Better abstractions for timed automata

B. Srivathsan[1]

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

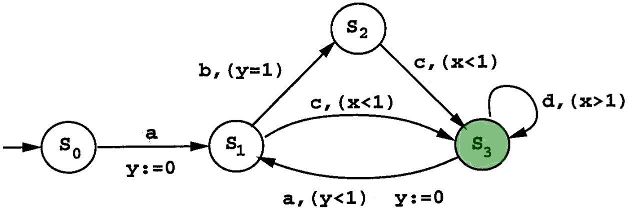
LaBRI, Université Bordeaux 1

IIT Bombay / UIUC

LSV-ENS Cachan

February 2012

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 ...

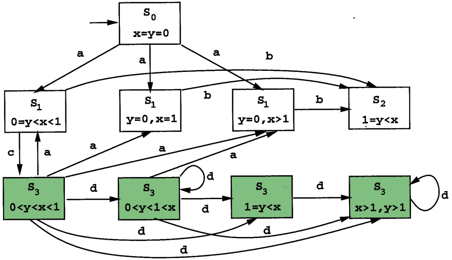
Given a TA, does there **exist** an **accepting run**?

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**



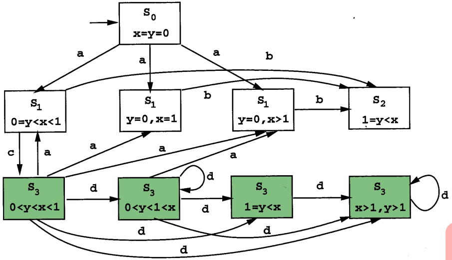
- Region: set of valuations satisfying the **same guards** w.r.t. time
- Finiteness: Parametrized by **maximal constant**

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**

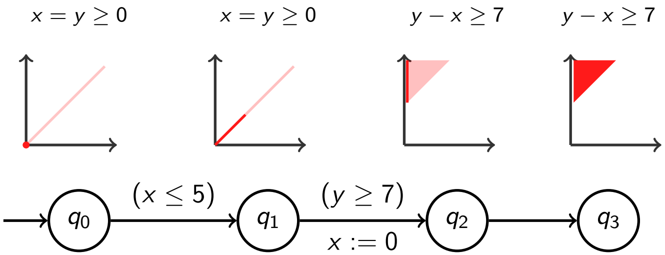
$\mathcal{O}(|X|!.M^{|X|})$ many regions!

Sound and complete [AD94]

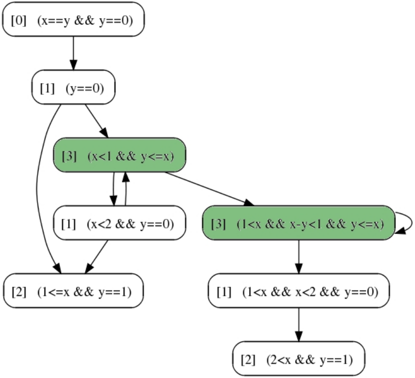
Region graph preserves state **reachability**

A more efficient solution...

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



Zones and zone graph

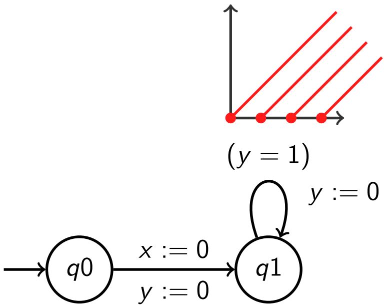


- Zone: set of valuations defined by conjunctions of constraints:
  - $x \sim c$
  - $x - y \sim c$
  - e.g. $(x - y \geq 1) \wedge y < 2$
- Representation: by DBM

Sound and complete [DT98]

Zone graph preserves state **reachability**

But the zone graph could be infinite ...



Use finite abstractions

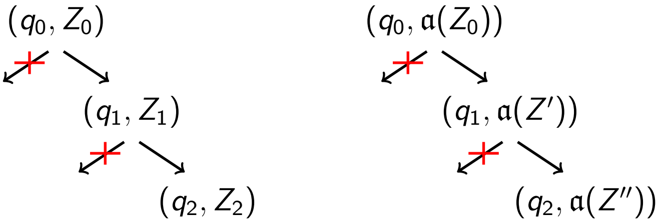
Key idea: **Abstract** each zone in a **sound** manner

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

- Number of **abstracted zones** is **finite**

Use finite abstractions

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



- Number of **abstracted zones** is **finite**
- **Coarser** abstraction $\rightarrow$ smaller **abstract zone graph**

Abstractions in literature [Bou04, BBLP06]

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

Sound and complete

All the above abstractions preserve state **reachability**

Abstractions in literature [Bou04, BBLP06]

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

Sound and complete

All the above abstractions preserve state **reachability**

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

Abstractions in literature [Bou04, BBLP06]

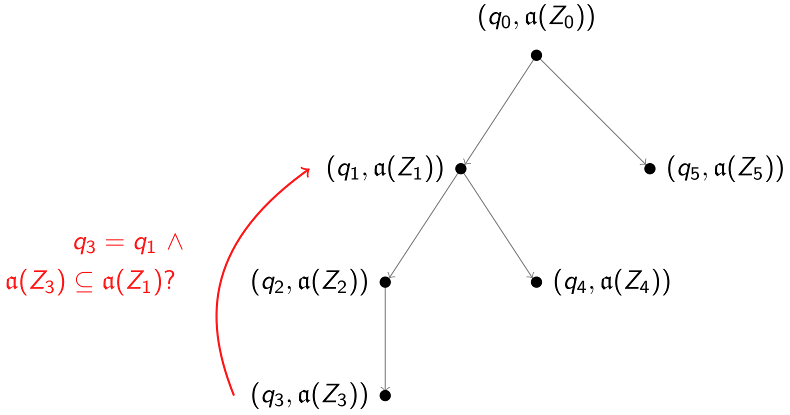
$$\begin{tikzcd}
 & \mathfrak{a}_{\preccurlyeq LU} & \\
\mathrm{Closure}_{\alpha} \arrow[ur] & & \mathrm{Extra}^{+}_{LU} \checkmark \arrow[ul] \\
 & \mathrm{Extra}^{+}_{\alpha} \checkmark \arrow[ul] \arrow[ur] & \\
 & Z \arrow[u] &
\end{tikzcd}$$

Only convex abstractions in **implementations**!

