Efficient On-the-fly Algorithms for the Analysis of Timed Games

F. Cassez[1], A. David[2], E. Fleury[2], K. G. Larsen[2], D. Lime[2]

[1] IRCCyN/CNRS, Nantes, France
[2] CISS - Aalborg University, Denmark

August 23, 2005

Outline

Outline

Control Problems

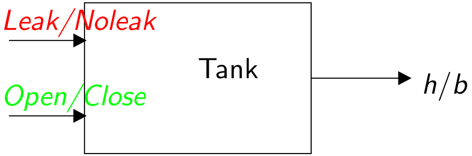
Reachability Control

On-the-fly Algorithms for Reachability Control

Implementation, Optimizations, Time Optimality

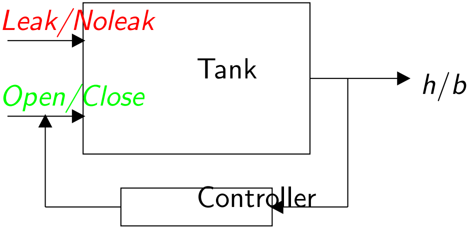
Experiments

Automated Systems Viewpoint



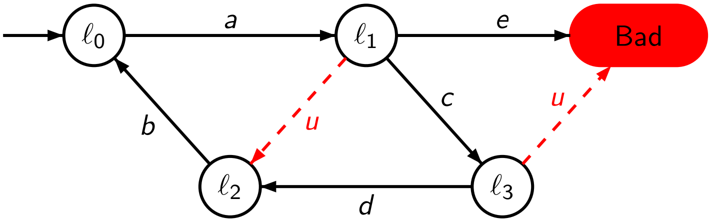
- Open System = plant to be controlled
  Uncontrollable events and Controllable events
- Goal: e.g. "level of the tank always between h and b"
  or "enforce the level of the tank above h"

Automated Systems Viewpoint



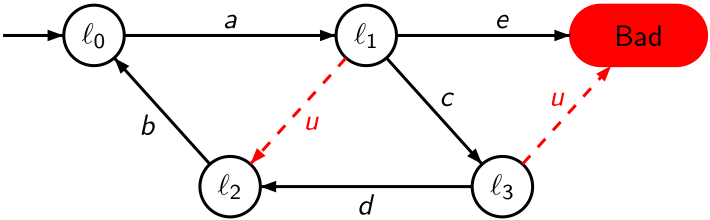
- Open System = plant to be controlled
  Uncontrollable events and Controllable events
- Goal: e.g. "level of the tank always between h and b"
  or "enforce the level of the tank above h"
- Closed (loop) = Controlled System

Control Problems as Games



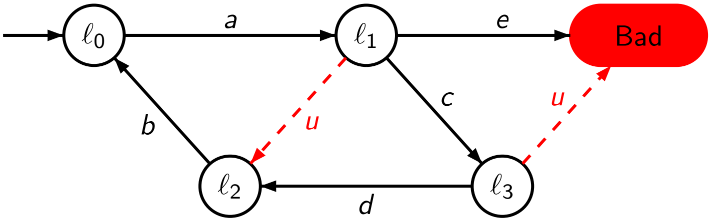
- Control problem = game = controller (C) vs environment (E)

Control Problems as Games



- Control problem = game = controller (C) vs environment (E)
- Various types of game models for C and E
  - Finite or pushdown or counter automata ...
  - Timed (or hybrid) automata

Control Problems as Games



- Control problem = game = controller (C) vs environment (E)
- Various types of game models for C and E
  - Finite or pushdown or counter automata . . .
  - Timed (or hybrid) automata
- Goal: find a strategy for the controller to win
  Avoid bad states: safety control
  Enforce good states: reachability control

Outline

Control Problems

Reachability Control

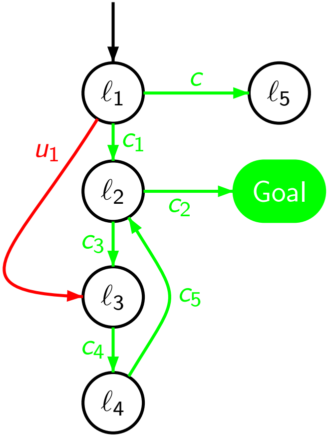
- Finite Games & Backward Algorithm
- Timed Games & Backward Algorithm
- Summary of the Results for Reachability Control

On-the-fly Algorithms for Reachability Control

Implementation, Optimizations, Time Optimality

Experiments

Backward Computation of Winning States



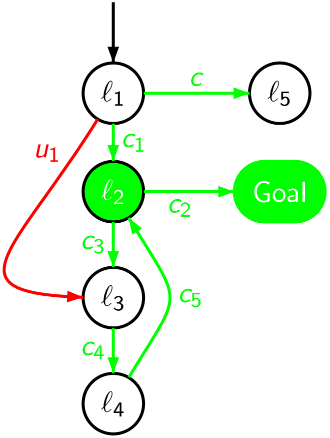
Controllable Predecessors:

$$\pi(X) = \big(\text{cPred}(X) \setminus \text{uPred}(\overline{X})\big)$$

Iterate π:

1. $X_0 = \{\text{Goal}\}$

Backward Computation of Winning States



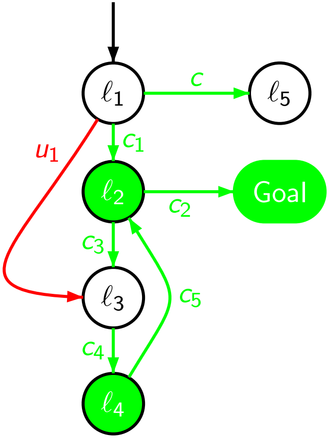
Controllable Predecessors:

$$\pi(X) = \big(\mathrm{cPred}(X) \setminus \mathrm{uPred}(\overline{X})\big)$$

Iterate π:

1. $X_0 = \{\mathrm{Goal}\}$
2. $X_1 = \{\mathrm{Goal}, \ell_2\}$

Backward Computation of Winning States



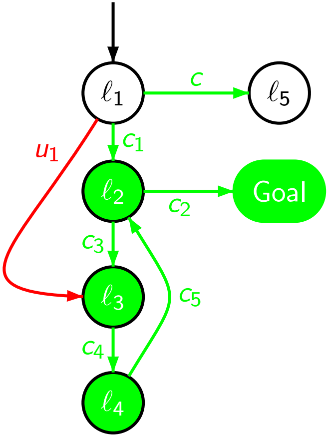
Controllable Predecessors:

$$\pi(X) = \left(\text{cPred}(X) \setminus \text{uPred}(\overline{X})\right)$$

Iterate π:

1. $X_0 = \{\text{Goal}\}$
2. $X_1 = \{\text{Goal}, \ell_2\}$
3. $X_2 = \{\text{Goal}, \ell_2, \ell_4\}$

Backward Computation of Winning States



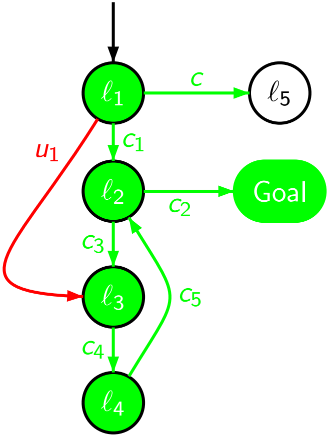
Controllable Predecessors:

$$\pi(X) = \left(\mathrm{cPred}(X) \setminus \mathrm{uPred}(\overline{X})\right)$$

Iterate π:

1. $X_0 = \{\mathrm{Goal}\}$
2. $X_1 = \{\mathrm{Goal}, \ell_2\}$
3. $X_2 = \{\mathrm{Goal}, \ell_2, \ell_4\}$
4. $X_3 = \{\mathrm{Goal}, \ell_2, \ell_4, \ell_3\}$

Backward Computation of Winning States



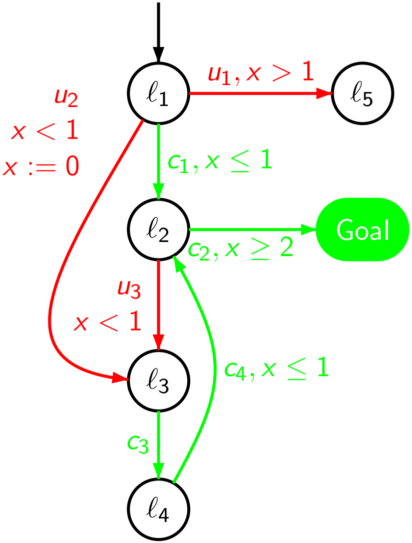
Controllable Predecessors:

$$\pi(X) = \big(\mathrm{cPred}(X) \setminus \mathrm{uPred}(\overline{X})\big)$$

Iterate π:

