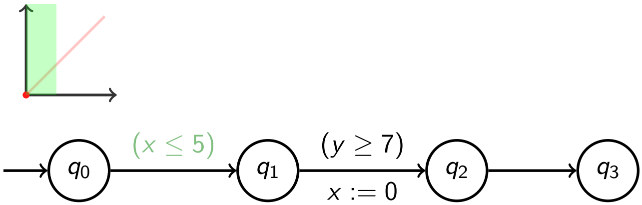
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



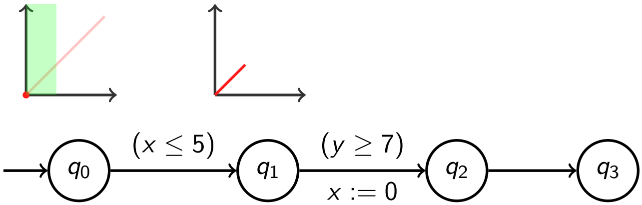
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



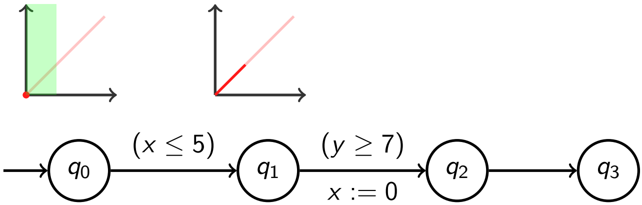
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



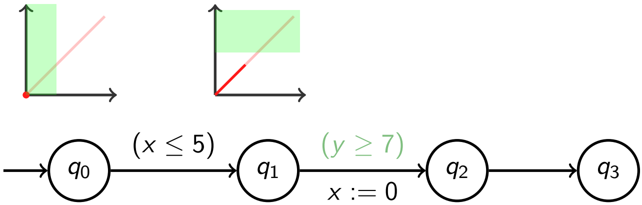
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



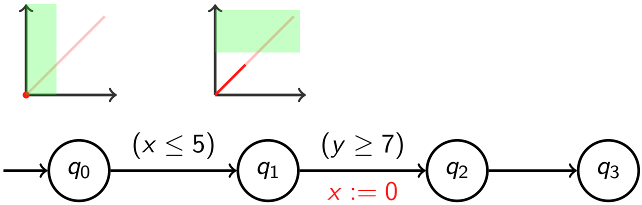
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



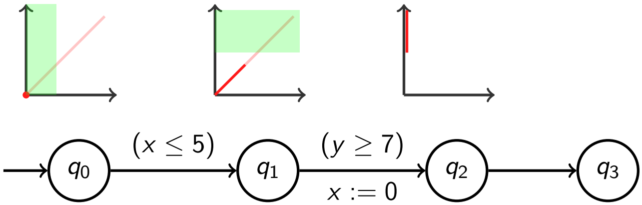
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



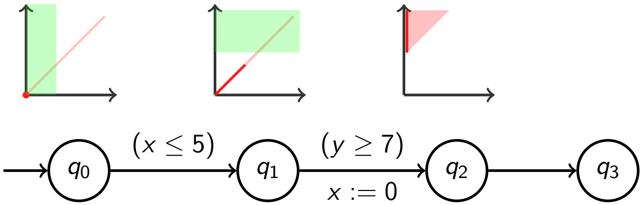
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



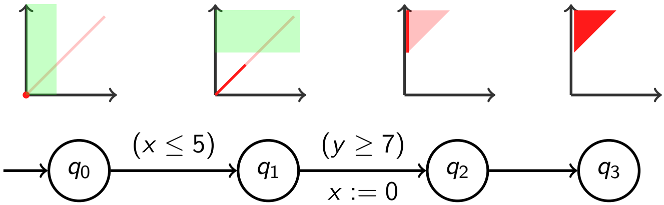
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



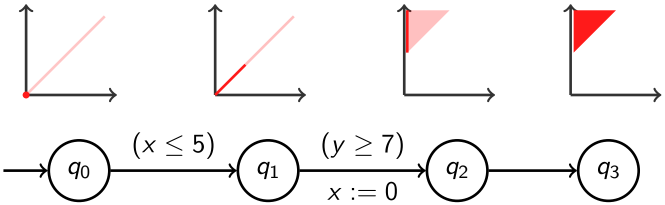
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



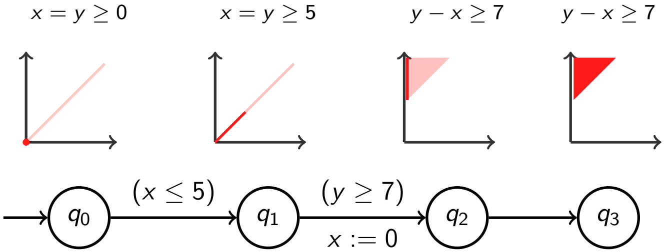
A more efficient solution...