Here...

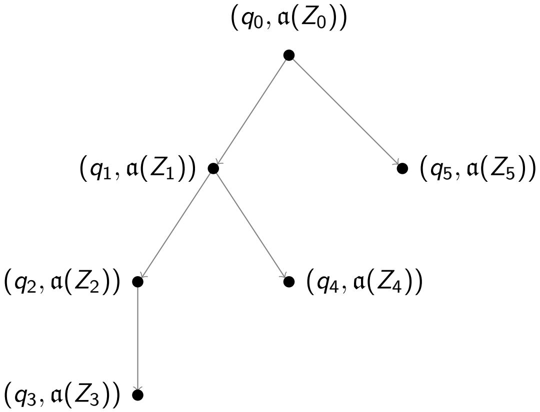
Efficient use of the **non-convex** $\mathfrak{a}_{\preccurlyeq_{LU}}$ abstraction!

Using $\mathfrak{a}_{\preccurlyeq LU}$ for reachability



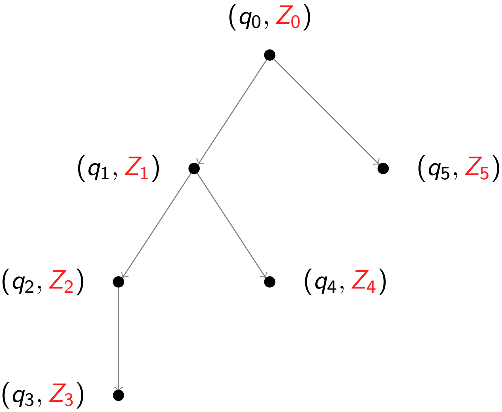
Standard algorithm: **covering tree**

Using $\mathfrak{a}_{\preccurlyeq_{LU}}$ for reachability



$\mathfrak{a}_{\preccurlyeq_{LU}}(Z)$ **cannot be** efficiently **stored**

Using $\mathfrak{a}_{\preccurlyeq LU}$ for reachability

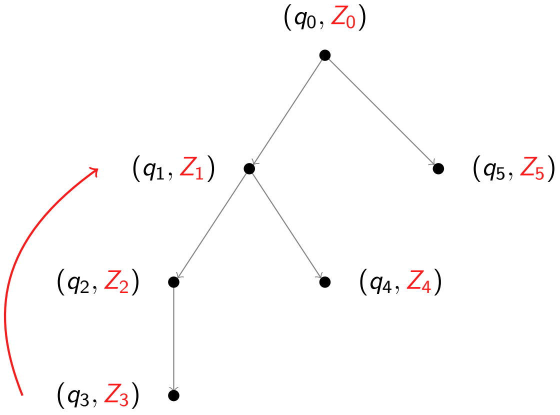


Do **not store** abstracted zones!

Using $\mathfrak{a}_{\preccurlyeq LU}$ for reachability

$q_3 = q_1 \ \wedge$

$Z_3 \subseteq \mathfrak{a}_{\preccurlyeq LU}(Z_1)$?



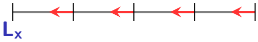
Use $\mathfrak{a}_{\preccurlyeq LU}$ for termination!

Missing piece...

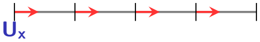
Efficient $Z \subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x} :=$ max c over guards $x > c$ and $x \geq c$

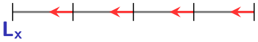


$\mathbf{U_x} :=$ max c over guards $x < c$ and $x \leq c$

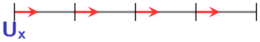


What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x} :=$ max c over guards $x > c$ and $x \geq c$



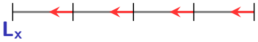
$\mathbf{U_x} :=$ max c over guards $x < c$ and $x \leq c$



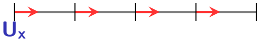
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v'
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x}$:= max c over guards $x > c$ and $x \geq c$



$\mathbf{U_x}$:= max c over guards $x < c$ and $x \leq c$



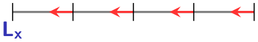
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$

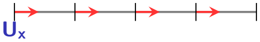
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x}$:= max c over guards $x > c$ and $x \geq c$

$\mathbf{U_x}$:= max c over guards $x < c$ and $x \leq c$





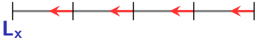
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$
  - $v(x) > v'(x)$

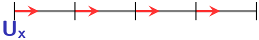
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x}$:= max c over guards $x > c$ and $x \geq c$

$\mathbf{U_x}$:= max c over guards $x < c$ and $x \leq c$





bad guard!

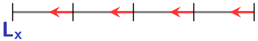
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$
  - $v(x) > v'(x)$

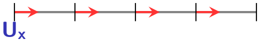
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x}$:= max c over guards $x > c$ and $x \geq c$

$\mathbf{U_x}$:= max c over guards $x < c$ and $x \leq c$



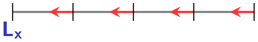


- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$
  - $v(x) > v'(x) \quad \Rightarrow \quad v'(x) > L_x,$ and

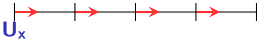
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x}$:= max c over guards $x > c$ and $x \geq c$



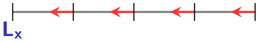
$\mathbf{U_x}$:= max c over guards $x < c$ and $x \leq c$



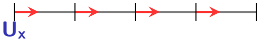
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$
  - $v(x) > v'(x) \quad \Rightarrow \quad v'(x) > L_x$, and
  - $v(x) < v'(x)$
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x}$:= max c over guards $x > c$ and $x \geq c$



$\mathbf{U_x}$:= max c over guards $x < c$ and $x \leq c$

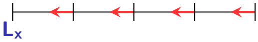


bad guard!

