Nash Equilibria in Timed Games

Patricia Bouyer-Decitre

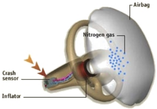
LSV, CNRS & ENS Cachan, France

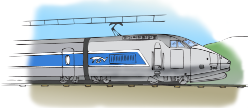
This talk is based on joint works with Romain Brenguier and Nicolas Markey

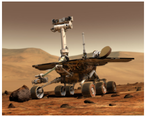
Outline

- Context: formal verification of systems
- Why timed systems?
  - to model time-constrained systems (eg embedded systems)
  - to model systems running in real-time (for instance GPS)

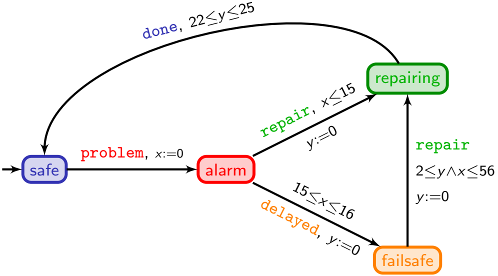




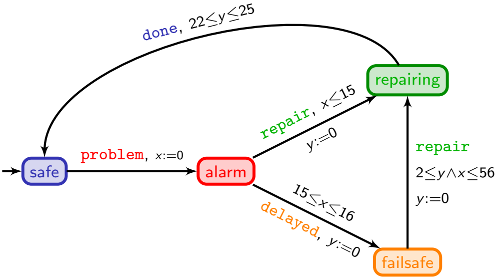





An example of a timed automaton

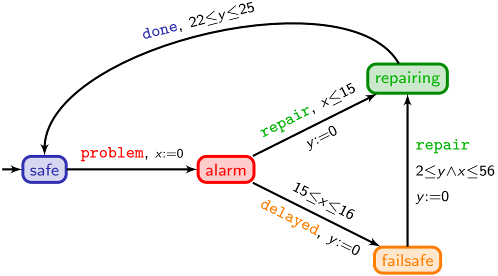


An example of a timed automaton



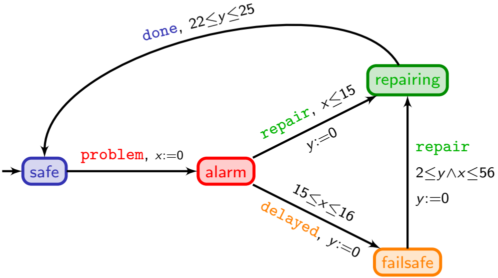
	safe
x	0
y	0

An example of a timed automaton



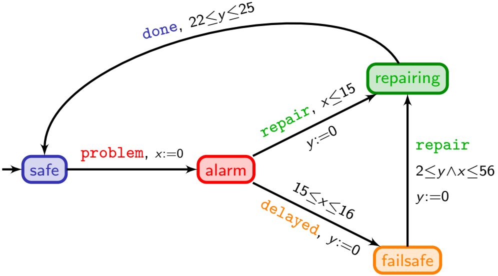
	safe	$\xrightarrow{23}$	safe
x	0		23
y	0		23

An example of a timed automaton



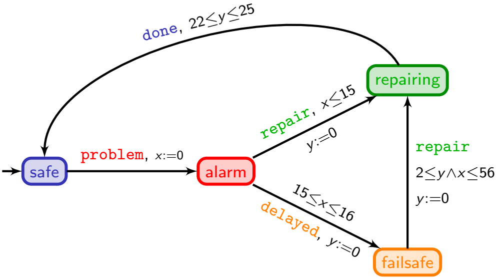
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm
x	0		23		0
y	0		23		23

An example of a timed automaton



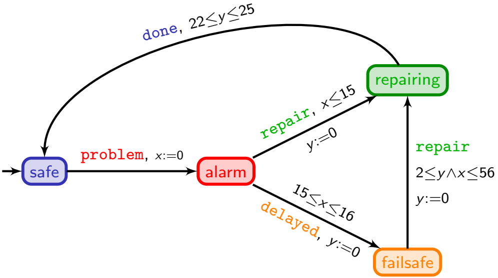
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm
x	0		23		0		15.6
y	0		23		23		38.6

An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe
x	0		23		0		15.6		15.6
y	0		23		23		38.6		0

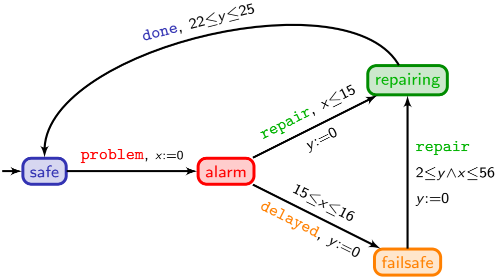
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	⋯
y	0		23		23		38.6		0	

	failsafe
⋯	15.6
	0

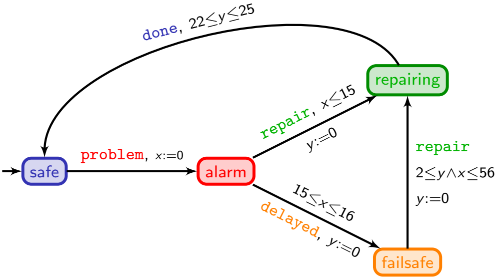
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	···
y	0		23		23		38.6		0	

	failsafe	$\xrightarrow{2.3}$	failsafe
···	15.6		17.9
	0		2.3

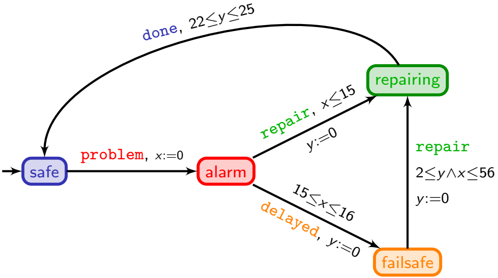
An example of a timed automaton



$$
\begin{array}{cccccccccccc}
 & \text{safe} & \xrightarrow{23} & \text{safe} & \xrightarrow{\texttt{problem}} & \text{alarm} & \xrightarrow{15.6} & \text{alarm} & \xrightarrow{\texttt{delayed}} & \text{failsafe} & \\
\text{x} & 0 & & 23 & & 0 & & 15.6 & & 15.6 & \cdots \\
\text{y} & 0 & & 23 & & 23 & & 38.6 & & 0 &
\end{array}
$$

$$
\begin{array}{cccccc}
 & \text{failsafe} & \xrightarrow{2.3} & \text{failsafe} & \xrightarrow{\texttt{repair}} & \text{repairing} \\
\cdots & 15.6 & & 17.9 & & 17.9 \\
 & 0 & & 2.3 & & 0
\end{array}
$$

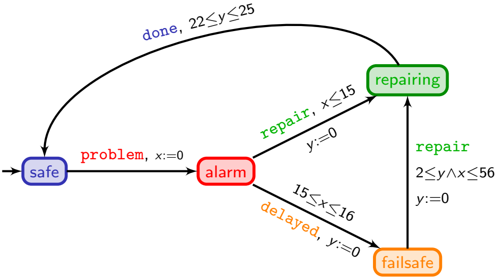
An example of a timed automaton



	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe	
x	0		23		0		15.6		15.6	$\cdots$
y	0		23		23		38.6		0	

	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\texttt{repair}}$	repairing	$\xrightarrow{22.1}$	repairing
$\cdots$	15.6		17.9		17.9		40
	0		2.3		0		22.1

An example of a timed automaton



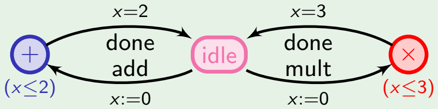
	safe	$\xrightarrow{23}$	safe	$\xrightarrow{\texttt{problem}}$	alarm	$\xrightarrow{15.6}$	alarm	$\xrightarrow{\texttt{delayed}}$	failsafe		
x	0		23		0		15.6		15.6		⋯
y	0		23		23		38.6		0		

	failsafe	$\xrightarrow{2.3}$	failsafe	$\xrightarrow{\texttt{repair}}$	repairing	$\xrightarrow{22.1}$	repairing	$\xrightarrow{\texttt{done}}$	safe
⋯	15.6		17.9		17.9		40		40
	0		2.3		0		22.1		22.1

Why (timed) games?

- to model uncertainty

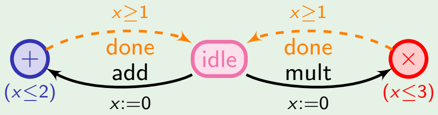
Example of a processor



Why (timed) games?

- to model uncertainty

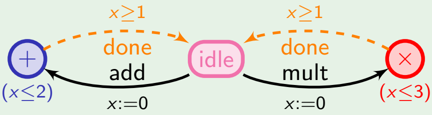
Example of a processor



Why (timed) games?

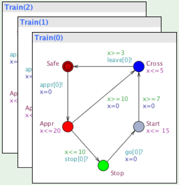
- to model uncertainty

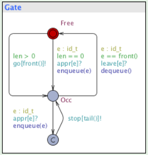
Example of a processor



- to model an interaction with an environment

Example of the gate in the train/gate example

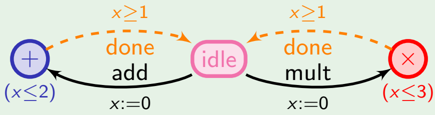




Why (timed) games?