Key idea: Maintain **all valuations** reaching a state via a path



A more efficient solution...

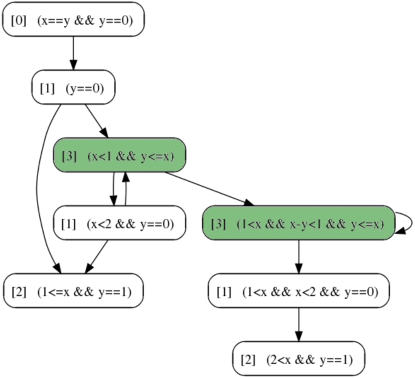
Key idea: Maintain **all valuations** reaching a state via a path



Zones and zone graph

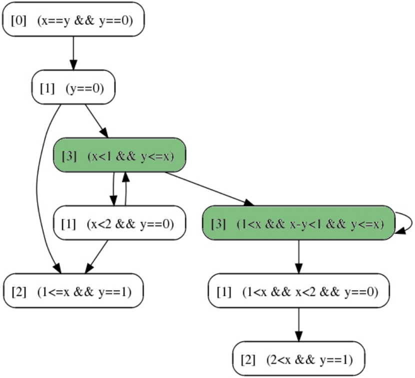
- Zone: set of valuations defined by conjunctions of constraints:
  - $x \sim c$
  - $x - y \sim c$
  - e.g. $(x - y \geq 1) \wedge y < 2$
- Representation: by DBM

Zones and zone graph



- Zone: set of valuations defined by conjunctions of constraints:
  - $x \sim c$
  - $x - y \sim c$
  - e.g. $(x - y \geq 1) \wedge y < 2$
- Representation: by DBM

Zones and zone graph

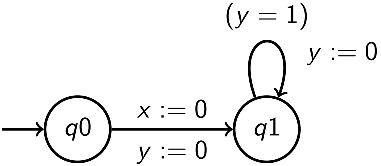


- Zone: set of valuations defined by conjunctions of constraints:
  - $x \sim c$
  - $x - y \sim c$
  - e.g. $(x - y \geq 1) \wedge y < 2$
- Representation: by DBM

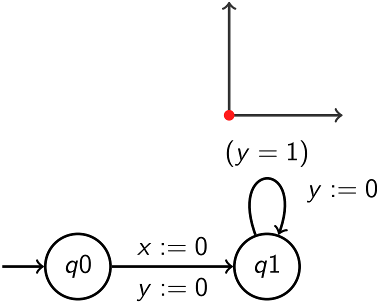
Sound and complete

Zone graph preserves state **reachability**

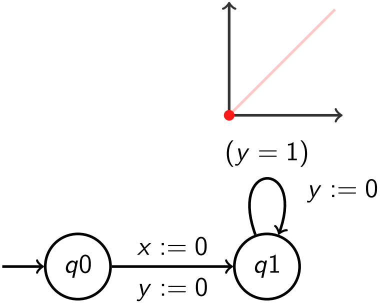
But the zone graph could be infinite ...



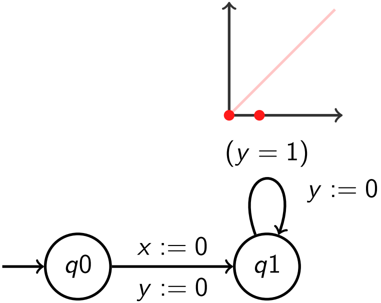
But the zone graph could be infinite ...



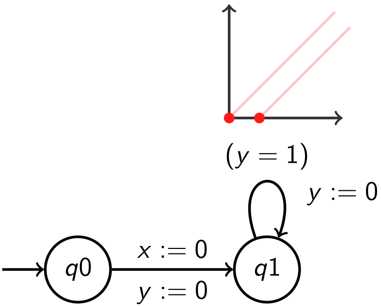
But the zone graph could be infinite ...



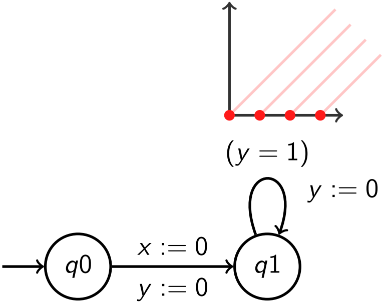
But the zone graph could be infinite ...



But the zone graph could be infinite ...

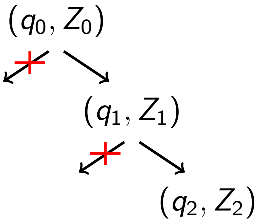


But the zone graph could be infinite ...