1. $X_0 = \{\mathrm{Goal}\}$
2. $X_1 = \{\mathrm{Goal}, \ell_2\}$
3. $X_2 = \{\mathrm{Goal}, \ell_2, \ell_4\}$
4. $X_3 = \{\mathrm{Goal}, \ell_2, \ell_4, \ell_3\}$
5. $X_4 = \{\mathrm{Goal}, \ell_2, \ell_4, \ell_3, \ell_1\}$

Reachability Control for Timed Games

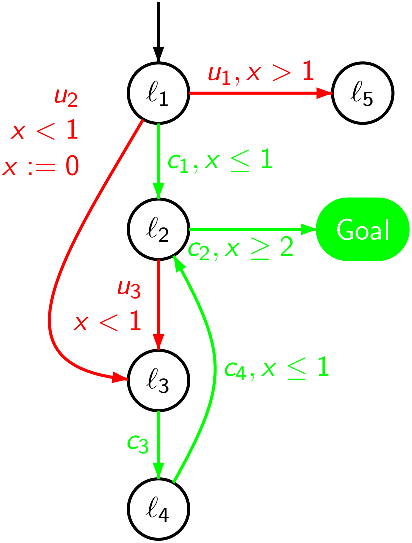


Safe Time Elapsing

When is it safe to let time elapse from q to q' ?

q $\qquad$ $q' \in X$

Reachability Control for Timed Games

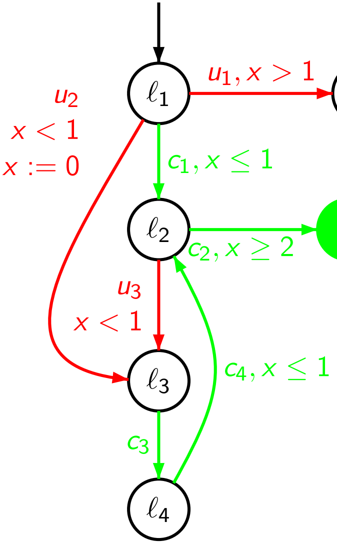


Safe Time Elapsing

When is it safe to let time elapse from q to q' ?

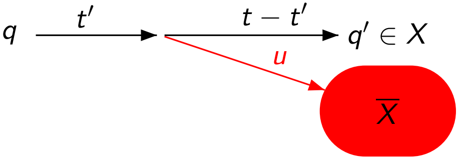
$$q \xrightarrow{t} q' \in X$$

Reachability Control for Timed Games

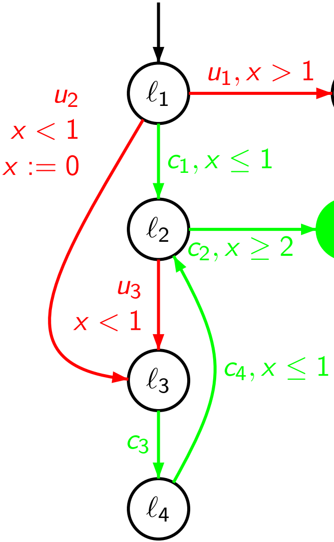


Safe Time Elapsing

When is it safe to let time elapse from q to q' ?

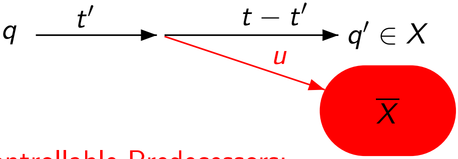


Reachability Control for Timed Games



Safe Time Elapsing

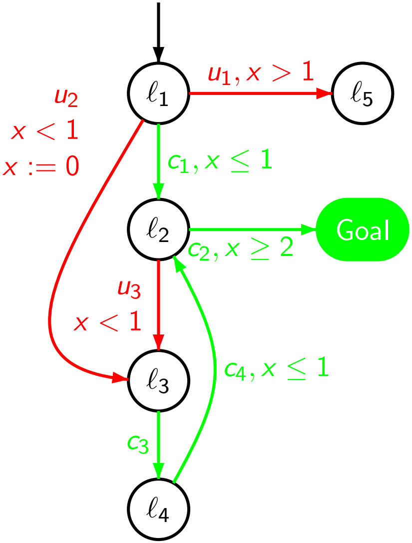
When is it safe to let time elapse from q to q' ?

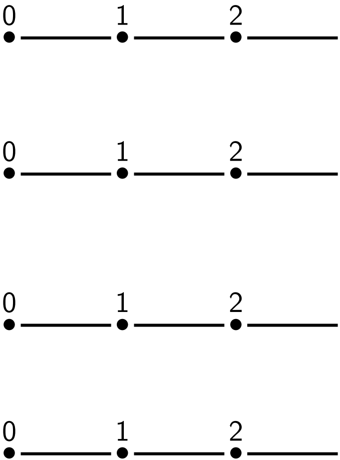


Controllable Predecessors:

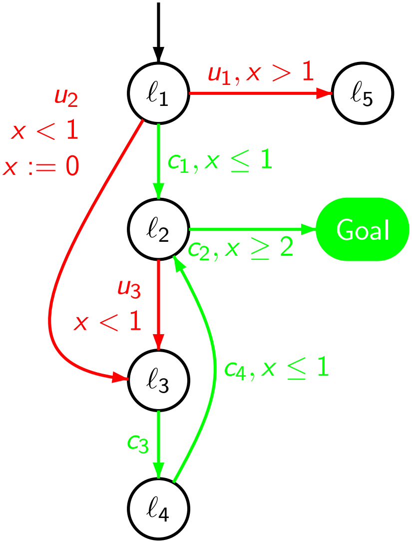
$$\pi(X) = \text{Pred}_t\big(X \cup \text{cPred}(X), \text{uPred}(\overline{X})\big)$$

