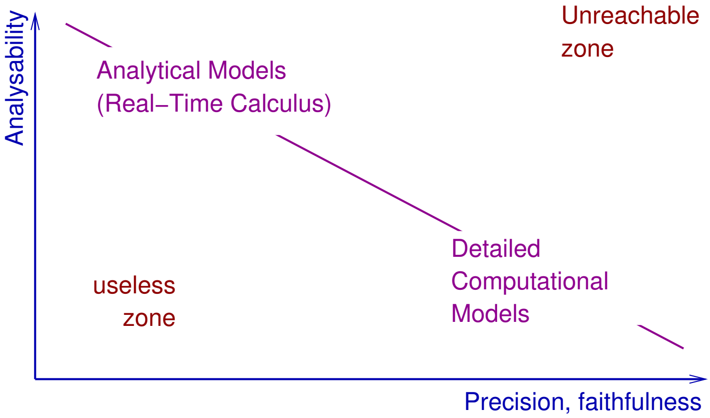
Arrival Curves for Real-Time Calculus: the Causality Problem and its Solutions

Karine Altisen and Matthieu Moy

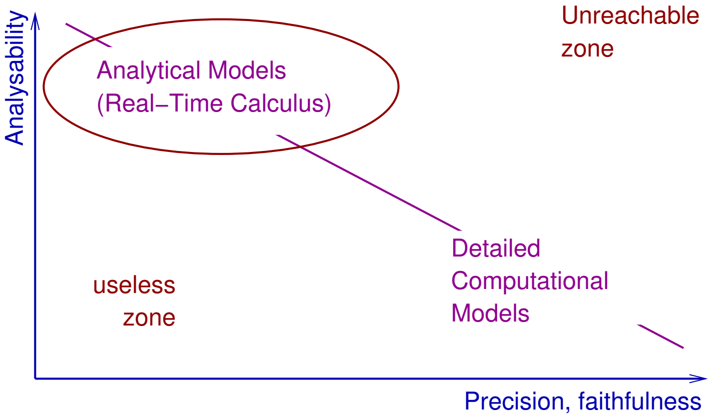
Verimag (Grenoble INP)
Grenoble
France

TACAS, 25 March 2010

Models for Performance Analysis



Models for Performance Analysis

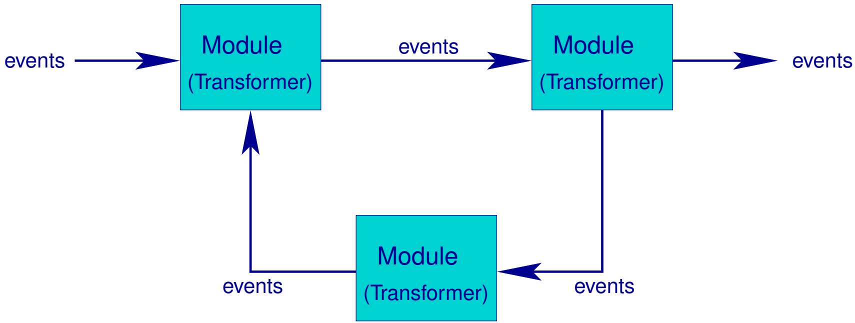


Summary

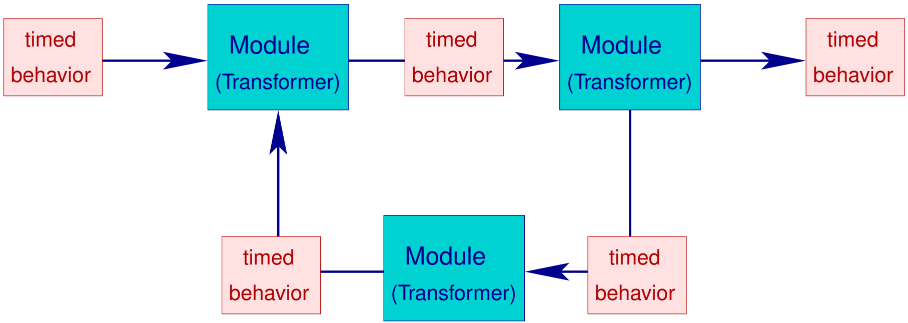
Summary

1. Introduction: Modular Performance Analysis
2. The Causality Problem for Arrival Curves
3. The Causality Closure: Solving the Causality Problem
4. Conclusion

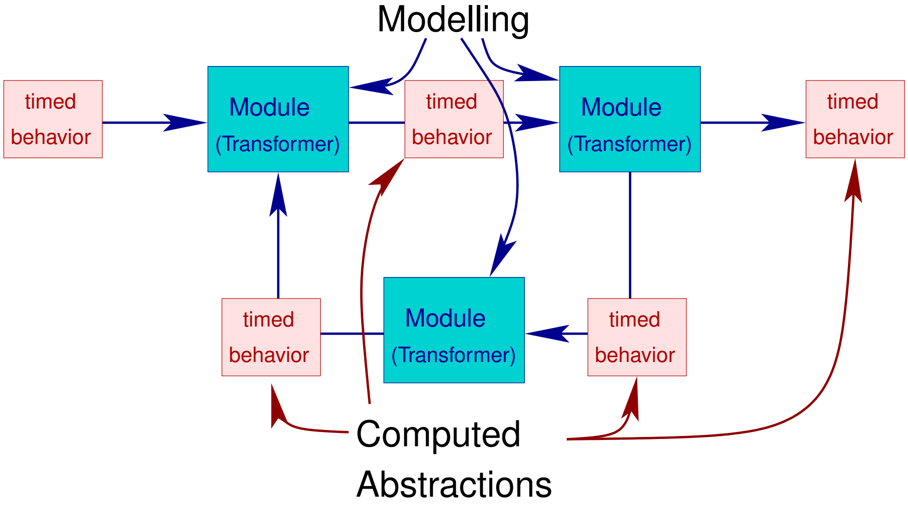
Modular Performance Analysis (MPA): The Big Picture



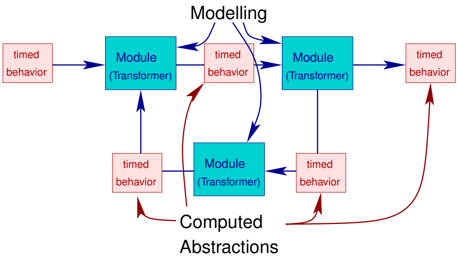
Modular Performance Analysis (MPA): The Big Picture



Modular Performance Analysis (MPA): The Big Picture

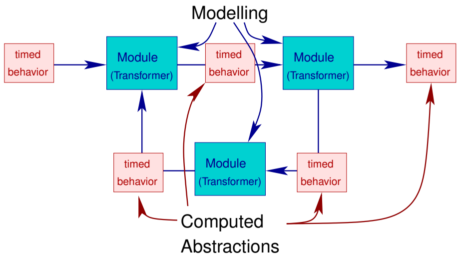


Modular Performance Analysis (MPA)



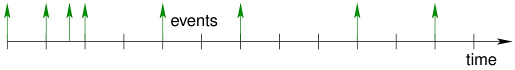
- What can "Timed Behavior" be?
  - Number of events per time unit?
  - Bounds for number of events?

