Using non-convex approximations for efficient analysis of timed automata

B. Srivathsan[1]

Joint work with F. Herbreteau[1], D. Kini[2] and I. Walukiewicz[1]

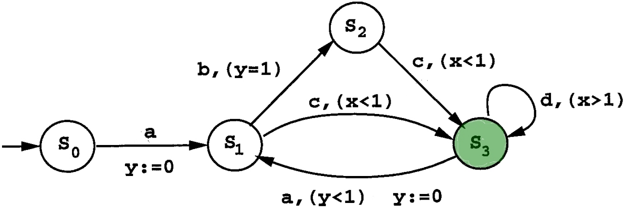
LaBRI, Université Bordeaux 1

Indian Institute of Technology Bombay, India

Verification Seminar

Université Libre de Bruxelles

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 ...

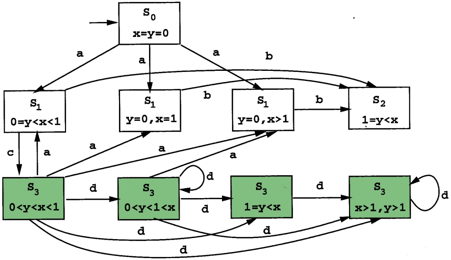
Given a TA, does there **exist** an **accepting run**?

Theorem [AD94, CY92]

This problem is **PSPACE-complete**

First solution to this problem

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



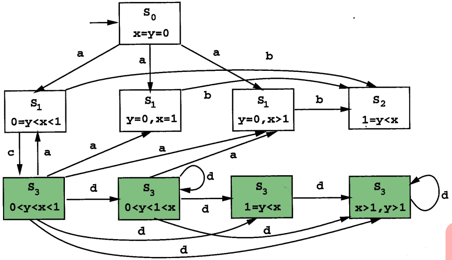
- Region: set of valuations satisfying the **same guards** w.r.t. time
- Finiteness: Parametrized by **maximal constant**

Sound and complete [AD94]

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 [AD94]

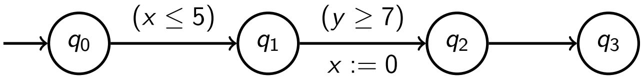
Region graph preserves state **reachability**

A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path

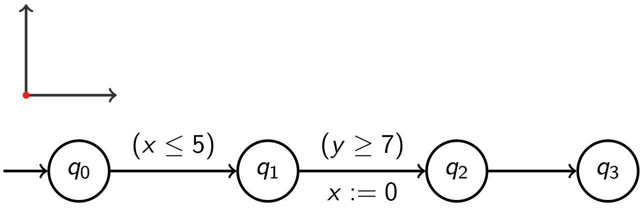
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



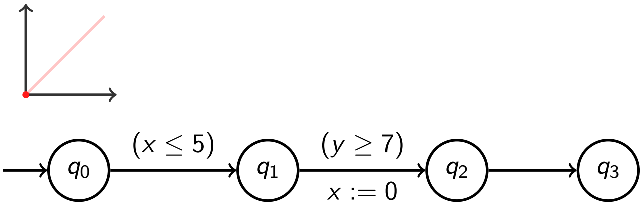
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



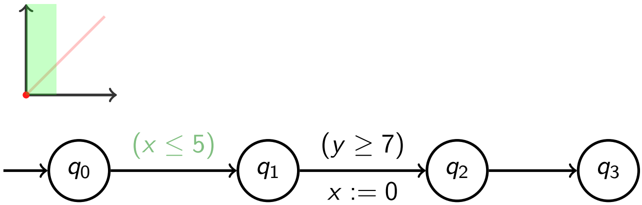
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



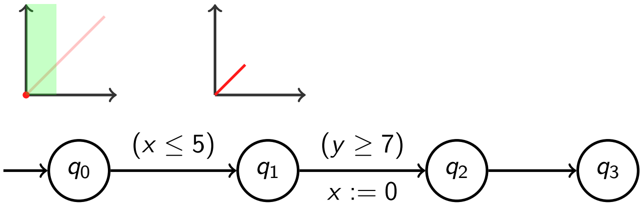
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



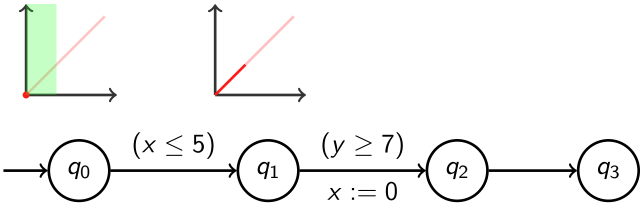
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



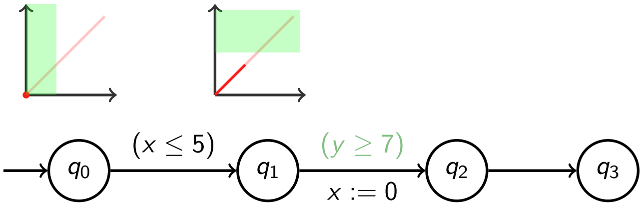
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



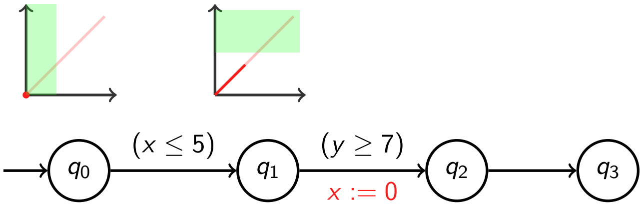
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



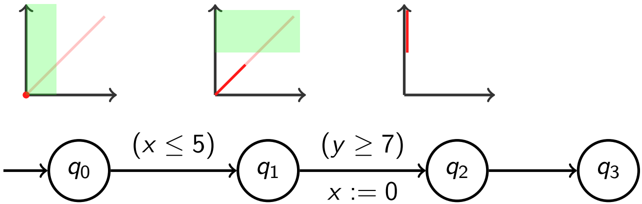
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



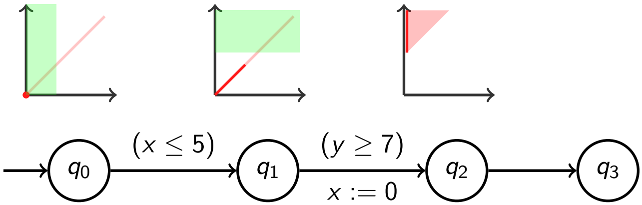
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



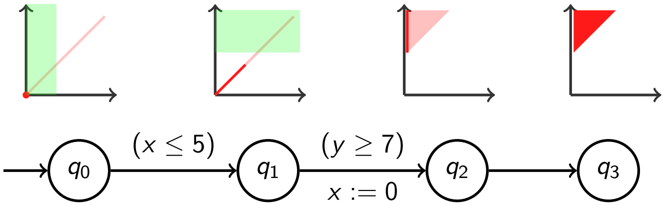
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



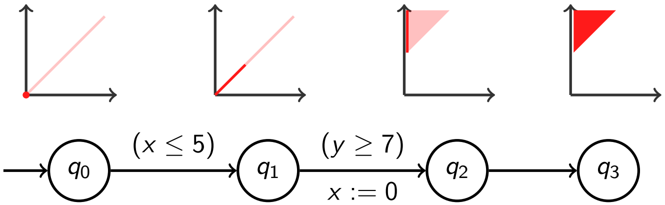
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



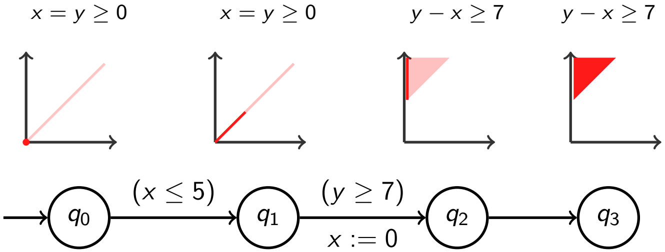
A more efficient solution...

Key idea: Maintain **all valuations** reachable along a path



A more efficient solution...

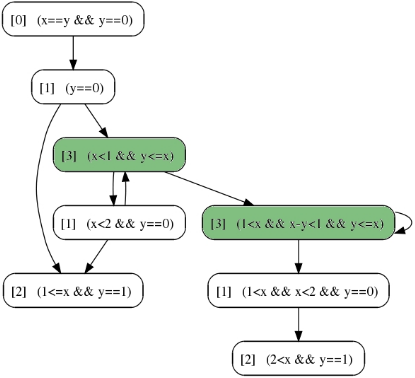
Key idea: Maintain **all valuations** reachable along a path



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

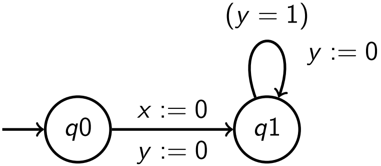


- Zone: set of valuations defined by conjunctions of constraints:
  - $x \sim c$
  - $x - y \sim c$
  - e.g. $(x - y \geq 1) \wedge y < 2$
- Representation: by DBM

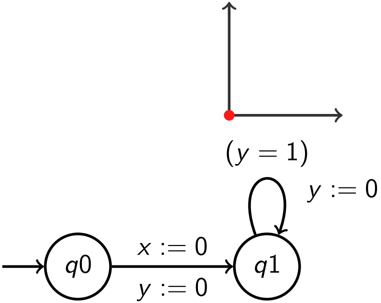
Sound and complete [DT98]

Zone graph preserves state **reachability**

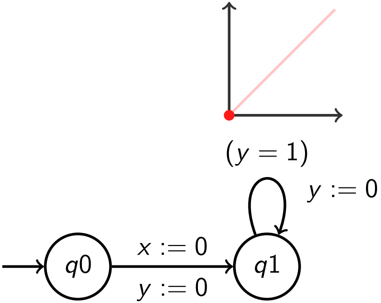
But the zone graph could be infinite ...