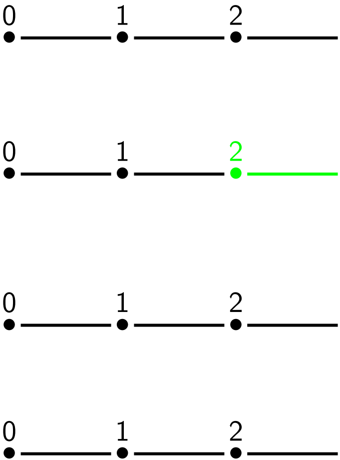
Reachability Control for Timed Games



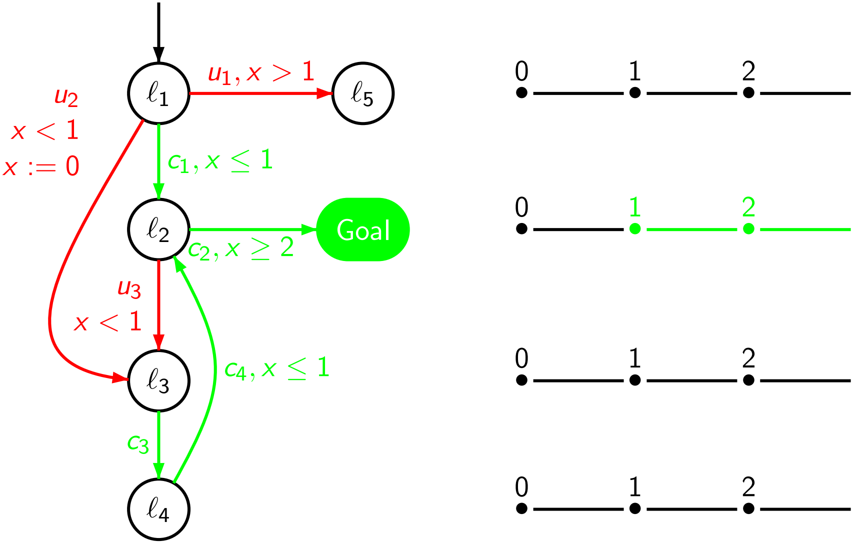


Reachability Control for Timed Games

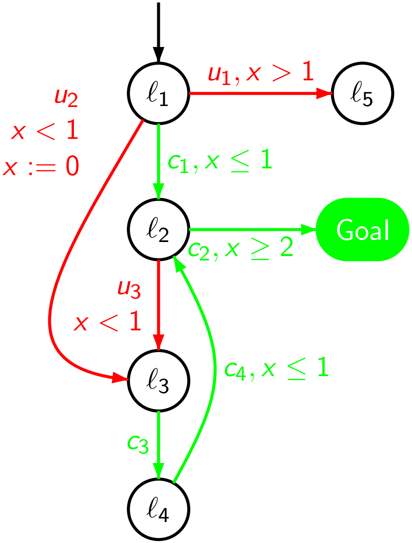


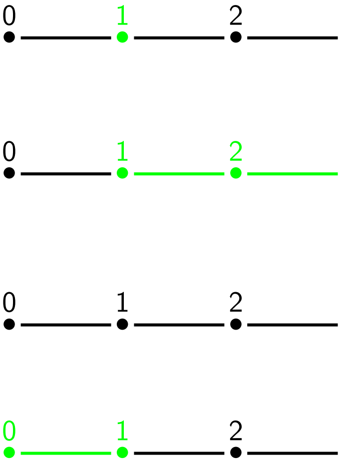


Reachability Control for Timed Games

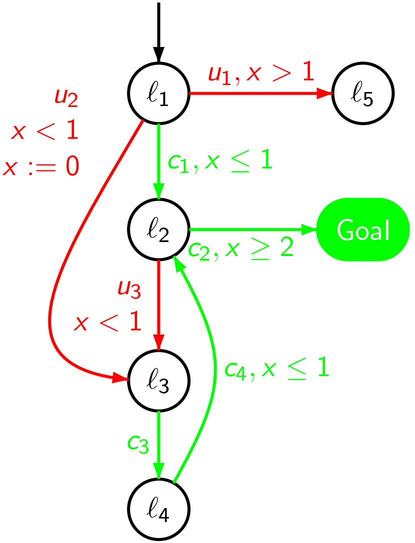


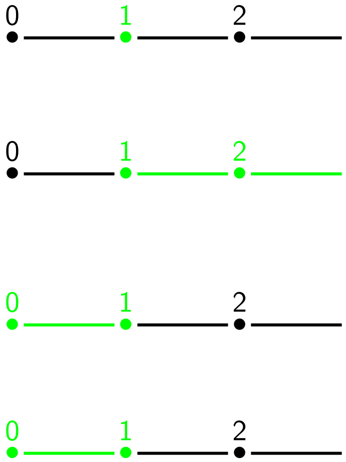
Reachability Control for Timed Games



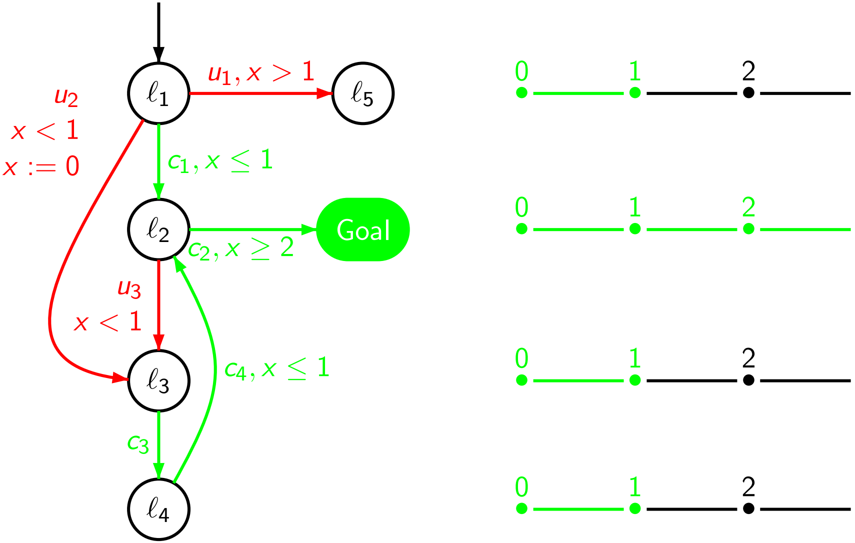


Reachability Control for Timed Games





Reachability Control for Timed Games



State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata
  [Maler (STACS'95), Asarin (SSC'98)]

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

Our Contribution: True On-the-fly algorithm for reachability timed games

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

Our Contribution: True On-the-fly algorithm for reachability timed games

- Advantages:

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

Our Contribution: True On-the-fly algorithm for reachability timed games

- Advantages:
  - avoid constructing all backwards & forwards reachable states

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

Our Contribution: True On-the-fly algorithm for reachability timed games

- Advantages:
  - avoid constructing all backwards & forwards reachable states
  - allows for use of discrete variables (e.g. $i := i + 1$)

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

Our Contribution: True On-the-fly algorithm for reachability timed games

- Advantages:
  - avoid constructing all backwards & forwards reachable states
  - allows for use of discrete variables (e.g. $i := i + 1$)
- Extends to Time (sub)-Optimal Control

State-of-the-art

Known Results for Timed Games:

- Reachability in *Timed Automata* (Alur & Dill, 1994)
- Büchi Control for Timed Game Automata [Maler (STACS'95), Asarin (SSC'98)]
- Time Optimal Control [Asarin (HSCC'99)]
- Price Optimal Control for (cost non-zeno) Timed Automata [Bouyer (FSTTCS'04)]
- Half on-the-fly algorithm [Altisen (TOOLS'02)]

Our Contribution: True On-the-fly algorithm for reachability timed games

- Advantages:
  - avoid constructing all backwards & forwards reachable states
  - allows for use of discrete variables (e.g. $i := i + 1$)
- Extends to Time (sub)-Optimal Control
- Efficient implementation with UPPAAL libraries

Outline

Control Problems

Reachability Control

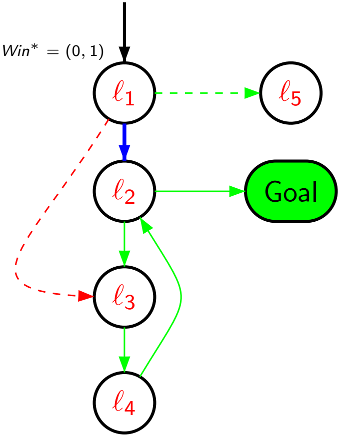
On-the-fly Algorithms for Reachability Control
- Finite State Games
- Timed Games

Implementation, Optimizations, Time Optimality

Experiments

Liu & Smolka Algorithm [Liu (ICALP'98)]

Liu & Smolka Algorithm [Liu (ICALP'98)]



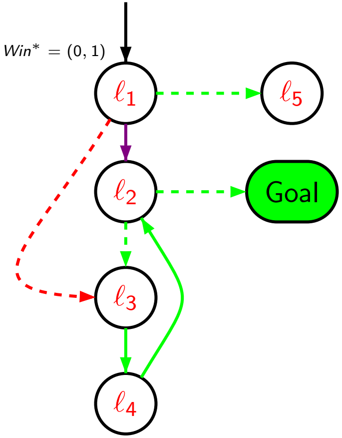
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal ? } 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal ? } 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
    }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



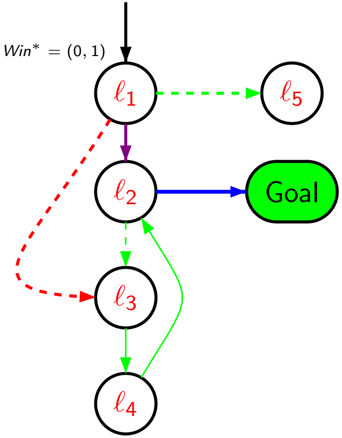
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



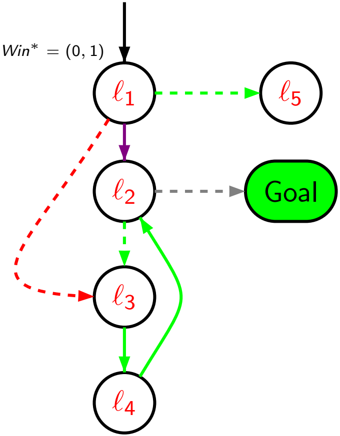
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal} ~?~ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal} ~?~ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



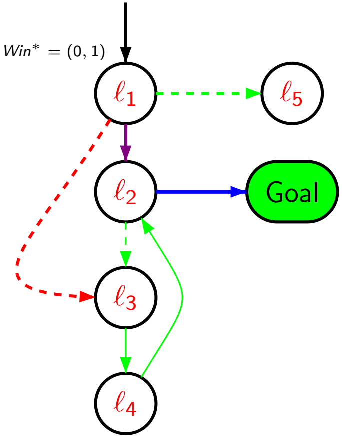
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



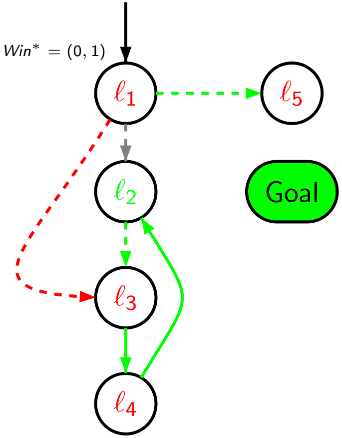
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1)$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



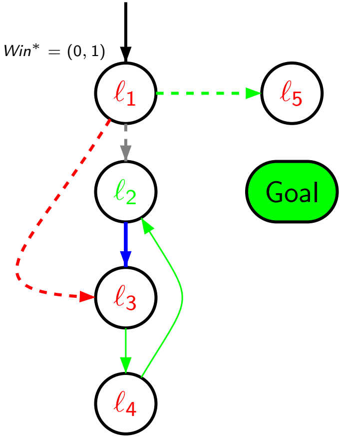
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