Modular Performance Analysis (MPA)



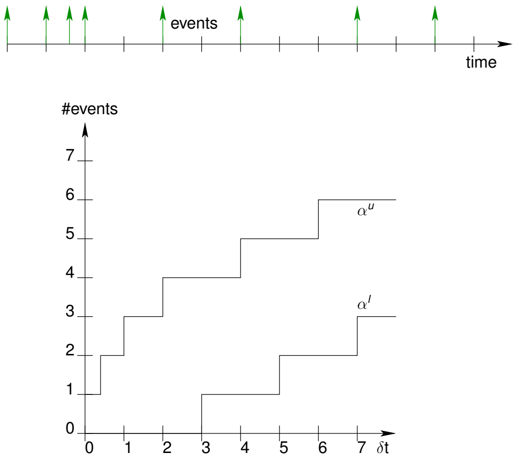
- What can "Timed Behavior" be?
  - Number of events per time unit?
  - Bounds for number of events?
  - MPA uses "Arrival Curves".
- "Modules" = Arrival Curve transformers:
  - FIFO + processing element (defined by "service curves")
  - Can also be a "program"

Arrival Curves in Real-Time Calculus (RTC)



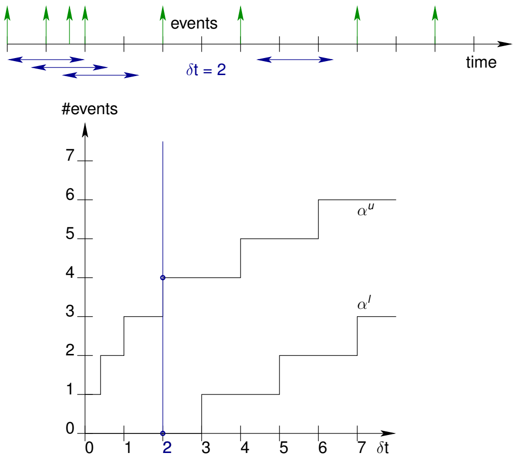
- $\alpha^u(\delta)$: max number of events in any window of size δ.
- $\alpha^l(\delta)$: min number of events in any window of size δ.

Arrival Curves in Real-Time Calculus (RTC)



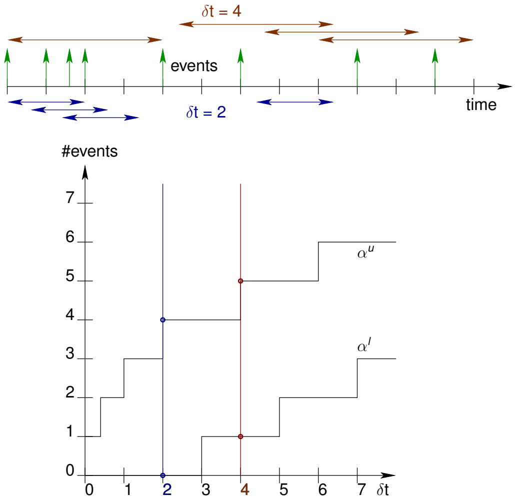
- $\alpha^u(\delta)$: max number of events in any window of size δ.
- $\alpha^l(\delta)$: min number of events in any window of size δ.

Arrival Curves in Real-Time Calculus (RTC)



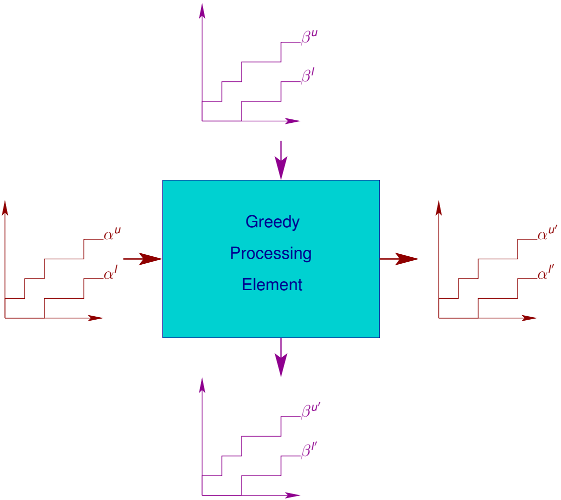
- $\alpha^u(\delta)$: max number of events in any window of size δ.
- $\alpha^l(\delta)$: min number of events in any window of size δ.

Arrival Curves in Real-Time Calculus (RTC)

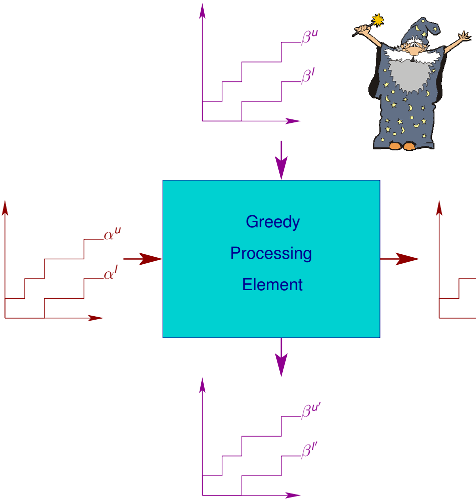


- $\alpha^u(\delta)$: max number of events in any window of size δ.
- $\alpha^l(\delta)$: min number of events in any window of size δ.

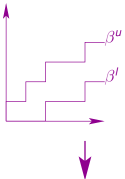
Service Curves

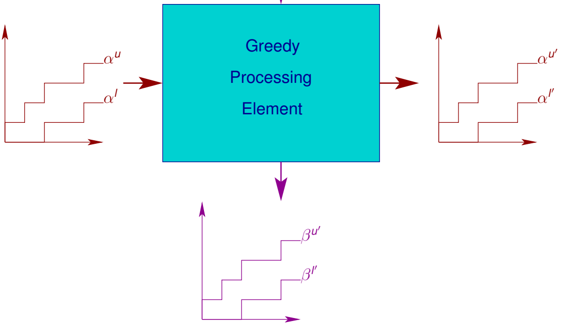


Service Curves



Service Curves



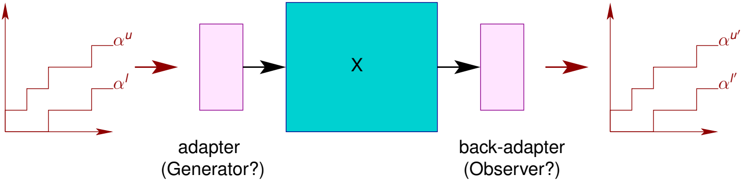


$$\alpha^{u'} = ((\alpha^u \underline{\oplus} \beta^u) \overline{\oplus} \beta^l) \wedge \beta^u$$
$$\alpha^{l'} = ((\alpha^l \overline{\otimes} \beta^u) \underline{\otimes} \beta^l) \wedge \beta^l$$
$$\beta^{u'} = (\beta^u - \alpha^l) \underline{\otimes} 0$$
$$\beta^{l'} = (\beta^l - \alpha^u) \overline{\otimes} 0$$