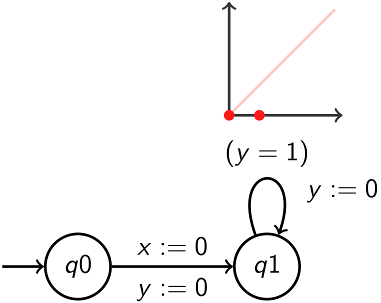
But the zone graph could be infinite ...



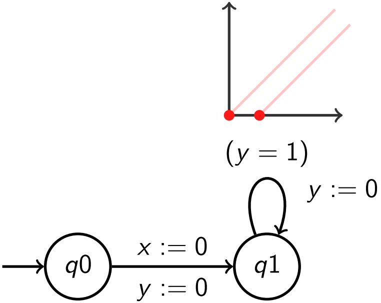
But the zone graph could be infinite ...



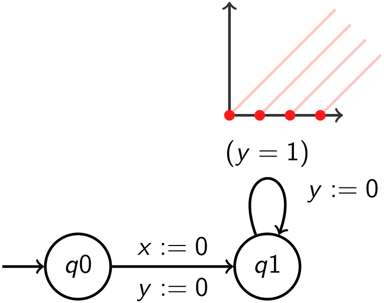
But the zone graph could be infinite ...



But the zone graph could be infinite ...



But the zone graph could be infinite ...



Use finite abstractions

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

$$\begin{tikzcd}
& (q_0, Z_0) \arrow[dl, "/" marking] \arrow[dr] & & \\
{} & & (q_1, Z_1) \arrow[dl, "/" marking] \arrow[dr] & \\
& {} & & (q_2, Z_2)
\end{tikzcd}$$

Use finite abstractions

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

$$
\begin{tikzcd}
 & (q_0, Z_0) \arrow[dl, "/" marking] \arrow[dr] & & \\
 {} & & (q_1, Z_1) \arrow[dl, "/" marking] \arrow[dr] & \\
 & {} & & (q_2, Z_2)
\end{tikzcd}
$$

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

Use finite abstractions

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

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

$$\begin{tikzcd} & {(q_0, \mathfrak{a}(Z_0))} \arrow[dl, "\times" marking] \\ {} & \end{tikzcd}$$

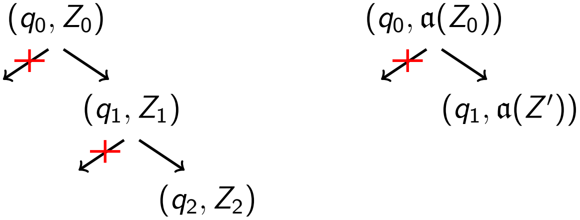
Use finite abstractions

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

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

Use finite abstractions

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



Use finite abstractions

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

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

Use finite abstractions

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

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

Use finite abstractions

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

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

$$(q_1, Z_1) \qquad (q_1, \mathfrak{a}(Z'))$$

$$(q_2, Z_2) \qquad (q_2, \mathfrak{a}(Z''))$$

- Number of **abstracted zones** is **finite**
- **Coarser** abstraction $\rightarrow$ fewer **abstracted zones**

Abstractions in literature [Bou04, BBLP06]

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

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**

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** abstracted zone should be a zone

Abstractions in literature [Bou04, BBLP06]

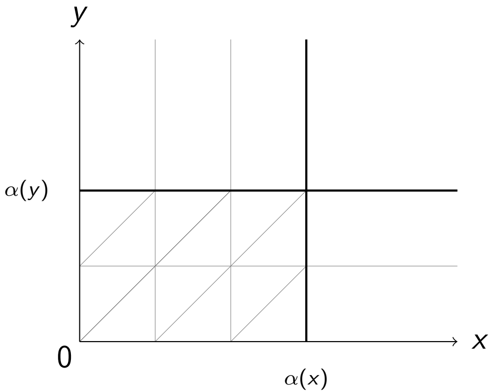
$$
\begin{tikzcd}
 & \mathfrak{a}_{\preccurlyeq LU} & \\
\mathrm{Closure}_{\alpha} \arrow[ur] & & \mathrm{Extra}^{+}_{LU}\ \checkmark \arrow[ul] \\
 & \mathrm{Extra}^{+}_{\alpha}\ \checkmark \arrow[ul] \arrow[ur] & \\
 & \mathrm{Extra}_{\alpha}\ \checkmark \arrow[u] &
\end{tikzcd}
$$

Only convex abstractions in **implementations**!

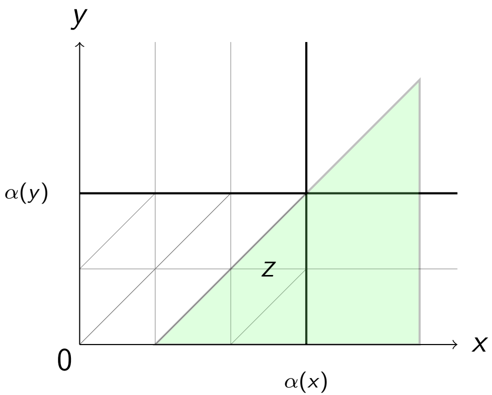
Here...

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

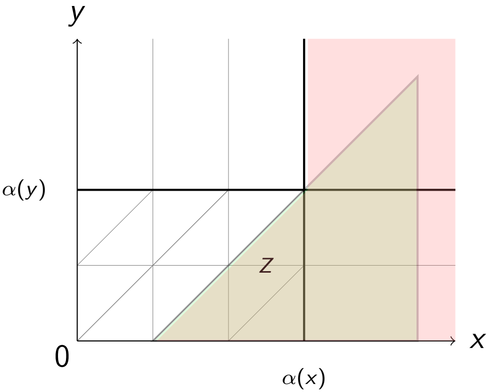
What is Closure$_\alpha$?



What is Closure$_\alpha$?

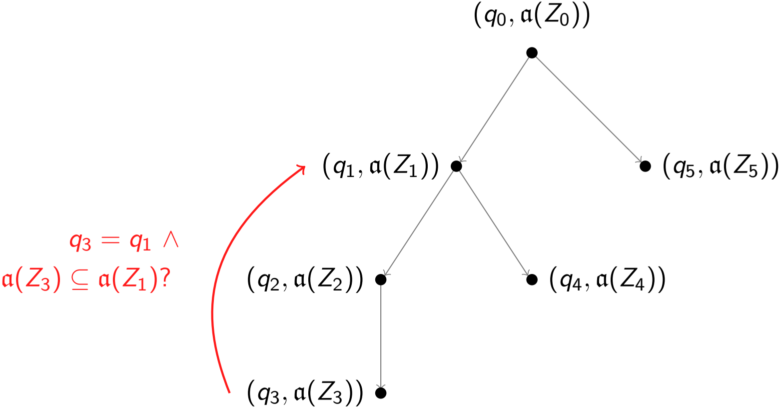


What is Closure$_\alpha$?



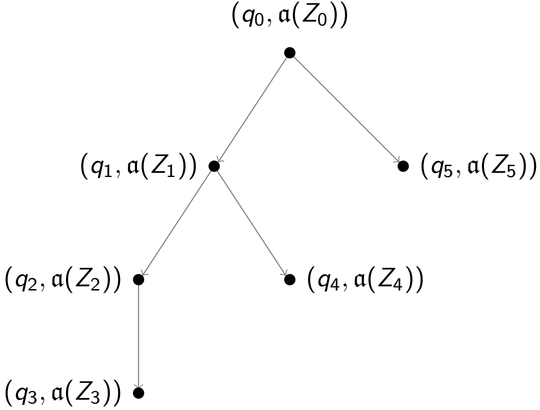
Closure$_\alpha(Z)$: set of regions that Z intersects

Using Closure$_\alpha$ for reachability



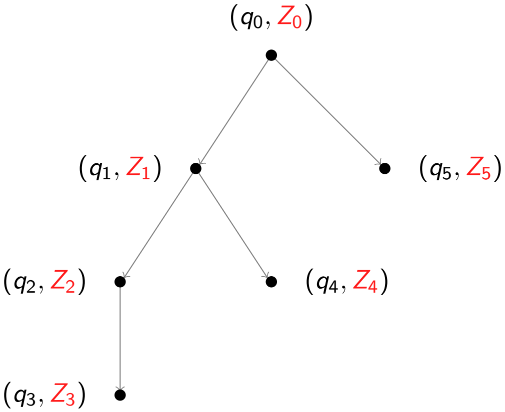
Standard algorithm: **covering tree**

Using Closure$_\alpha$ for reachability



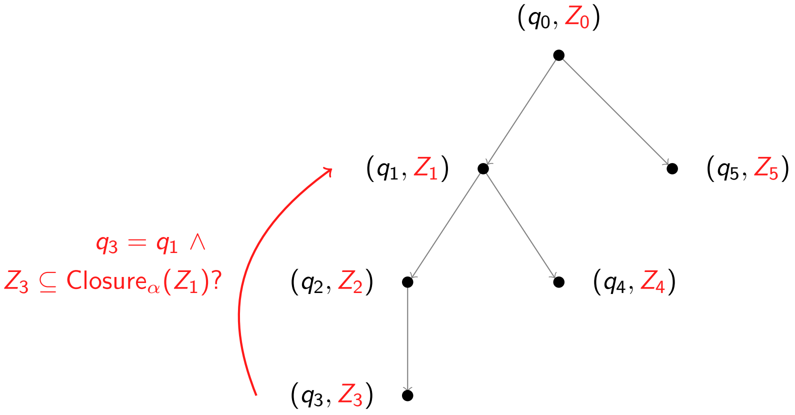
Closure$_\alpha(Z)$ **cannot be** efficiently **stored**