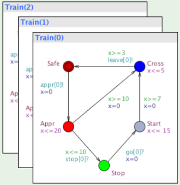
- to model uncertainty

Example of a processor



- to model an interaction with an environment

Example of the gate in the train/gate example

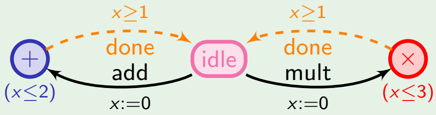


?

Why (timed) games?

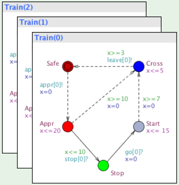
- to model uncertainty

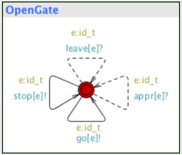
Example of a processor



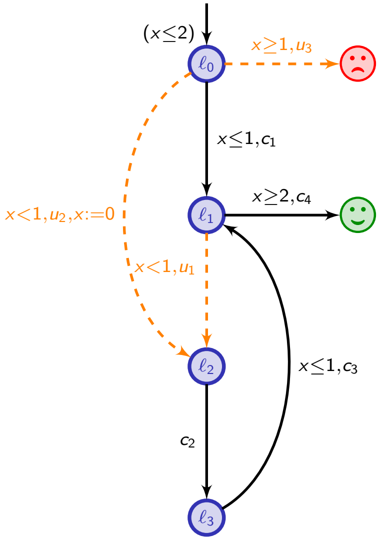
- to model an interaction with an environment

Example of the gate in the train/gate example





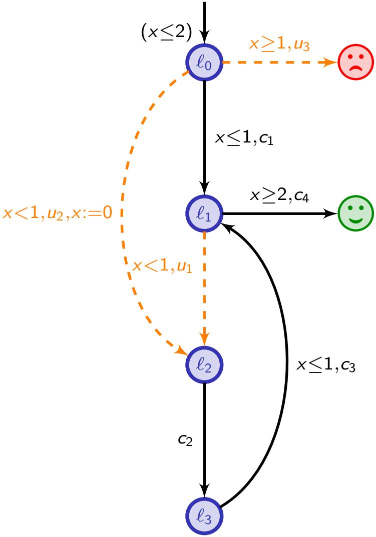
Control (two-player) timed games



Rule of the game

- Aim (for the controller): avoid ☹ and reach ☺

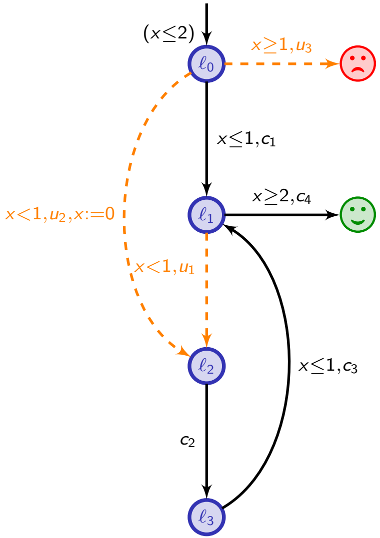
Control (two-player) timed games



Rule of the game

- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

Control (two-player) timed games



Rule of the game

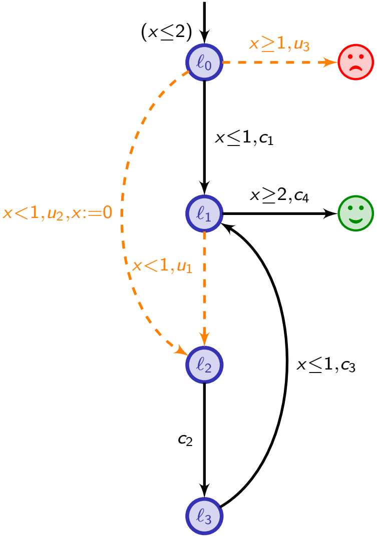
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

$$\|$$

move

Control (two-player) timed games

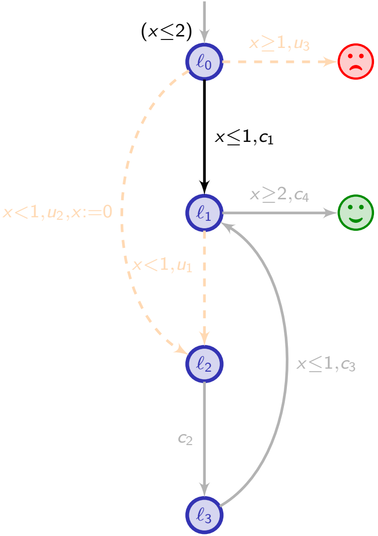


Rule of the game

- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

Control (two-player) timed games



Rule of the game

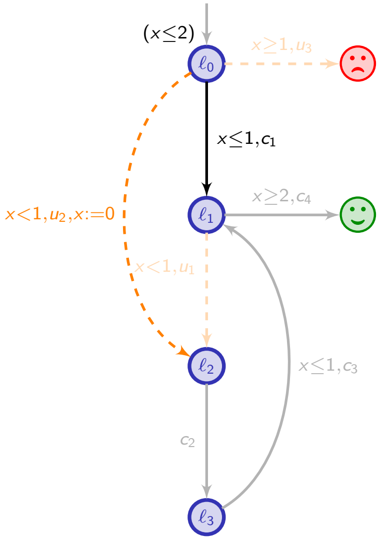
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$

Control (two-player) timed games



Rule of the game

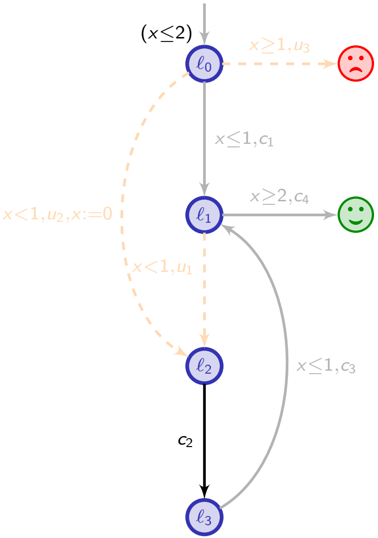
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2

Control (two-player) timed games



Rule of the game

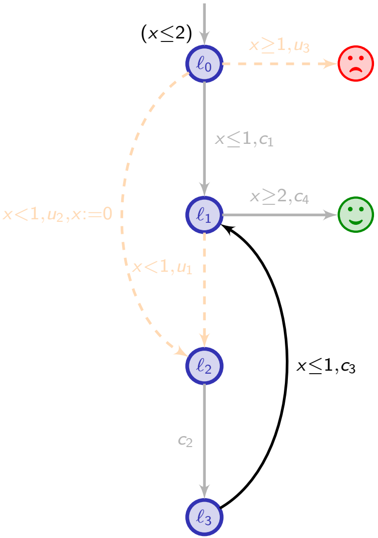
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2
- from $(\ell_2, \star)$, play $(1 - \star, c_2)$

Control (two-player) timed games



Rule of the game

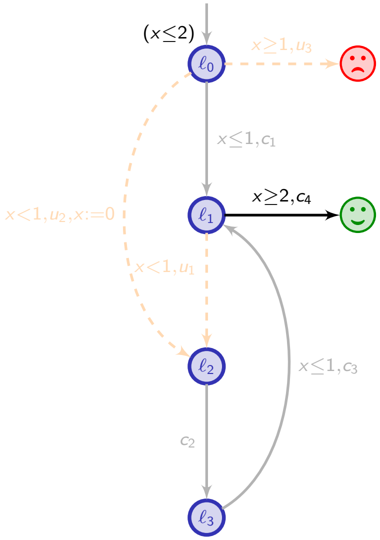
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2
- from $(\ell_2, \star)$, play $(1 - \star, c_2)$
- from $(\ell_3, 1)$, play $(0, c_3)$

Control (two-player) timed games



Rule of the game

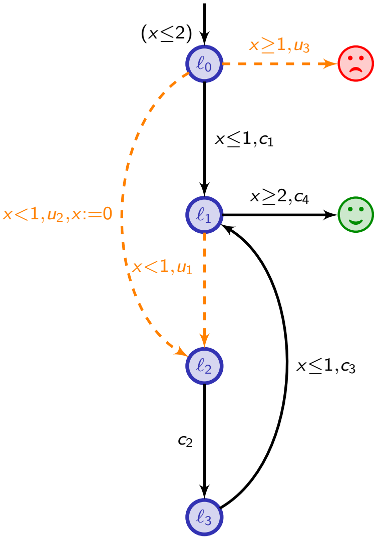
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2
- from $(\ell_2, \star)$, play $(1 - \star, c_2)$
- from $(\ell_3, 1)$, play $(0, c_3)$
- from $(\ell_1, 1)$, play $(1, c_4)$

Control (two-player) timed games



Rule of the game

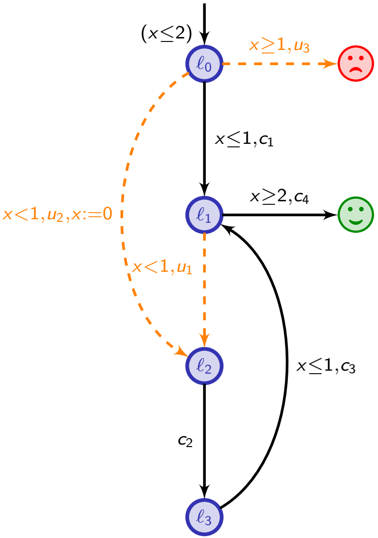
- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

What we are computing here

- (simple) worst-case winning strategies

Control (two-player) timed games



Rule of the game

- Aim (for the controller): avoid ☹ and reach ☺
- How do we play? According to strategies

$$f : \text{history} \mapsto (\text{delay}, \text{cont. transition})$$

What we are computing here

- (simple) worst-case winning strategies
- should win against any strategy of the opponent

A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)

[AMPS98] Asarin, Maler, Pnueli, Sifakis. Controller synthesis for timed automata *(SSC'98)*.
[HK99] Henzinger, Kopke. Discrete-time control for rectangular hybrid automata *(Theoretical Computer Science)*.