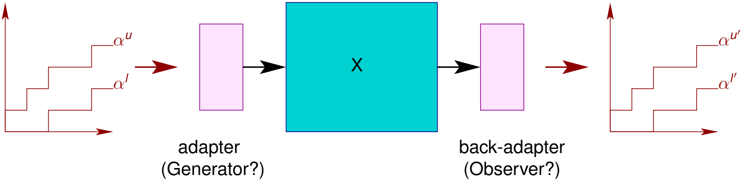
Real-Time Calculus (RTC): pros and cons

- Nice things with RTC
  - Can model: event streams, simple scheduling policies
  - Scales up nicely
  - Exact hard bounds
- Less nice thing with RTC
  - Limited expressiveness

Allowing more Complex Modules in MPA



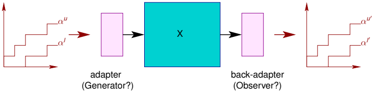
Allowing more Complex Modules in MPA



- X = Arbitrary program ⇒ testing (ETHZ)
- X = Timed Automata ⇒ model-checking (Verimag, ETHZ, Uppsala)
- X = Lustre ⇒ Abstract Interpretation, SMT Solving (Verimag)

Summary

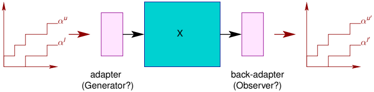
A Closer Look at the Generator



- The idea behind generators:

  "at each point in time, compute an interval $[l, u]$ on the number of events that can be emitted".

A Closer Look at the Generator

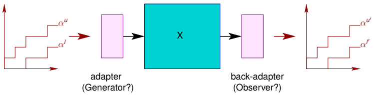


- The idea behind generators:

  "at each point in time, compute an interval $[l, u]$ on the number of events that can be emitted".

What if $l > u$? $\Rightarrow$ deadlock.

A Closer Look at the Generator



- The idea behind generators:

“at each point in time, compute an interval $[l, u]$ on the number of events that can be emitted”.

What if $l > u$? $\Rightarrow$ deadlock.

This talk:

How can we prevent these deadlocks?

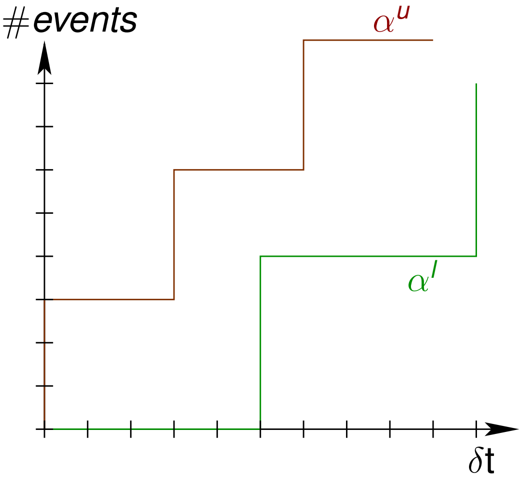
Causal and Non-Causal Curves

- A pair of arrival curve (α^u, α^l) is causal iff an event stream conformant with (α^u, α^l) *up to time t* can be extended into a stream conformant with (α^u, α^l) *forever*.

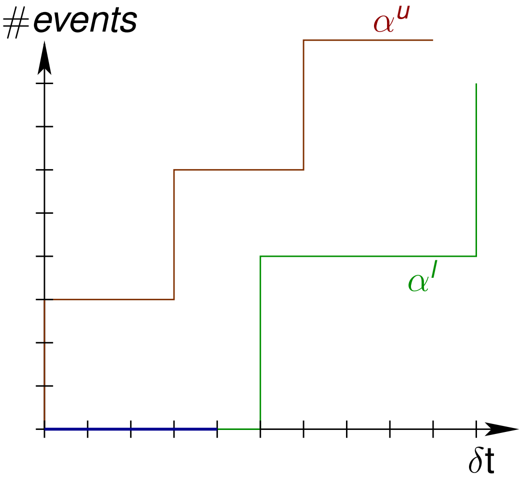
Causal and Non-Causal Curves

- A pair of arrival curve (α^u, α^l) is causal iff
  an event stream conformant with (α^u, α^l) *up to time t* can be extended into a stream conformant with (α^u, α^l) *forever*.
- i.e., (α^u, α^l) is causal iff
  the associated generator has no deadlock.