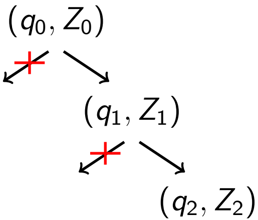
Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner



Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner



$(q_0, \mathfrak{a}(Z_0))$

Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner

$$
\begin{tikzcd}[column sep=small]
 & (q_0, Z_0) \arrow[dl, "\times" marking] \arrow[dr] & & \\
  & & (q_1, Z_1) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & & & (q_2, Z_2)
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
 & (q_0, \mathfrak{a}(Z_0)) \arrow[dl, "\times" marking] \\
  &
\end{tikzcd}
$$

Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner

$$\begin{tikzcd}
(q_0, Z_0) \arrow[dl, "\times" marking] \arrow[dr] & & \\
 & (q_1, Z_1) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & & (q_2, Z_2)
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
(q_0, \mathfrak{a}(Z_0)) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & (q_1, Z')
\end{tikzcd}$$

Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner

$$\begin{tikzcd}[column sep=small]
& (q_0, Z_0) \arrow[dl, "\times" marking] \arrow[dr] & & \\
{} & & (q_1, Z_1) \arrow[dl, "\times" marking] \arrow[dr] & \\
& {} & & (q_2, Z_2)
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}[column sep=small]
& (q_0, \mathfrak{a}(Z_0)) \arrow[dl, "\times" marking] \arrow[dr] & \\
{} & & (q_1, \mathfrak{a}(Z'))
\end{tikzcd}$$

Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner

$$(q_0, Z_0)$$

$$(q_1, Z_1)$$

$$(q_2, Z_2)$$

$$(q_0, \mathfrak{a}(Z_0))$$

$$(q_1, \mathfrak{a}(Z'))$$

$$(q_2, Z'')$$

Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner

$$
\begin{tikzcd}[column sep=small]
 & (q_0, Z_0) \arrow[dl, "\times" marking] \arrow[dr] & & \\
  & & (q_1, Z_1) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & & & (q_2, Z_2)
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}[column sep=small]
 & (q_0, \mathfrak{a}(Z_0)) \arrow[dl, "\times" marking] \arrow[dr] & & \\
  & & (q_1, \mathfrak{a}(Z')) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & & & (q_2, \mathfrak{a}(Z''))
\end{tikzcd}
$$

Use finite abstractions

Key idea: **Extrapolate** each zone in a **sound** manner

$$
\begin{tikzcd}
 & (q_0, Z_0) \arrow[dl, "\times" marking] \arrow[dr] & & \\
 & & (q_1, Z_1) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & & & (q_2, Z_2)
\end{tikzcd}
\qquad
\begin{tikzcd}
 & (q_0, \mathfrak{a}(Z_0)) \arrow[dl, "\times" marking] \arrow[dr] & & \\
 & & (q_1, \mathfrak{a}(Z')) \arrow[dl, "\times" marking] \arrow[dr] & \\
 & & & (q_2, \mathfrak{a}(Z''))
\end{tikzcd}
$$

- Number of **extrapolated zones** is **finite**
- **Bigger** extrapolated zones $\rightarrow$ smaller **simulation graph**

Abstractions in literature [Bou04, BBLP06]

$$\begin{tikzcd}
 & \mathfrak{a}_{\preccurlyeq LU} & \\
\mathrm{Closure}_{\alpha} \arrow[ur] & & \mathrm{Extra}^{+}_{LU} \arrow[ul] \\
 & \mathrm{Extra}^{+}_{\alpha} \arrow[ul] \arrow[ur] & \\
 & \mathrm{Extra}_{\alpha} \arrow[u] &
\end{tikzcd}$$

Abstractions in literature [Bou04, BBLP06]

$$\begin{tikzcd}
 & \mathfrak{a}_{\preccurlyeq LU} & \\
\text{Closure}_{\alpha} \arrow[ur] & & \text{Extra}^{+}_{LU} \arrow[ul] \\
 & \text{Extra}^{+}_{\alpha} \arrow[ul] \arrow[ur] & \\
 & \text{Extra}_{\alpha} \arrow[u] &
\end{tikzcd}$$

Sound and complete

All the above abstractions preserve state **reachability**

Abstractions in literature [Bou04, BBLP06]

$$
\begin{tikzcd}
 & \mathfrak{a}_{\preccurlyeq LU} & \\
\mathrm{Closure}_\alpha \arrow[ur] & & \mathrm{Extra}^+_{LU} \arrow[ul] \\
 & \mathrm{Extra}^+_\alpha \arrow[ul] \arrow[ur] & \\
 & \mathrm{Extra}_\alpha \arrow[u] &
\end{tikzcd}
$$

Sound and complete

All the above abstractions preserve state **reachability**

But for **implementation** extrapolated zone should be a zone

Abstractions in literature [Bou04, BBLP06]

