Prompt Delay*

Felix Klein[1] and Martin Zimmermann[2]

1 **Reactive Systems Group, Saarland University, Saarbrücken, Germany**
`klein@react.uni-saarland.de`

2 **Reactive Systems Group, Saarland University, Saarbrücken, Germany**
`zimmermann@react.uni-saarland.de`

Abstract

Delay games are two-player games of infinite duration in which one player may delay her moves to obtain a lookahead on her opponent's moves. Recently, such games with quantitative winning conditions in weak MSO with the unbounding quantifier were studied, but their properties turned out to be unsatisfactory. In particular, unbounded lookahead is in general necessary.

Here, we study delay games with winning conditions given by Prompt-LTL, Linear Temporal Logic equipped with a parameterized eventually operator whose scope is bounded. Our main result shows that solving Prompt-LTL delay games is complete for triply-exponential time. Furthermore, we give tight triply-exponential bounds on the necessary lookahead and on the scope of the parameterized eventually operator. Thus, we identify Prompt-LTL as the first known class of well-behaved quantitative winning conditions for delay games.

Finally, we show that applying our techniques to delay games with ω-regular winning conditions answers open questions in the cases where the winning conditions are given by non-deterministic, universal, or alternating automata.

1998 ACM Subject Classification D.2.4 Software/Program Verification

Keywords and phrases Infinite Games, Delay Games, Prompt-LTL, LTL

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2016.43

1 Introduction

The synthesis of reactive systems concerns the automatic construction of an implementation satisfying a given specification against every behavior of its possibly antagonistic environment. A prominent specification language is Linear Temporal Logic (LTL), describing the temporal behavior of an implementation [21]. The LTL synthesis problem has been intensively studied since the seminal work of Pnueli and Rosner [22, 23], theoretical foundations have been established [2, 19], and several tools have been developed [4, 7, 8].

However, LTL is not able to express quantitative properties. As an example, consider the classical request-response condition [13], where every request q has to be answered eventually by some response r. This property is expressible in LTL via the formula $\mathbf{G}\,(q \rightarrow \mathbf{F}\,r)$, but the property cannot guarantee any bound on the waiting times between a request and its earliest response. To specify such a behaviour, parameterized logics have been introduced [1, 6, 18, 26], which extend LTL by quantitative operators.

The simplest of these logics is Prompt-LTL [18], which extends LTL by the prompt eventually operator $\mathbf{F_P}$. The scope of this operator is bounded by some arbitrary but

* The first author was supported by an IMPRS-CS PhD Scholarship, the second by the project "TriCS" (ZI 1516/1-1) of the German Research Foundation (DFG).

36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016).
Editors: Akash Lal, S. Akshay, Saket Saurabh, and Sandeep Sen; Article No. 43; pp. 43:1–43:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

fixed number k. With this extension, we can express the aforementioned property by the PROMPT-LTL formula $\mathbf{G}\,(q \rightarrow \mathbf{F_P}\, r)$, expressing that every request is answered within k steps. To show that the PROMPT-LTL synthesis problem is as hard as the LTL synthesis problem, i.e., 2EXPTIME-complete, Kupferman et al. introduced the alternating-color technique to reduce the former problem to the latter [18]. Additionally, similar reductions have been proven to exist in other settings too, where PROMPT-LTL can be reduced to LTL using the alternating-color technique, e.g., for (assume-guarantee) model-checking [18]. Finally, the technique is also applicable to more expressive extensions of LTL, e.g., parametric LTL [25], parametric LDL [6], and their variants with costs [26].

Nevertheless, all these considerations assume that the specified implementation immediately reacts to inputs of the environment. However, this assumption might be too restrictive, e.g., in a buffered network, where the implementation may delay its outputs by several time steps. Delay games have been introduced by Hosch and Landweber [14] to overcome this restriction. In the setting of infinite games, the synthesis problem is viewed as a game between two players, the input player "Player I", representing the environment, and the output player "Player O", representing the implementation. The goal of Player O is to satisfy the specification, while Player I tries to violate it. Usually, the players move in strict alternation. On the contrary, in a delay game, Player O can delay her moves to obtain a lookahead on her opponent's moves. This way, she gains additional information on her opponent's strategy, which she can use to achieve her goal. Hence, many specifications are realizable, when allowing lookahead, which are unrealizable otherwise.