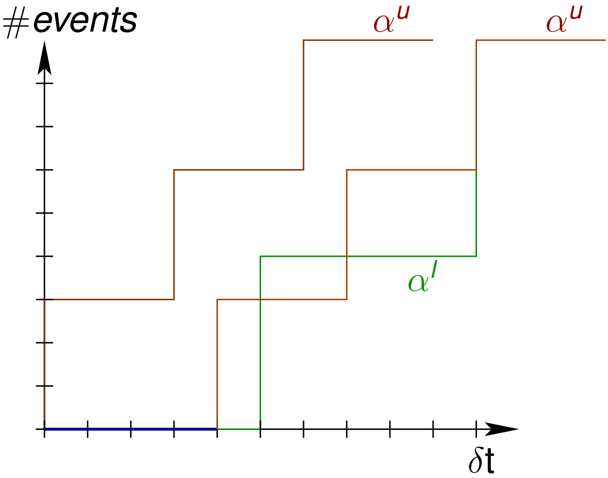
Causality problem: Forbidden Regions



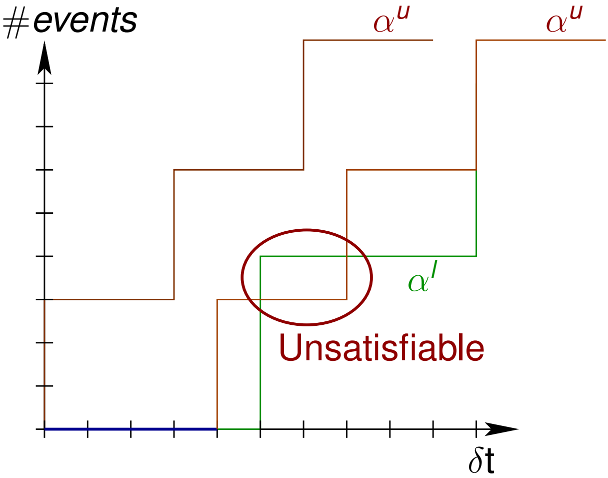
Causality problem: Forbidden Regions



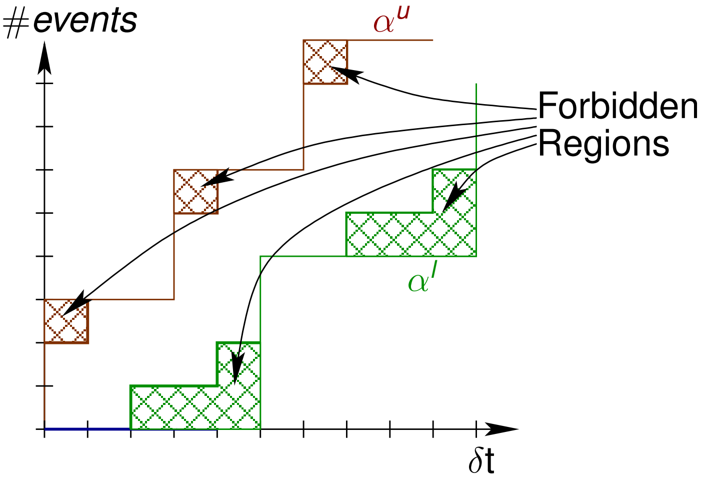
Causality problem: Forbidden Regions



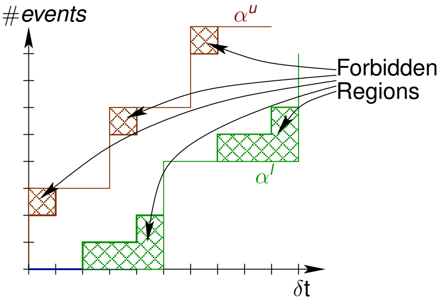
Causality problem: Forbidden Regions



Causality problem: Forbidden Regions



Causality problem: Forbidden Regions



If a stream gets in a forbidden region,
it will eventually reach a dead-end

Problems with Non-Causal Curves

- Simulating a Generator: the generator may stop with "you shouldn't have been there, I can't continue"
- Formal verification: spurious counter-examples (finite event stream that cannot be extended into an infinite one)

Problems with Non-Causal Curves

- Simulating a Generator: the generator may stop with "you shouldn't have been there, I can't continue"
- Formal verification: spurious counter-examples (finite event stream that cannot be extended into an infinite one)
- Practical issue: non-causal curves are hard to think with!

Problems with Non-Causal Curves

- Simulating a Generator: the generator may stop with “you shouldn’t have been there, I can’t continue”
- Formal verification: spurious counter-examples (finite event stream that cannot be extended into an infinite one)
- Practical issue: non-causal curves are hard to think with!

(Side question:
How come have people worked with RTC
for 10 years avoiding the problem?)

Problems with Non-Causal Curves

- Simulating a Generator: the generator may stop with "you shouldn't have been there, I can't continue"
- Formal verification: spurious counter-examples (finite event stream that cannot be extended into an infinite one)
- Practical issue: non-causal curves are hard to think with!

(Side question:
How come have people worked with RTC
for 10 years avoiding the problem?)
$\rightsquigarrow$ Possible answer at end of talk.

Summary

Causality Closure: Making Curves Causal

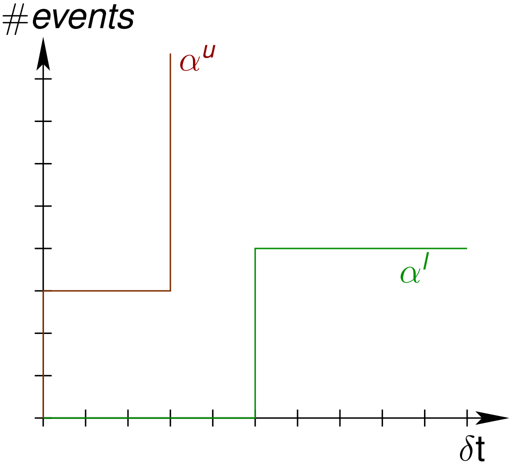
- The causality closure of (α^u, α^l) is a pair of curves that is:
  - Equivalent to (α^u, α^l) (i.e. same set of accepted event streams)
  - Causal (i.e. finite accepted event streams can be extended infinitely)
- How to compute it?
- Intuition: Causal curves are curves without forbidden regions (?)

Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)

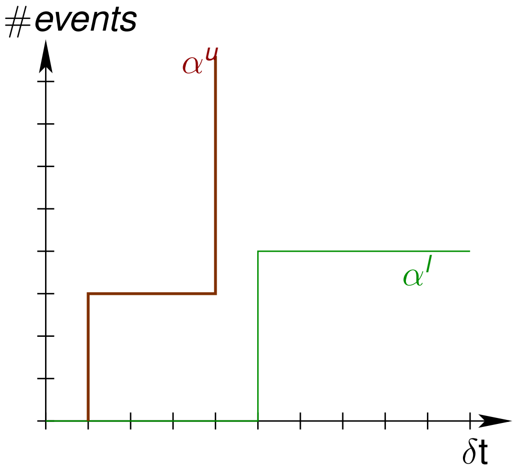
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



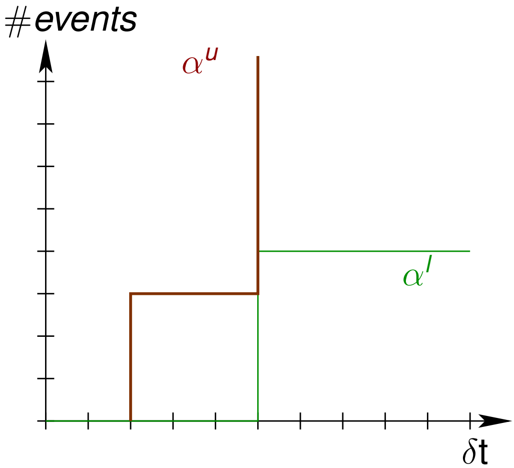
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



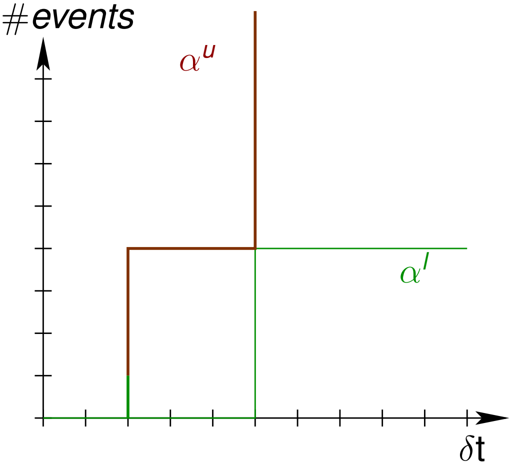
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