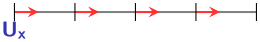
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$
  - $v(x) > v'(x) \quad \Rightarrow \quad v'(x) > L_x$, and
  - $v(x) < v'(x)$
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

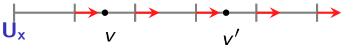
What is $\mathfrak{a}_{\preccurlyeq_{LU}}$? [BBLP06]

$\mathbf{L_x} :=$ max c over guards $x > c$ and $x \geq c$



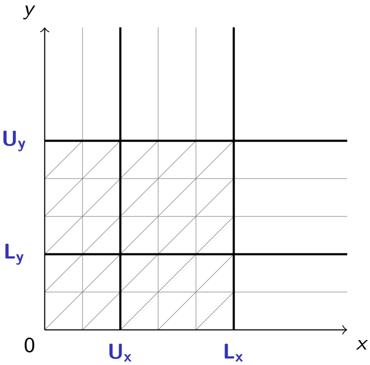
$\mathbf{U_x} :=$ max c over guards $x < c$ and $x \leq c$





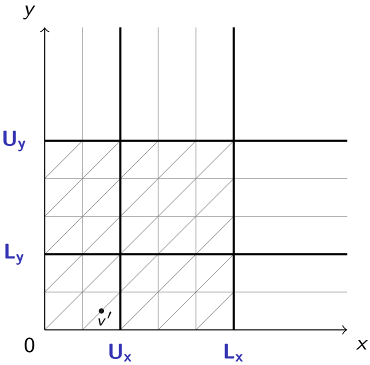
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: v is **simulated by** v' when for all $x \in X$
  - $v(x) > v'(x) \quad \Rightarrow \quad v'(x) > L_x, \qquad$ and
  - $v(x) < v'(x) \quad \Rightarrow \quad v(x) > U_x$
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z) := \{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

Example



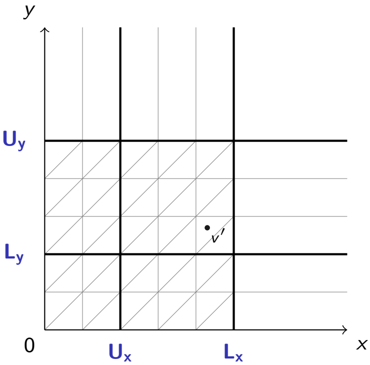
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Example



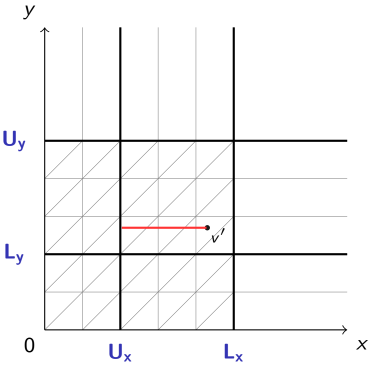
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Example



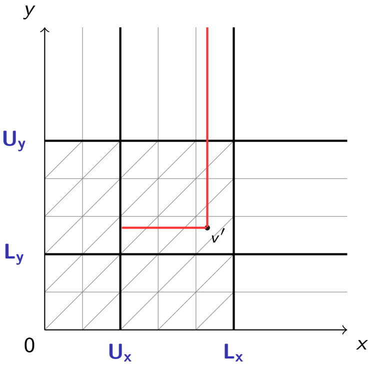
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Example



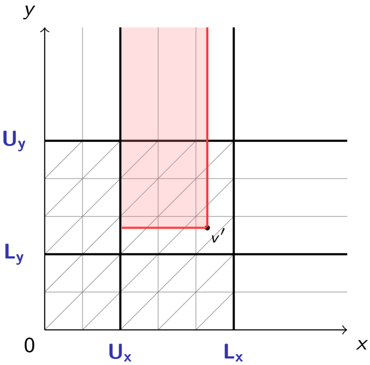
- $\mathbf{v} \preccurlyeq_{LU} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$ ✓

Example



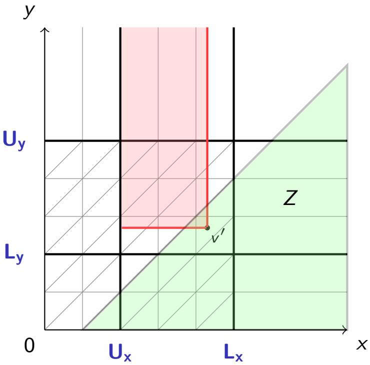
- $\mathbf{v} \preccurlyeq_{LU} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ ✓ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Example



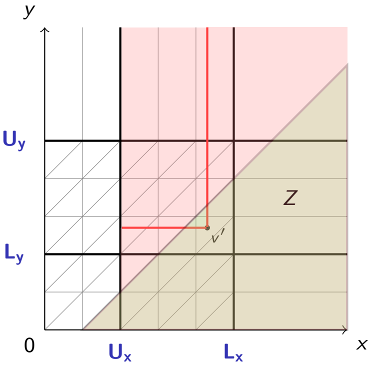
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Example



- $\mathbf{v} \preccurlyeq_{LU} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Example



- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$
- $\mathfrak{a}_{\preccurlyeq_{LU}}(Z)$: $\{v \mid \exists v' \in Z \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

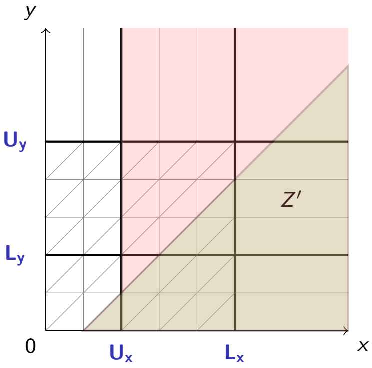
Coming next...

$$Z \subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$$

Step 1: Focus on regions

Union of regions

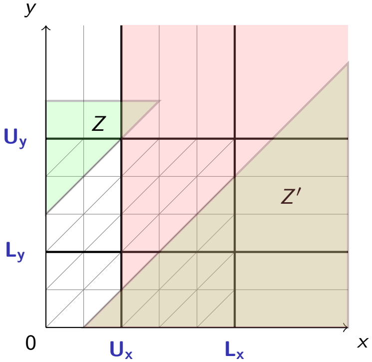
Every **region** R that **intersects** $\mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ is **included** in $\mathfrak{a}_{\preccurlyeq_{LU}}(Z')$.