Using Closure$_\alpha$ for reachability



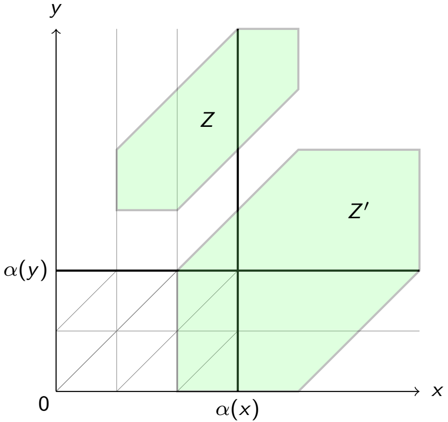
Do **not store** abstracted zones!

Using Closure$_\alpha$ for reachability

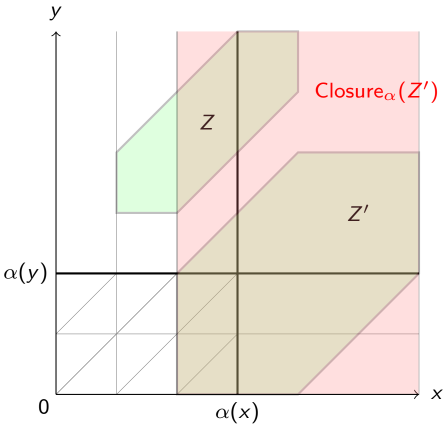


Use Closure for termination!

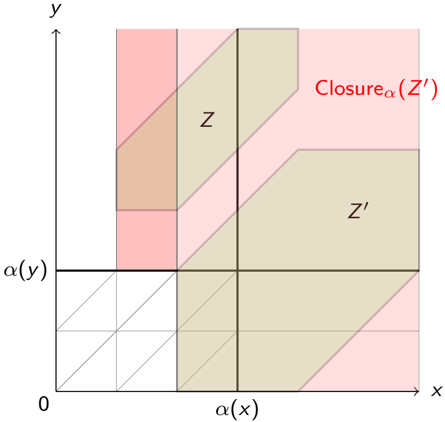
$Z \subseteq \text{Closure}_\alpha(Z')$?



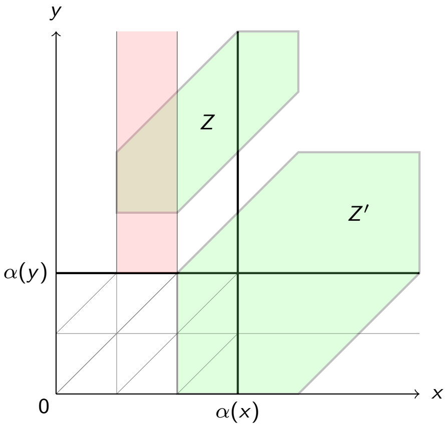
$Z \subseteq \text{Closure}_\alpha(Z')$?



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

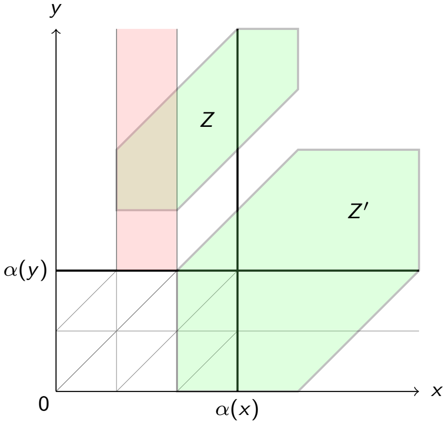


$Z \subseteq \mathrm{Closure}_\alpha(Z')$?



$Z \not\subseteq \mathrm{Closure}_\alpha(Z') \Leftrightarrow \exists R.$ R **intersects Z**, R does **not intersect Z′**

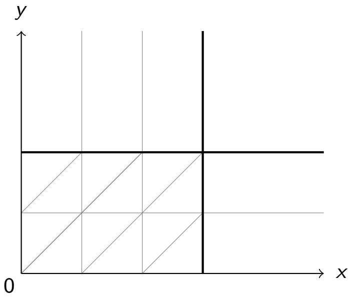
$Z \subseteq \text{Closure}_\alpha(Z')$?



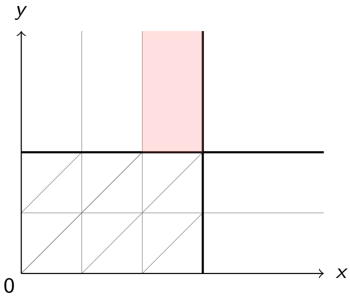
$Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ **intersects Z**, R does **not intersect Z′**

Coming next: An **efficient algorithm** for $Z \not\subseteq \text{Closure}_\alpha(Z')$