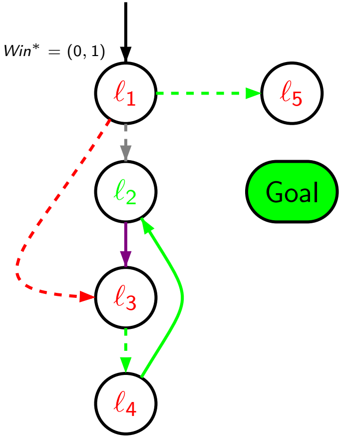
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal} \ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
$e = (q, \alpha, q') \leftarrow pop(Waiting)$;
if $q' \notin Passed$ **then** {
$Passed \leftarrow Passed \cup \{q'\}$;
$Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
$Win[q'] \leftarrow (q' \in \text{Goal} \ ?\ 1 : 0)$;
$Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
$Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
}
else (* reevaluate *)
$Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
$Waiting \leftarrow Waiting \cup Depend[q]$;
$Win[q] \leftarrow 1$;
}
if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



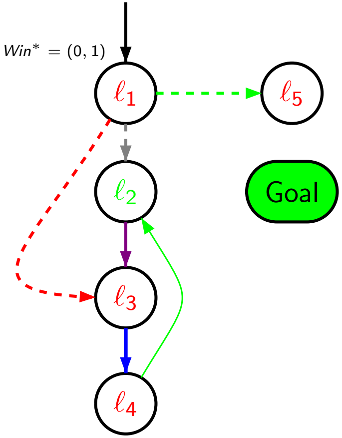
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal} ? 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
$e = (q, \alpha, q') \leftarrow pop(Waiting)$;
if $q' \notin Passed$ **then** {
$Passed \leftarrow Passed \cup \{q'\}$;
$Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
$Win[q'] \leftarrow (q' \in \text{Goal} ? 1 : 0)$;
$Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
$Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
}
else (* reevaluate *)
$Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
$Waiting \leftarrow Waiting \cup Depend[q]$;
$Win[q] \leftarrow 1$;
}
if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



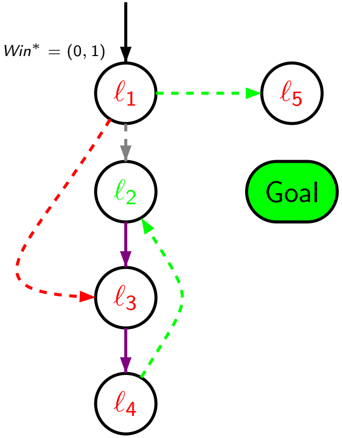
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal} ~?~ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal} ~?~ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow \textbf{Update}(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



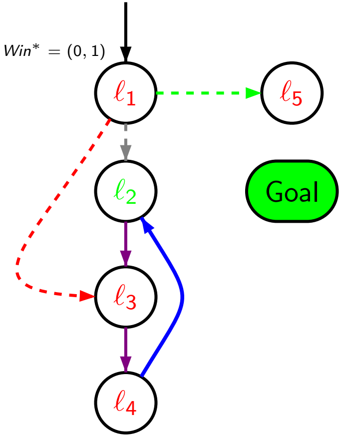
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \mathrm{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

```
while ((Waiting ≠ ∅) ∧ Win[q0] ≠ 1)) do
    e = (q, α, q') ← pop(Waiting);
    if q' ∉ Passed then {
        Passed ← Passed ∪ {q'};
        Depend[q'] ← {(q, α, q')};
        Win[q'] ← (q' ∈ Goal ? 1 : 0);
        Waiting ← Waiting ∪ {(q', α, q'') | q' --α--> q''};
        Win*[q] ← (0, #{q --u-->});
        if Win[q'] then Waiting ← Waiting ∪ {e};
    }
    else (* reevaluate *)
        Win*[q] ← Update(Win*[q]) ;
        if (Win*[q] = (k, 0) ∧ k ≥ 1) then {
            Waiting ← Waiting ∪ Depend[q];
            Win[q] ← 1;
            }
        if Win[q'] = 0 then Depend[q'] ← Depend[q'] ∪ {e};
    endif
endwhile
```

Liu & Smolka Algorithm [Liu (ICALP'98)]



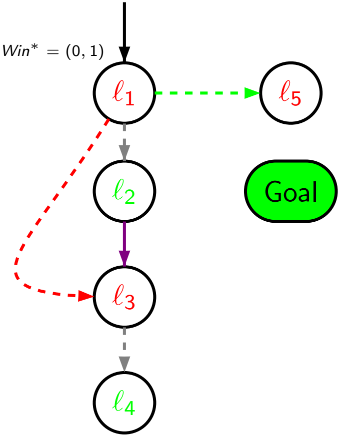
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal} ? 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
$e = (q, \alpha, q') \leftarrow pop(Waiting)$;
if $q' \notin Passed$ **then** {
$Passed \leftarrow Passed \cup \{q'\}$;
$Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
$Win[q'] \leftarrow (q' \in \text{Goal} ? 1 : 0)$;
$Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
$Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
}
else (* reevaluate *)
$Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
$Waiting \leftarrow Waiting \cup Depend[q]$;
$Win[q] \leftarrow 1$;
}
if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



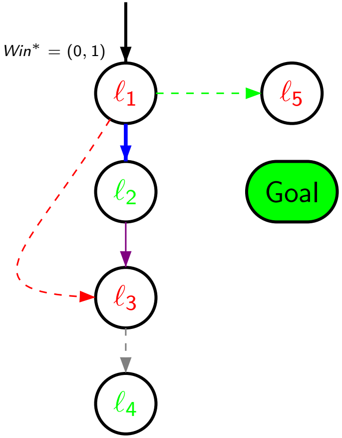
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1)$ **do**
$e = (q, \alpha, q') \leftarrow pop(Waiting)$;
if $q' \notin Passed$ **then** {
$Passed \leftarrow Passed \cup \{q'\}$;
$Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
$Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
$Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
$Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
}
else (* reevaluate *)
$Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
$Waiting \leftarrow Waiting \cup Depend[q]$;
$Win[q] \leftarrow 1$;
}
if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



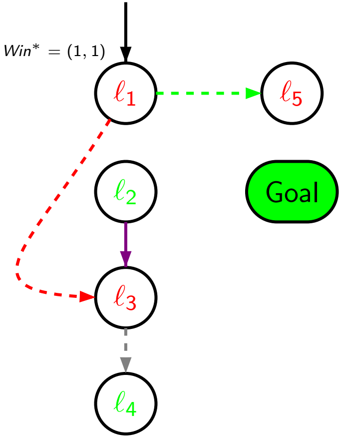
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

```
while ((Waiting ≠ ∅) ∧ Win[q0] ≠ 1)) do
   e = (q, α, q') ← pop(Waiting);
   if q' ∉ Passed then {
      Passed ← Passed ∪ {q'};
      Depend[q'] ← {(q, α, q')};
      Win[q'] ← (q' ∈ Goal ? 1 : 0);
      Waiting ← Waiting ∪ {(q', α, q'') | q' −α→ q''};
      Win*[q] ← (0, #{q −u→});
      if Win[q'] then Waiting ← Waiting ∪ {e};
   }
   else (* reevaluate *)
      Win*[q] ← Update(Win*[q]) ;
      if (Win*[q] = (k, 0) ∧ k ≥ 1) then {
         Waiting ← Waiting ∪ Depend[q];
         Win[q] ← 1;
         }
      if Win[q'] = 0 then Depend[q'] ← Depend[q'] ∪ {e};
   endif
endwhile
```

Liu & Smolka Algorithm [Liu (ICALP'98)]



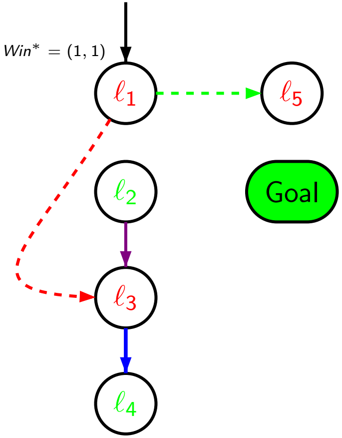
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

```
while ((Waiting ≠ ∅) ∧ Win[q0] ≠ 1)) do
    e = (q, α, q′) ← pop(Waiting);
    if q′ ∉ Passed then {
        Passed ← Passed ∪ {q′};
        Depend[q′] ← {(q, α, q′)};
        Win[q′] ← (q′ ∈ Goal ? 1 : 0);
        Waiting ← Waiting ∪ {(q′, α, q′′) | q′ --α--> q′′};
        Win*[q] ← (0, #{q --u-->});
        if Win[q′] then Waiting ← Waiting ∪ {e};
    }
    else (* reevaluate *)
        Win*[q] ← Update(Win*[q]) ;
        if (Win*[q] = (k, 0) ∧ k ≥ 1) then {
            Waiting ← Waiting ∪ Depend[q];
            Win[q] ← 1;
            }
        if Win[q′] = 0 then Depend[q′] ← Depend[q′] ∪ {e};
    endif
endwhile
```