For delay games with ω-regular winning conditions (given by deterministic parity automata) exponential lookahead is always sufficient and in general necessary, and determining the winner is EXPTIME-complete [16]. As LTL formulas can be translated into equivalent deterministic parity automata of doubly-exponential size, these results imply a triply-exponential upper bound on the necessary lookahead in delay games with LTL winning condition and yield an algorithm solving such games with triply-exponential running time. However, no matching lower bounds are known.

Recently, based on the techniques developed for the ω-regular case, the investigation of delay games with quantitative winning conditions was initiated by studying games with winning conditions specified in weak monadic second order logic with the unbounding quantifier (WMSO+U) [3]. This logic extends the weak variant of monadic second order logic (WMSO), where only quantification over finite sets is allowed, with an additional unbounding quantifier that allows to express (un)boundedness properties. The resulting logic subsumes all parameterized logics mentioned above. The winner of a WMSO+U delay game with respect to bounded lookahead is effectively computable [27]. However, in general, Player O needs unbounded lookahead to win such games and the decidability of such games with respect to arbitrary lookahead remains an open problem. In the former aspect, delay games with WMSO+U winning conditions behave worse than those with ω-regular ones.

Our Contribution. The results on WMSO+U delay games show the relevance of exploring more restricted classes of quantitative winning conditions which are better-behaved. In particular, bounded lookahead should always suffice and the winner should be effectively computable. To this end, we investigate delay games with PROMPT-LTL winning conditions. Formally, we consider the following synthesis problem: given some PROMPT-LTL formula φ, does there exist some lookahead and some bound k such that Player O has a strategy producing only outcomes that satisfy φ with respect to the bound k (and, if yes, compute such a strategy)?

We present the first results for delay games with Prompt-LTL winning conditions. First, we show that the synthesis problem is in 3ExpTime by tailoring the alternating-color technique to delay games and integrating it into the algorithm developed for the ω-regular case. In the end, we obtain a reduction from delay games with Prompt-LTL winning conditions to delay-free parity games of triply-exponential size.

Second, from this construction, we derive triply-exponential upper bounds on the necessary lookahead for Player O, i.e., bounded lookahead always suffices, as well as a triply-exponential upper bound on the necessary scope of the prompt eventually operator. Thus, we obtain the same upper bounds as for LTL.

Third, we complement all three upper bounds by matching lower bounds, e.g., the problem is 3ExpTime-complete and there are triply-exponential lower bounds on the necessary lookahead and on the scope of the prompt eventually operator. The former two lower bounds already hold for the special case of LTL delay games. Thereby, we settle the case of delay games with LTL winning conditions as well as the case of delay games with Prompt-LTL winning conditions and show that they are of equal complexity and that the same bounds on the necessary lookahead hold. Thus, we prove that delay games with Prompt-LTL winning conditions are not harder than those with LTL winning conditions. The complexity of solving LTL games increases exponentially when adding lookahead, which is in line with the results in the ω-regular case [16], where one also observes an exponential blowup.

Fourth, our proofs are all applicable to the stronger extensions of LTL like parametric LTL [25], parametric LDL [6], and their variants with costs [26], as the alternating-color technique is applicable to them as well and as their formulas can be compiled into equivalent exponential Büchi automata.

Fifth, we show that our lower bounds also answer open questions in the ω-regular case mentioned in [16], e.g., on the influence of the branching mode of the specification automaton on the complexity. Recall that the tight exponential bounds on the complexity and the necessary lookahead for ω-regular delay games were shown for winning conditions given by *deterministic* automata. Our lower bounds proven here can be adapted to show that both these bounds are doubly-exponential for non-deterministic and universal automata and triply-exponential for alternating automata. Hence, the lower bounds match the trivial upper bounds obtained by determinizing the automata and applying the results from [16]. Thus, we complete the picture in the ω-regular case with regard to the branching mode of the specification automaton.

