Prompt Delay

Joint Work with Felix Klein (Saarland University)

Martin Zimmermann

Saarland University

December 15th, 2016

FSTTCS 2016, Chennai, India

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

I: b
O:

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b
***O*:**	a

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a
***O*:**	a	

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a
***O*:**	a	a

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b
***O*:**	a	a	

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	$\cdots$
***O*:**	a	a	$\cdots$	

I wins!

Motivation

Büchi-Landweber: The winner of a zero-sum two-player game of infinite duration with ω-regular winning condition can be determined effectively.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	$\cdots$
***O*:**	a	a	$\cdots$	

I wins!

- Many possible extensions... we consider two:
  Interaction: one player may delay her moves.
  Winning condition: quantitative instead of qualitative.

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:** b
***O*:**

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:** b a
***O*:**

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:** b a b

***O*:**

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b
***O*:**	b		

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b
***O*:**	b			

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b
***O*:**	b	b		

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a
***O*:**	b	b			

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a
***O*:**	b	b	a		

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a	a
***O*:**	b	b	a			

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a	a
***O*:**	b	b	a	a		

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a	a	b
***O*:**	b	b	a	a			

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a	a	b
***O*:**	b	b	a	a	b		

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	*b*	*a*	*b*	*b*	*a*	*a*	*b*	*b*
***O*:**	*b*	*b*	*a*	*a*	*b*			

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a	a	b	b
***O*:**	b	b	a	a	b	b		

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	b	a	b	b	a	a	b	b	$\cdots$
***O*:**	b	b	a	a	b	b	$\cdots$		

O wins!

Motivation

- Allow Player O to delay her moves.

$$\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\cdots \in L, \text{ if } \beta(i) = \alpha(i+2) \text{ for every } i$$

***I*:**	*b*	*a*	*b*	*b*	*a*	*a*	*b*	*b*	⋯
***O*:**	*b*	*b*	*a*	*a*	*b*	*b*	⋯		

O wins!

- Winning conditions in PROMPT-LTL, LTL with parameterized temporal operators:

$$\mathbf{G}\,(q \rightarrow \mathbf{F_P}\; p)$$

holds if every request q is answered by a response p within some arbitrary, but fixed bound k.

Prompt-LTL

Syntax:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F_P}\,\varphi$$

where p ranges over a finite set AP of atomic propositions.

Prompt-LTL

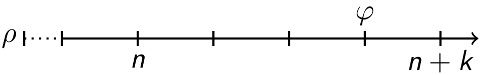
Syntax:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\,\varphi \mid \varphi\,\mathbf{U}\,\varphi \mid \varphi\,\mathbf{R}\,\varphi \mid \mathbf{F_P}\,\varphi$$

where p ranges over a finite set AP of atomic propositions.

Semantics: defined with respect to a fixed bound $k \in \mathbb{N}$

$(\rho, n, k) \models \mathbf{F_P}\,\varphi$:



Prompt-LTL Delay Games

A PROMPT-LTL delay game $\Gamma_f(\varphi)$ consists of

- a winning condition φ over $\mathrm{AP} = I \cup O$, and
- a delay function: $f \colon \mathbb{N} \to \mathbb{N}_+$.

Prompt-LTL Delay Games

A PROMPT-LTL delay game $\Gamma_f(\varphi)$ consists of

- a winning condition φ over $\mathrm{AP} = I \cup O$, and
- a delay function: $f \colon \mathbb{N} \to \mathbb{N}_+$.
- f is constant, if $f(i) = 1$ for all $i > 0$.

Prompt-LTL Delay Games

A PROMPT-LTL delay game $\Gamma_f(\varphi)$ consists of

- a winning condition φ over $\mathrm{AP} = I \cup O$, and
- a delay function: $f \colon \mathbb{N} \to \mathbb{N}_+$.
- f is constant, if $f(i) = 1$ for all $i > 0$.