Liu & Smolka Algorithm [Liu (ICALP'98)]



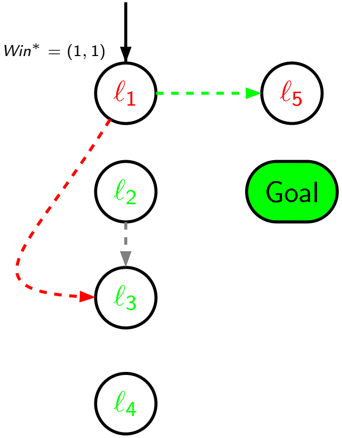
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



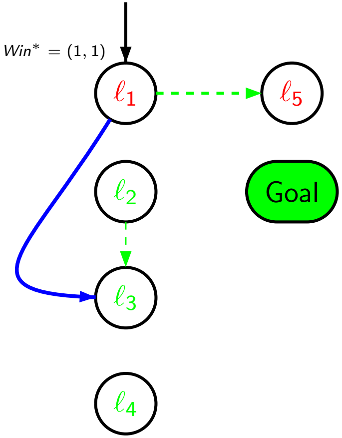
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



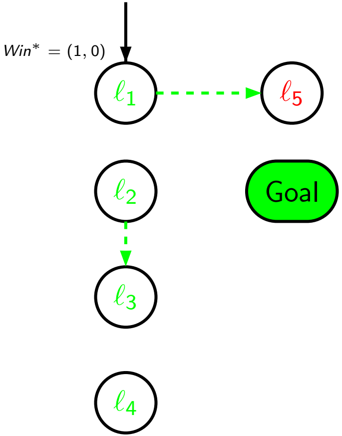
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal}\ ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
  $e = (q, \alpha, q') \leftarrow pop(Waiting)$;
  if $q' \notin Passed$ **then** {
    $Passed \leftarrow Passed \cup \{q'\}$;
    $Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
    $Win[q'] \leftarrow (q' \in \text{Goal}\ ?\ 1 : 0)$;
    $Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
    $Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
    if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  }
  else (* reevaluate *)
    $Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
    if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
      $Waiting \leftarrow Waiting \cup Depend[q]$;
      $Win[q] \leftarrow 1$;
      }
    if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
  endif
endwhile

Liu & Smolka Algorithm [Liu (ICALP'98)]



Linear in # transitions

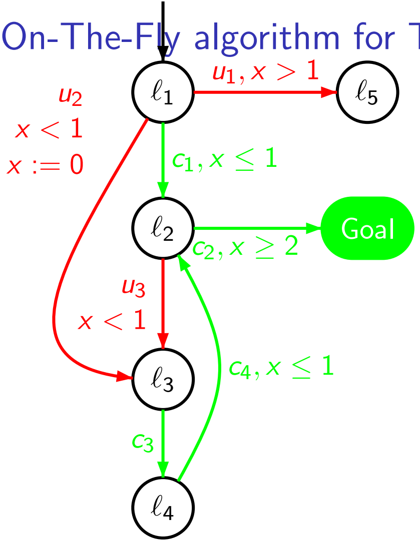
Initialization:

$Passed \leftarrow \{q_0\}$;
$Waiting \leftarrow \{(q_0, \alpha, q') \mid \alpha \in Act\ q \xrightarrow{\alpha} q'\}$;
$Win[q_0] \leftarrow (q_0 \in \text{Goal} ?\ 1 : 0)$;
$Depend[q_0] \leftarrow \emptyset$;

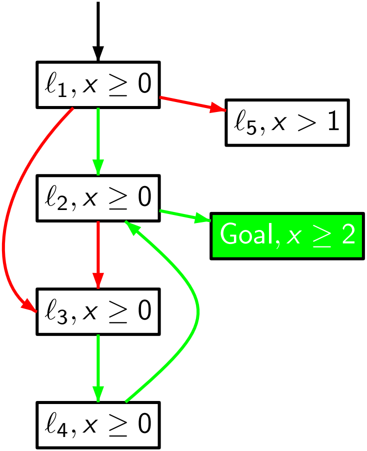
Main:

while $((Waiting \neq \emptyset) \wedge Win[q_0] \neq 1))$ **do**
$e = (q, \alpha, q') \leftarrow pop(Waiting)$;
if $q' \notin Passed$ **then** {
$Passed \leftarrow Passed \cup \{q'\}$;
$Depend[q'] \leftarrow \{(q, \alpha, q')\}$;
$Win[q'] \leftarrow (q' \in \text{Goal} ?\ 1 : 0)$;
$Waiting \leftarrow Waiting \cup \{(q', \alpha, q'') \mid q' \xrightarrow{\alpha} q''\}$;
$Win^*[q] \leftarrow (0, \#\{q \xrightarrow{u}\})$;
if $Win[q']$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
}
else (* reevaluate *)
$Win^*[q] \leftarrow$ **Update**$(Win^*[q])$;
if $(Win^*[q] = (k, 0) \wedge k \geq 1)$ **then** {
$Waiting \leftarrow Waiting \cup Depend[q]$;
$Win[q] \leftarrow 1$;
}
if $Win[q'] = 0$ **then** $Depend[q'] \leftarrow Depend[q'] \cup \{e\}$;
endif
endwhile

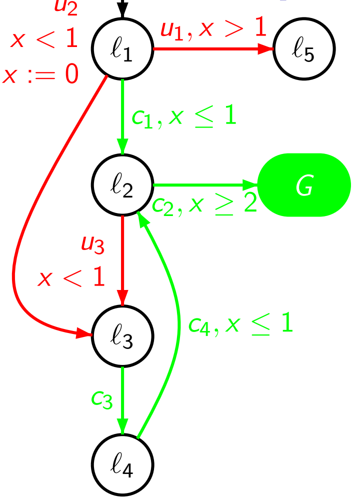
On-The-Fly algorithm for Timed Games: First Attempt



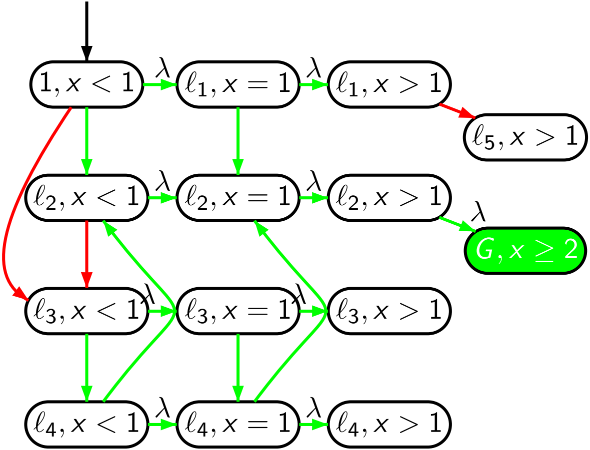
Using the Simulation Graph



Second Attempt [Tripakis (FM'99), Altisen (TOOLS'02)]



Stable Partitionning



Towards a True On-The-Fly Algorithm

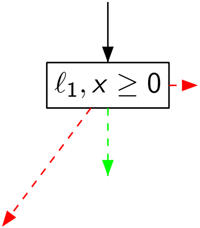
Our Approach:

- Write a Symbolic version of Liu & Smolka
- Use Symbolic states and Transitions
- Apply this to Timed Games

Key issues to be adressed:

- Symbolic States can be partially winning compared to FSG where 0 or 1
- When to propagate backward ?
- Termination, Complexity ?