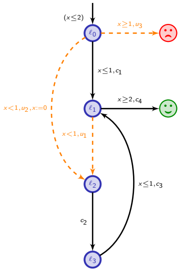
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



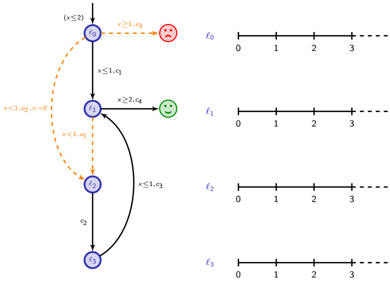
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



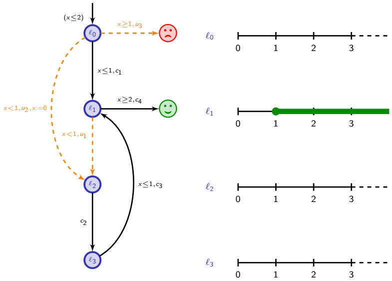
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



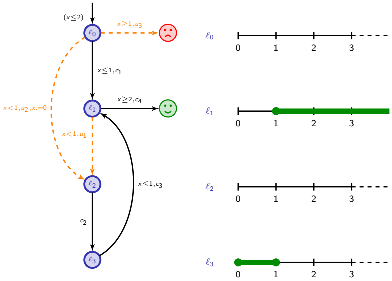
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



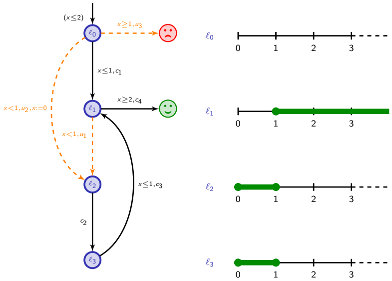
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



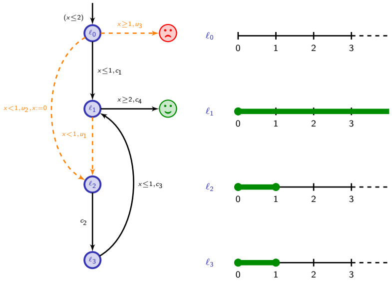
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



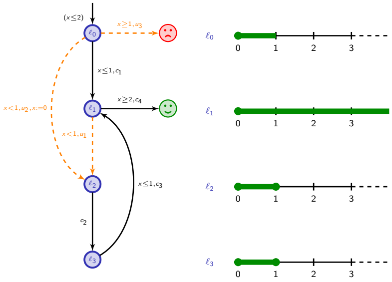
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



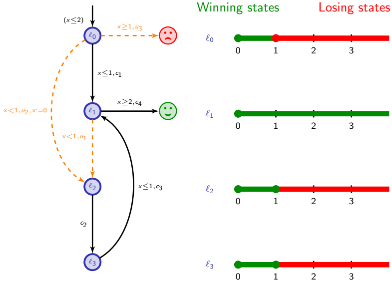
A short visit to control (two-player) timed games

Theorem [AMPS98,HK99,...]

For ω-regular objectives, it is decidable whether the controller has a winning strategy. It is EXPTIME-complete for safety and reachability objectives.

$\rightsquigarrow$ classical regions are sufficient to solve those games

(we can for instance compute the so-called attractor)



Outline

Multi-agent systems

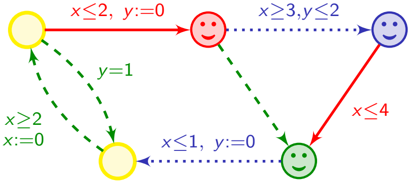
- Why multi-agent systems?
  - development of distributed systems (distributed scheduling problems, power control problems, mobile systems, ...)

Multi-agent systems

- Why multi-agent systems?
  - development of distributed systems (distributed scheduling problems, power control problems, mobile systems, ...)
- How can we model those systems?
  - with distributed automata
  - with multiplayer games
  - ...

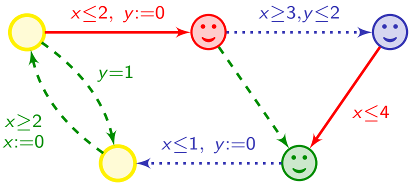
Multiplayer timed games

As a natural extension of two-player timed games:



Multiplayer timed games

As a natural extension of two-player timed games:

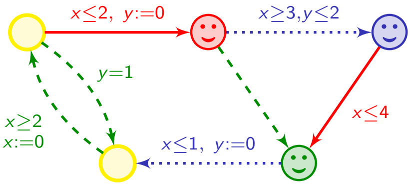


How do we play those games?

- Each player plays according to standard strategies (which specify moves at each step)
- Once all moves are given, the shortest delay is chosen and a corresponding transition is non-deterministically selected

Multiplayer timed games

As a natural extension of two-player timed games:



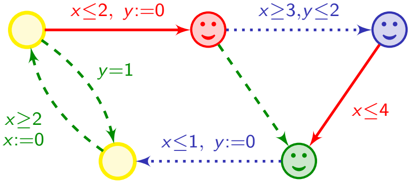
How do we play those games?

- Each player plays according to standard strategies (which specify moves at each step)
- Once all moves are given, the shortest delay is chosen and a corresponding transition is non-deterministically selected

$\leadsto$ They are infinite-state non-deterministic concurrent games
Strategies are assumed to be pure (*i.e.* not stochastic)

Multiplayer timed games

As a natural extension of two-player timed games:



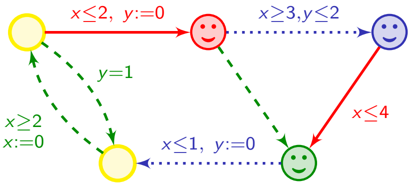
What is the aim of the game?

- Each player has her own objective (*via* a preference relation or a payoff function).

  NB: qualitative objective $=$ 0/1 payoff function

Multiplayer timed games

As a natural extension of two-player timed games:

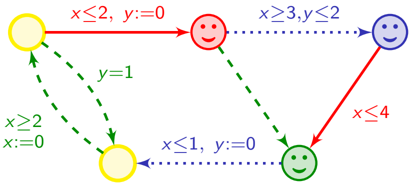


What is the aim of the game?

- Each player has her own objective (*via* a preference relation or a payoff function).
  NB: qualitative objective = 0/1 payoff function
- The notions of winning strategies are no more relevant, as those games are not zero-sum

Multiplayer timed games

As a natural extension of two-player timed games:



What is the aim of the game?

- Each player has her own objective (*via* a preference relation or a payoff function).
  NB: qualitative objective $= 0/1$ payoff function
- The notions of winning strategies are no more relevant, as those games are not zero-sum
- The players should play rationally, for instance according to equilibria (eg Nash equilibria, subgame-perfect equilibria, secure equilibria, ...)

Nash equilibria

Definition

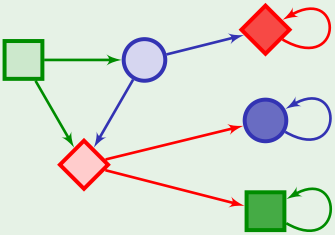
A strategy profile $(\sigma_j)_{A_j \in \mathsf{Agt}}$ is a Nash equilibrium if no player can improve her payoff by unilaterally changing her strategy.

Nash equilibria (in deterministic games)

Definition

A strategy profile $(\sigma_j)_{A_j \in \mathsf{Agt}}$ is a Nash equilibrium if no player can improve her payoff by unilaterally changing her strategy.

Example

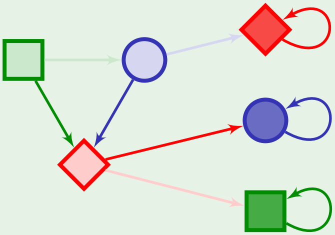


Nash equilibria (in deterministic games)

Definition

A strategy profile $(\sigma_j)_{A_j \in \mathsf{Agt}}$ is a Nash equilibrium if no player can improve her payoff by unilaterally changing her strategy.

Example



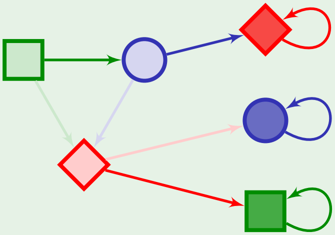
is a Nash equilibrium with payoff $(0, 1, 0)$

Nash equilibria (in deterministic games)

Definition

A strategy profile $(\sigma_j)_{A_j \in \mathsf{Agt}}$ is a Nash equilibrium if no player can improve her payoff by unilaterally changing her strategy.

Example



is not a Nash equilibrium

Nash equilibria (in non-deterministic games)

Definition

A strategy profile $(\sigma_j)_{A_j \in \mathsf{Agt}}$ is a Nash equilibrium if no player can improve her payoff by unilaterally changing her strategy.

What if there are several outcomes due to non-determinism?

Nash equilibria (in non-deterministic games)

Definition

A strategy profile $(\sigma_j)_{A_j \in \mathsf{Agt}}$ is a Nash equilibrium if no player can improve her payoff by unilaterally changing her strategy.