Related Work. Delay games with ω-regular winning conditions have been introduced by Hosch and Landweber, who proved that the winner w.r.t. bounded lookahead can be determined effectively [14]. Later, they were revisited by Holtmann et al. who showed that bounded lookahead is always sufficient and who gave a streamlined algorithm with doubly-exponential running time and a doubly-exponential upper bound on the necessary lookahead [12]. Recently, the tight exponential bounds on the running time and on the lookahead mentioned above were proven [16]. Delay games with context-free winning conditions turned out to be undecidable for very small fragments [9]. The results of going beyond the ω-regular case by considering WMSO+U winning conditions are mentioned above. Furthermore, all delay games with Borel winning conditions are determined [15]. Finally, from a more theoretical point of view, Holtman et al. also showed that delay games are a suitable representation of uniformization problems for relations by continuous functions [12].

All proofs omitted due to space constraints can be found in the full version [17].

2 Preliminaries

The set of non-negative (positive) integers is denoted by $\mathbb{N}$ ($\mathbb{N}_+$). An alphabet Σ is a non-empty finite set of letters, Σ^* is the set of finite words over Σ, Σ^i the set of words of length i, and Σ^ω the set of infinite words. The empty word is denoted by ε and the length of a finite word w by $|w|$. For $w \in \Sigma^* \cup \Sigma^\omega$ we write $w(i)$ for the i-th letter of w. Given two infinite words $\alpha \in \Sigma_I^\omega$ and $\beta \in \Sigma_O^\omega$ we write $\binom{\alpha}{\beta}$ for the word $\binom{\alpha(0)}{\beta(0)}\binom{\alpha(1)}{\beta(1)}\binom{\alpha(2)}{\beta(2)}\cdots \in (\Sigma_I \times \Sigma_O)^\omega$. Analogously, we write $\binom{x}{y}$ for finite words x and y, provided they are of equal length.

Parity Games. An arena $\mathcal{A}$ is a tuple (V, V_I, V_O, E), where (V, E) is a finite directed graph without terminal vertices and $\{V_I, V_O\}$ is a partition of V into the positions of Player I and Player O. A parity game $\mathcal{G} = (\mathcal{A}, \Omega)$ consists of an arena $\mathcal{A}$ with vertex set V and of a priority function $\Omega\colon V \to \mathbb{N}$. A play ρ is an infinite sequence $v_0v_1v_2\cdots$ of vertices such that $(v_i, v_{i+1}) \in E$ for all i. A strategy for Player O is a map $\sigma\colon V^*V_O \to V$ such that $(v_i, \sigma(v_0\cdots v_i)) \in E$ for all $v_i \in V_O$. The strategy σ is positional, if $\sigma(wv) = \sigma(v)$ for all $wv \in V^*V_O$. Hence, we denote it as mapping from V_O to V. A play $v_0v_1v_2\cdots$ is consistent with σ, if $v_{i+1} = \sigma(v_0\cdots v_i)$ for every i with $v_i \in V_O$. The strategy σ is winning from a vertex $v \in V$, if every play $v_0v_1v_2\cdots$ with $v_0 = v$ that is consistent with σ satisfies the parity condition, i.e., the maximal priority appearing infinitely often in $\Omega(v_0)\Omega(v_1)\Omega(v_2)\cdots$ is even. The definition of (winning) strategies for Player I is dual. Parity games are positionally determined [5, 20], i.e., from every vertex one of the players has a positional winning strategy.

Delay Games. A delay function is a mapping $f\colon \mathbb{N} \to \mathbb{N}_+$, which is said to be constant, if $f(i) = 1$ for every $i > 0$. Given a winning condition $L \subseteq (\Sigma_I \times \Sigma_O)^\omega$ and a delay function f, the game $\Gamma_f(L)$ is played by two players, Player I and Player O, in rounds $i = 0, 1, 2, \ldots$ as follows: in round i, Player I picks a word $u_i \in \Sigma_I^{f(i)}$, then Player O picks one letter $v_i \in \Sigma_O$. We refer to the sequence $(u_0, v_0), (u_1, v_1), (u_2, v_2), \ldots$ as a play of $\Gamma_f(L)$. Player O wins the play if the outcome $\binom{u_0u_1u_2\cdots}{v_0v_1v_2\cdots}$ is in L, otherwise Player I wins.