Liu & Smolka for Timed Games



Initialization:

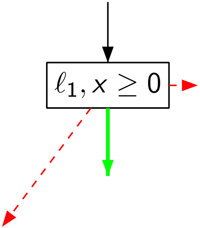
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

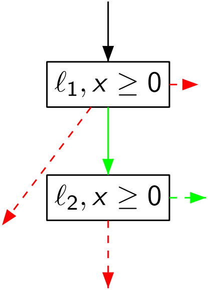
Passed $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
Waiting $\leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
Win$[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
Depend$[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

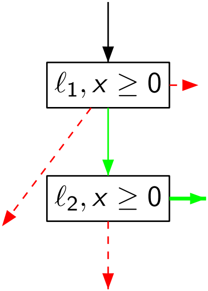
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}_{\geq 0}^X)$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}_{\geq 0}^X)$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

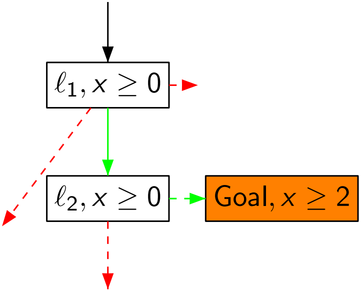
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

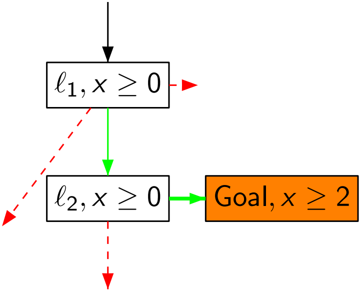
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

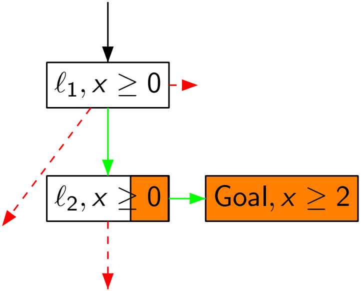
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

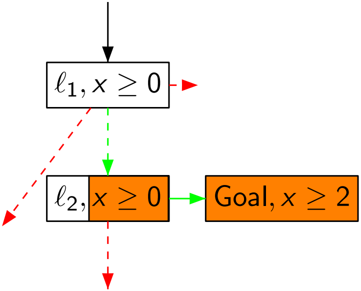
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
 $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
 if $S' \notin Passed$ **then**
  $Passed \leftarrow Passed \cup \{S'\}$;
  $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
  $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
  $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
  if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
 else (* reevaluate *)
  $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
  if $(Win[S] \subsetneq Win^*)$ **then**
   $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
  $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
 endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

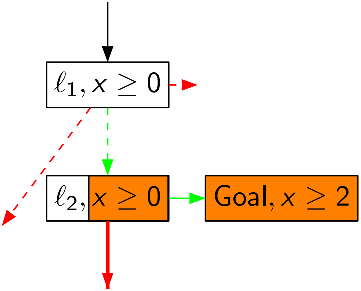
Passed $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
Waiting $\leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
Win$[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}_{\geq 0}^X)$;
Depend$[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
 $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
 if $S' \notin Passed$ **then**
  $Passed \leftarrow Passed \cup \{S'\}$;
  $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
  $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}_{\geq 0}^X)$;
  $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
  if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
 else (* reevaluate *)
  $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
  if $(Win[S] \subsetneq Win^*)$ **then**
   $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
  $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
 endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

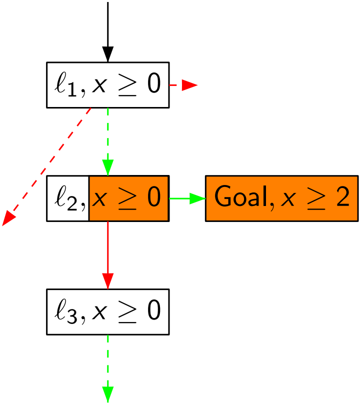
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

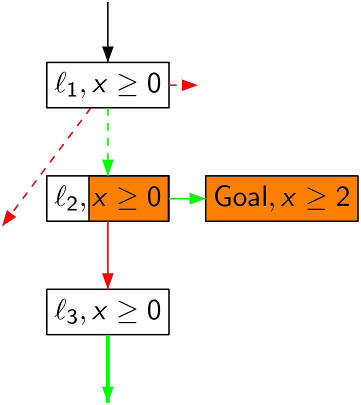
$\quad$ *Passed* $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$\quad$ *Waiting* $\leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$\quad$ $Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$\quad$ $Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$\quad$ $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
$\quad$ **if** $S' \notin Passed$ **then**
$\quad\quad$ $Passed \leftarrow Passed \cup \{S'\}$;
$\quad\quad$ $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$\quad\quad$ $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$\quad\quad$ $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
$\quad\quad$ **if** $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
$\quad$ **else (* reevaluate *)**
$\quad\quad$ $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]),$
$\quad\quad\quad$ $\bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
$\quad\quad$ **if** $(Win[S] \subsetneq Win^*)$ **then**
$\quad\quad\quad$ $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$\quad\quad$ $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
$\quad$ **endif**
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

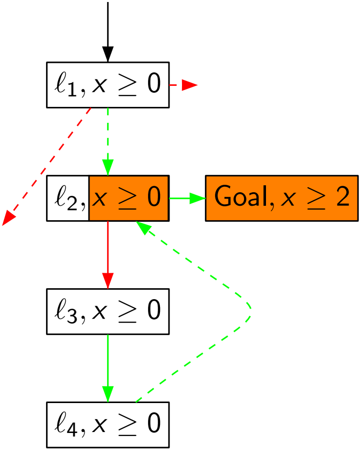
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

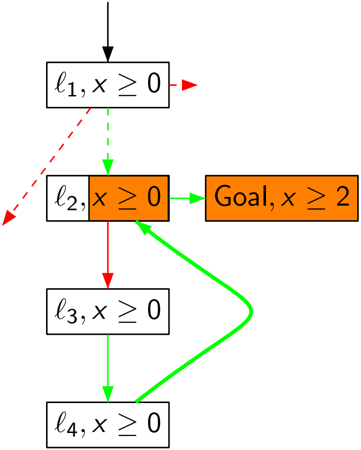
Passed $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
Waiting $\leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

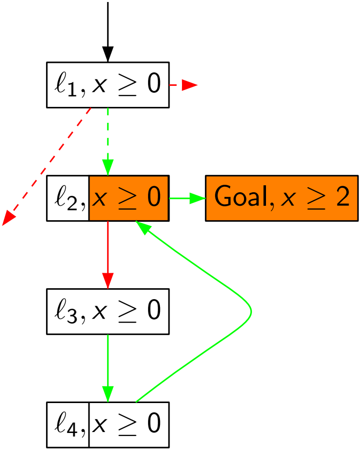
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

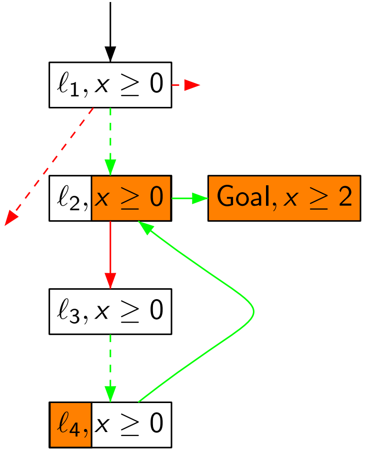
Passed $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
Waiting $\leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

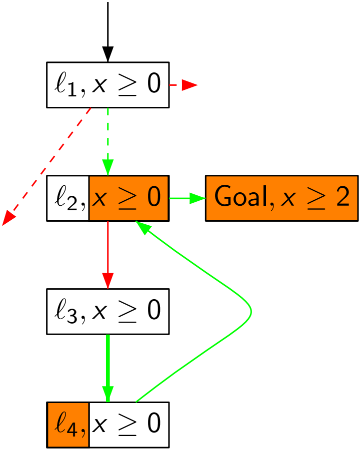
$\quad$ *Passed* $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$\quad$ *Waiting* $\leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$\quad$ $Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}_{\geq 0}^X)$;
$\quad$ $Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$\quad$ $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
$\quad$ **if** $S' \notin Passed$ **then**
$\quad\quad$ $Passed \leftarrow Passed \cup \{S'\}$;
$\quad\quad$ $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$\quad\quad$ $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}_{\geq 0}^X)$;
$\quad\quad$ $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
$\quad\quad$ **if** $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
$\quad$ **else** **(* reevaluate *)**
$\quad\quad$ $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
$\quad\quad$ **if** $(Win[S] \subsetneq Win^*)$ **then**
$\quad\quad\quad$ $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$\quad\quad$ $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
$\quad$ **endif**
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

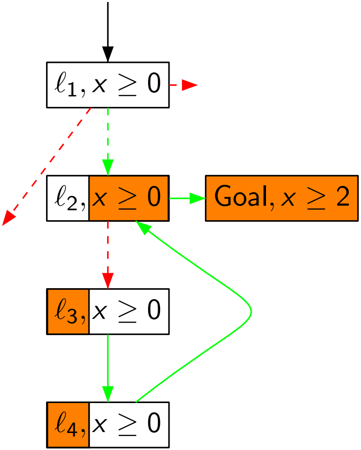
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

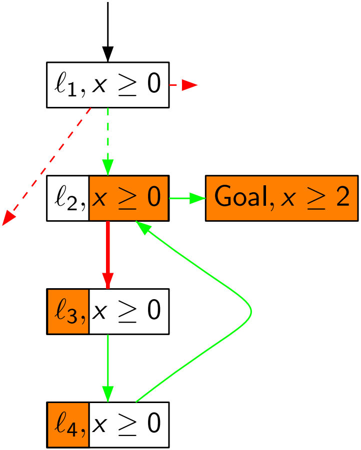
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