What if there are several outcomes due to non-determinism?

Definition

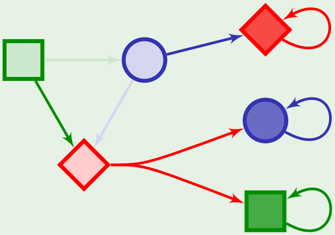
A pair $((\sigma_j)_{A_j \in \mathsf{Agt}}, \pi)$ is a pseudo-Nash equilibrium if π is an outcome of $(\sigma_j)_{A_j \in \mathsf{Agt}}$ and no player can improve her payoff by unilaterally changing her strategy (*i.e.*, no outcome improves the payoff of π).

Nash equilibria (in non-deterministic games)

Definition

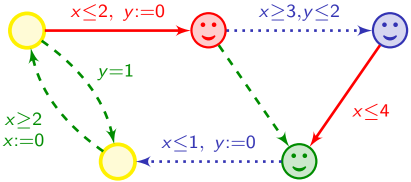
A pair $((\sigma_j)_{A_j \in \mathsf{Agt}}, \pi)$ is a pseudo-Nash equilibrium if π is an outcome of $(\sigma_j)_{A_j \in \mathsf{Agt}}$ and no player can improve her payoff by unilaterally changing her strategy (*i.e.*, no outcome improves the payoff of π).

Example

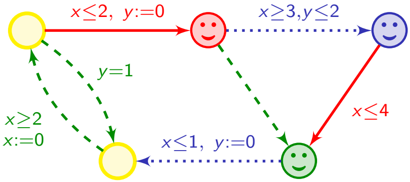


is not a pseudo-Nash equilibrium.
There is no pseudo-Nash equilibirum in this game.

Back to the timed game

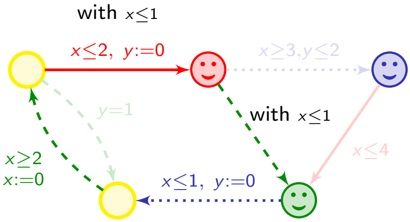


Back to the timed game



There are several (pseudo-)Nash equilibria:

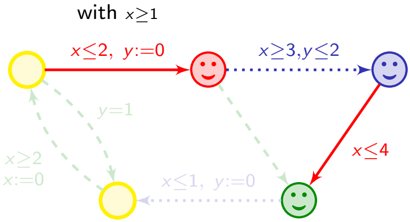
Back to the timed game



There are several (pseudo-)Nash equilibria:

- Some with payoff $(1, 1, 0)$

Back to the timed game



There are several (pseudo-)Nash equilibria:

- Some with payoff $(1, 1, 0)$
- Some with payoff $(1, 1, 1)$

Two different approaches to solve the problem

- use the specificity of two-player timed games [BBM10a]
  - can be turned into two twin turn-based finite games
  - apply to ω-regular objectives (and even more!)

[BBM10a] Bouyer, Brenguier, Markey. Computing equilibria in two-player timed games via turn-based finite games *(FORMATS'10)*.
[BBM10b] Bouyer, Brenguier, Markey. Nash equilibria for reachability objectives in multi-player timed games *(CONCUR'10)*.

Two different approaches to solve the problem

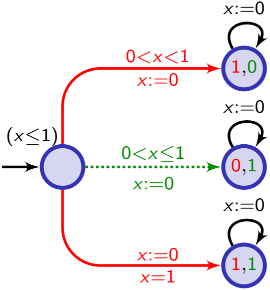
- use the specificity of two-player timed games [BBM10a]
  - can be turned into two twin turn-based finite games
  - apply to ω-regular objectives (and even more!)
- consider a general abstract framework [BBM10b]
  - non-deterministic concurrent multiplayer (infinite) games
  - restrict to qualitative reachability objectives
  - apply the general framework to finite games and then to timed games (through a simulation theorem), and get “optimal” algorithms

[BBM10a] Bouyer, Brenguier, Markey. Computing equilibria in two-player timed games via turn-based finite games *(FORMATS'10)*.
[BBM10b] Bouyer, Brenguier, Markey. Nash equilibria for reachability objectives in multi-player timed games *(CONCUR'10)*.

Outline

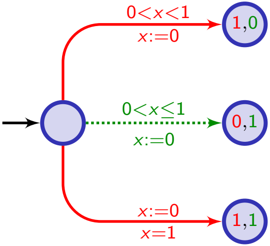
From timed games to finite concurrent games...

Compute the Nash equilibria in the following timed game:



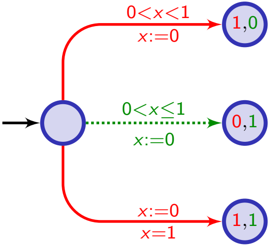
From timed games to finite concurrent games...

Compute the Nash equilibria in the following timed game:

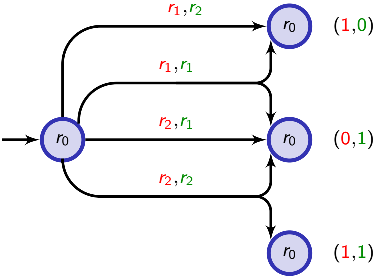


From timed games to finite concurrent games...

Compute the Nash equilibria in the following timed game:

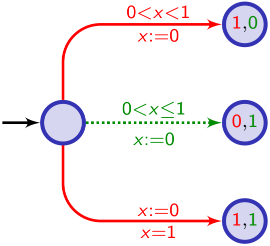


r_0	region $x = 0$
r_1	region $0 < x < 1$
r_2	region $x = 1$

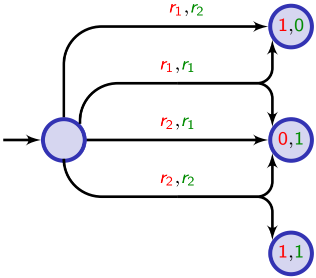


From timed games to finite concurrent games...

Compute the Nash equilibria in the following timed game:

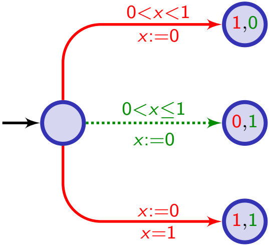


r_0	region $x = 0$
r_1	region $0 < x < 1$
r_2	region $x = 1$

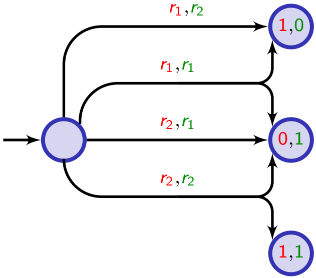


From timed games to finite concurrent games...

Compute the Nash equilibria in the following timed game:



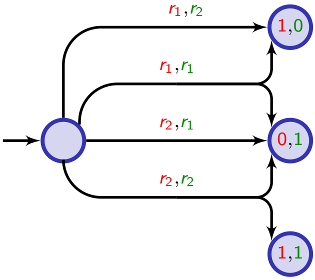
r_0	region $x = 0$
r_1	region $0 < x < 1$
r_2	region $x = 1$



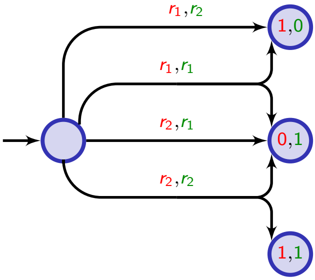
Proposition

(Pseudo-)Nash equilibria "coincide" in both games.

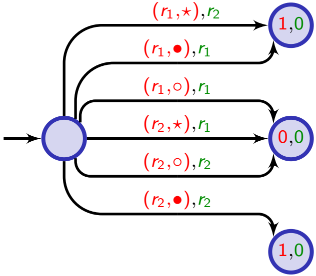
Removing non-determinism



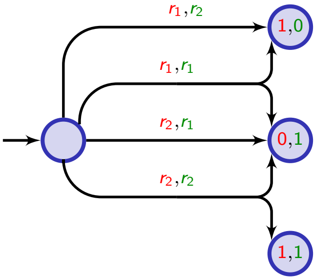
Removing non-determinism



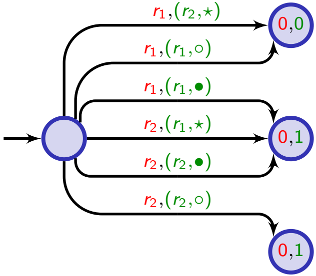
Game where Player 1 has an advantage
(and all payoffs for Player 2 are set to 0)



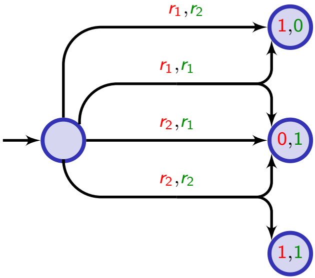
Removing non-determinism

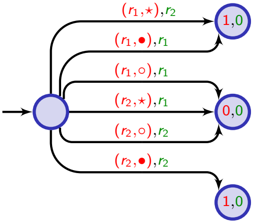


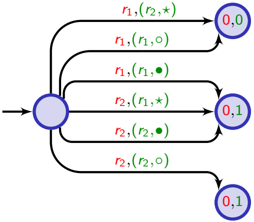
Game where Player 2 has an advantage
(and all payoffs for Player 1 are set to 0)



Removing non-determinism





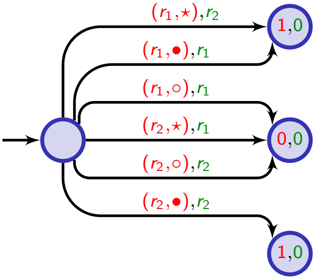


Proposition

(Pseudo-)Nash equilibria in the game on the left correspond to twin Nash equilibria in the two games on the right.