Given a delay function f, a strategy for Player I is a mapping $\tau_I\colon \Sigma_O^* \to \Sigma_I^*$ where $|\tau_I(w)| = f(|w|)$, and a strategy for Player O is a mapping $\tau_O\colon \Sigma_I^* \to \Sigma_O$. Consider a play $(u_0, v_0), (u_1, v_1), (u_2, v_2), \ldots$ of $\Gamma_f(L)$. Such a play is consistent with τ_I, if $u_i = \tau_I(v_0\cdots v_{i-1})$ for every $i \in \mathbb{N}$. It is consistent with τ_O, if $v_i = \tau_O(u_0\cdots u_i)$ for every $i \in \mathbb{N}$. A strategy τ for Player $P \in \{I, O\}$ is winning, if every play that is consistent with τ is winning for Player P. We say that a player wins $\Gamma_f(L)$, if she has a winning strategy.

Prompt LTL. Fix a set AP of atomic propositions. Prompt-LTL formulas are given by

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

where $p \in \mathrm{AP}$. We use $\varphi \to \psi$ as shorthand for $\neg\varphi \vee \psi$, where we require φ to be a $\mathbf{F_P}$-free formula (for which the negation can be pushed to the atomic propositions using the dualities of the classical temporal operators). The size $|\varphi|$ of φ is the number of subformulas of φ.

The satisfaction relation is defined for an ω-word $w \in \left(2^{\mathrm{AP}}\right)^\omega$, a position i of w, a bound k for the prompt eventually operators, and a Prompt-LTL formula. The definition is standard for the classical operators and defined as follows for the prompt eventually:

$$(w, i, k) \models \mathbf{F_P}\,\varphi \text{ if, and only if, there exists a } j \text{ with } 0 \leq j \leq k \text{ such that } (w, i + j, k) \models \varphi.$$

For the sake of brevity, we write $(w, k) \models \varphi$ instead of $(w, 0, k) \models \varphi$. Note that φ is an LTL formula [21], if it does not contain the prompt eventually operator. Then, we write $w \models \varphi$.

The Alternating-color Technique. Let $p \notin \mathrm{AP}$ be a fixed fresh proposition. An ω-word $w' \in \left(2^{\mathrm{AP}\cup\{p\}}\right)^\omega$ is a p-coloring of $w \in \left(2^{\mathrm{AP}}\right)^\omega$ if $w'(i) \cap \mathrm{AP} = w(i)$ for all i.

A position i of a word in $\left(2^{\mathrm{AP}\cup\{p\}}\right)^\omega$ is a change point, if $i = 0$ or if the truth value of p at positions $i-1$ and i differs. A p-block is an infix $w'(i)\cdots w'(i+j)$ of w' such that i and $i+j+1$ are adjacent change points. Let $k \geq 1$: we say that w' is k-spaced, if w has infinitely many changepoints and each p-block has length at least k; we say that w' is k-bounded, if each p-block has length at most k (which implies that w' has infinitely many change points).

Given a Prompt-LTL formula φ, let $\mathrm{rel}'(\varphi)$ denote the formula obtained by inductively replacing every subformula $\mathbf{F_P}\,\psi$ by

$$(p \rightarrow (p\,\mathbf{U}\,(\neg p\,\mathbf{U}\,\mathrm{rel}'(\psi)))) \wedge (\neg p \rightarrow (\neg p\,\mathbf{U}\,(p\,\mathbf{U}\,\mathrm{rel}'(\psi))))$$

and let $\mathrm{rel}(\varphi) = \mathrm{rel}'(\varphi) \wedge \mathbf{G}\,\mathbf{F}\,p \wedge \mathbf{G}\,\mathbf{F}\,\neg p$, i.e., we additionally require infinitely many change points. Intuitively, instead of requiring ψ to be satisfied within a bounded number of steps, $\mathrm{rel}(\varphi)$ requires it to be satisfied within at most one change point. The relativization $\mathrm{rel}(\varphi)$ is an LTL formula of size $\mathcal{O}(|\varphi|)$. Kupferman et al. showed that φ and $\mathrm{rel}(\varphi)$ are "equivalent" on ω-words which are bounded and spaced.