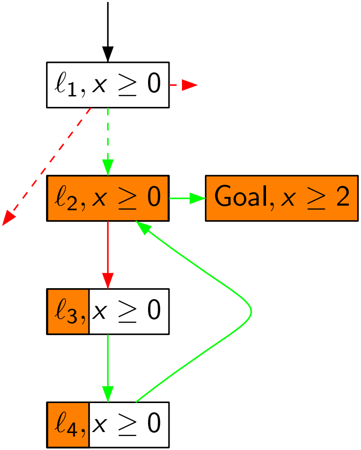
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

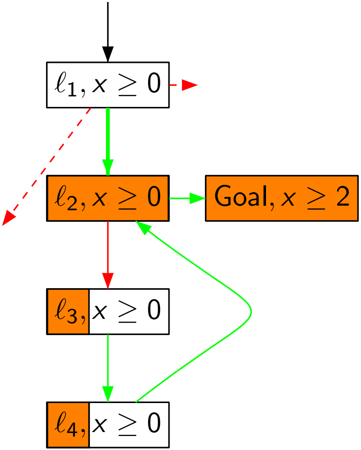
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

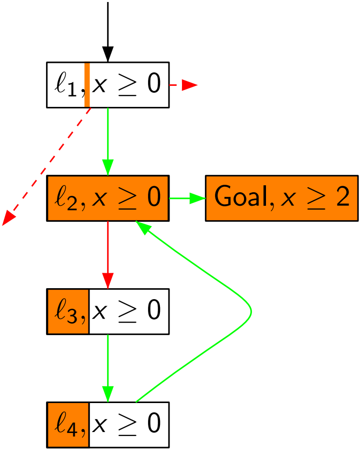
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

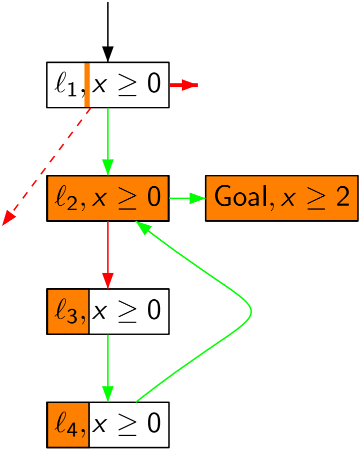
Passed $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
Waiting $\leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

▸▸ Skip algorithm

Liu & Smolka for Timed Games



Initialization:

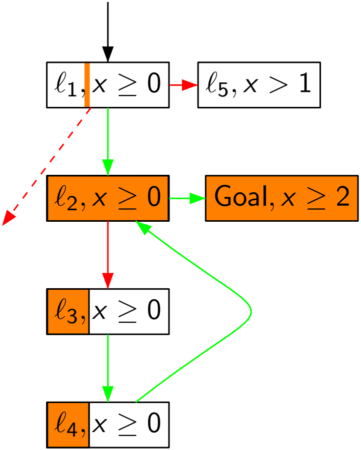
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  else (* reevaluate *)
    $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

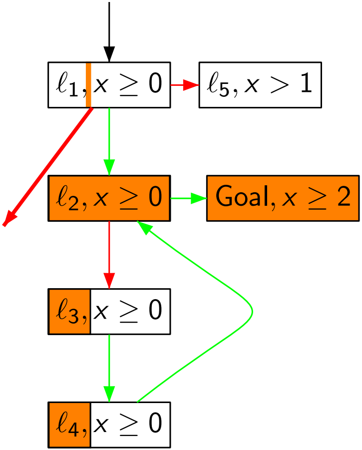
$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* **reevaluate** *)
$Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

▸▸ Skip algorithm

Liu & Smolka for Timed Games



Initialization:

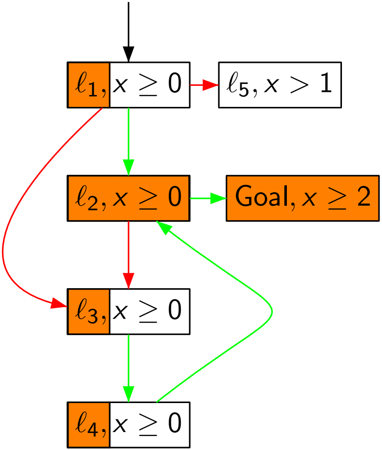
$\quad$ *Passed* $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
$\quad$ *Waiting* $\leftarrow \{(S_0, \alpha, S') \mid S' = \mathrm{Post}_\alpha(S_0)^{\nearrow}\}$;
$\quad$ $Win[S_0] \leftarrow S_0 \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$\quad$ $Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$\quad$ $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
$\quad$ **if** $S' \notin Passed$ **then**
$\quad\quad$ $Passed \leftarrow Passed \cup \{S'\}$;
$\quad\quad$ $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$\quad\quad$ $Win[S'] \leftarrow S' \cap (\{\mathrm{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$\quad\quad$ $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathrm{Post}_\alpha(S')^{\nearrow}\}$;
$\quad\quad$ **if** $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
$\quad$ **else (* reevaluate *)**
$\quad\quad$ $Win^* \leftarrow \mathrm{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathrm{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \mathrm{Pred}_u(T \setminus Win[T])) \cap S$;
$\quad\quad$ **if** $(Win[S] \subsetneq Win^*)$ **then**
$\quad\quad\quad$ $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$\quad\quad$ $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
$\quad$ **endif**
endwhile

Skip algorithm

Liu & Smolka for Timed Games



Initialization:

Passed $\leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \bar{0})\}^{\nearrow}$;
Waiting $\leftarrow \{(S_0, \alpha, S') \mid S' = \text{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

Main:

while $((Waiting \neq \emptyset) \wedge ((\ell_0, \bar{0}) \notin Win[S_0]))$ **do**
$e = (S, \alpha, S') \leftarrow pop(Waiting)$;
if $S' \notin Passed$ **then**
$Passed \leftarrow Passed \cup \{S'\}$;
$Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
$Win[S'] \leftarrow S' \cap (\{\text{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \text{Post}_\alpha(S')^{\nearrow}\}$;
if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
else (* reevaluate *)
$Win^* \leftarrow \text{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \text{Pred}_c(Win[T]), \bigcup_{S \xrightarrow{u} T} \text{Pred}_u(T \setminus Win[T])) \cap S$;
if $(Win[S] \subsetneq Win^*)$ **then**
$Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
$Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
endif
endwhile

Skip algorithm

Summary of the Results

- A True on-the-fly algorithm
  for reachability control and safety control
- Strategies can be computed
- Termination
  A symbolic edge (S, α, T)
  will be at most (1+ # regions(T)) times in *Waiting*
- Complexity
  A region may be in many symbolic states
  Our algorithm is Not linear in the size of the region graph
  Still it is theoretically optimal (EXPTIME)
- ... seems also good in practice!

Outline

Control Problems

Reachability Control

On-the-fly Algorithms for Reachability Control

Implementation, Optimizations, Time Optimality

Experiments

Efficient Implementation of Pred_t

Theorem

The following distribution law holds:

$$\mathrm{Pred}_t(\bigcup_i G_i, \bigcup_j B_j) = \bigcup_i \bigcap_j \mathrm{Pred}_t(G_i, B_j) \quad (1)$$

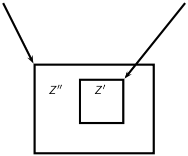
Theorem

If B is a convex set, then:

$$\mathrm{Pred}_t(G, B) = (G^{\swarrow} \setminus B^{\swarrow}) \cup ((G \cap B^{\swarrow}) \setminus B)^{\swarrow} \quad (2)$$