Step 1: Representing regions and zones



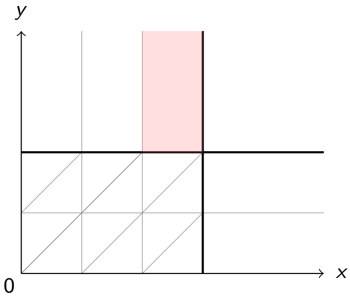
Step 1: Representing regions and zones



$x < 3$ $\qquad$ $y < \infty$

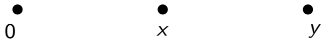
$x > 2$ $\qquad$ $y > 2$

Step 1: Representing regions and zones

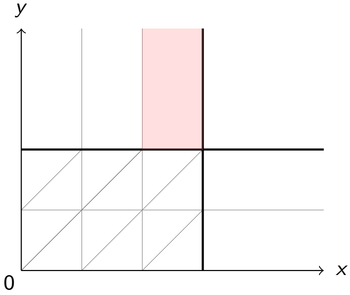


$x < 3$ $\quad$ $y < \infty$

$x > 2$ $\quad$ $y > 2$

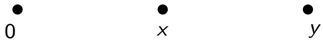


Step 1: Representing regions and zones

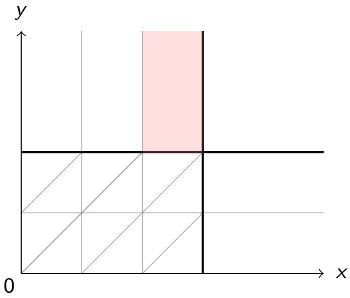


$x < 3$ $\quad y < \infty$

$x > 2$ $\quad y > 2$

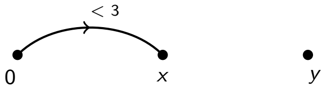


Step 1: Representing regions and zones

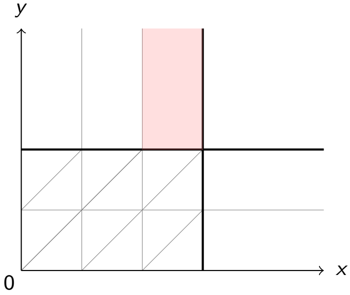


$x - 0 < 3$ $\quad$ $y < \infty$

$x > 2$ $\quad$ $y > 2$

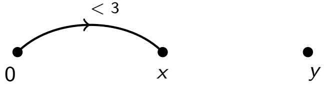


Step 1: Representing regions and zones

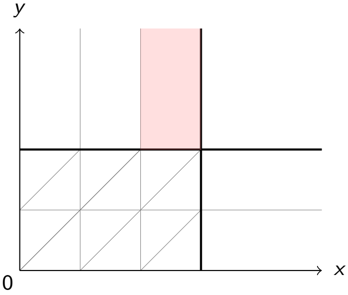


$x - 0 < 3$ $\qquad y < \infty$

$x > 2$ $\qquad y > 2$

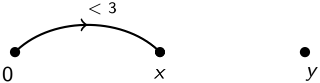


Step 1: Representing regions and zones

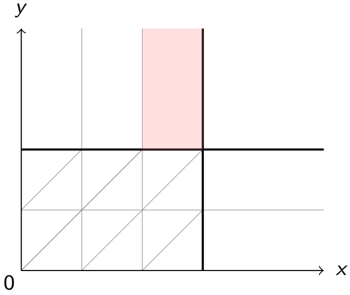


$x - 0 < 3$ $\quad$ $y < \infty$

$0 - x < -2$ $\quad$ $y > 2$

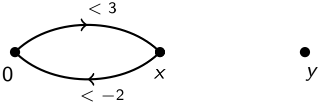


Step 1: Representing regions and zones

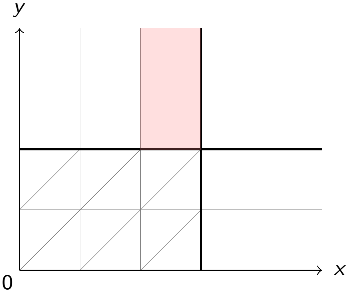


$x - 0 < 3$ $\qquad y < \infty$

$0 - x < -2$ $\qquad y > 2$

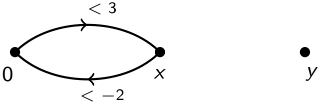


Step 1: Representing regions and zones

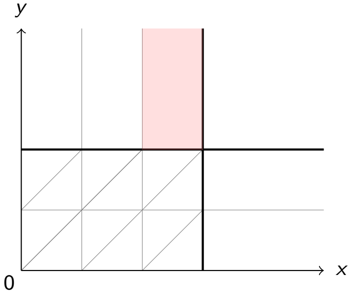


$x - 0 < 3$ $\quad$ $y < \infty$

$0 - x < -2$ $\quad$ $y > 2$

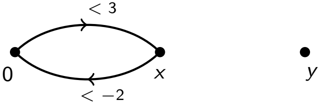


Step 1: Representing regions and zones

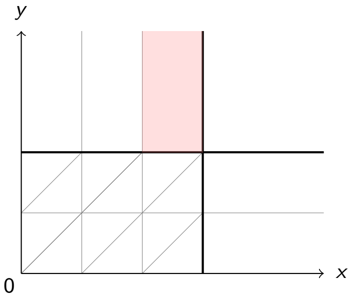


$x - 0 < 3$ $\quad$ $y - 0 < \infty$

$0 - x < -2$ $\quad$ $0 - y < -2$

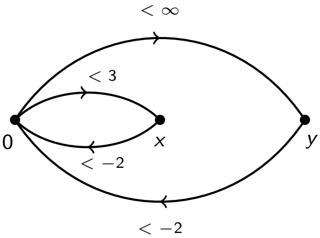


Step 1: Representing regions and zones

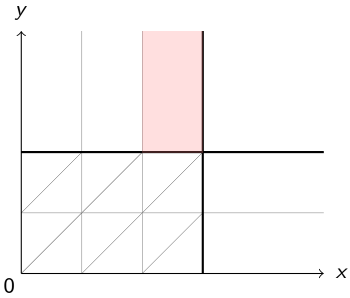


$x - 0 < 3 \qquad y - 0 < \infty$

$0 - x < -2 \qquad 0 - y < -2$

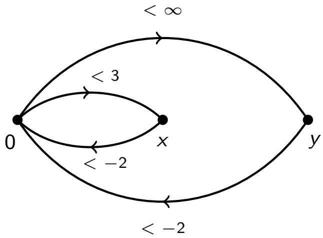


Step 1: Representing regions and zones

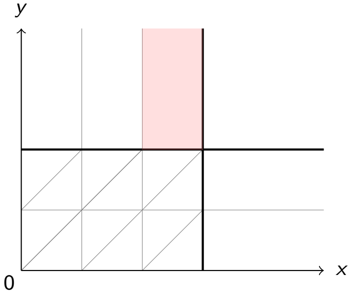


$x - 0 < 3 \qquad y - 0 < \infty$

$0 - x < -2 \qquad 0 - y < -2$

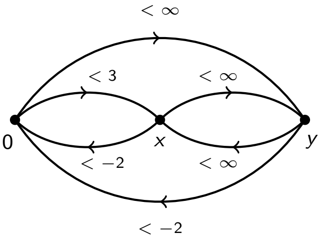


Step 1: Representing regions and zones

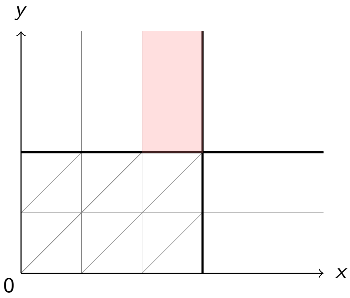


$$x - 0 < 3 \qquad y - 0 < \infty$$

$$0 - x < -2 \qquad 0 - y < -2$$

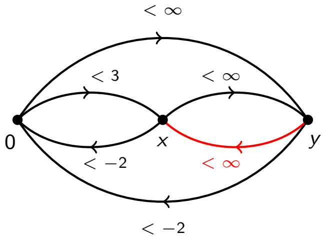


Step 1: Representing regions and zones



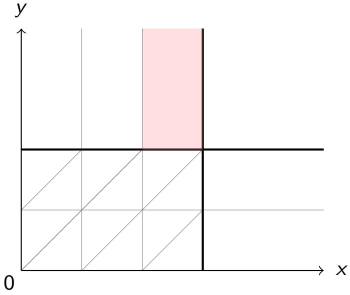
$$x - 0 < 3 \qquad y - 0 < \infty$$

$$0 - x < -2 \qquad 0 - y < -2$$



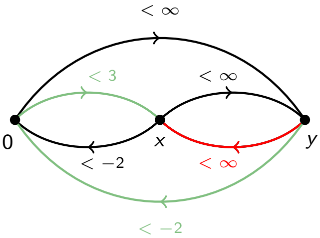
Need a **canonical** representation

Step 1: Representing regions and zones



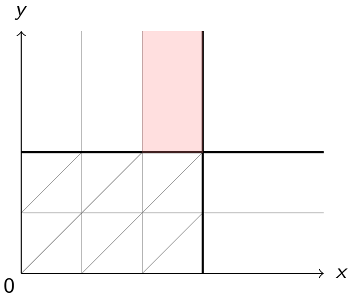
$x - 0 < 3$ $\quad$ $y - 0 < \infty$

$0 - x < -2$ $\quad$ $0 - y < -2$



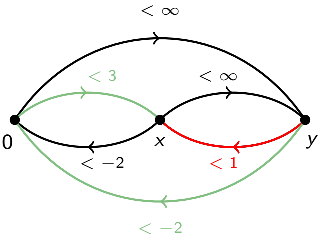
Shortest path should be given by the **direct edge**

Step 1: Representing regions and zones



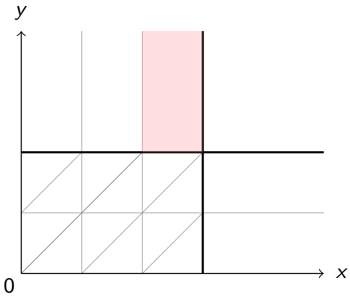
$x - 0 < 3$ $\quad$ $y - 0 < \infty$

$0 - x < -2$ $\quad$ $0 - y < -2$



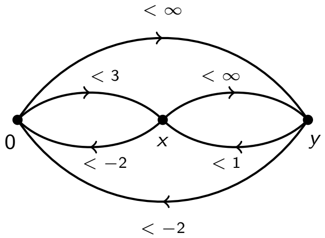
Shortest path should be given by the **direct edge**

Step 1: Representing regions and zones



$x - 0 < 3 \qquad y - 0 < \infty$

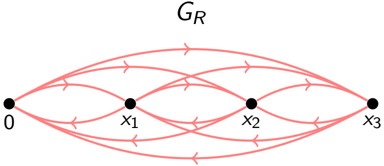
$0 - x < -2 \qquad 0 - y < -2$

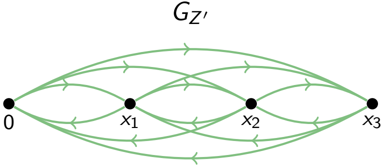


For every zone Z, **canonical distance graph** G_Z

Step 2: When is $R \cap Z'$ empty?

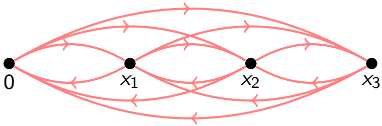
Step 2: When is $R \cap Z'$ empty?



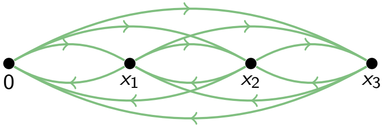


Step 2: When is $R \cap Z'$ empty?

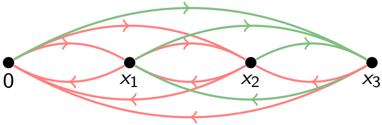
G_R



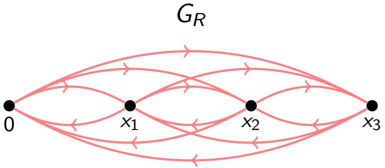
$G_{Z'}$

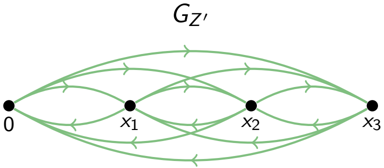


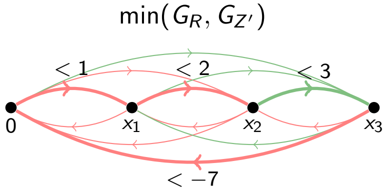
$\min(G_R, G_{Z'})$



Step 2: When is $R \cap Z'$ empty?