Step 1: Focus on regions

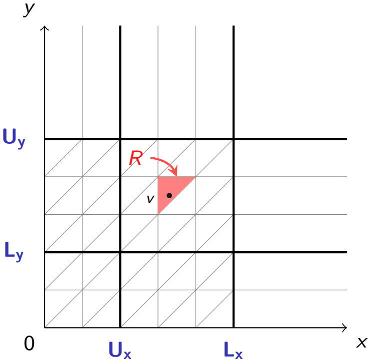
Union of regions

Every **region** R that **intersects** $\mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ is **included** in $\mathfrak{a}_{\preccurlyeq_{LU}}(Z')$.



$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z') \quad \Leftrightarrow \quad \exists R.\ R$ **intersects** Z, R **not included** in $\mathfrak{a}_{\preccurlyeq_{LU}}(Z')$

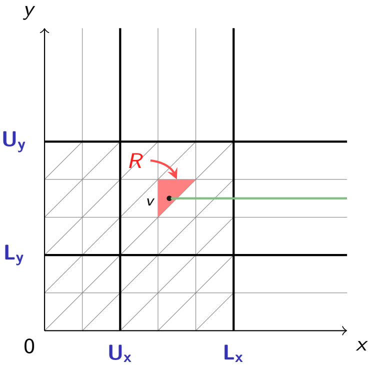
Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$?



- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Collect all v' that **simulate** v

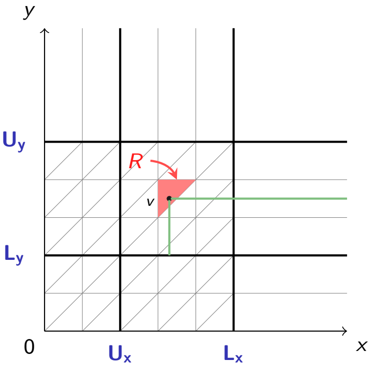
Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$?



- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$ ✓

Collect all v' that **simulate** v

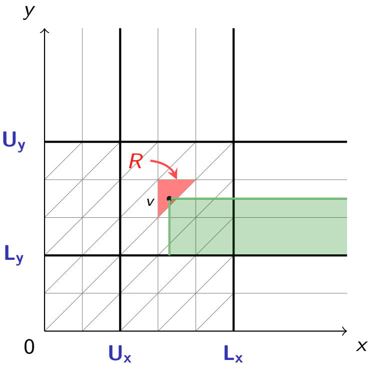
Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$?



- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ ✓ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Collect all v' that **simulate** v

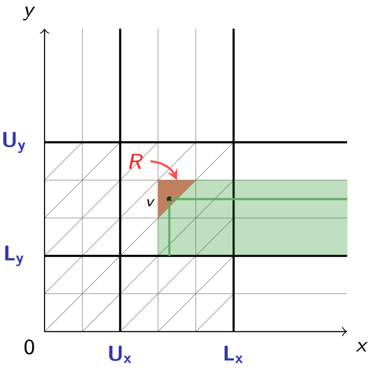
Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$?



- $\mathbf{v} \preccurlyeq_{LU} \mathbf{v}'$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$

Collect all v' that **simulate** v

Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$?

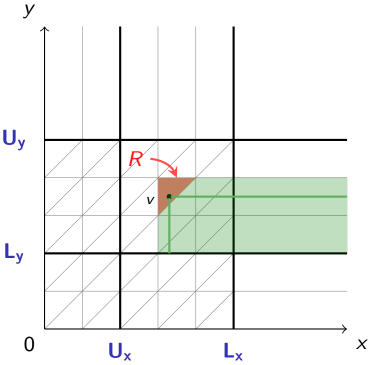


- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$
- $\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(R)$: $\{v' \mid \exists v \in R \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

Inclusion to intersection

$$\mathbf{R} \not\subseteq \mathfrak{a}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{Z'}) \quad \Leftrightarrow \quad \mathfrak{a}^{-1}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{R}) \cap \mathbf{Z'} \text{ is } \textbf{empty}$$

Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq LU}(Z')$?



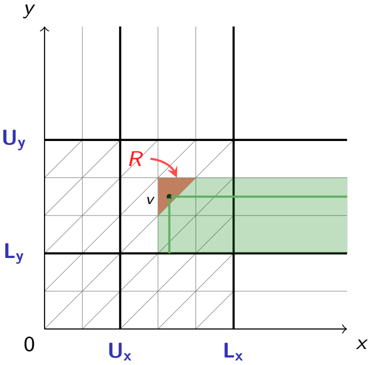
- $\mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$
- $\mathfrak{a}^{-1}_{\preccurlyeq LU}(R)$: $\{v' \mid \exists v \in R \text{ s.t. } \mathbf{v} \preccurlyeq_{\mathbf{LU}} \mathbf{v'}\}$

Inclusion to intersection

$$\mathbf{R} \not\subseteq \mathfrak{a}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{Z'}) \quad \Leftrightarrow \quad \mathfrak{a}^{-1}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{R}) \cap \mathbf{Z'} \text{ is } \textbf{empty}$$

For **every** region R, the set $\mathfrak{a}^{-1}_{\preccurlyeq LU}(R)$ is a **zone**!

Step 2: When is $R \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$?



- $\mathbf{v \preccurlyeq_{LU} v'}$: when for all $x \in X$
  - $v(x) > v'(x) \Rightarrow v'(x) > L_x$ and
  - $v(x) < v'(x) \Rightarrow v(x) > U_x$
- $\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(R)$: $\{v' \mid \exists v \in R \text{ s.t. } \mathbf{v \preccurlyeq_{LU} v'}\}$

Inclusion to intersection

$$\mathbf{R} \not\subseteq \mathfrak{a}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{Z'}) \quad \Leftrightarrow \quad \mathfrak{a}^{-1}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{R}) \cap \mathbf{Z'} \text{ is } \textbf{empty}$$

$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z') \Leftrightarrow \exists R.\ R$ **intersects** Z, $\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(R)$ **does not intersect** Z'

Step 3: When is $\mathfrak{a}^{-1}_{\preccurlyeq LU}(R) \cap Z'$ empty?

Reduction to two clocks

$\mathfrak{a}^{-1}_{\preccurlyeq LU}(R) \cap Z'$ is empty iff there **exist 2 clocks** x, y s.t.

$$\mathfrak{a}^{-1}_{\preccurlyeq LU}(\mathbf{Proj}_{xy}(R)) \cap \mathbf{Proj}_{xy}(Z') \text{ is empty}$$

Step 3: When is $\mathfrak{a}_{\preccurlyeq LU}^{-1}(R) \cap Z'$ empty?