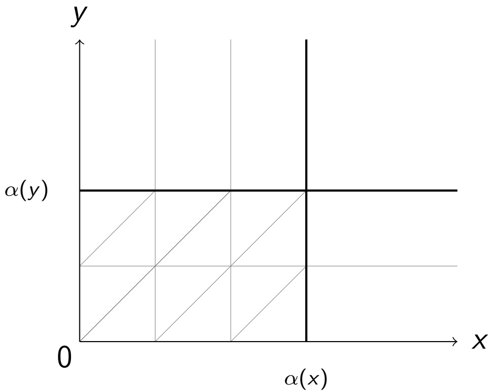
$$
\begin{tikzcd}
 & \mathfrak{a}_{\preccurlyeq LU} & \\
\mathrm{Closure}_\alpha \arrow[ur] & & \mathrm{Extra}^+_{LU}\ \checkmark \arrow[ul] \\
 & \mathrm{Extra}^+_\alpha\ \checkmark \arrow[ul] \arrow[ur] & \\
 & \mathrm{Extra}_\alpha\ \checkmark \arrow[u] &
\end{tikzcd}
$$

Only convex abstractions in **implementations**!

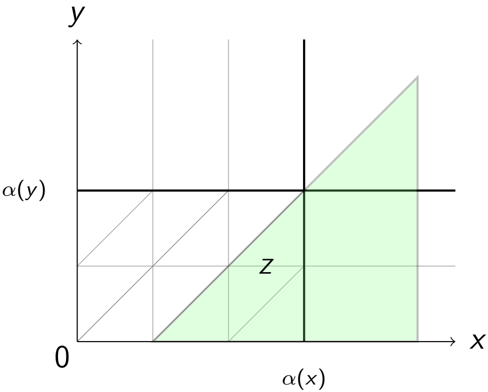
Here...

Efficient use of the **non-convex** Closure abstraction!

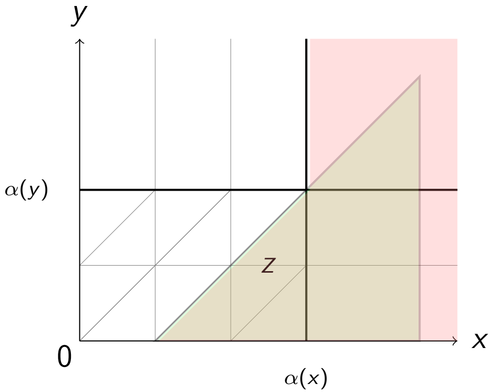
What is Closure$_\alpha$?



What is Closure$_\alpha$?

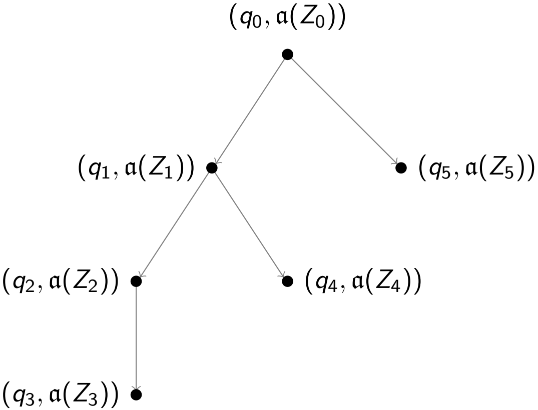


What is Closure$_\alpha$?



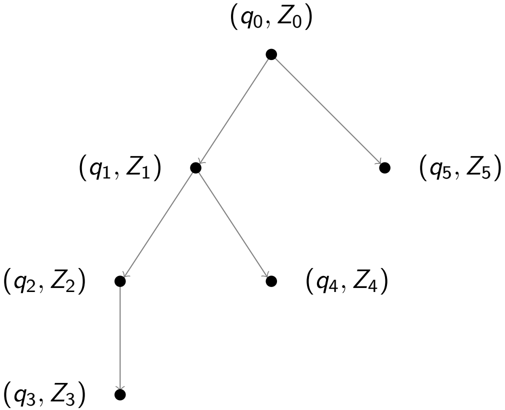
Closure$_\alpha(Z)$: set of regions that Z intersects

Schema

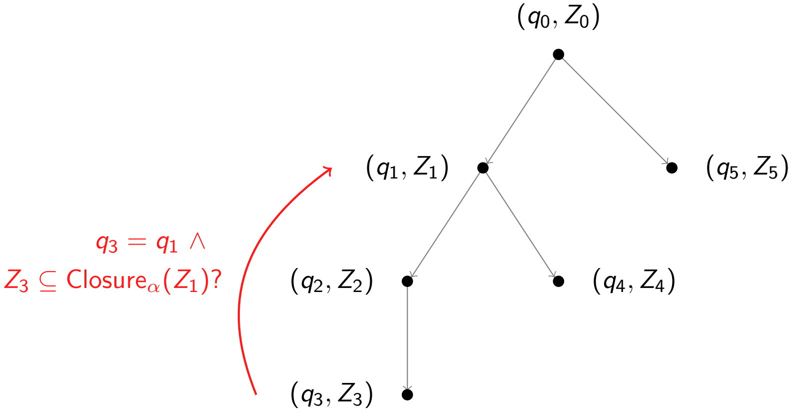


$(q_0, \mathfrak{a}(Z_0))$
$(q_1, \mathfrak{a}(Z_1))$
$(q_5, \mathfrak{a}(Z_5))$
$(q_2, \mathfrak{a}(Z_2))$
$(q_4, \mathfrak{a}(Z_4))$
$(q_3, \mathfrak{a}(Z_3))$