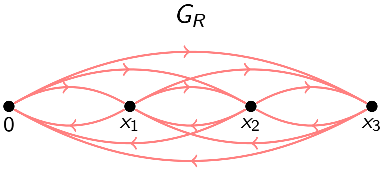
$$\begin{array}{cccc} x_1 & - & 0 & < 1 \\ x_2 & - & x_1 & < 2 \\ x_3 & - & x_2 & < 3 \\ 0 & - & x_3 & < -7 \end{array}$$

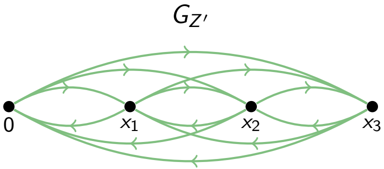
$0 < -1!$

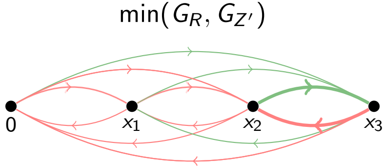
Lemma

$R \cap Z'$ is **empty** $\Leftrightarrow$ $\min(G_R, G_{Z'})$ has a **negative cycle**

Step 2: When is $R \cap Z'$ empty?



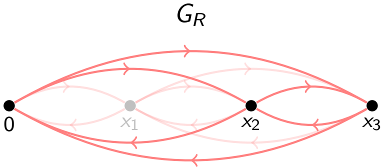


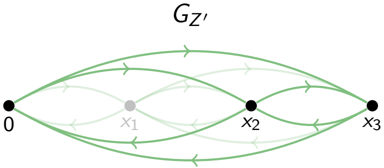


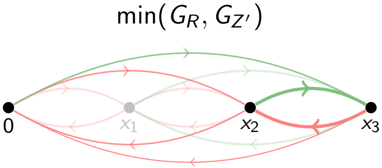
Lemma [Bou04]

$R \cap Z'$ is **empty** $\Leftrightarrow$ $\min(G_R, G_{Z'})$ has a **negative cycle** involving **2 clocks!**

Step 2: When is $R \cap Z'$ empty?





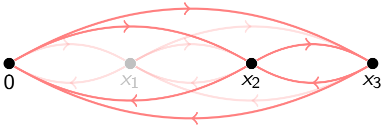


Lemma

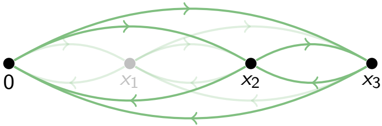
$R \cap Z'$ is **empty** $\Leftrightarrow$ $\min(G_R, G_{Z'})$ has a **negative cycle** involving **2 clocks!**

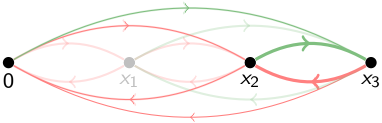
Step 2: When is $R \cap Z'$ empty?

$G_{\mathrm{Proj}_{x_2 x_3}(R)}$



$G_{\mathrm{Proj}_{x_2 x_3}(Z')}$



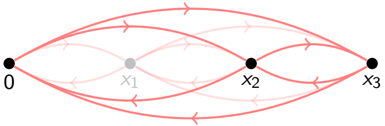


Lemma

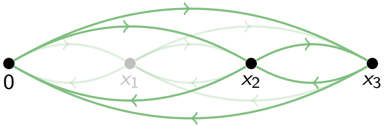
$R \cap Z'$ is **empty** $\quad\Leftrightarrow\quad$ $\min(G_R, G_{Z'})$ has a **negative cycle** involving **2 clocks!**

Step 2: When is $R \cap Z'$ empty?

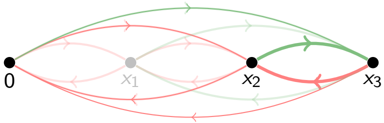
$G_{\mathrm{Proj}_{x_2 x_3}(R)}$



$G_{\mathrm{Proj}_{x_2 x_3}(Z')}$



$\min\left(G_{\mathrm{Proj}_{x_2 x_3}(R)}, G_{\mathrm{Proj}_{x_2 x_3}(Z')}\right)$

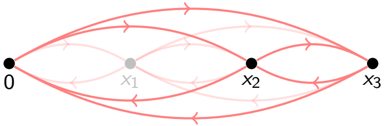


Lemma

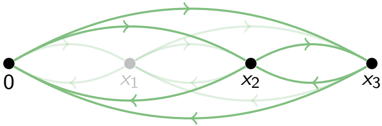
$R \cap Z'$ is **empty** $\Leftrightarrow$ $\min(G_R, G_{Z'})$ has a **negative cycle** involving **2 clocks!**

Step 2: When is $R \cap Z'$ empty?

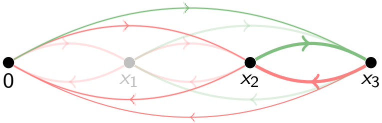
$G_{\mathrm{Proj}_{x_2 x_3}(R)}$



$G_{\mathrm{Proj}_{x_2 x_3}(Z')}$



$\min(G_{\mathrm{Proj}_{x_2 x_3}(R)}, G_{\mathrm{Proj}_{x_2 x_3}(Z')})$



Lemma

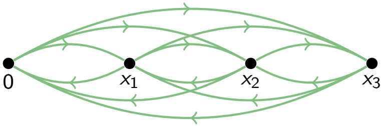
$R \cap Z'$ is **empty** $\Leftrightarrow$ $\exists\, x, y.\ \mathrm{Proj}_{\mathbf{xy}}(R) \cap \mathrm{Proj}_{\mathbf{xy}}(Z')$ is **empty**

Step 3: Reduction to two clocks

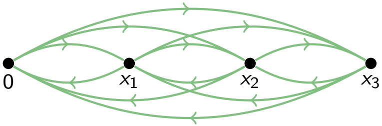
Recall: $Z \not\subseteq \mathrm{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'

Step 3: Reduction to two clocks

Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'



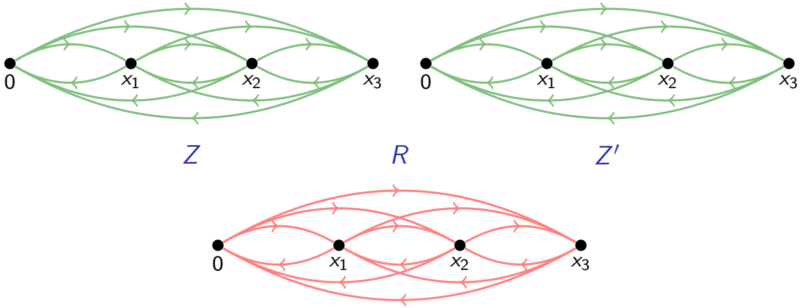
Z



Z'

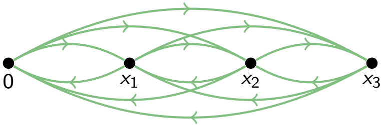
Step 3: Reduction to two clocks

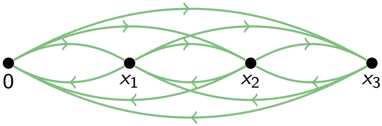
Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'



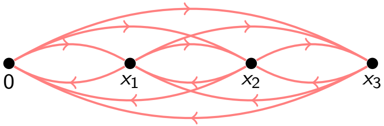
Step 3: Reduction to two clocks

Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'



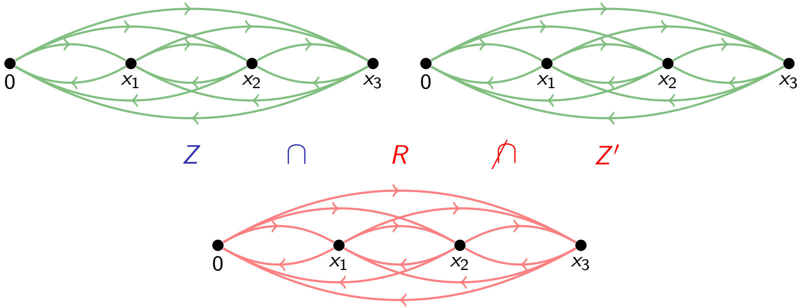


$$Z \quad \cap \quad R \quad \not\cap \quad Z'$$



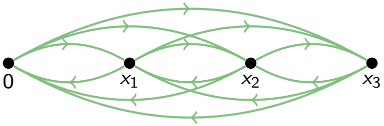
Step 3: Reduction to two clocks

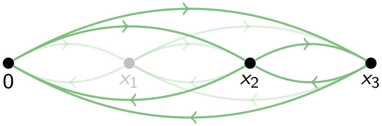
Recall: $Z \not\subseteq \mathrm{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'



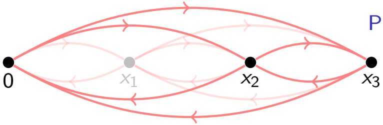
Step 3: Reduction to two clocks

Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'





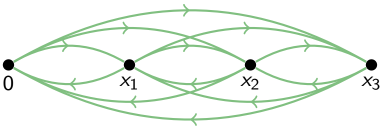
$Z \qquad \cap \qquad R \qquad \not\cap \qquad Z'$

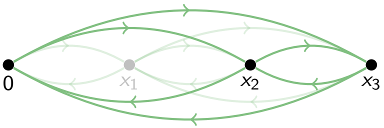


$\text{Proj}_{x_2x_3}(R) \not\cap \text{Proj}_{x_2x_3}(Z')$

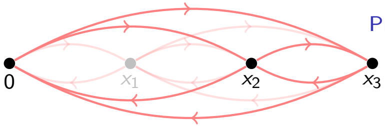
Step 3: Reduction to two clocks

Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'