Reduction to two clocks

$\mathfrak{a}_{\preccurlyeq LU}^{-1}(R) \cap Z'$ is empty iff there **exist 2 clocks** x, y s.t.

$$\mathfrak{a}_{\preccurlyeq LU}^{-1}(\mathbf{Proj}_{xy}(R)) \cap \mathbf{Proj}_{xy}(Z') \text{ is empty}$$

$\mathbf{Proj}_{xy}(R)$ is a region, $\mathbf{Proj}_{xy}(Z')$ is a zone

Step 3: When is $\mathfrak{a}^{-1}_{\preccurlyeq LU}(R) \cap Z'$ empty?

Reduction to two clocks

$\mathfrak{a}^{-1}_{\preccurlyeq LU}(R) \cap Z'$ is empty iff there **exist 2 clocks** x, y s.t.

$$\mathfrak{a}^{-1}_{\preccurlyeq LU}(\mathbf{Proj}_{xy}(R)) \cap \mathbf{Proj}_{xy}(Z') \text{ is empty}$$

$\mathbf{Proj}_{xy}(R)$ is a region, $\mathbf{Proj}_{xy}(Z')$ is a zone

$$\mathfrak{a}^{-1}_{\preccurlyeq LU}(\mathbf{Proj}_{xy}(R)) \;\cap\; \mathbf{Proj}_{xy}(Z') \text{ is empty} \;\Leftrightarrow\; \mathbf{Proj}_{xy}(R) \not\subseteq \mathfrak{a}_{\preccurlyeq LU}(\mathbf{Proj}_{xy}(Z'))$$

Step 3: When is $\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(R) \cap Z'$ empty?

Reduction to two clocks

$\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(R) \cap Z'$ is empty iff there **exist 2 clocks** x, y s.t.

$$\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(R)) \cap \mathbf{Proj}_{xy}(Z') \text{ is empty}$$

$\mathbf{Proj}_{xy}(R)$ is a region, $\mathbf{Proj}_{xy}(Z')$ is a zone

$$\mathfrak{a}^{-1}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(R)) \cap \mathbf{Proj}_{xy}(Z') \text{ is empty} \Leftrightarrow \mathbf{Proj}_{xy}(R) \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(Z'))$$

$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z') \Leftrightarrow \exists R, x, y.\ R$ **intersects** Z, $\mathbf{Proj_{xy}(R)} \not\subseteq \mathfrak{a}_{\preccurlyeq_{\mathbf{LU}}}(\mathbf{Proj_{xy}(Z')})$

Efficient inclusion testing

Theorem

$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj}_{xy}(Z) \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(Z'))$$

Efficient inclusion testing

Theorem

$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj}_{xy}(Z) \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(Z'))$$

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

Efficient inclusion testing

Theorem

$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj}_{xy}(Z) \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(Z'))$$

Same complexity as $Z \subseteq Z'$!

Efficient inclusion testing

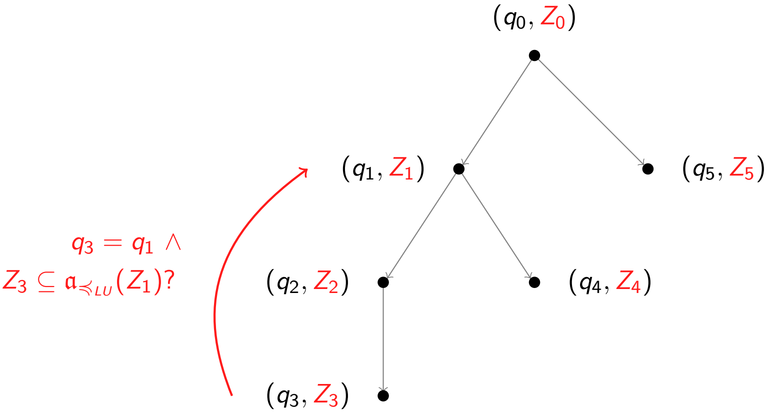
Theorem

$Z \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ if and only if there **exist 2 clocks** x, y s.t.

$$\mathbf{Proj}_{xy}(Z) \not\subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(\mathbf{Proj}_{xy}(Z'))$$

Slightly modified comparison works!

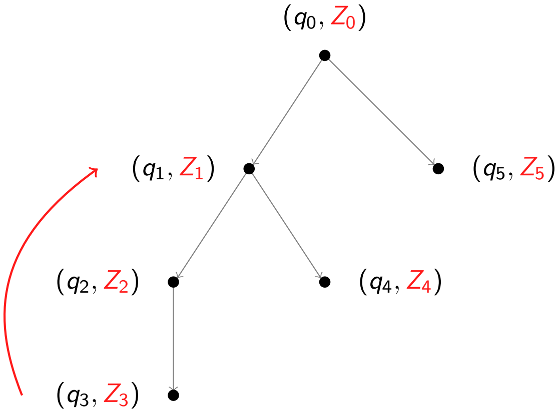
So what do we have now...



Efficient algorithm for $Z \subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$

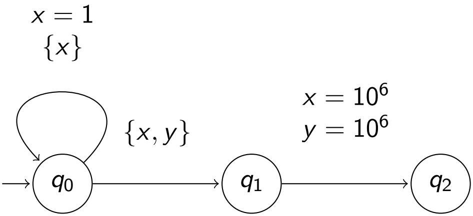
So what do we have now...

$q_3 = q_1 \ \wedge$
$Z_3 \subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z_1)$?



Coming next: **prune** the **LU bounds**!