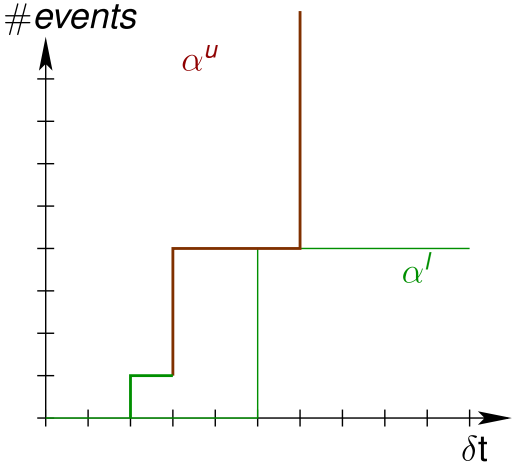
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



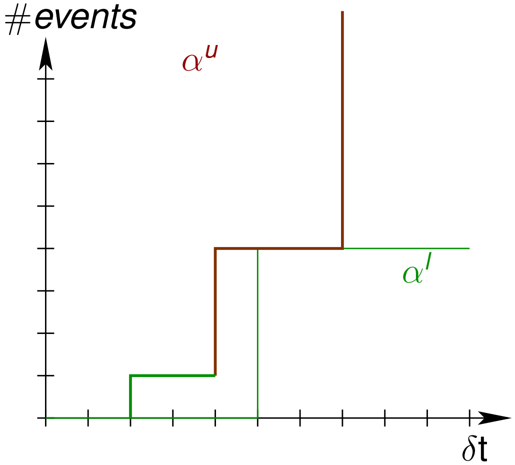
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



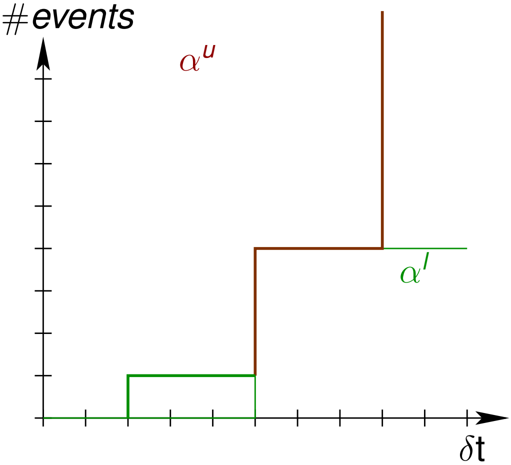
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



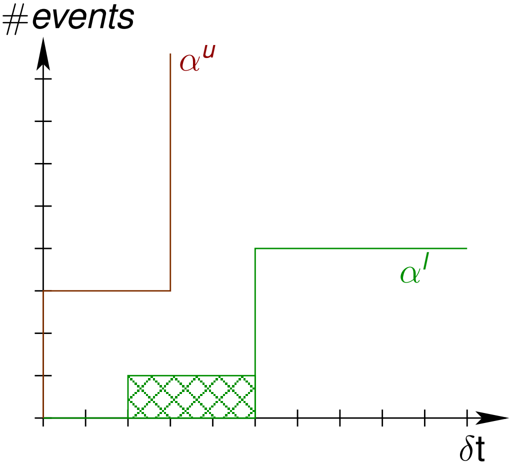
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



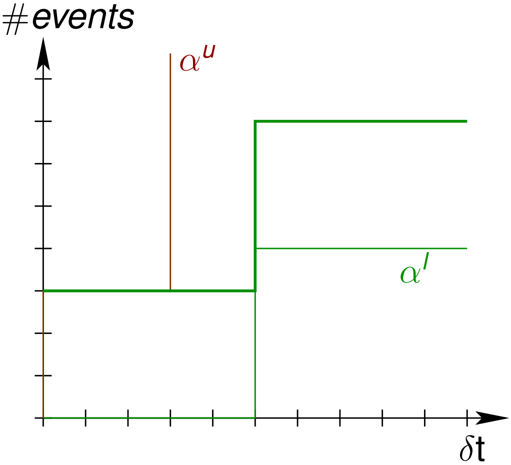
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



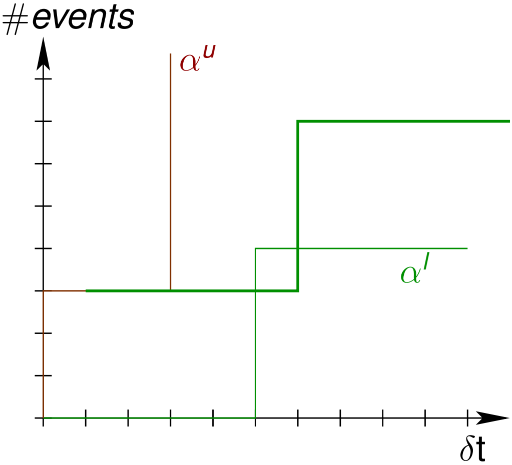
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



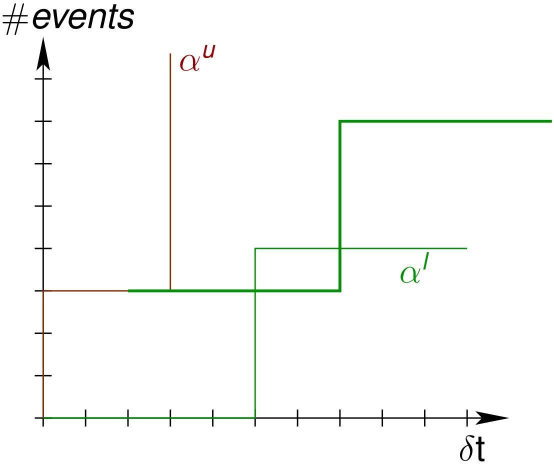
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



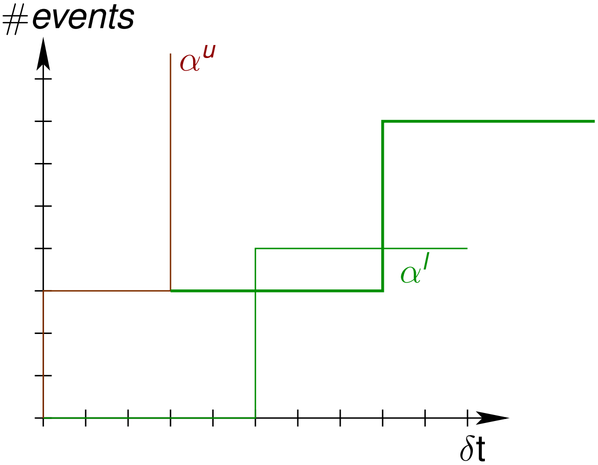
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