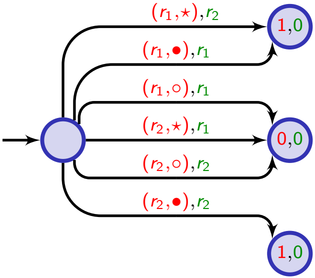
Building turn-based games

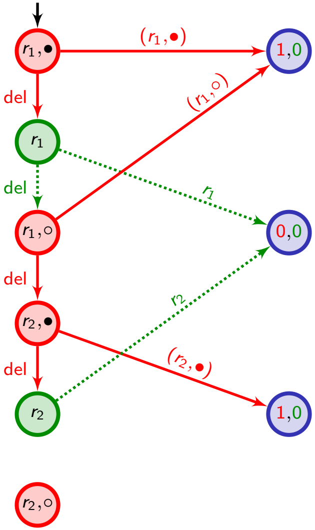
Games where Player 1 has an advantage.



Building turn-based games

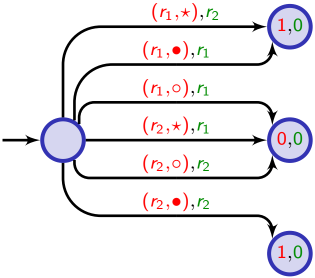
Games where Player 1 has an advantage.

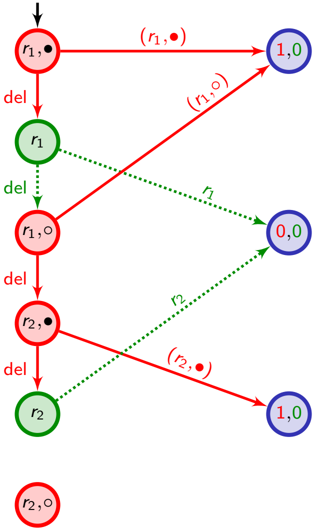




Building turn-based games

Games where Player 1 has an advantage.



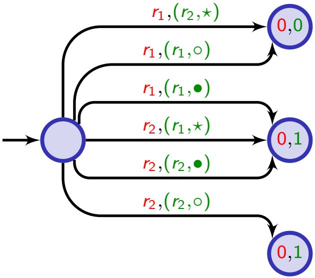


Proposition

Nash equilibria “coincide”.

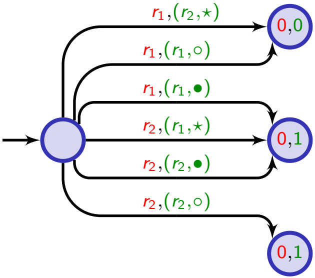
Building turn-based games

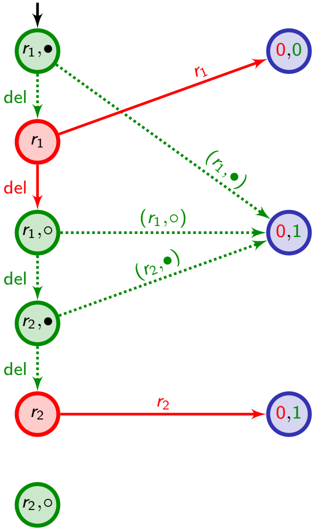
Games where Player 2 has an advantage.



Building turn-based games

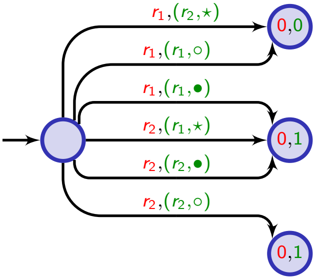
Games where Player 2 has an advantage.

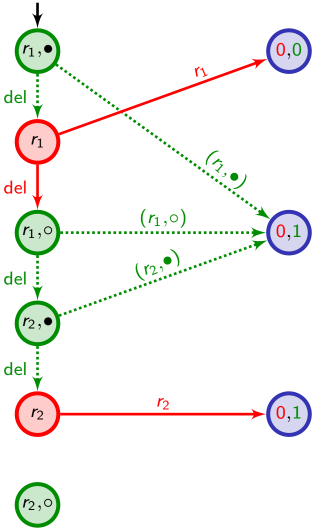




Building turn-based games

Games where Player 2 has an advantage.

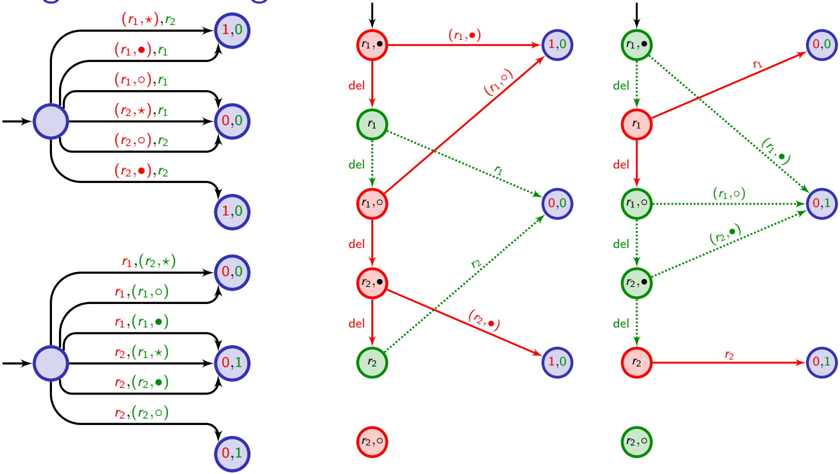




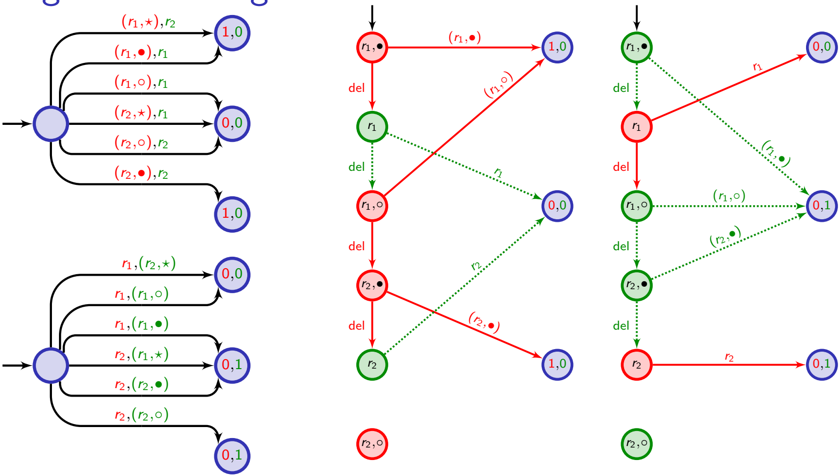
Proposition

Nash equilibria "coincide".

Building turn-based games



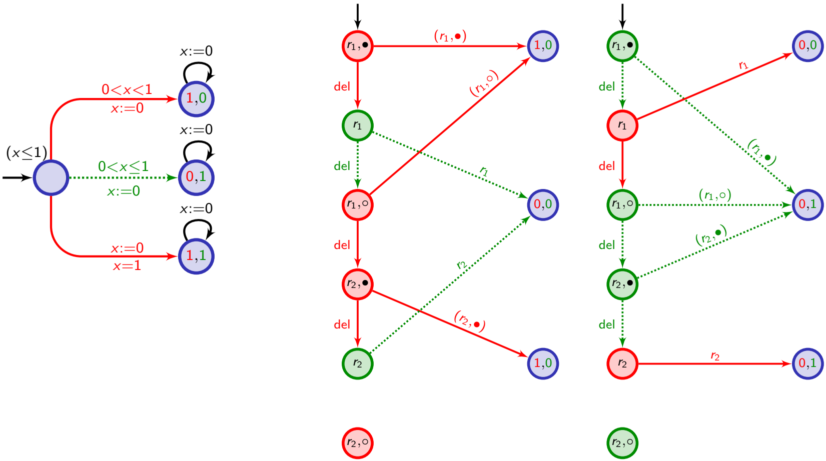
Building turn-based games



Proposition

The twin Nash equilibria in the two games on the left coincide with the twin Nash equilibria in the two turn-based games on the right.