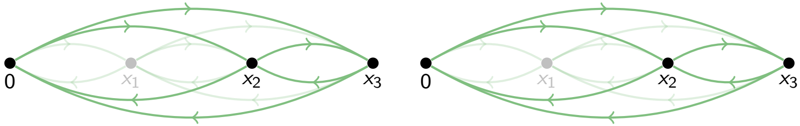
$Z \quad \cap \quad R \quad \not\cap \quad Z'$



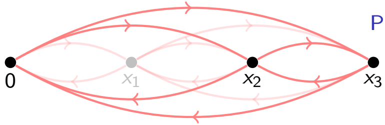
$\text{Proj}_{x_2 x_3}(R) \not\cap \text{Proj}_{x_2 x_3}(Z')$

Step 3: Reduction to two clocks

Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'



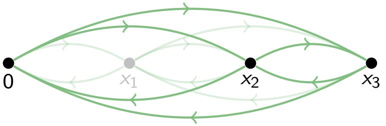
$Z \quad \cap \quad R \quad \not\cap \quad Z'$

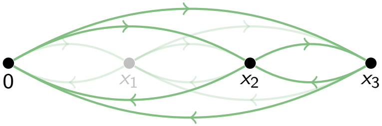


$\text{Proj}_{x_2 x_3}(R) \not\cap \text{Proj}_{x_2 x_3}(Z')$

Step 3: Reduction to two clocks

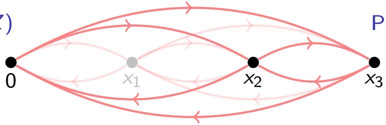
Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'





$Z \quad \cap \quad R \quad \not\cap \quad Z'$

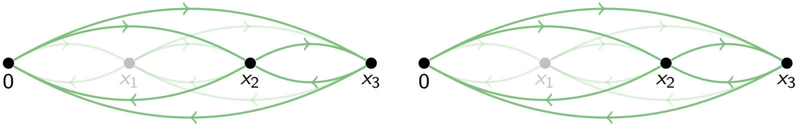
$\text{Proj}_{x_2x_3}(R) \cap \text{Proj}_{x_2x_3}(Z)$



$\text{Proj}_{x_2x_3}(R) \not\cap \text{Proj}_{x_2x_3}(Z')$

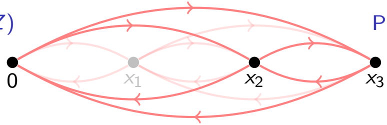
Step 3: Reduction to two clocks

Recall: $Z \not\subseteq \text{Closure}_\alpha(Z') \Leftrightarrow \exists R.\ R$ intersects Z, R does not intersect Z'



$Z \quad \cap \quad R \quad \not\cap \quad Z'$

$\text{Proj}_{x_2x_3}(R) \cap \text{Proj}_{x_2x_3}(Z)$



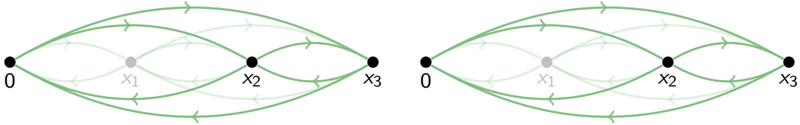
$\text{Proj}_{x_2x_3}(R) \not\cap \text{Proj}_{x_2x_3}(Z')$

Theorem

$Z \not\subseteq \text{Closure}_\alpha(Z')$ if and only if there **exist 2 clocks** x, y s.t.

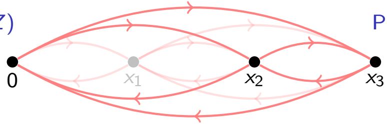
$$\mathbf{Proj_{xy}}(Z) \not\subseteq \text{Closure}_\alpha(\mathbf{Proj}_{xy}(Z'))$$

Step 3: Reduction to two clocks



$Z \quad \cap \quad R \quad \not\cap \quad Z'$

$\text{Proj}_{x_2x_3}(R) \cap \text{Proj}_{x_2x_3}(Z)$



$\text{Proj}_{x_2x_3}(R) \not\cap \text{Proj}_{x_2x_3}(Z')$

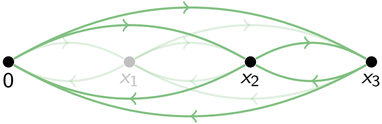
Theorem

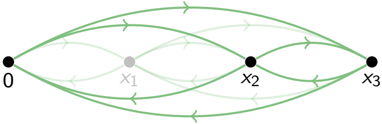
$Z \not\subseteq \text{Closure}_\alpha(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj_{xy}}(Z) \not\subseteq \text{Closure}_\alpha(\mathbf{Proj}_{xy}(Z'))$$

Slightly **modified edge-edge comparison** is enough

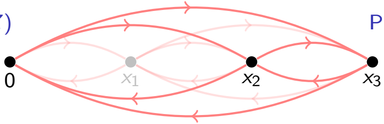
Step 3: Reduction to two clocks





$Z \quad \cap \quad R \quad \not\cap \quad Z'$

$\mathsf{Proj}_{x_2x_3}(R) \cap \mathsf{Proj}_{x_2x_3}(Z)$



$\mathsf{Proj}_{x_2x_3}(R) \not\cap \mathsf{Proj}_{x_2x_3}(Z')$

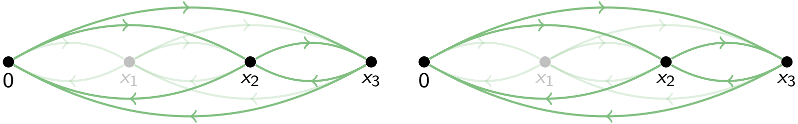
Theorem

$Z \not\subseteq \text{Closure}_\alpha(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj_{xy}}(Z) \not\subseteq \text{Closure}_\alpha(\mathbf{Proj}_{xy}(Z'))$$

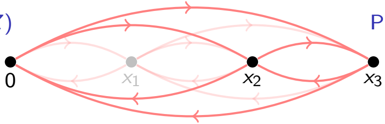
Complexity: $\mathcal{O}(|X|^2)$, where X is the set of clocks

Step 3: Reduction to two clocks



$Z \quad \cap \quad R \quad \not\cap \quad Z'$

$\mathrm{Proj}_{x_2 x_3}(R) \cap \mathrm{Proj}_{x_2 x_3}(Z)$



$\mathrm{Proj}_{x_2 x_3}(R) \not\cap \mathrm{Proj}_{x_2 x_3}(Z')$

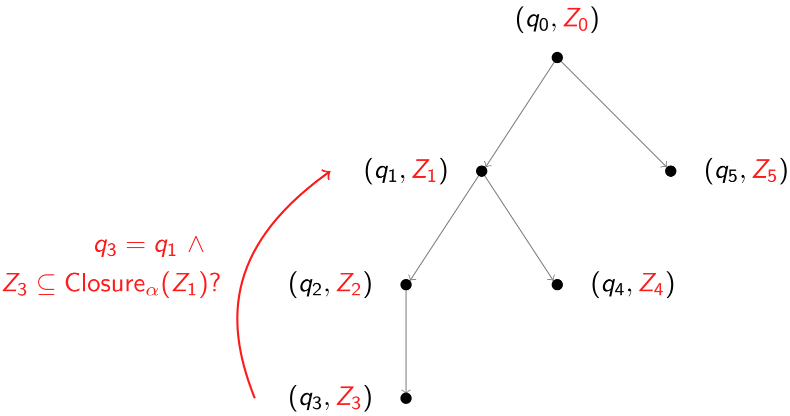
Theorem

$Z \not\subseteq \mathrm{Closure}_\alpha(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj_{xy}}(Z) \not\subseteq \mathrm{Closure}_\alpha(\mathbf{Proj}_{xy}(Z'))$$

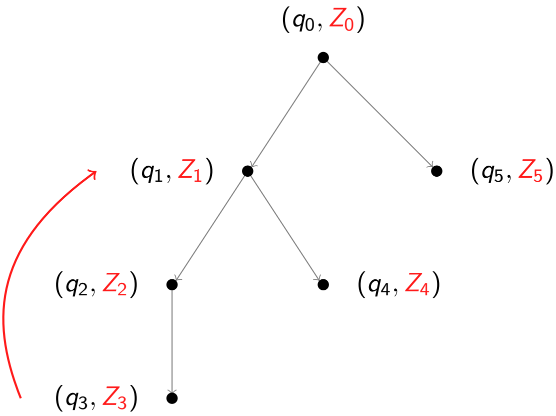
Same complexity as $Z \subseteq Z'$!

So what do we have now...



Efficient algorithm for $Z \subseteq \mathrm{Closure}_\alpha(Z')$

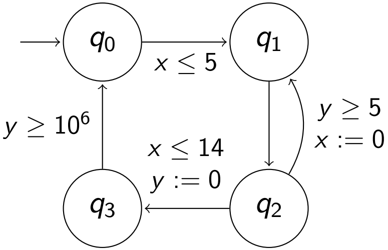
So what do we have now...



$q_3 = q_1 \wedge$
$Z_3 \subseteq \text{Closure}_\alpha(Z_1)$?