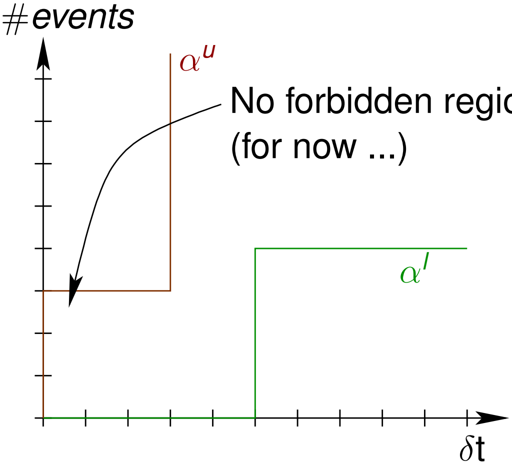
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



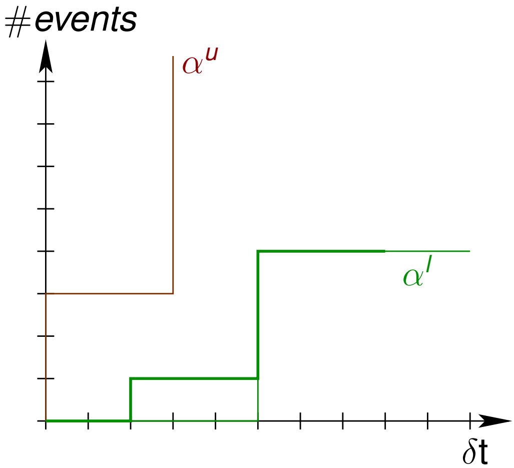
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



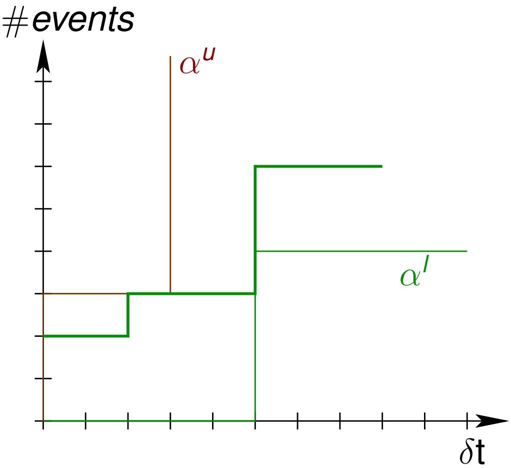
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



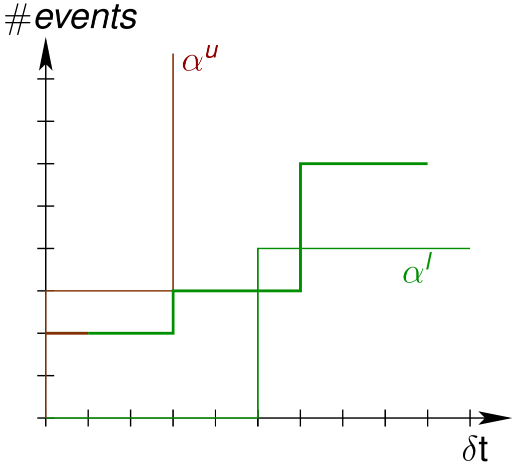
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



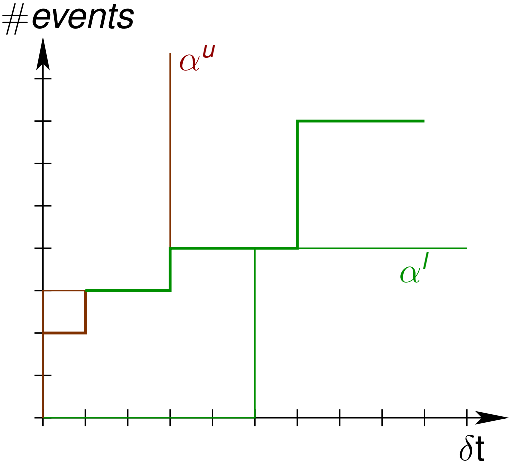
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



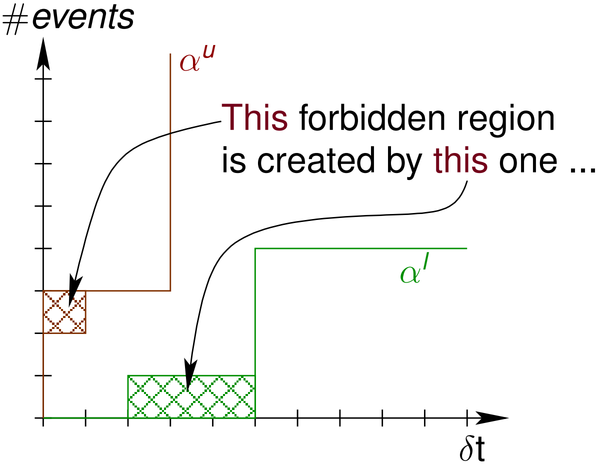
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



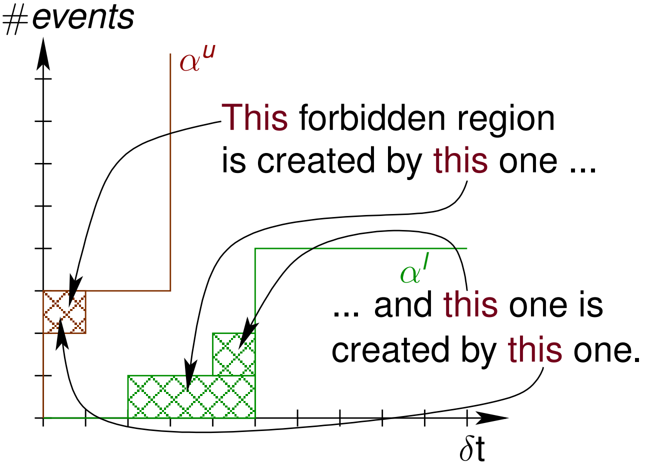
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



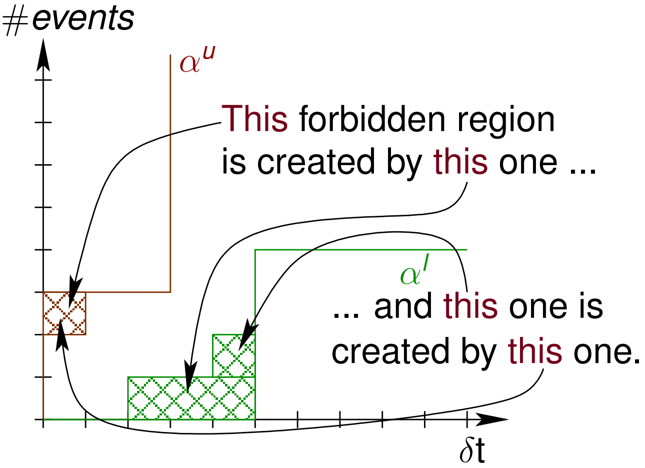
Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



Towards an Algorithm for Causality Closure

- First idea: remove forbidden regions and we're done (?)
- Insufficient:



- Technically: Forbidden region removal = deconvolution = $\oslash$, $\overline{\oslash}$.