Schema



Schema


(q_0, Z_0)
(q_1, Z_1)
(q_5, Z_5)
$q_3 = q_1 \wedge$
$Z_3 \subseteq \text{Closure}_\alpha(Z_1)$?
(q_2, Z_2)
(q_4, Z_4)
(q_3, Z_3)

$Z \subseteq \text{Closure}_\alpha(Z')$

Reduction to 2 clocks

$Z \subseteq \text{Closure}_\alpha(Z')$ iff for all pairs of clocks x, y, we have

$$Proj_{xy}(Z) \subseteq \text{Closure}_\alpha(Proj_{xy}(Z'))$$

Complexity: $\mathcal{O}(|X|^2)$ where X is the set of clocks

$Z \subseteq \text{Closure}_\alpha(Z')$

Reduction to 2 clocks

$Z \subseteq \text{Closure}_\alpha(Z')$ iff for all pairs of clocks x, y, we have

$$Proj_{xy}(Z) \subseteq \text{Closure}_\alpha(Proj_{xy}(Z'))$$

Complexity: $\mathcal{O}(|X|^2)$ where X is the set of clocks

Same complexity as $Z \subseteq Z'$!

So what do we have now...

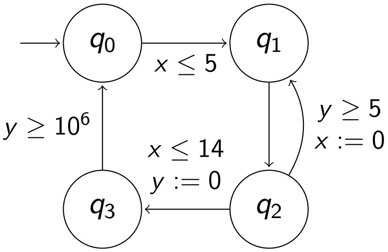
- **No need** to restrict to **convex** abstractions
- Compute $ZG(\mathcal{A})$: $Z \subseteq \mathrm{Closure}_\alpha(Z')$ to **terminate**

So what do we have now...

- **No need** to restrict to **convex** abstractions
- Compute $ZG(\mathcal{A})$: $Z \subseteq \text{Closure}_\alpha(Z')$ to **terminate**

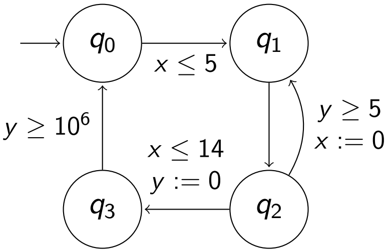
Coming next: **prune** the **bound function** α !

Bound function α



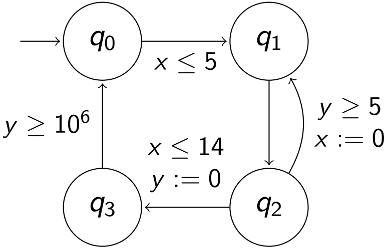
Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

Static analysis: bound function for every q [BBFL03]



Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

Static analysis: bound function for every q [BBFL03]

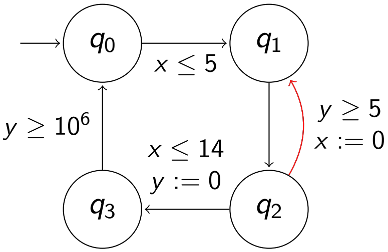


$\nu(y) = 10$

$\nu'(y) = 10^7$

Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

Static analysis: bound function for every q [BBFL03]

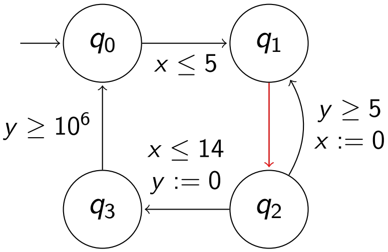


$\nu(y) = 10$

$\nu'(y) = 10^7$

Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

Static analysis: bound function for every q [BBFL03]

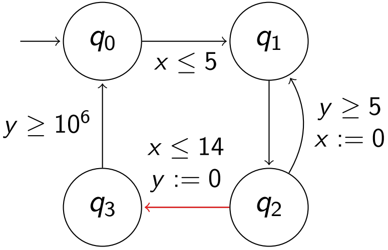


$\nu(y) = 10$

$\nu'(y) = 10^7$

Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

Static analysis: bound function for every q [BBFL03]