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

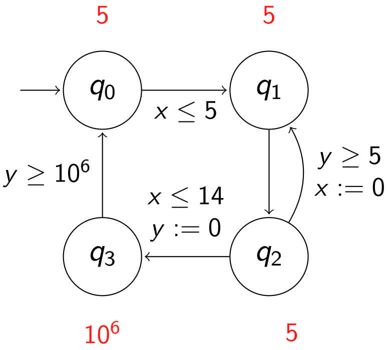
Bound function α



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

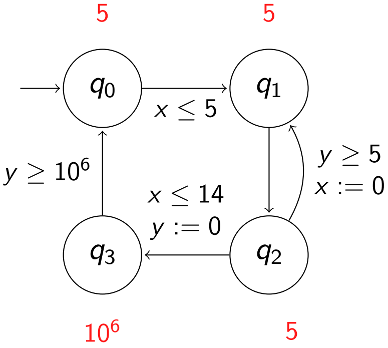
Size of graph $\sim 10^5$

Static analysis: bound function for every q [BBFL03]



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

Static analysis: bound function for every q [BBFL03]

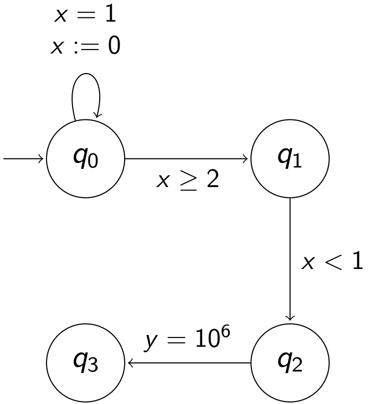


Naive: $\alpha(x) = 14$, $\alpha(y) = 10^6$

But this is not enough!

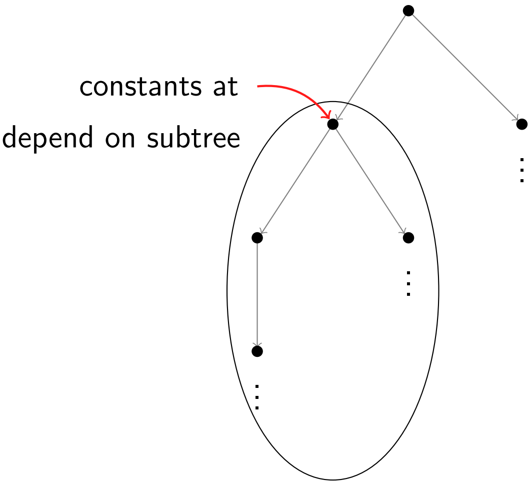
Need to look at semantics...