Rules:

- Two players: Input (Player I) vs. Output (Player O).
- In round i:
  - Player I picks word $u_i \in (2^I)^{f(i)}$ (building $\alpha = u_0 u_1 \cdots$).
  - Player O picks letter $v_i \in 2^O$ (building $\beta = v_0 v_1 \cdots$).

Prompt-LTL Delay Games

A Prompt-LTL delay game $\Gamma_f(\varphi)$ consists of

- a winning condition φ over $\mathrm{AP} = I \cup O$, and
- a delay function: $f \colon \mathbb{N} \to \mathbb{N}_+$.
- f is constant, if $f(i) = 1$ for all $i > 0$.

Rules:

- Two players: Input (Player I) vs. Output (Player O).
- In round i:
  - Player I picks word $u_i \in (2^I)^{f(i)}$ (building $\alpha = u_0 u_1 \cdots$).
  - Player O picks letter $v_i \in 2^O$ (building $\beta = v_0 v_1 \cdots$).
- Player O wins w.r.t. bound k iff
  $((\alpha(0) \cup \beta(0))\,(\alpha(1) \cup \beta(1)) \cdots, k) \models \varphi$.

Prompt-LTL Delay Games

A PROMPT-LTL delay game $\Gamma_f(\varphi)$ consists of

- a winning condition φ over $\mathrm{AP} = I \cup O$, and
- a delay function: $f : \mathbb{N} \to \mathbb{N}_+$.
- f is constant, if $f(i) = 1$ for all $i > 0$.

Rules:

- Two players: Input (Player I) vs. Output (Player O).
- In round i:
  - Player I picks word $u_i \in (2^I)^{f(i)}$ (building $\alpha = u_0 u_1 \cdots$).
  - Player O picks letter $v_i \in 2^O$ (building $\beta = v_0 v_1 \cdots$).
- Player O wins w.r.t. bound k iff $((\alpha(0) \cup \beta(0))(\alpha(1) \cup \beta(1)) \cdots, k) \models \varphi$.

Note:

Definition here is equivalent to O skipping moves.

Prompt-LTL Delay Games

Problems we are interested in:

- Given φ, is there an f such that O wins $\Gamma_f(\varphi)$ w.r.t some k?
- How *large* do f and k have to be?
- How hard is it to determine the winner?

An Example

- $I = \{1, \ldots, n\}$ and $O = \{1_O, \ldots, n_O\}$
- We assume that both players pick exactly one proposition in each round (expressible in LTL)
- $\varphi_n = \bigvee_{j \in [n]} j_O \rightarrow \psi_j$ with $\psi_j = \mathbf{F_P}\, (j \wedge \mathbf{X}\, ((\bigwedge_{j' > j} \neg j')\, \mathbf{U}\, j))$

Example

- $1\,2\,3\,2\,1\,1\,1\,1\,1 \cdots$ satisfies ψ_1, but not ψ_2 and not ψ_3
- In general, every word satisfies some ψ_j
- $1\,2\,1\,3\,1\,2\,1\,3\,3\,3 \cdots$ satisfies ψ_3, but not ψ_1 and not ψ_2

An Example

- $I = \{1, \ldots, n\}$ and $O = \{1_O, \ldots, n_O\}$
- We assume that both players pick exactly one proposition in each round (expressible in LTL)
- $\varphi_n = \bigvee_{j \in [n]} j_O \rightarrow \psi_j$ with $\psi_j = \mathbf{F_P} \left(j \wedge \mathbf{X} \left(\left(\bigwedge_{j' > j} \neg j' \right) \mathbf{U}\, j \right) \right)$

Then:

- Player O wins $\Gamma_f(\varphi_n)$, if $f(0) \geq 2^n$: every word of length 2^n satisfies ψ_j for some j. Player O just picks j_O in round 0.