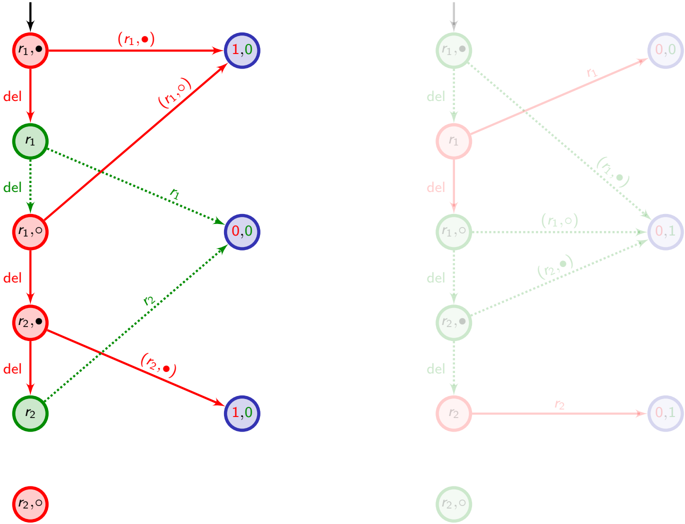
Summary of the construction



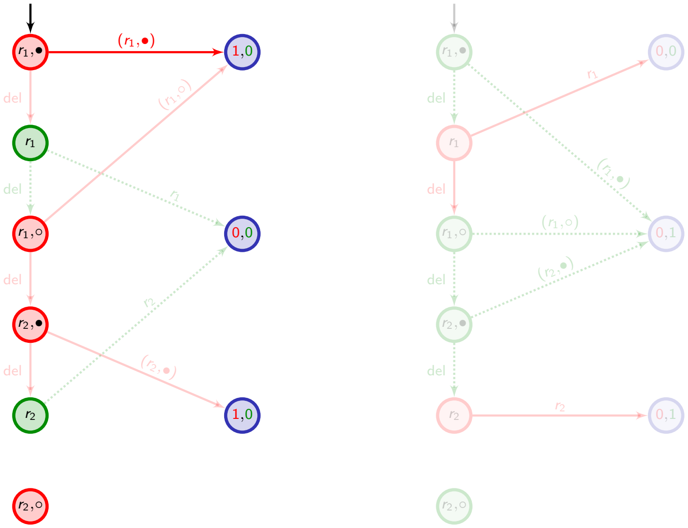
Theorem

(Pseudo-)Nash equilibria in the original timed game coincide with the twin Nash equilibria in the two turn-based games.

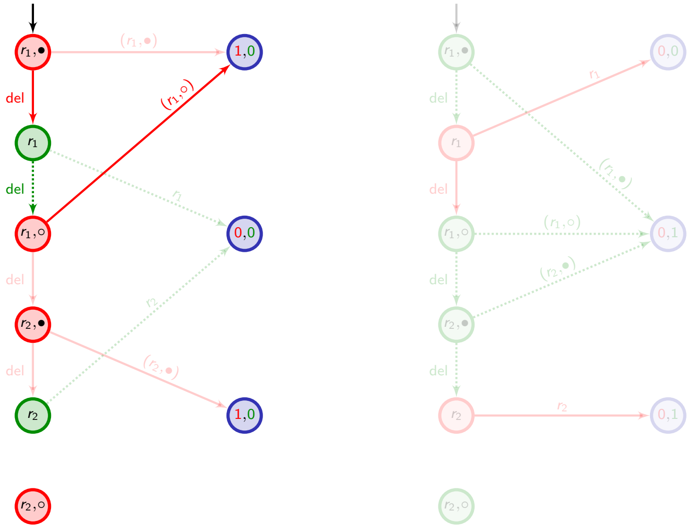
Computing (twin) Nash equilibria in the turn-based games



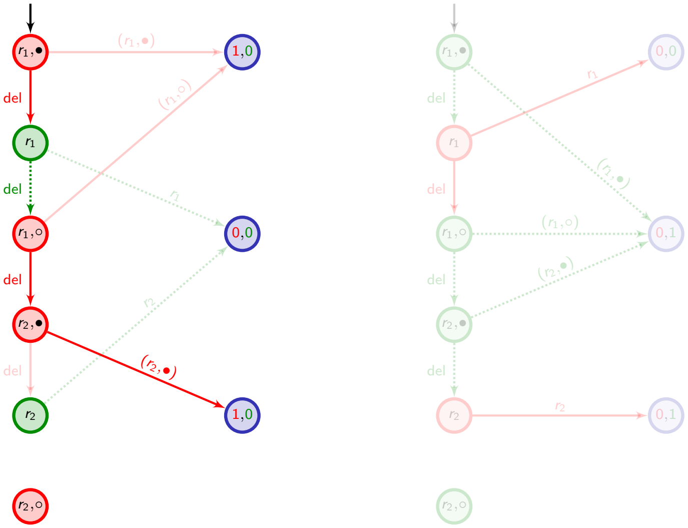
Computing (twin) Nash equilibria in the turn-based games



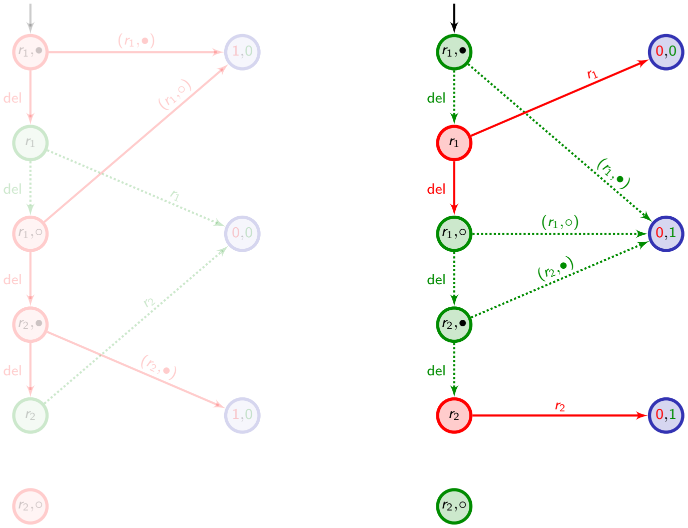
Computing (twin) Nash equilibria in the turn-based games



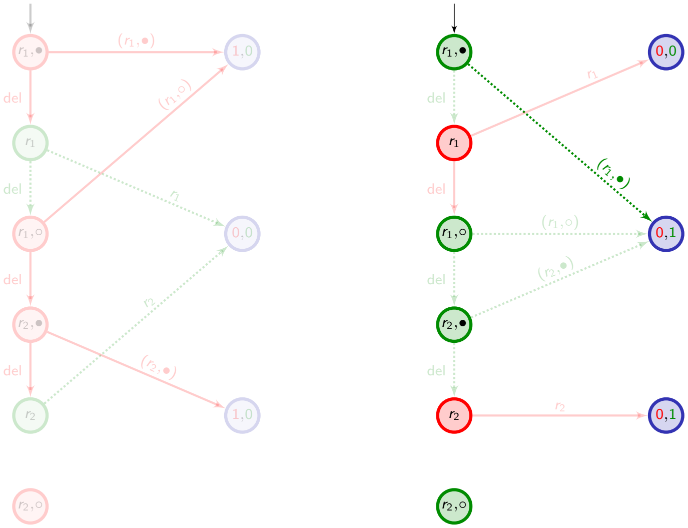
Computing (twin) Nash equilibria in the turn-based games



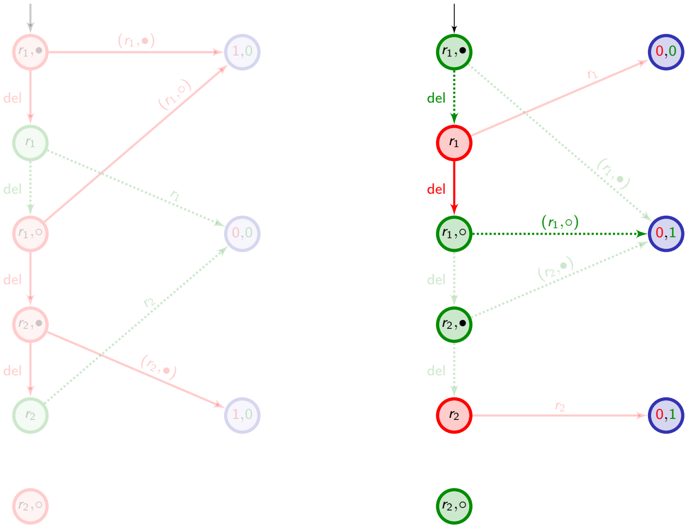
Computing (twin) Nash equilibria in the turn-based games



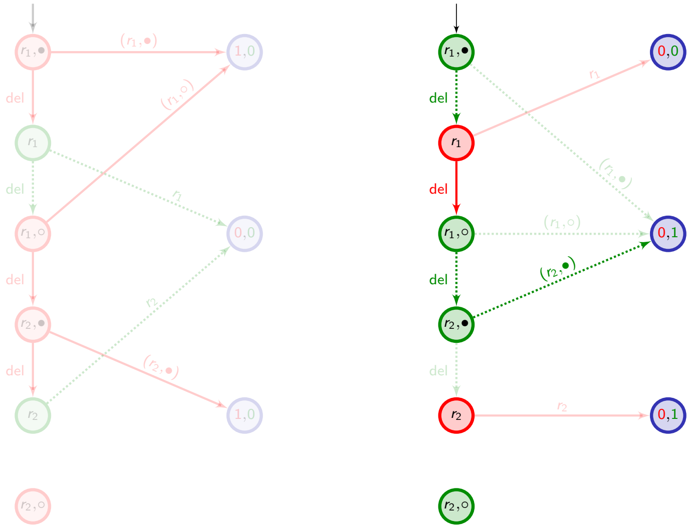
Computing (twin) Nash equilibria in the turn-based games