Towards an Algorithm for Causality Closure

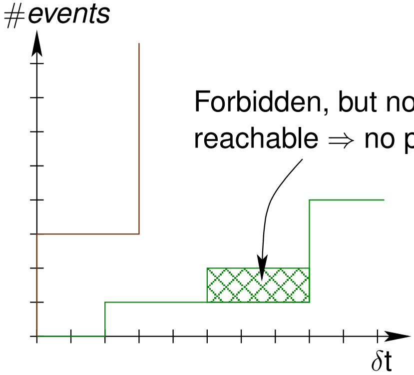
- Second idea: Curves without forbidden regions are (?) causal, let's iterate forbidden region removal until fix-point and we're done (?)

Towards an Algorithm for Causality Closure

- Second idea: Curves without forbidden regions are (?) causal, let's iterate forbidden region removal until fix-point and we're done (?)
- Issue 1: Will it terminate?

Towards an Algorithm for Causality Closure

- Second idea: Curves without forbidden regions are (?) causal, let's iterate forbidden region removal until fix-point and we're done (?)
- Issue 1: Will it terminate?
- Issue 2: some causal curves do have forbidden regions!



Unreachable Regions: Another (Well Known) Issue

- $\alpha^l(\delta_1 + \delta_2)$ = minimum number of events in any time window of duration $\delta_1 + \delta_2$.
- $\alpha^l(\delta_1) + \alpha^l(\delta_2)$ is another valid bound. It may be better.
- $\Rightarrow$ If so, we say that $\alpha^l(\delta_1 + \delta_2)$ is unreachable.

Unreachable Regions: Another (Well Known) Issue

- $\alpha^l(\delta_1 + \delta_2)$ = minimum number of events in any time window of duration $\delta_1 + \delta_2$.
- $\alpha^l(\delta_1) + \alpha^l(\delta_2)$ is another valid bound. It may be better.
- $\Rightarrow$ If so, we say that $\alpha^l(\delta_1 + \delta_2)$ is unreachable.
- Technically, (α^u, α^l) have no unreachable regions iff
  - α^l is super-additive,
  - α^u is sub-additive.
- $\overline{\alpha^u}$ = sub-additive closure of α^u
- $\underline{\alpha^l}$ = super-additive closure of α^l

Theorem 1: Causality and Forbidden Region

For sub-additive/super-additive pair of curves,
Causality $\Leftrightarrow$ Absence of forbidden region

Theorem 1: Causality and Forbidden Region

For sub-additive/super-additive pair of curves,
Causality $\Leftrightarrow$ Absence of forbidden region

$\Rightarrow$ Causal curve can be obtained by
the fix-point of forbidden region removal
for sub-additive/super-additive curves.

Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property

Theorem 2: no Need to Iterate!

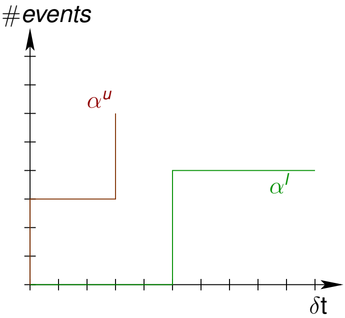
- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property

$\Rightarrow$ Applying forbidden region removal from $(\overline{\alpha^u}, \underline{\alpha^l})$ gives a causal pair of curves. This is the causality closure.

(both forbidden region removal and sub-additive/super-additive closures are cheap algorithms)

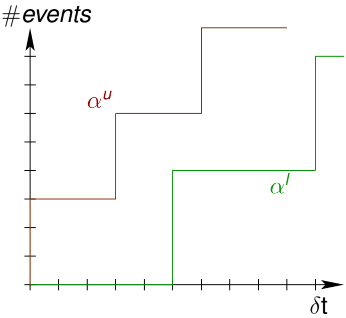
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



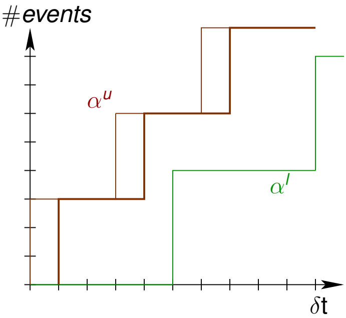
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



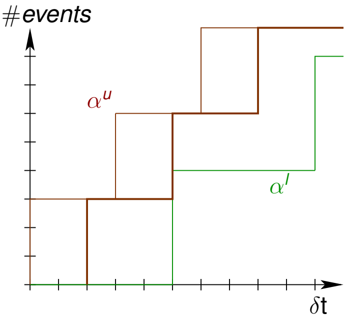
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



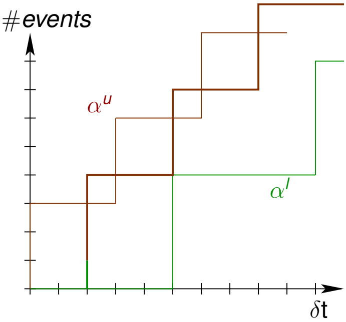
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



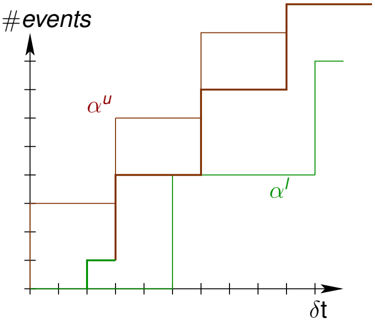
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



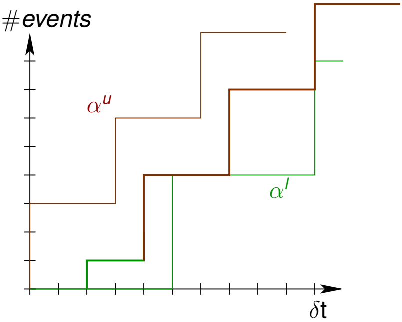
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



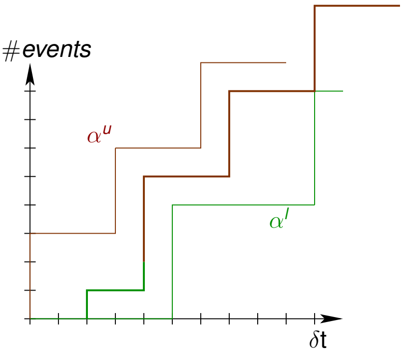
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