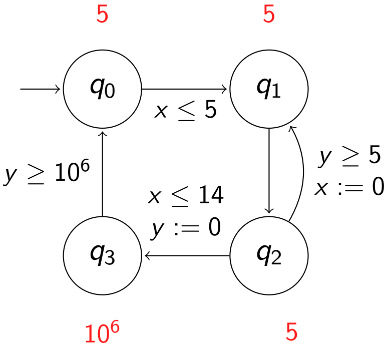


$\nu(y) = 10$

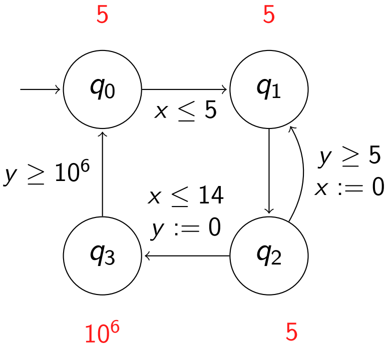
$\nu'(y) = 10^7$

Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

Static analysis: bound function for every q [BBFL03]

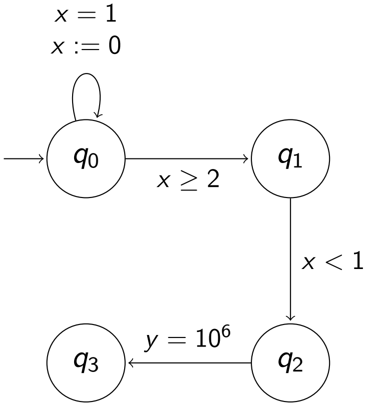


Static analysis: bound function for every q [BBFL03]

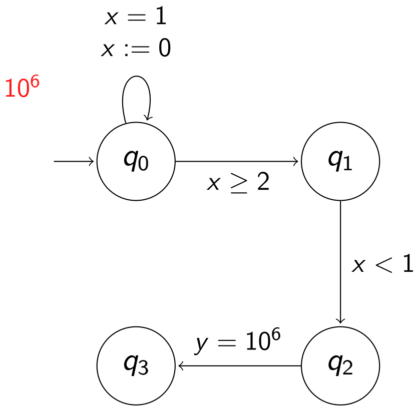


But this is not enough!

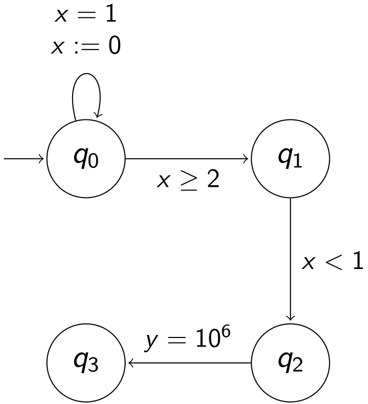
Need to look at semantics...



Need to look at semantics...

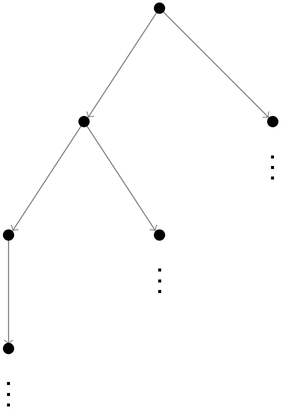


Need to look at semantics...

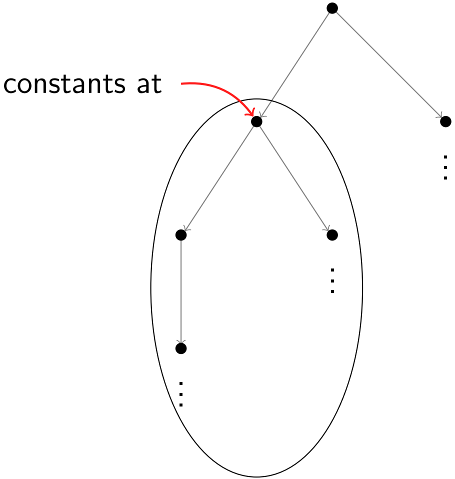


More than $\mathbf{10^6}$ zones at q_0 **not necessary**!

Bound function for every (q, Z) in $ZG(\mathcal{A})$



Bound function for every (q, Z) in $ZG(\mathcal{A})$



Constant propagation

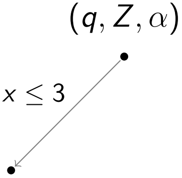
$\alpha(x) = -\infty$

(q, Z, α)