An Example

- $I = \{1, \ldots, n\}$ and $O = \{1_O, \ldots, n_O\}$
- We assume that both players pick exactly one proposition in each round (expressible in LTL)
- $\varphi_n = \bigvee_{j \in [n]} j_O \rightarrow \psi_j$ with $\psi_j = \mathbf{F_P}\,(j \wedge \mathbf{X}\,((\bigwedge_{j' > j} \neg j')\,\mathbf{U}\,j))$

Then:

- Player O wins $\Gamma_f(\varphi_n)$, if $f(0) \geq 2^n$: every word of length 2^n satisfies ψ_j for some j. Player O just picks j_O in round 0.
- Player I wins $\Gamma_f(\varphi_n)$, if $f(0) < 2^n$: there is a word w_n of length $2^n - 1$ that does not satisfy ψ_j for any j.
  - Player I picks prefix of length $f(0)$ of w_n in round 0, Player O answers by some j_O.
  - Player I picks j' for some $j' \neq j$ in each following round.

Special Cases

Theorem (Pnueli, Rosner '89 / Kupferman et al. 07)

Determining the winner of delay-free Prompt-LTL *games is* 2ExpTime*-complete.*

Special Cases

Theorem (Pnueli, Rosner '89 / Kupferman et al. 07)

Determining the winner of delay-free PROMPT-LTL *games is* 2EXPTIME*-complete.*

Theorem (Klein, Z. '15)

The following problem is EXPTIME*-complete: given a deterministic parity automaton $\mathcal{A}$, does Player O win $\Gamma_f(L(\mathcal{A}))$ for some delay function f? If yes, a constant f with $f(0) \leq 2^{\mathcal{O}(|\mathcal{A}|)}$ suffices.*

Special Cases

Theorem (Pnueli, Rosner '89 / Kupferman et al. 07)

Determining the winner of delay-free Prompt-LTL *games is* 2ExpTime*-complete.*

Theorem (Klein, Z. '15)

The following problem is ExpTime*-complete: given a deterministic parity automaton $\mathcal{A}$, does Player O win $\Gamma_f(L(\mathcal{A}))$ for some delay function f? If yes, a constant f with $f(0) \leq 2^{\mathcal{O}(|\mathcal{A}|)}$ suffices.*

Corollary

The following problem is in 3ExpTime*: given an* LTL *formula φ, does Player O win $\Gamma_f(\varphi)$ for some delay function f? If yes, a constant f with $f(0) \leq 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ suffices.*

Roadmap

Condition	complexity	lookahead	bound k
LTL	in 3EXPTIME	$\leq$ triply-exp.	NA
PROMPT-LTL	?	?	?

Solving Prompt-LTL Delay Games

Theorem

The following problem is in $3\textsc{ExpTime}$*: given a* $\textsc{Prompt}$-LTL *formula* φ*, does Player O win* $\Gamma_f(\varphi)$ *for some delay function* f*? If yes, a constant* f *with* $f(0) \in 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ *and some bound* $k \in 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ *suffice simultaneously.*

Solving Prompt-LTL Delay Games

Theorem

The following problem is in $\textsc{3ExpTime}$*: given a* $\textsc{Prompt-LTL}$ *formula* φ*, does Player* O *win* $\Gamma_f(\varphi)$ *for some delay function* f*? If yes, a constant* f *with* $f(0) \in 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ *and some bound* $k \in 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ *suffice simultaneously.*

Proof Idea: by a reduction to LTL delay games.

- Add fresh proposition p to $O \subseteq \mathrm{AP}$ and inductively replace every subformula $\mathbf{F_P}\,\psi$ by

$$(p \rightarrow p\,\mathbf{U}\,(\neg p\,\mathbf{U}\,\psi)) \wedge (\neg p \rightarrow \neg p\,\mathbf{U}\,(p\,\mathbf{U}\,\psi)).$$

- **Lemma** Player O wins $\Gamma_f(\varphi)$ for some f $\Leftrightarrow$ Player O wins $\Gamma_f(\mathrm{rel}(\varphi) \wedge \mathbf{G}\,\mathbf{F}\,p \wedge \mathbf{G}\,\mathbf{F}\,\neg p)$ for some f.

Roadmap

Condition	complexity	lookahead	bound k
LTL	in 3ExpTime	$\leq$ triply-exp.	NA
Prompt-LTL	in 3ExpTime	$\leq$ triply-exp.	$\leq$ triply-exp.