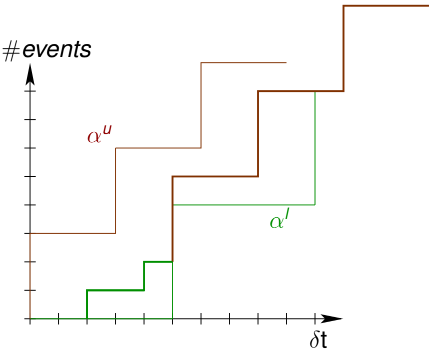
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



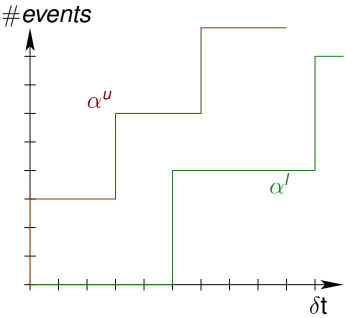
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



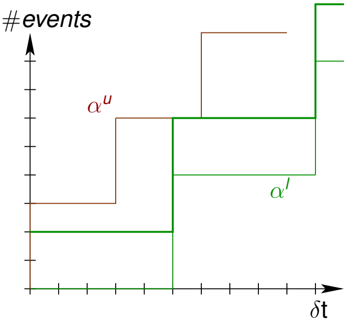
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



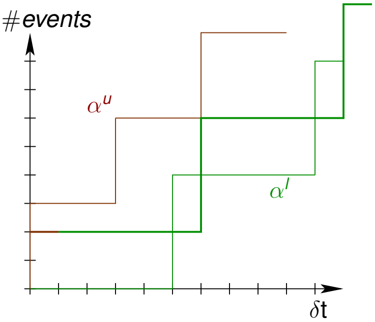
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



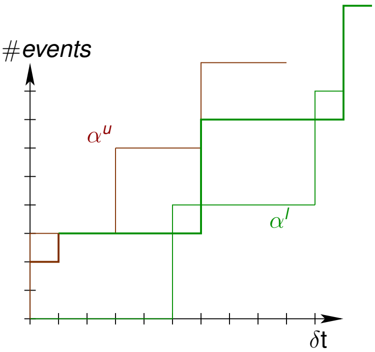
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



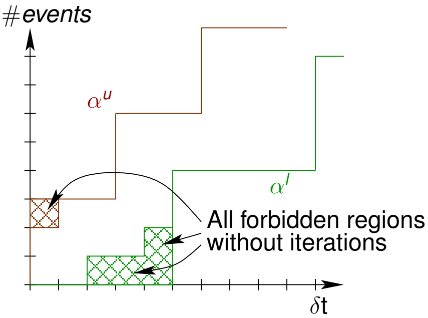
Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



Theorem 2: no Need to Iterate!

- Removing forbidden regions from a sub-additive/super-additive pair of curves
  - Doesn't create new forbidden regions
  - Preserves sub-additive/super-additive property



Causality Closure Algorithm

- Causality Closure Algorithm:
  1. Compute sub-additive/super-additive closure $(\overline{\alpha^u}, \underline{\alpha^l})$.
  2. Remove forbidden regions : $\overline{\alpha^l} \oslash \underline{\alpha^u}$ and $\overline{\alpha^u} \,\overline{\oslash}\, \underline{\alpha^l}$.
- Implementable in any framework implementing $\overline{\alpha}$, $\underline{\alpha}$, $\oslash$ and $\overline{\oslash}$.

Theorem 3: Optimality

Applying forbidden region removal from $(\overline{\alpha^u}, \underline{\alpha^l})$ gives the tightest pair of curves equivalent to (α^u, α^l)

Theorem 3: Optimality

Applying forbidden region removal from $(\overline{\alpha^u}, \underline{\alpha^l})$ gives the tightest pair of curves equivalent to (α^u, α^l)

$\Rightarrow$ Give me a pair of curves,
and I'll give you a better one,
(almost) for free!

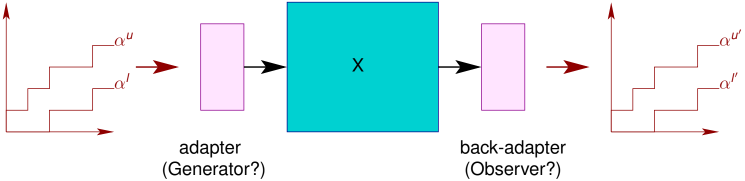
Corollary 4: Converse of Optimality

(Reminder: Theorem 3 = Causality closure is optimal)

Conversely, the tightest pair of curves is causal and sub-additive/super-additive.

$\Rightarrow$ optimal computations do not have the causality problem.
Non-causal curves are caused by over-approximations.

Connection from RTC to X, Revisited



- Compute the causality closure of (α^u, α^l) beforehand $\Rightarrow$ avoids deadlocks in the generator (and spurious counter-examples in proofs)
- Compute the causality closure of $(\alpha^{u'}, \alpha^{l'})$ afterwards $\Rightarrow$ may increase precision
- (Same applies for service curves)

Summary

Summary of Contributions

- Identification and formalization of the causality problem in RTC,
- Causality closure: make curves causal/remove deadlocks with a cheap algorithm,
- Interesting side-effect: optimality.
- Implementation for finite, discrete curves.

Summary of Contributions

- Identification and formalization of the causality problem in RTC,
- Causality closure: make curves causal/remove deadlocks with a cheap algorithm,
- Interesting side-effect: optimality.
- Implementation for finite, discrete curves.

Future Works

Future Works

Problem solved, move to another one ;-)

Future and Related Works

Problem solved, move to another one ;-)

- Work on connection of RTC to other formalisms :
  - Lustre
  - Timed Automata (QAPL talk on Sunday)
  - $\Rightarrow$ Causality closure is definitely useful there.
- Define causality closure for more classes of curves (mixed discrete curves + piecewise affine)

Details I've spared you ...

- Algorithm for finite, discrete curves
  (which do need iterations)
- Proofs
  (surprisingly tricky, full paper is 28 pages mostly in LaTeX math mode!)

Questions?

Backup Slide

$$
\begin{array}{ccc}
\begin{array}{c} \alpha^{l} = \alpha^{l} \oslash \alpha^{u} \\ \text{and} \\ \alpha^{u} = \alpha^{u} \mathbin{\overline{\oslash}} \alpha^{l} \end{array} & \Longrightarrow & (\alpha^{u}, \alpha^{l}) \text{ is causal} \\
\\
\Downarrow & \Nearrow & \Updownarrow \\
\\
\begin{array}{c} \underline{\alpha^{l}} = \underline{\alpha^{l}} \oslash \overline{\alpha^{u}} \\ \text{and} \\ \overline{\alpha^{u}} = \overline{\alpha^{u}} \mathbin{\overline{\oslash}} \underline{\alpha^{l}} \end{array} & \Longleftrightarrow & (\overline{\alpha^{u}}, \underline{\alpha^{l}}) \text{ is causal}
\end{array}
$$