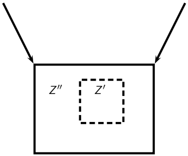
Inclusion Checking & Losing States

Inclusion checking



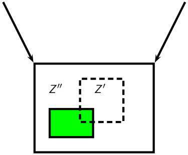
Inclusion Checking & Losing States

Inclusion checking



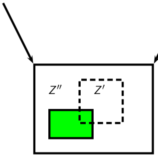
Inclusion Checking & Losing States

Inclusion checking

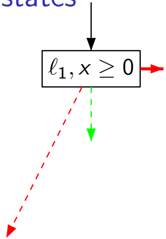


Inclusion Checking & Losing States

Inclusion checking

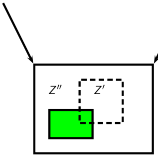


Losing states

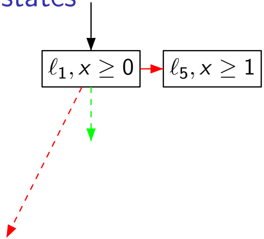


Inclusion Checking & Losing States

Inclusion checking

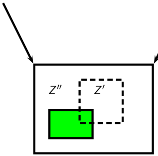


Losing states

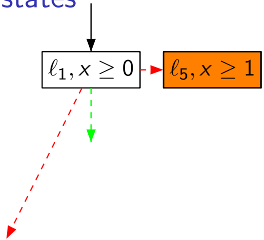


Inclusion Checking & Losing States

Inclusion checking

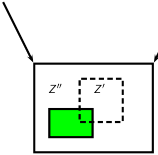


Losing states

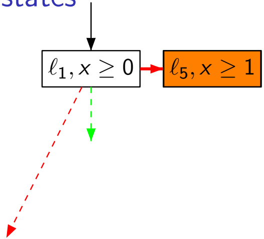


Inclusion Checking & Losing States

Inclusion checking

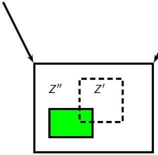


Losing states

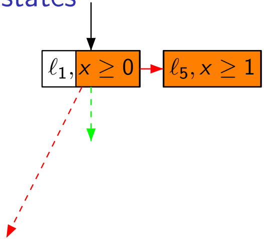


Inclusion Checking & Losing States

Inclusion checking

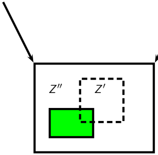


Losing states

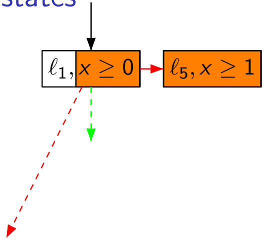


Inclusion Checking & Losing States

Inclusion checking



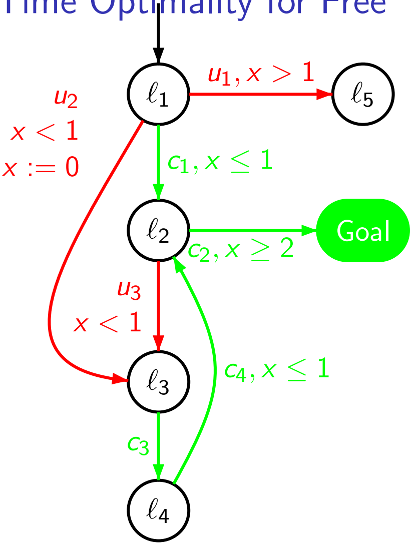
Losing states



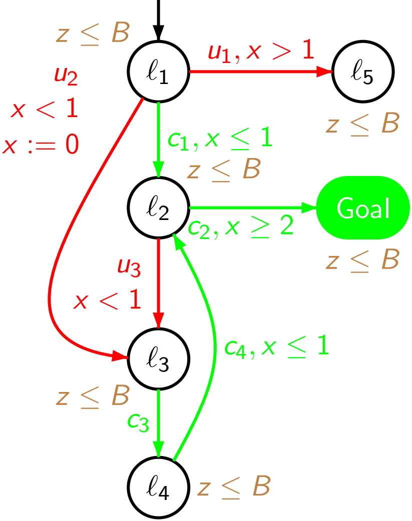
Pruning

Main:
while $((Waiting \neq \emptyset) \wedge (s_0 \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $Win[S] \subsetneq S$ **then**
    if $S' \notin Passed$ **then**
      $(\ldots)$
    else (* reevaluate *)
      $(\ldots)$
    endif
  endif
endwhile

Time Optimality for Free



Time Optimality for Free

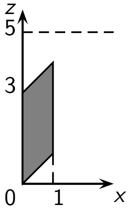


Assume:

- The initial state is winning
- We know an upper bound B of the optimal time needed to reach the winning state

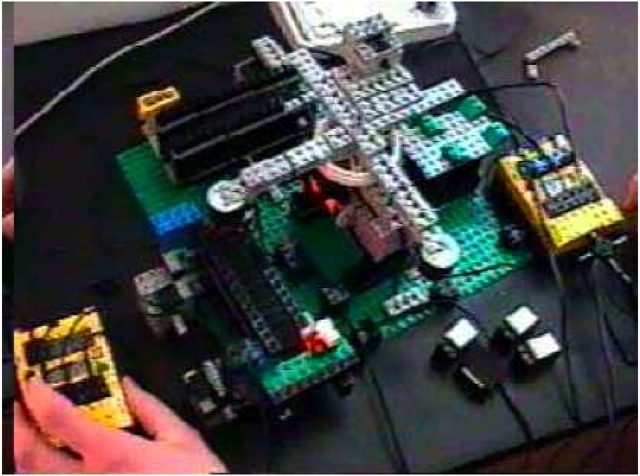
To compute the optimal time:

- Add a clock z (unconstrained at the beginning)
- Add a global invariant $z \leq B$

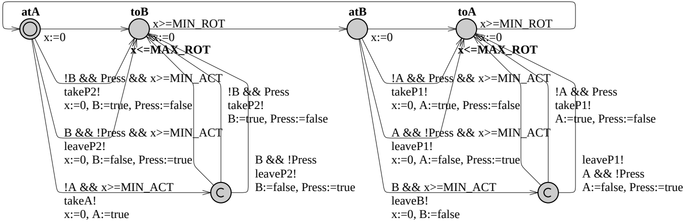


Outline

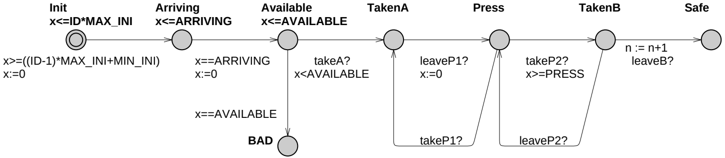
Experimental Results



Model of the robot



Model of the plates



Comparisons of Different Optimizations

Plates		Basic		Basic +inc		Basic +inc +pruning		Basic+lose +inc +pruning		Basic+lose +inc +topt	
		time	mem	time	mem	time	mem	time	mem	time	mem
2	win	0.0s	1M	0.0s	1M	0.0s	1M	0.0s	1M	0.04s	1M
	lose	0.0s	1M	0.0s	1M	0.0s	1M	0.0s	1M	n/a	n/a
3	win	0.5s	19M	0.0s	1M	0.0s	1M	0.1s	1M	0.27s	4M
	lose	1.1s	45M	0.1s	1M	0.0s	1M	0.2s	3M	n/a	n/a
4	win	33.9s	1395M	0.2s	8M	0.1s	6M	0.4s	5M	1.88s	13M
	lose	-	-	0.5s	11M	0.4s	10M	0.9s	9M	n/a	n/a
5	win	-	-	3.0s	31M	1.5s	22M	2.0s	16M	13.35s	59M
	lose	-	-	11.1s	61M	5.9s	46M	7.0s	41M	n/a	n/a
6	win	-	-	89.1s	179M	38.9s	121M	12.0s	63M	220.3s	369M
	lose	-	-	699s	480M	317s	346M	135.1s	273M	n/a	n/a
7	win	-	-	3256s	1183M	1181s	786M	124s	319M	6188s	2457M
	lose	-	-	-	-	16791s	2981M	4075s	2090M	n/a	n/a

Even better results are coming soon!

Conclusion and Future work

Conclusions:

- Successful development of a truly on-the-fly algorithm for reachability and safety games
- Efficient implementation using the UPPAAL DBM library
- Promising experimental results

Future work:

- Integration with the UPPAAL GUI.
- Guiding of the exploration by ordering the waiting list (heuristics) e.g.: Application to Job-Shop problems
- Distributed implementation
- Extension to deal with Partial observability criteria

References I

K. Altisen and S. Tripakis.
Tools for controller synthesis of timed systems.
In *Proc. 2nd Work. on Real-Time Tools (RT-TOOLS'02)*, 2002.
Proc. published as Technical Report 2002-025, Uppsala University, Sweden.

E. Asarin and O. Maler.
As Soon as Possible: Time Optimal Control for Timed Automata.
In *Proc. 2nd Work. Hybrid Systems: Computation & Control (HSCC'99)*, volume 1569 of *LNCS*, pages 19–30. Springer, 1999.

E. Asarin, O. Maler, A. Pnueli, and J. Sifakis.
Controller Synthesis for Timed Automata.
In *Proc. IFAC Symp. on System Structure & Control*, pages 469–474. Elsevier Science, 1998.

References II

P. Bouyer, F. Cassez, E. Fleury and K. Larsen.
Synthesis of Optimal Strategies for Timed Games.
In Proc. FSTTCS'04, LNCS. Springer, 2004.

X. Liu and S. Smolka.
Simple Linear-Time Algorithm for Minimal Fixed Points.
In *Proc. 26^{th} Conf. on Automata, Languages and Programming (ICALP'98)*, volume 1443 of *LNCS*, pages 53–66. Springer, 1998.

O. Maler, A. Pnueli, and J. Sifakis.
On the synthesis of discrete controllers for timed systems.
In *Proc. 12^{th} Symp. on Theoretical Aspects of Computer Science (STACS'95)*, volume 900, pages 229–242. Springer, 1995.

References III

S. Tripakis and K. Altisen.
On-the-Fly Controller Synthesis for Discrete and Timed Systems.
In *Proc. of World Congress on Formal Methods (FM'99)*, volume 1708 of *LNCS*, pages 233–252. Springer, 1999.