Lower Bounds: Lookahead

Theorem

For every $n > 0$, there is an LTL formula φ_n of size $\mathcal{O}(n^2)$ s.t.

- *Player O wins $\Gamma_f(\varphi_n)$ for some delay function f, but*
- *Player I wins $\Gamma_f(\varphi_n)$ for every delay function f with $f(0) \leq 2^{2^{2^n}}$.*

Lower Bounds: Lookahead

Theorem

For every $n > 0$, there is an LTL *formula φ_n of size $\mathcal{O}(n^2)$ s.t.*

- *Player O wins $\Gamma_f(\varphi_n)$ for some delay function f, but*
- *Player I wins $\Gamma_f(\varphi_n)$ for every delay function f with $f(0) \leq 2^{2^{2^n}}$.*

Proof Idea: blow up the introductory example

Recall:

- Both players pick a sequence of numbers from $\{1, \ldots, n\}$.
- Player O has to pick j in first move such that Player I's sequence contains two j's without larger number in between.
- Player O has winning strategy, but only with lookahead 2^n.

Lower Bounds: Lookahead

Theorem

For every $n > 0$, there is an LTL *formula φ_n of size $\mathcal{O}(n^2)$ s.t.*

- *Player O wins $\Gamma_f(\varphi_n)$ for some delay function f, but*
- *Player I wins $\Gamma_f(\varphi_n)$ for every delay function f with $f(0) \leq 2^{2^{2^n}}$.*

Proof Idea: blow up the introductory example

Recall:

- Both players pick a sequence of numbers from $\{1, \ldots, n\}$.
- Player O has to pick j in first move such that Player I's sequence contains two j's without larger number in between.
- Player O has winning strategy, but only with lookahead 2^n.

$\Rightarrow$ Construct φ_n to encode game with range $\{1, \ldots, 2^{2^{|\varphi_n|}}\}$.

Lower Bounds: Lookahead

- $I = \{b_0, \ldots, b_{n-1}, b_I, \#\}$ and $O = \{b_O, \rightarrow, \leftarrow\}$
- Require the b_j implement cyclic addressing of positions with domain $\{0, \ldots, 2^n - 1\}$
- Interpret truth values of b_I and b_O in one cycle of the addressing as sequence of numbers from $\{0, \ldots, 2^{2^n} - 1\}$
- Player O marks two numbers by $\rightarrow, \leftarrow$

Lower Bounds: Lookahead

- $I = \{b_0, \ldots, b_{n-1}, b_I, \#\}$ and $O = \{b_O, \rightarrow, \leftarrow\}$
- Require the b_j implement cyclic addressing of positions with domain $\{0, \ldots, 2^n - 1\}$
- Interpret truth values of b_I and b_O in one cycle of the addressing as sequence of numbers from $\{0, \ldots, 2^{2^n} - 1\}$
- Player O marks two numbers by $\rightarrow, \leftarrow$
- Require Player O to always pick the same number (*) $\Rightarrow$ checking correctness of her marks straightforward

Lower Bounds: Lookahead

- $I = \{b_0, \ldots, b_{n-1}, b_I, \#\}$ and $O = \{b_O, \rightarrow, \leftarrow\}$
- Require the b_j implement cyclic addressing of positions with domain $\{0, \ldots, 2^n - 1\}$
- Interpret truth values of b_I and b_O in one cycle of the addressing as sequence of numbers from $\{0, \ldots, 2^{2^n} - 1\}$
- Player O marks two numbers by $\rightarrow, \leftarrow$
- Require Player O to always pick the same number (*) $\Rightarrow$ checking correctness of her marks straightforward
- But: cannot check (*) with *small* formula, we need the help of Player I
- Copy-error manifests itself at one address. Player I uses # to specify such an address to force Player O to copy honestly

Roadmap

Condition	complexity	lookahead	bound k
LTL	in 3ExpTime	triply-exp.	NA
Prompt-LTL	in 3ExpTime	triply-exp.	$\leq$ triply-exp.