•

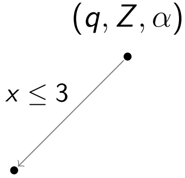
Constant propagation

$\alpha(x) = -\infty$



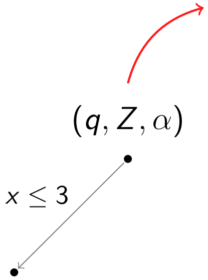
Constant propagation

$\alpha(x) = \ 3$



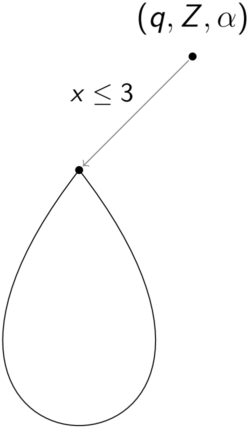
Constant propagation

$\alpha(x) = \ 3$



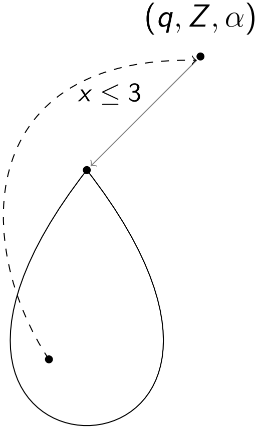
Constant propagation

$\alpha(x) = \quad 5$



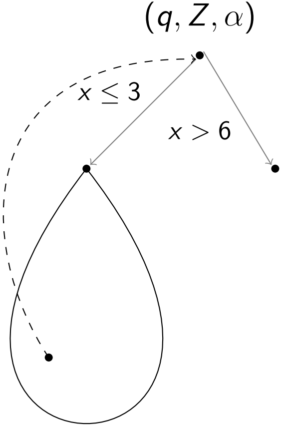
Constant propagation

$\alpha(x) = \quad 5$



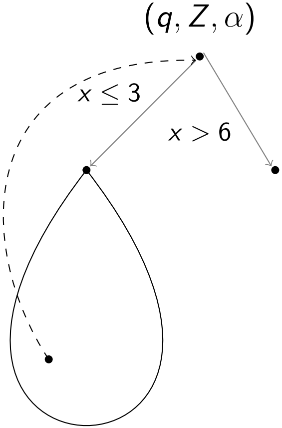
Constant propagation

$\alpha(x) = \quad 5$



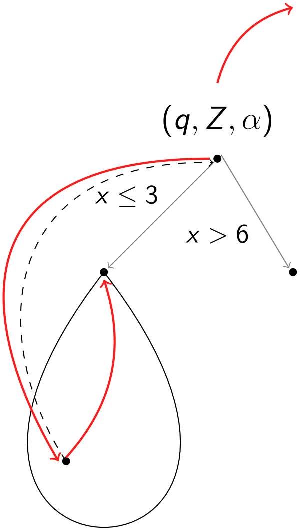
Constant propagation

$\alpha(x) = \quad 6$



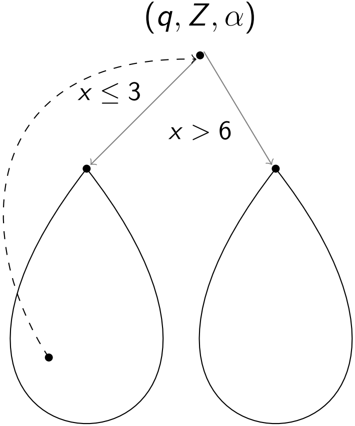
Constant propagation

$\alpha(x) = \quad 6$



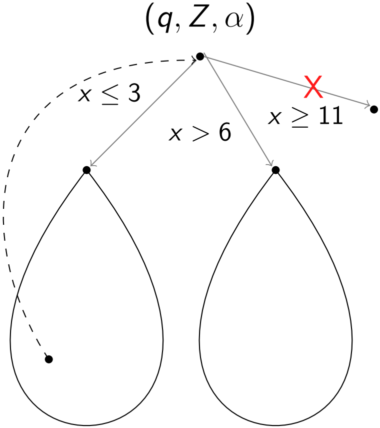
Constant propagation

$\alpha(x) = \quad 6$



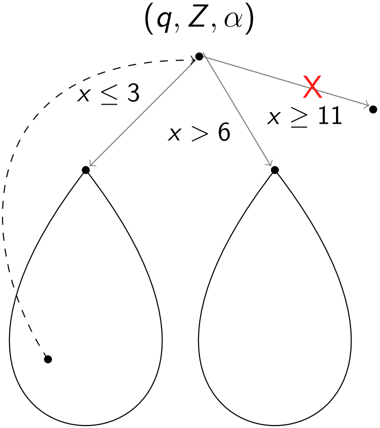
Constant propagation

$\alpha(x) = \quad 6$



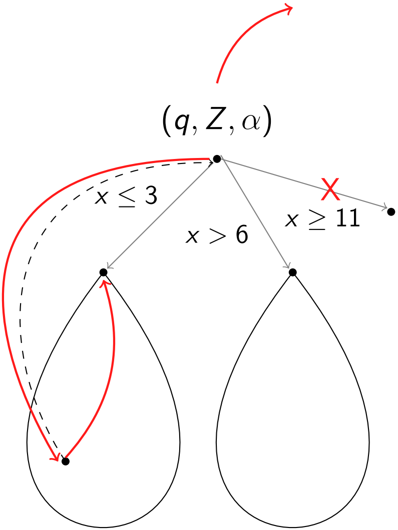
Constant propagation

$\alpha(x) = \quad 11$



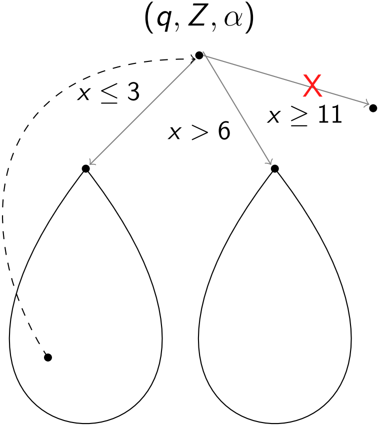
Constant propagation

$\alpha(x) = \quad 11$



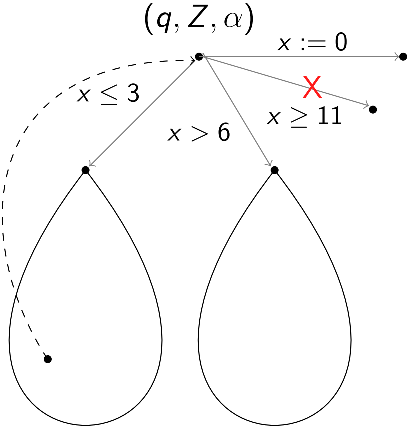
Constant propagation

$\alpha(x) = \quad 11$



Constant propagation

$\alpha(x) = \quad 11$



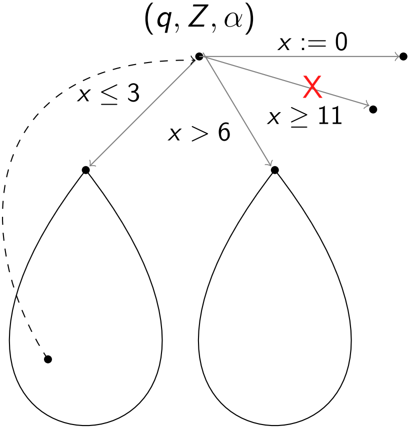
Constant propagation

All **tentative nodes** consistent

$+$ No more **exploration**

$\rightarrow$ Terminate!

$\alpha(x) = \quad 11$



Invariants on the bounds

- Non tentative nodes: $\alpha = max\{\alpha_{succ}\}$ (modulo resets)
- Tentative nodes: $\alpha = \alpha_{master}$

Invariants on the bounds

- Non tentative nodes: $\alpha = max\{\alpha_{succ}\}$ (modulo resets)
- Tentative nodes: $\alpha = \alpha_{master}$

Theorem (Correctness)

An accepting state is reachable in $ZG(\mathcal{A})$ iff the algorithm reaches a node with an accepting state and a non-empty zone.