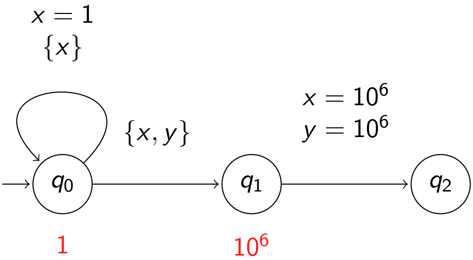
LU-bounds



Naive: $L_x = U_x = 10^6$, $L_y = U_y = 10^6$

Size of graph $\sim 10^6$

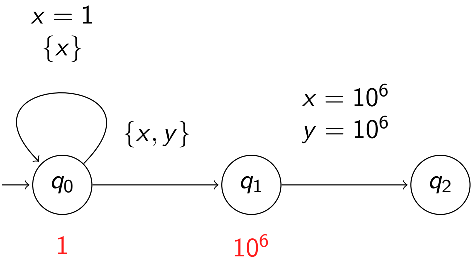
Static analysis: bounds for every q [BBFL03]



Naive: $L_x = U_x = 10^6$, $L_y = U_y = 10^6$

Size of graph < 10

Static analysis: bounds for every q [BBFL03]

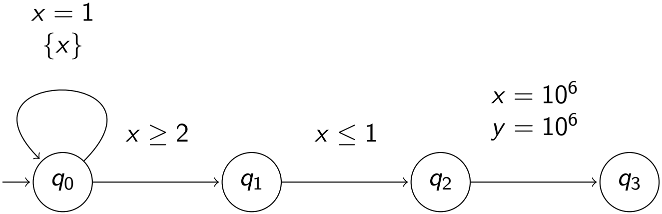


Naive: $L_x = U_x = 10^6$, $L_y = U_y = 10^6$

Size of graph < 10

But this is not enough!

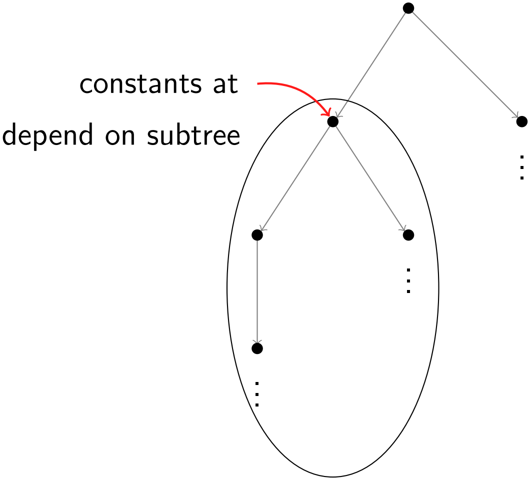
Need to look at semantics...



Static analysis: 10^6

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

LU bounds for every (q, Z) in $ZG(\mathcal{A})$



Constant propagation

$$L(x) = \quad -\infty$$
$$U(x) = \quad -\infty$$

$$(q, Z, LU)$$
$$\bullet$$

Constant propagation

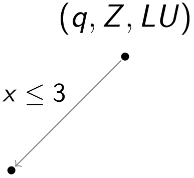
$$L(x) = \quad -\infty$$
$$U(x) = \quad -\infty$$

(q, Z, LU)