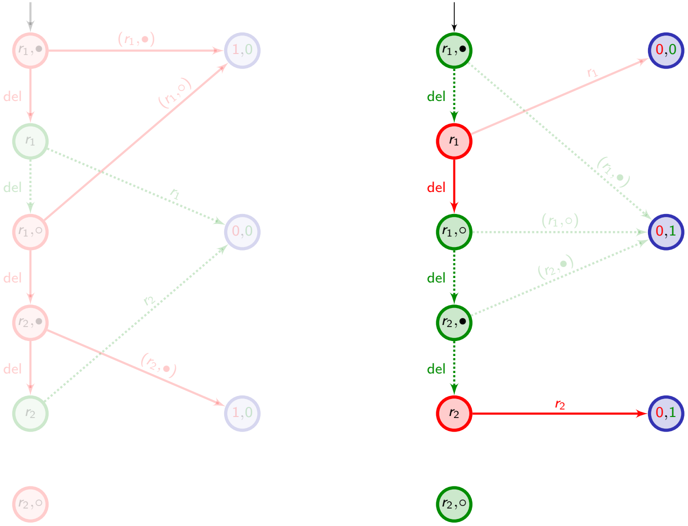
Computing (twin) Nash equilibria in the turn-based games



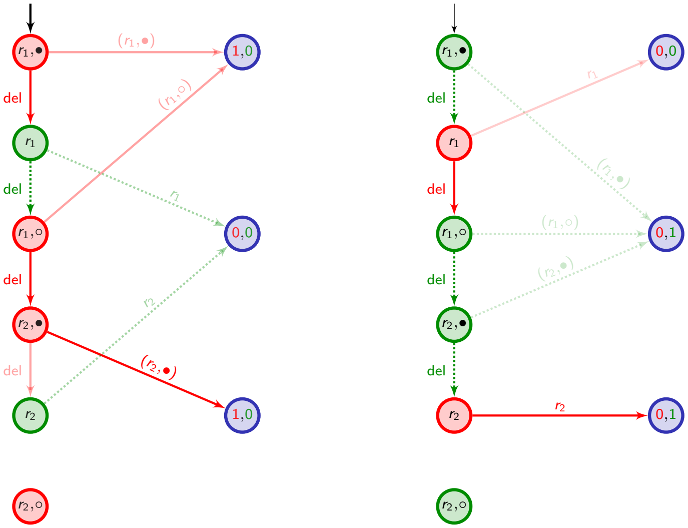
Computing (twin) Nash equilibria in the turn-based games



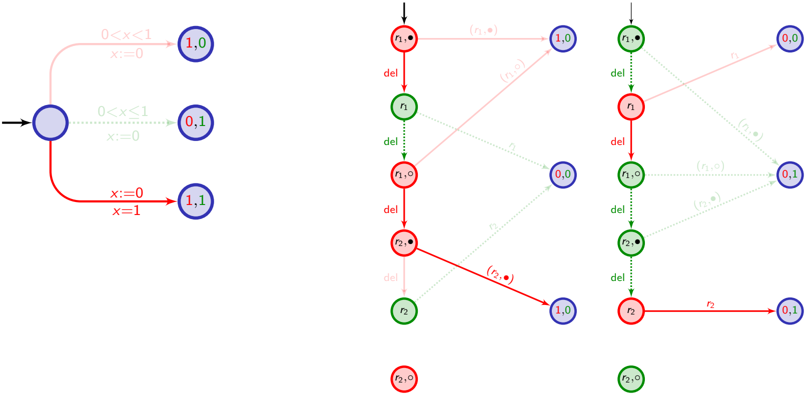
Computing (twin) Nash equilibria in the turn-based games



Computing (twin) Nash equilibria in the turn-based games



Computing (twin) Nash equilibria in the turn-based games



There is a single Nash equilibrium in the original timed game.

Summary of the construction

Theorem

Using this construction, we can compute (pseudo-)Nash equilibria:

- in two-player timed games
- with large classes of objectives

Summary of the construction

Theorem

Using this construction, we can compute (pseudo-)Nash equilibria:

- in two-player timed games
- with large classes of objectives

Examples of objectives

- ω-regular objectives
- optimal paths
- ...

Summary of the construction

Theorem

Using this construction, we can compute (pseudo-)Nash equilibria:

- in two-player timed games
- with large classes of objectives

An interesting side-result

In two-player control timed games, the controller (Player 1) has a winning strategy iff she has a winning strategy in the turn-based game where Player 2 has the advantage.

Summary of the construction

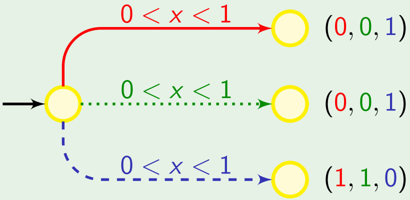
Theorem

Using this construction, we can compute (pseudo-)Nash equilibria:

- in two-player timed games
- with large classes of objectives

Example

This technique does not extend to three-player timed games:



Outline

1. The context
2. Multiplayer timed games
3. Focusing on two-player timed games
4. A generic approach for reachability objectives
5. Conclusion

Context of these developments

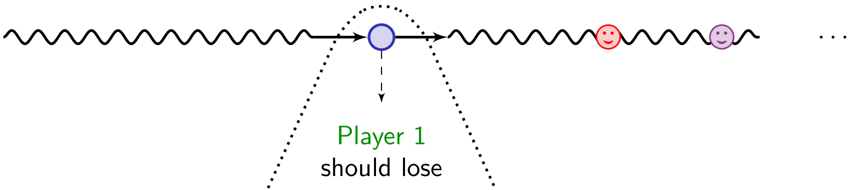
- general (possibly infinite-state) concurrent non-deterministic multiplayer games
- qualitative reachability objectives

At which condition a play *is* an equilibrium where only Player 1 loses?

At which condition a play *is* an equilibrium where only Player 1 loses?

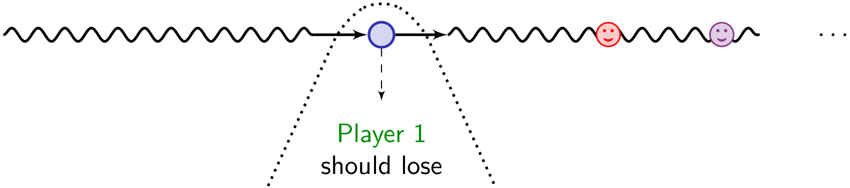
- the path visits all winning states for all other players
- the path doesn't visit the winning states of Player 1

At which condition a play *is* an equilibrium where only Player 1 loses?



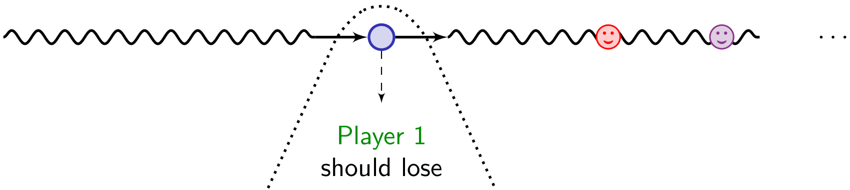
- from any state along the path, Player 1 cannot enforce visiting his winning state

At which condition a play *is* an equilibrium where only Player 1 loses?



- from any state along the path, Player 1 cannot enforce visiting his winning state
    - all states along the path are out of the attractor of Player 1

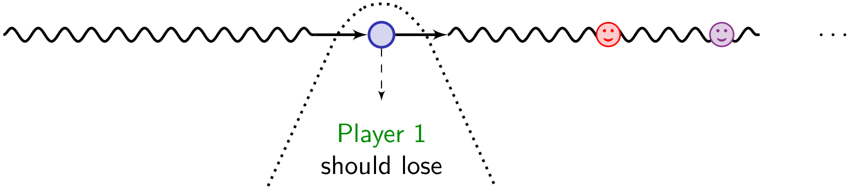
At which condition a play *is* an equilibrium where only Player 1 loses?



- from any state along the path, Player 1 cannot enforce visiting his winning state
  - all states along the path are out of the attractor of Player 1

$\rightsquigarrow$ this is ok for deterministic turn-based games

At which condition a play *is* an equilibrium where only Player 1 loses?

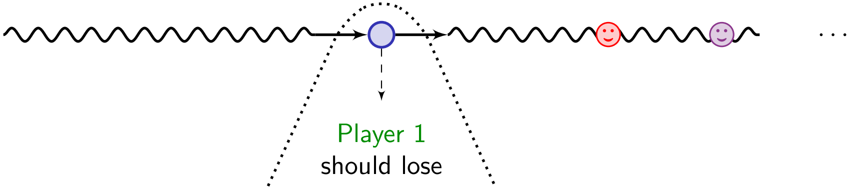


- from any state along the path, Player 1 cannot enforce visiting his winning state
  - all states along the path are out of the attractor of Player 1

$\rightsquigarrow$ this is ok for deterministic turn-based games

$\rightsquigarrow$ this is not ok for the general case!

At which condition a play *is* an equilibrium where only Player 1 loses?

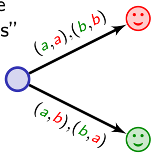


- from any state along the path, Player 1 cannot enforce visiting his winning state
  - all states along the path are out of the attractor of Player 1

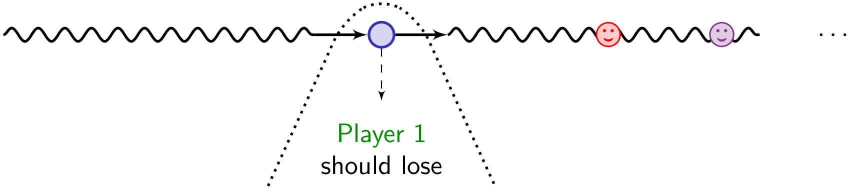
    $\rightsquigarrow$ this is ok for deterministic turn-based games

    $\rightsquigarrow$ this is not ok for the general case!
  - Two counter-examples:

concurrent game
"matching pennies"



At which condition a play *is* an equilibrium where only Player 1 loses?