Static analysis: $\alpha(y) = 10^6$



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

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



Constant propagation

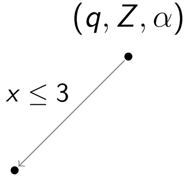
$\alpha(x) = -\infty$

$$(q, Z, \alpha)$$

•

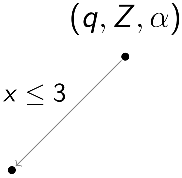
Constant propagation

$\alpha(x) = -\infty$



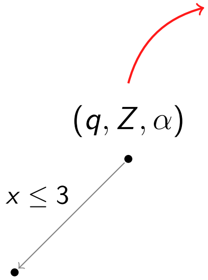
Constant propagation

$\alpha(x) = \ 3$



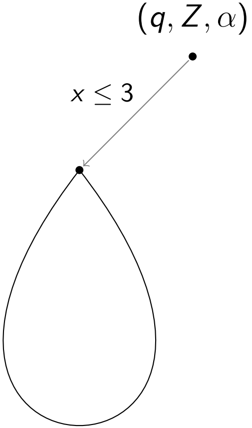
Constant propagation

$\alpha(x) = \ 3$



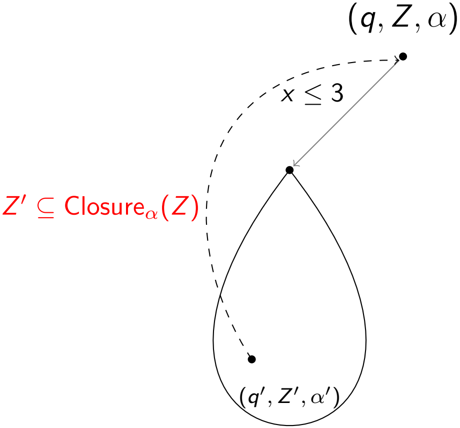
Constant propagation

$\alpha(x) = \quad 5$



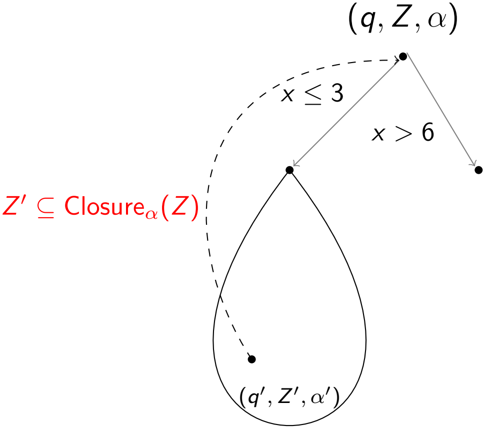
Constant propagation

$\alpha(x) = \quad 5$

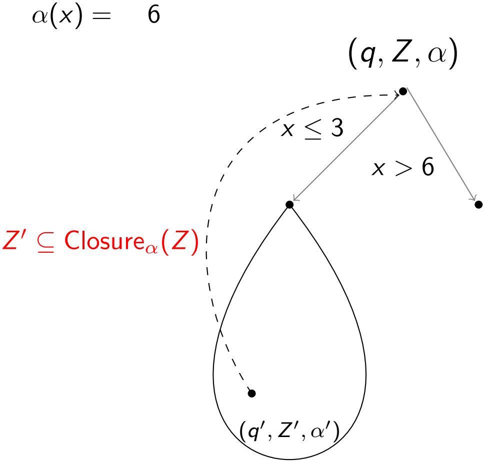


Constant propagation

$\alpha(x) = \quad 5$

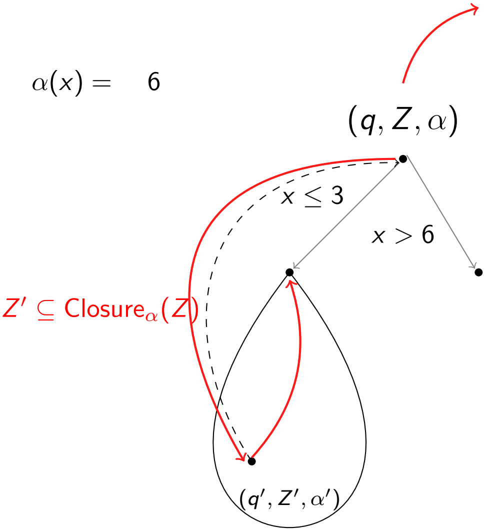


Constant propagation



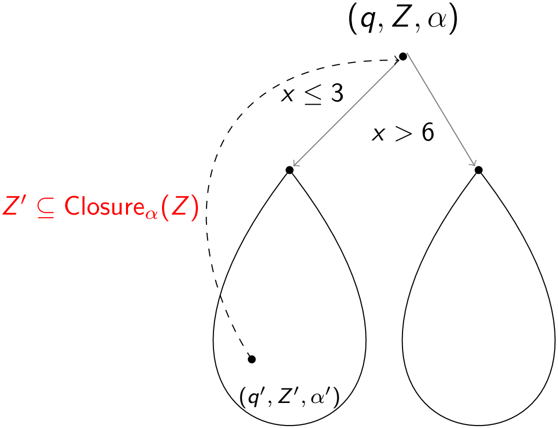
Constant propagation

$\alpha(x) = \quad 6$



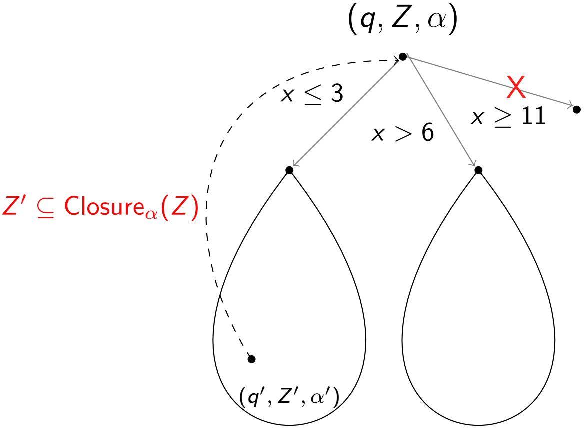
Constant propagation

$\alpha(x) = \quad 6$



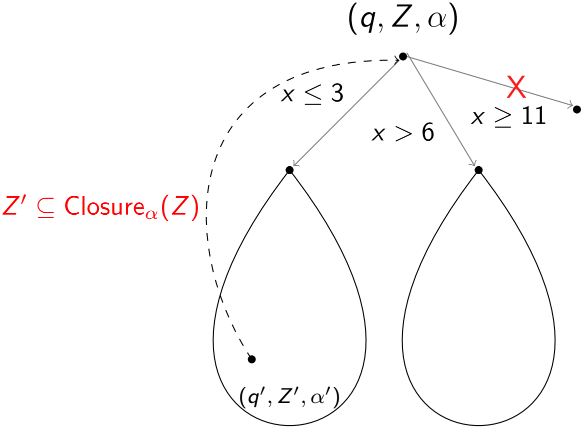
Constant propagation

$\alpha(x) = \quad 6$



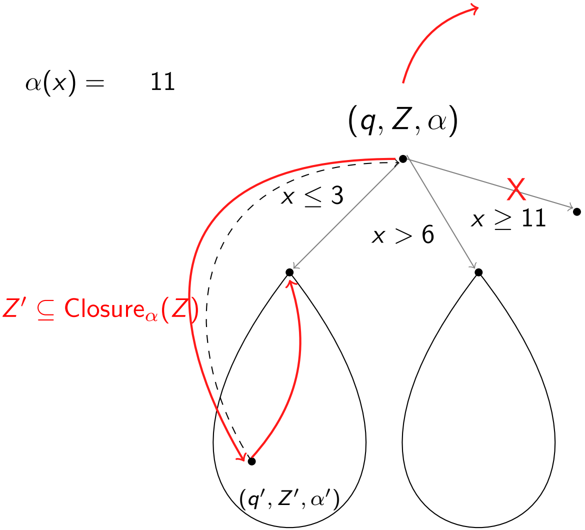
Constant propagation

$\alpha(x) = \quad 11$



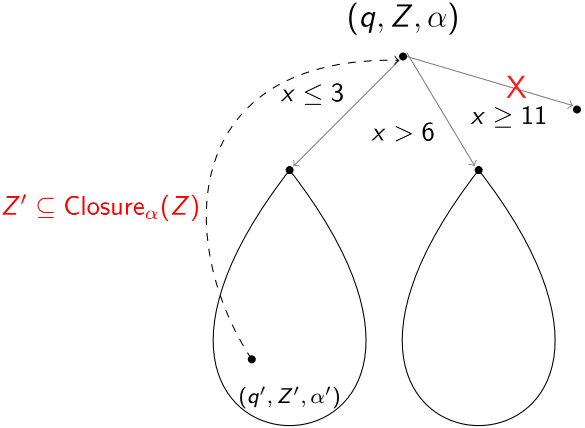
Constant propagation

$\alpha(x) = \quad 11$



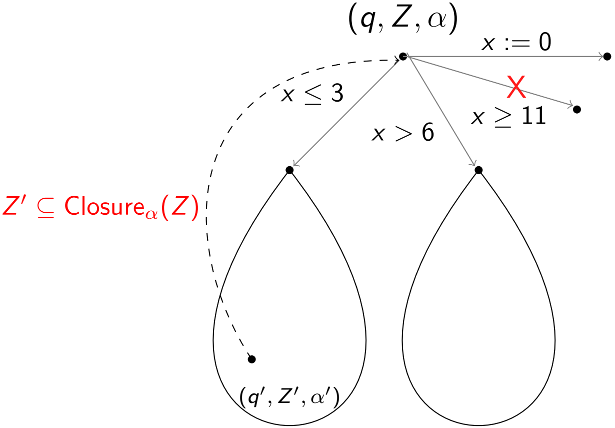
Constant propagation

$\alpha(x) = \quad 11$



Constant propagation

$\alpha(x) = \quad 11$



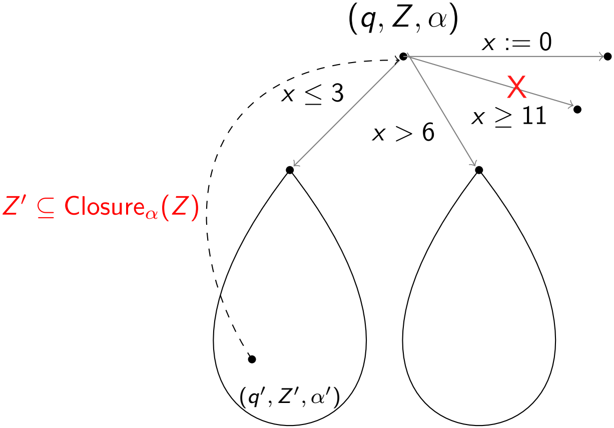
Constant propagation

All **tentative nodes** consistent

+ No more **exploration**

→ Terminate!

$\alpha(x) = \quad 11$



Invariants on the bounds

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

Invariants on the bounds

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

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 $ZG(\mathcal{A})$: $Z \subseteq \mathrm{Closure}_{\alpha'}(Z')$ for **termination**
- **Bounds** α calculated **on-the-fly**
- Abstraction Extra^{+}_{LU} can **also** be **handled**:

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

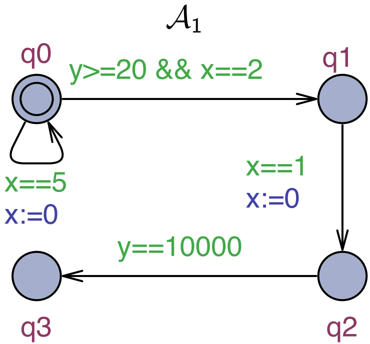
An **efficient** $\mathcal{O}(|X|^2)$ procedure for $Z \subseteq \mathrm{Closure}_{\alpha}(\mathit{Extra}^{+}_{LU}(Z'))$!

Benchmarks

Model	Our algorithm		UPPAAL's algorithm		UPPAAL 4.1.3 (-n4 -C -o1)	
	nodes	s.	nodes	s.	nodes	s.
CSMA/CD7	5031	0.32	5923	0.27	–	T.O.
CSMA/CD8	16588	1.36	19017	1.08	–	T.O.
CSMA/CD9	54439	6.01	60783	4.19	–	T.O.
FDDI10	459	0.02	525	0.06	12049	2.43
FDDI20	1719	0.29	2045	0.78	–	T.O.
FDDI30	3779	1.29	4565	4.50	–	T.O.
Fischer7	7737	0.42	18374	0.53	18374	0.35
Fischer8	25080	1.55	85438	2.48	85438	1.53
Fischer9	81035	5.90	398685	12.54	398685	8.95
Fischer10	–	T.O.	–	T.O.	1827009	53.44

- **Extra**$^+_{LU}$ and **static** analysis bounds in UPPAAL
- **Closure**$_\alpha$**(Extra**$^+_{LU}$**)** and **otf** bounds in our algorithm

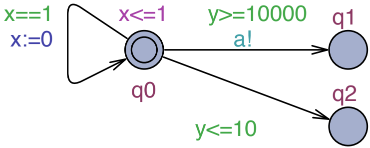
Experiments I



$\mathcal{A}_1$	nodes	s.
Our algorithm	7	0.0
UPPAAL's algorithm	2003	0.60
UPPAAL 4.1.3	2003	0.01

Experiments II

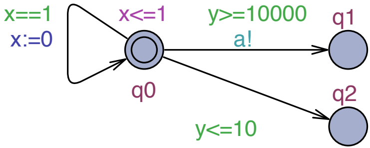
$\mathcal{A}_2$



$\mathcal{A}_2$	nodes	s.
Our algorithm	2	0.0
UPPAAL's algorithm	10003	0.07
UPPAAL 4.1.3	10003	0.07

Experiments II

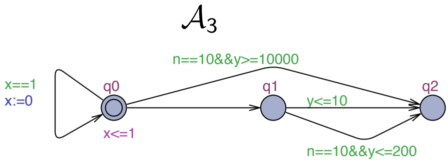
$\mathcal{A}_2$



$\mathcal{A}_2$	nodes	s.
Our algorithm	2	0.0
UPPAAL's algorithm	10003	0.07
UPPAAL 4.1.3	10003	0.07

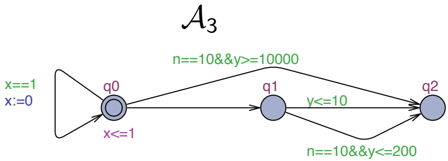
Occurs in **CSMA/CD**!

Experiments III



$\mathcal{A}_3$	nodes	s.
Our algorithm	3	0.0
UPPAAL's algorithm	10004	0.37
UPPAAL 4.1.3	10004	0.32

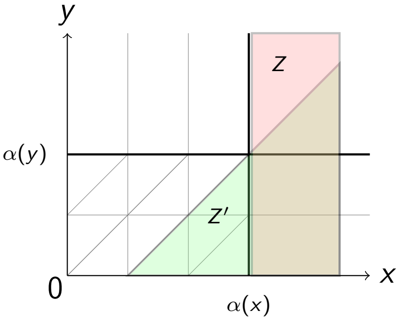
Experiments III



$\mathcal{A}_3$	nodes	s.
Our algorithm	3	0.0
UPPAAL's algorithm	10004	0.37
UPPAAL 4.1.3	10004	0.32

Occurs in **Fischer**!

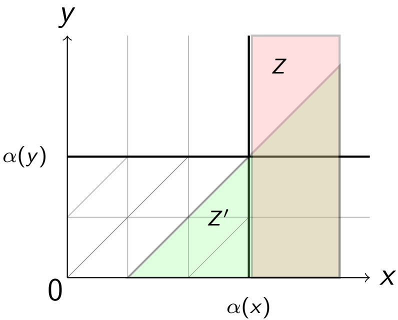
Experiments IV



Z' : $x - y \geq 1$

Z : $x > \alpha(x)$

Experiments IV



$Z' : \; x - y \geq 1$
$Z : \; x > \alpha(x)$

Occurs in **FDDI**!

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
- Propagating **more** than constants
- Automata with **diagonal** constraints

References

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.

C. Courcoubetis and M. Yannakakis.
Minimum and maximum delay problems in real-time systems.
Form. Methods Syst. Des., 1(4):385–415, 1992.

C. Daws and S. Tripakis.
Model checking of real-time reachability properties using abstractions.
In *TACAS'98*, volume 1384 of *LNCS*, pages 313–329. Springer, 1998.