$x \leq 3$

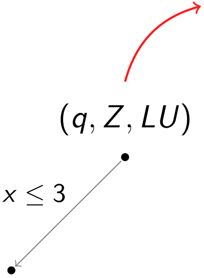
Constant propagation

$$L(x) = -\infty$$
$$U(x) = 3$$



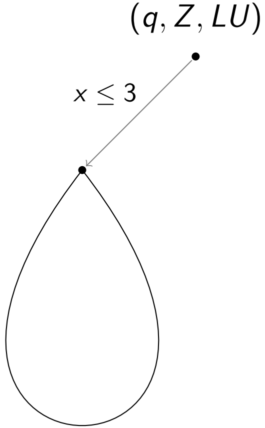
Constant propagation

$L(x) = \quad -\infty$
$U(x) = \quad 3$



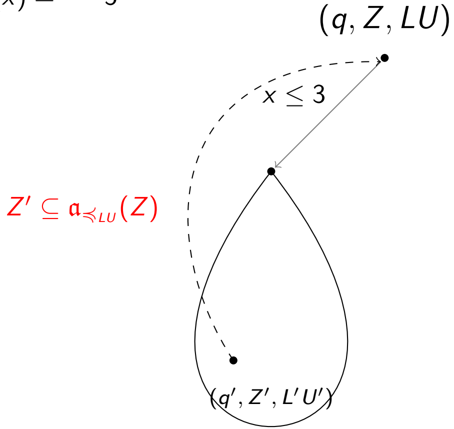
Constant propagation

$L(x) = \quad 4$
$U(x) = \quad 5$



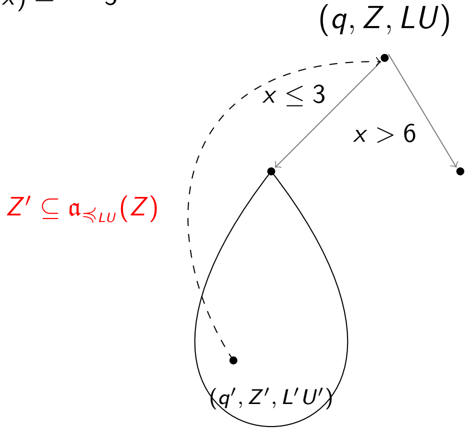
Constant propagation

$L(x) = \quad 4$
$U(x) = \quad 5$



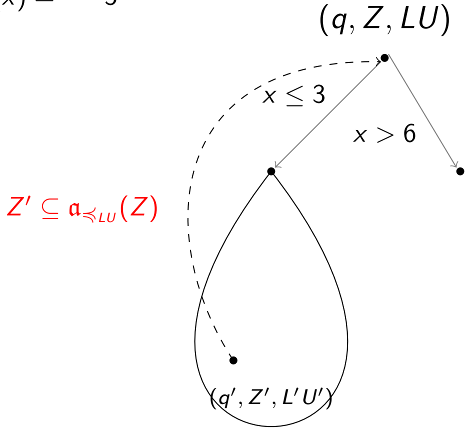
Constant propagation

$$\begin{aligned} L(x) &= \quad 4 \\ U(x) &= \quad\ 5 \end{aligned}$$



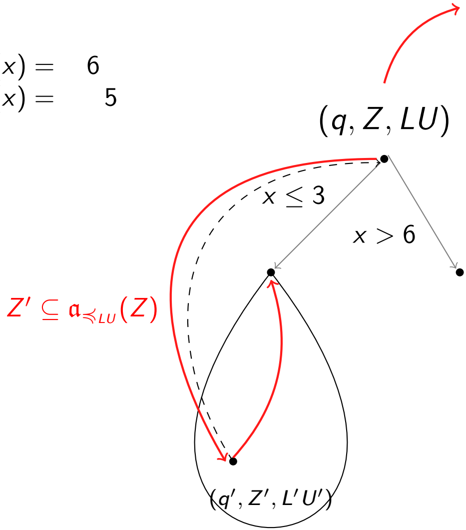
Constant propagation

$$L(x) = \quad 6$$
$$U(x) = \quad 5$$



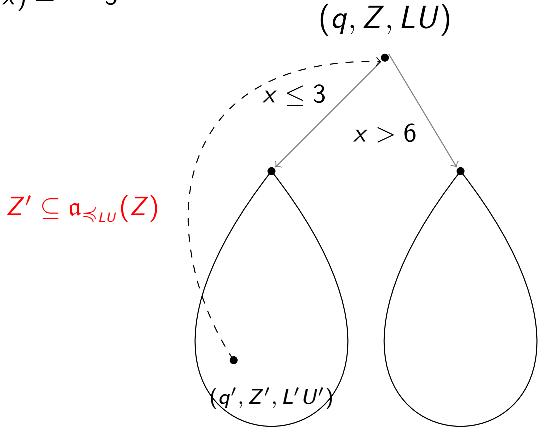
Constant propagation

$L(x) = \quad 6$
$U(x) = \quad 5$



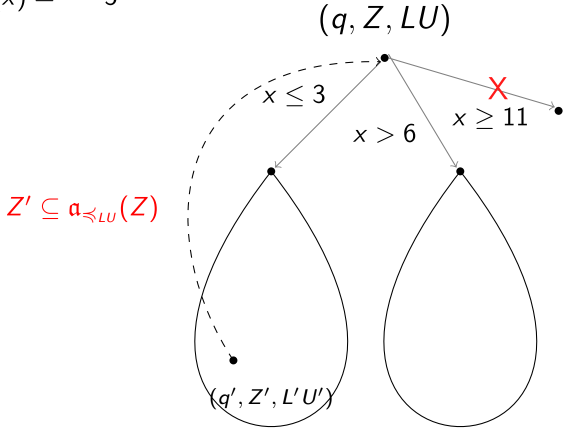
Constant propagation

$L(x) = \quad 6$
$U(x) = \quad 5$



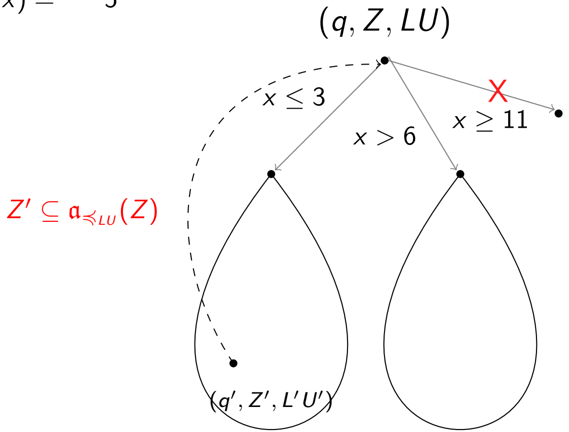
Constant propagation

$L(x) = \quad 6$
$U(x) = \quad 5$



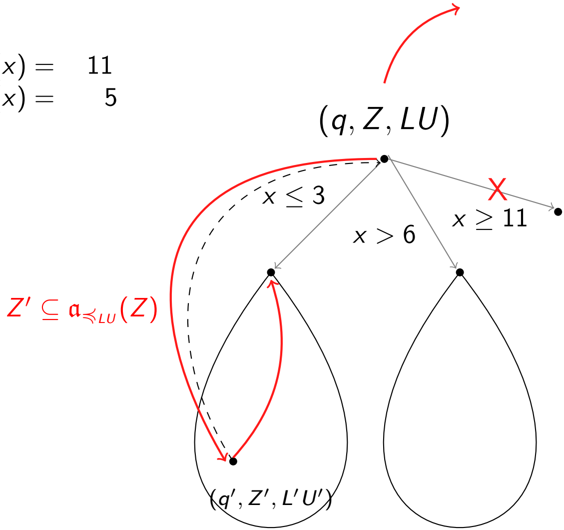
Constant propagation

$$L(x) = 11$$
$$U(x) = 5$$



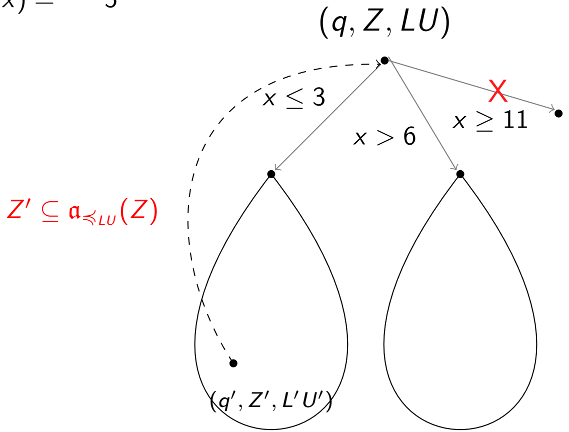
Constant propagation

$$L(x) = 11$$
$$U(x) = 5$$



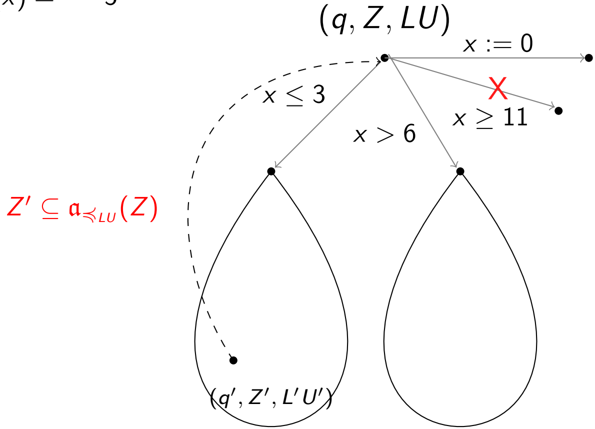
Constant propagation

$L(x) = \quad 11$
$U(x) = \quad 5$



Constant propagation

$$L(x) = \quad 11$$
$$U(x) = \quad 5$$



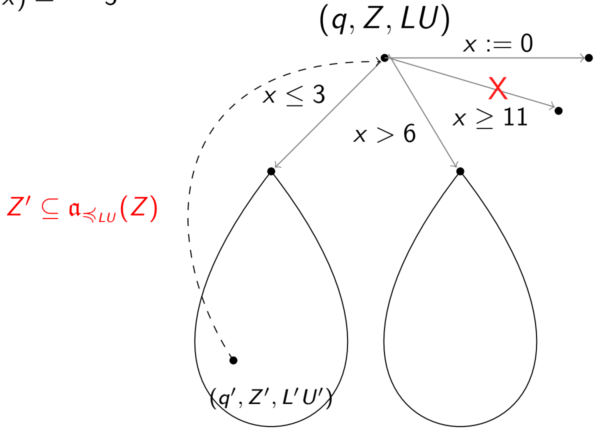
Constant propagation

$L(x) = \quad 11$
$U(x) = \quad 5$

All **tentative nodes** consistent

$+$ No more **exploration**

$\rightarrow$ Terminate!



Invariants on the bounds

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

Invariants on the bounds

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

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 $ZG(\mathcal{A})$: $Z \subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z')$ for **termination**
- **LU-bounds** calculated **on-the-fly**

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

A bonus

- LU-automata: automata with guards **determined by** L and U
- Z: an arbitrary **reachable zone** in some LU-automaton

A bonus

- LU-automata: automata with guards **determined by** L and U