Lower Bounds: Bound k

Theorem

For every $n > 0$, there is a Prompt-LTL *formula* φ'_n *of size* $\mathcal{O}(n^2)$ *s.t.*

- *Player O wins* $\Gamma_f(\varphi'_n)$ *for some delay function* f *and some* k*, but*
- *Player I wins* $\Gamma_f(\varphi'_n)$ *for every delay function* f *and every* $k \leq 2^{2^{2^n}}$*.*

Lower Bounds: Bound k

Theorem

For every $n > 0$, there is a $\textsc{Prompt-LTL}$ *formula* φ'_n *of size* $\mathcal{O}(n^2)$ *s.t.*

- *Player O wins* $\Gamma_f(\varphi'_n)$ *for some delay function f and some k, but*
- *Player I wins* $\Gamma_f(\varphi'_n)$ *for every delay function f and every* $k \leq 2^{2^{2^n}}$.

Proof Idea: adapt formula for lookahead from last slide

- Require Player O to play second mark ⬅ promptly

Roadmap

Condition	complexity	lookahead	bound k
LTL	in 3EXPTIME	triply-exp.	NA
PROMPT-LTL	in 3EXPTIME	triply-exp.	triply-exp.

Lower Bounds: Complexity

Theorem

The following problem is $\textsc{3ExpTime}$*-complete: given an* LTL *formula* φ*, does Player* O *win* $\Gamma_f(\varphi)$ *for some delay function* f*?*

Lower Bounds: Complexity

Theorem

The following problem is 3ExpTime*-complete: given an* LTL *formula* φ*, does Player O win* $\Gamma_f(\varphi)$ *for some delay function* f*?*

Proof Idea: encode alternating doubly-exponential space TM

- Use previous tricks and then some more...

Roadmap

Condition	complexity	lookahead	bound k
LTL	3EXPTIME-compl.	triply-exp.	NA
PROMPT-LTL	3EXPTIME-compl.	triply-exp.	triply-exp.

Non-determinism and Alternation

- The lower bounds for LTL can be adapted to solve several open problems for ω-regular delay games on non-deterministic, universal, and alternating automata
- The results obtained by determinization are optimal:

Automaton type	complexity	lookahead
deterministic parity	ExpTime-compl.	exponential

Non-determinism and Alternation

- The lower bounds for LTL can be adapted to solve several open problems for ω-regular delay games on non-deterministic, universal, and alternating automata
- The results obtained by determinization are optimal:

Automaton type	complexity	lookahead
deterministic parity	ExpTime-compl.	exponential
non-deterministic parity	2ExpTime-compl.	doubly-exp.
universal parity	2ExpTime-compl.	doubly-exp.

Non-determinism and Alternation

- The lower bounds for LTL can be adapted to solve several open problems for ω-regular delay games on non-deterministic, universal, and alternating automata
- The results obtained by determinization are optimal:

Automaton type	complexity	lookahead
deterministic parity	EXPTIME-compl.	exponential
non-deterministic parity	2EXPTIME-compl.	doubly-exp.
universal parity	2EXPTIME-compl.	doubly-exp.
alternating parity	3EXPTIME-compl.	triply-exp.

Conclusion

Results

- Determining the winner of PROMPT-LTL delay games is 3EXPTIME-complete
- Triply-exponential lookahead and a triply-exponential bound for the prompt-eventually are necessary and sufficient
- All results hold for stronger parametric logics as well (e.g., PLTL and PLDL)
- doubly-exponential complexity for non-deterministic and universal parity automata, triply-exponential for alternating parity automata

Conclusion

Results

- Determining the winner of Prompt-LTL delay games is 3ExpTime-complete
- Triply-exponential lookahead and a triply-exponential bound for the prompt-eventually are necessary and sufficient
- All results hold for stronger parametric logics as well (e.g., PLTL and PLDL)
- doubly-exponential complexity for non-deterministic and universal parity automata, triply-exponential for alternating parity automata

Open problem

- What about more succinct acceptance conditions than parity?