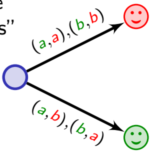


- from any state along the path, Player 1 cannot enforce visiting his winning state
  - all states along the path are out of the attractor of Player 1

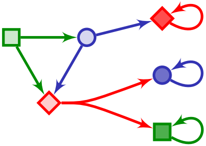
    $\rightsquigarrow$ this is ok for deterministic turn-based games

    $\rightsquigarrow$ this is not ok for the general case!
  - Two counter-examples:

concurrent game
"matching pennies"



turn-based non-det. game

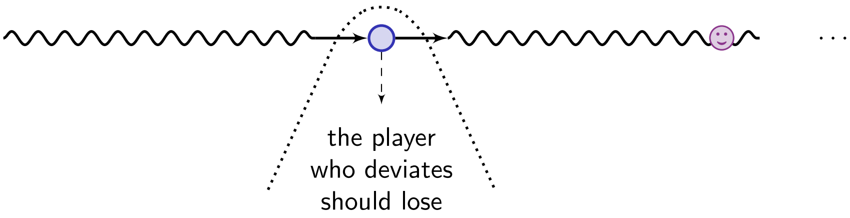


And what if only Player 1 and Player 2 lose?

And what if only Player 1 and Player 2 lose?

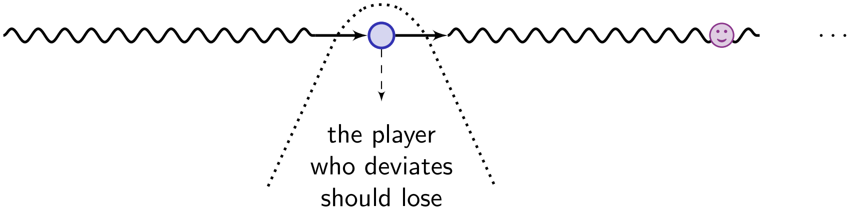
- the path visits all winning states for all other players
- the path doesn't visit the winning states of Player 1 and of Player 2

And what if only Player 1 and Player 2 lose?



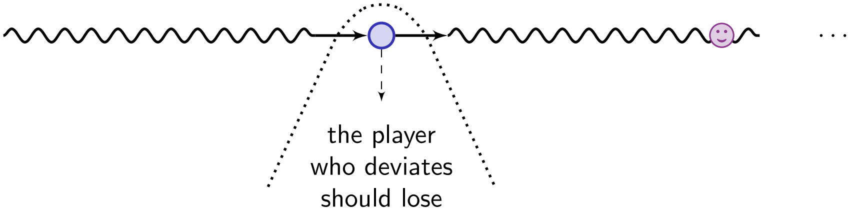
- from any state along the path, if Player 1 deviates, he should not be able to enforce visiting his winning states. Idem for Player 2.

And what if only Player 1 and Player 2 lose?



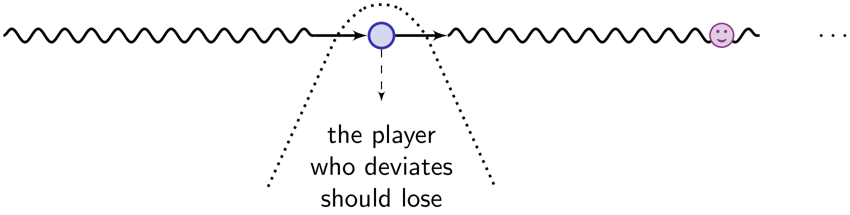
- from any state along the path, if Player 1 deviates, he should not be able to enforce visiting his winning states. Idem for Player 2.
  - $\rightsquigarrow$ requires that we are able to compute who is to be blamed for the deviation

And what if only Player 1 and Player 2 lose?



- from any state along the path, if Player 1 deviates, he should not be able to enforce visiting his winning states. Idem for Player 2.
  - $\rightsquigarrow$ requires that we are able to compute who is to be blamed for the deviation
  - $\rightsquigarrow$ if there is a unique suspect for the deviation, we should blame him (as in the previous case)

And what if only Player 1 and Player 2 lose?



- from any state along the path, if Player 1 deviates, he should not be able to enforce visiting his winning states. Idem for Player 2.
  - $\rightsquigarrow$ requires that we are able to compute who is to be blamed for the deviation
  - $\rightsquigarrow$ if there is a unique suspect for the deviation, we should blame him (as in the previous case)
  - $\rightsquigarrow$ if both players are suspect for the deviation, we should be able to blame each of them.

If we want to formalize...

Definition

If P is a set of players, the *repellor set* for P (written $\mathrm{Rep}(P)$) is defined inductively by $\mathrm{Rep}(\emptyset) =$ "all states", and then as the largest set satisfying:

- $\forall A \in P$, $\mathrm{Rep}(P) \cap ☺_A = \emptyset$
- $\forall s \in \mathrm{Rep}(P)$, $\exists m$ s.t. $\forall s'$, $s' \in \mathrm{Rep}(P \cap \mathrm{Susp}((s, s'), m))$

If we want to formalize...

Definition

If P is a set of players, the *repellor set* for P (written $\mathrm{Rep}(P)$) is defined inductively by $\mathrm{Rep}(\emptyset) =$ "all states", and then as the largest set satisfying:

- $\forall A \in P$, $\mathrm{Rep}(P) \cap ☺_A = \emptyset$
- $\forall s \in \mathrm{Rep}(P)$, $\exists m$ s.t. $\forall s'$, $s' \in \mathrm{Rep}(P \cap \mathrm{Susp}((s, s'), m))$

$\rightsquigarrow$ we define in a natural way the repellor transition system $\mathcal{S}_{\mathrm{Rep}}(P)$

If we want to formalize...

Definition

If P is a set of players, the *repellor set* for P (written $\text{Rep}(P)$) is defined inductively by $\text{Rep}(\emptyset) =$ "all states", and then as the largest set satisfying:

- $\forall A \in P$, $\text{Rep}(P) \cap ☺_A = \emptyset$
- $\forall s \in \text{Rep}(P)$, $\exists m$ s.t. $\forall s'$, $s' \in \text{Rep}(P \cap \text{Susp}((s, s'), m))$

$\rightsquigarrow$ we define in a natural way the repellor transition system $\mathcal{S}_{\text{Rep}}(P)$

Theorem

There is a pseudo-Nash equilibrium with payoff ν_P (*i.e.* 0 for all players in P and 1 for all players out of P) iff there is an infinite path in $\mathcal{S}_{\text{Rep}}(P)$ which visits a winning state $☺_A$ for each $A \notin P$.

NB: this equivalence is furthermore constructive.

A generic procedure to compute (pseudo-)Nash equilibria in concurrent non-deterministic games

1. Guess the payoff function ν_P (if not already given);
2. Build the repellor transition system $\mathcal{S}_{\mathsf{Rep}}(P)$;
3. Find a path in $\mathcal{S}_{\mathsf{Rep}}(P)$ visiting the objectives of $\mathsf{Agt} \setminus P$.

Side results: the case of finite games

	$\mathcal{C}^{\mathsf{nd}}$, $\mathcal{C}^{\mathsf{d}}$, $\mathcal{TB}^{\mathsf{nd}}$		$\mathcal{TB}^{\mathsf{d}}$	
	bounded	general	bounded	general
Existence	P-c.	NP-c.	True	True
Verification	P-c.	NP-c.	P-c.	NP-c.
Constr. Ex.	P-c.	NP-c.	P-c.	NP-c.

- $\mathcal{TB}^{\mathsf{d}}$, $\mathcal{TB}^{\mathsf{nd}}$: turn-based games (deterministic and non-deterministic, resp.)
- $\mathcal{C}^{\mathsf{d}}$,$\mathcal{C}^{\mathsf{nd}}$: concurrent games (determinstic and non-deterministic, resp.)
- bounded: bounded number of players (hence not a parameter in the complexity)
- general: the number of players is a parameter of the problem.

The case of timed games

How do we proceed?

- Construct an appropriate region game
- Define a simulation that preserves all ☺'s and suspect players, hence repellors
- Prove that the two games (original timed game and region game) simulate each other

The case of timed games

How do we proceed?

- Construct an appropriate region game
- Define a simulation that preserves all ☺'s and suspect players, hence repellors
- Prove that the two games (original timed game and region game) simulate each other

Theorem

The verification, existence and constrained-existence problems for pseudo-Nash equilibria in timed games are EXPTIME-complete.

Outline

Conclusion and ongoing/further work

Summary of the results

- A specific technique for computing Nash equilibria for large classes of objectives in two-player timed games

Conclusion and ongoing/further work

Summary of the results

- A specific technique for computing Nash equilibria for large classes of objectives in two-player timed games
- A general technique for computing Nash equilibria for qualitative reachability objectives in multiplayer (potentially infinite) games:
  - can be applied to finite games
  - can be applied to timed games

  In both cases, yields optimal complexity bounds.

Conclusion and ongoing/further work

Summary of the results

- A specific technique for computing Nash equilibria for large classes of objectives in two-player timed games
- A general technique for computing Nash equilibria for qualitative reachability objectives in multiplayer (potentially infinite) games:
  - can be applied to finite games
  - can be applied to timed games

  In both cases, yields optimal complexity bounds.

What's going on now?

- Use the general technique to compute Nash equilibria in other families of systems (eg pushdown games)
- Extend the repellor idea beyond reachability
- Consider randomized strategies
- Compute other kinds of equilibria