- Z: an arbitrary **reachable zone** in some LU-automaton

Every **sound** and **complete** abstraction $\mathfrak{b}$ satisfies $\mathfrak{b}(Z) \subseteq \mathfrak{a}_{\preccurlyeq_{LU}}(Z)$

Theorem

In the context of **reachability**, the $\mathfrak{a}_{\preccurlyeq_{LU}}$ abstraction is the **biggest abstraction** that is **sound** and **complete** for all LU-automata.

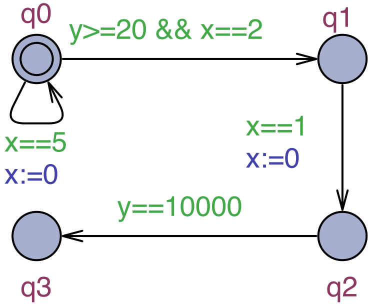
Benchmarks

Model	Our algorithm		UPPAAL's algorithm		UPPAAL 4.1.3 (-n4 -C -o1)	
	nodes	s.	nodes	s.	nodes	s.
CSMA/CD7	5046	0.39	5923	0.30	–	T.O.
CSMA/CD8	16609	0.75	19017	1.16	–	T.O.
CSMA/CD9	54467	9.40	60783	4.53	–	T.O.
FDDI10	459	0.04	525	0.05	12049	2.43
FDDI20	1719	0.41	2045	0.82	–	T.O.
FDDI30	3779	1.70	4565	3.90	–	T.O.
Fischer7	7737	0.40	18353	0.48	18374	0.35
Fischer8	25080	1.50	85409	2.31	85438	1.53
Fischer9	81035	5.70	397989	12.05	398685	8.95
Fischer10	–	T.O.	–	T.O.	1827009	53.44

- $\mathbf{Extra}^{+}_{LU}$ and **static** analysis bounds in UPPAAL
- $\mathfrak{a}_{\preccurlyeq_{LU}}$ and **otf** bounds in our algorithm

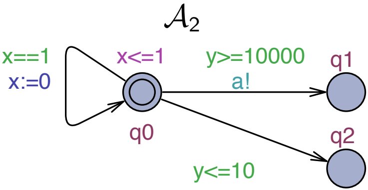
Experiments I

$\mathcal{A}_1$



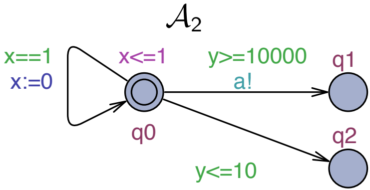
$\mathcal{A}_1$	nodes	s.
Our algorithm	7	0.0
UPPAAL's algorithm	2003	0.60
UPPAAL 4.1.3	2003	0.01

Experiments II



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

Experiments II

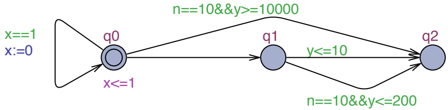


$\mathcal{A}_2$	nodes	s.
Our algorithm	2	0.0
UPPAAL's algorithm	10003	0.07
UPPAAL 4.1.3	10003	0.07

Occurs in **CSMA/CD**!

Experiments III

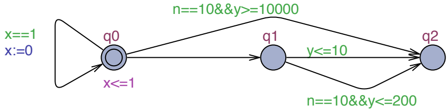
$\mathcal{A}_3$



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

Experiments III

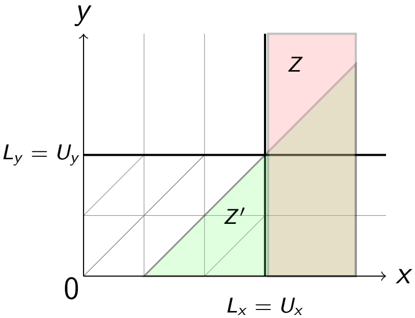
$\mathcal{A}_3$



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

Occurs in **Fischer**!

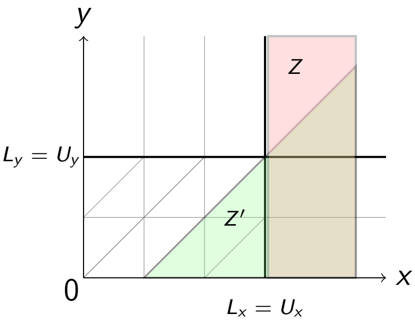
Experiments IV



$Z' : \; x - y \geq 1$

$Z : \; x > L_x$

Experiments IV



$$Z' : \; x - y \geq 1$$
$$Z : \; x > L_x$$

Occurs in **FDDI**!

Conclusions & Perspectives

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

- 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.