Overall algorithm

- Compute $ZG(\mathcal{A})$: $Z \subseteq \text{Closure}_{\alpha'}(Z')$ for **termination**
- **Bounds** α calculated **on-the-fly**
- Extra^{+}_{LU} can also be handled:
  - $\mathcal{O}(|X|^2)$ procedure for $Z \subseteq \text{Closure}_{\alpha'}(\mathit{Extra}^{+}_{L'U'}(Z'))$

Benchmarks

Model	E^+_{LU},sa		Cl^+_{LU},sa		Cl^+_{LU},otf	
	nodes	s.	nodes	s.	nodes	s.
Fi7	48535	6.24	48535	4.85	26405	2.76
Fi8	229890	63.97	229890	33.78	95353	12.49
Fi9	1024697	558.90	1024697	250.42	339211	55.29
Fi10	–	–	–	–	1191211	322.01
C7	23137	6.07	23137	6.74	18034	5.94
C8	86157	40.45	86157	37.18	65745	30.92
C9	317326	283.56	317326	201.02	238594	156.56
FD10	726	1.89	640	3.22	640	3.35
FD20	2846	70.13	2430	86.27	2430	90.99
FD30	6366	670.31	5370	622.14	5370	655.89

E^+_{LU}, sa is currently used in **UPPAAL**

- Closure can be **efficiently implemented**
- **Both** Closure and otf bounds help

Conclusions & Perspectives

- **Efficient implementation** of a non-convex approximation that **subsumes** current ones in use
- **On-the-fly learning** of bounds that is **better** than the current static analysis

- More **sophisticated** non-convex approximations
- **Strategies** for constraint propagation

Bibliography

R. Alur and D.L. Dill.
A theory of timed automata.
Theoretical Computer Science, 126(2):183–235, 1994.

G. Behrmann, P. Bouyer, E. Fleury, and K. G. Larsen.
Static guard analysis in timed automata verification.
In *TACAS'03*, volume 2619 of *LNCS*, pages 254–270. Springer, 2003.

G. Behrmann, P. Bouyer, K. G. Larsen, and R. Pelanek.
Lower and upper bounds in zone-based abstractions of timed automata.
Int. Journal on Software Tools for Technology Transfer, 8(3):204–215, 2006.

P. Bouyer.
Forward analysis of updatable timed automata.
Form. Methods in Syst. Des., 24(3):281–320, 2004.