▶ **Lemma 1** ([18]). *Let φ be a* Prompt-LTL *formula and $k \in \mathbb{N}$.*
1. *If $(w, k) \models \varphi$, then $w' \models \mathrm{rel}(\varphi)$ for every k-spaced p-coloring w' of w.*
2. *If w' is a k-bounded p-coloring of w such that $w' \models \mathrm{rel}(\varphi)$, then $(w, 2k) \models \varphi$.*

3 Delay Games with Prompt-LTL Winning Conditions

In this section, we study delay games with Prompt-LTL winning conditions. Player O's goal in such games is to satisfy the winning condition φ with respect to a bound k which is uniform among all plays consistent with the strategy. We show that such games are reducible to delay games with LTL winning conditions by tailoring the alternating-color technique to delay games and integrating it into the algorithm for solving ω-regular delay games [16].

Throughout this section, we fix a partition $\mathrm{AP} = I \cup O$ of the set of atomic propositions into input propositions I under Player $I's$ control and output propositions O under Player O's control. Let $\Sigma_I = 2^I$ and $\Sigma_O = 2^O$. Given $\binom{\alpha}{\beta} \in (\Sigma_I \times \Sigma_O)^\omega$, we write $(\binom{\alpha}{\beta}, k) \models \varphi$ for $((\alpha(0) \cup \beta(0))\,(\alpha(1) \cup \beta(1))\,(\alpha(2) \cup \beta(2)) \cdots, k) \models \varphi$. Given φ and a bound k, we define $L(\varphi, k) = \{\binom{\alpha}{\beta} \in (\Sigma_I \times \Sigma_O)^\omega \mid (\binom{\alpha}{\beta}, k) \models \varphi\}$. If φ is an LTL formula, then this language is independent of k and will be denoted by $L(\varphi)$.

A Prompt-LTL delay game $\Gamma_f(\varphi)$ consists of a delay function f and a Prompt-LTL formula φ. We say that Player $P \in \{I, O\}$ wins $\Gamma_f(\varphi)$ for the bound k, if she wins $\Gamma_f(L(\varphi, k))$. If we are not interested in the bound itself, but only in the existence of some bound, then we also say that Player O wins $\Gamma_f(\varphi)$, if there is some k such that she wins $\Gamma_f(\varphi)$ for k. If φ is an LTL formula, then we call $\Gamma_f(\varphi)$ an LTL delay game. The winning condition $L(\varphi)$ of such a game is ω-regular and independent of k.

In this section, we solve the following decision problem: given a Prompt-LTL formula φ, does Player O win $\Gamma_f(\varphi)$ for some delay function f? Furthermore, we obtain upper bounds on the necessary lookahead and the necessary bound k, which are complemented by matching lower bounds in the next section.

With all definitions at hand, we state our main theorem of this section.

▶ **Theorem 2.** *The following problem is in* 3ExpTime*: given a* Prompt-LTL *formula φ, does Player O win $\Gamma_f(\varphi)$ for some delay function f?*

Proof. We reduce Prompt-LTL to LTL delay games using the alternating-color technique. To this end, we add the proposition p, which induces the coloring, to O, i.e., in a game with winning condition $\mathrm{rel}(\varphi)$ Player O's alphabet is $2^{O\cup\{p\}}$. In Lemma 4, we prove that Player O wins $\Gamma_f(\varphi)$ for some delay function f if, and only if, Player O wins $\Gamma_f(\mathrm{rel}(\varphi))$ for some delay function f. This equivalence proves our claim: Determining whether Player O wins a delay game (for some f) whose winning condition is given by a deterministic parity automaton is ExpTime-complete [16]. We obtain an algorithm with triply-exponential running time by constructing a doubly-exponential deterministic parity automaton recognizing $L(\mathrm{rel}(\varphi))$ and then running the exponential-time algorithm on it. ◀

Thus, it remains to prove the equivalence between the delay games with winning conditions φ and $\mathrm{rel}(\varphi)$. The harder implication is the one from the LTL delay game to the Prompt-LTL delay game. There is a straightforward extension of the solution to the delay-free case. There, one proves that a finite-state strategy for the LTL game with winning condition $\mathrm{rel}(\varphi)$ (which always exists, if Player O wins the game) only produces k-bounded outcomes, for some k that only depends on the size of the strategy. Hence, by projecting away the additional proposition p inducing the coloring, we obtain a winning strategy for the Prompt-LTL game with winning condition φ with bound $2k$ by applying Lemma 1.2.

Now, consider the case with lookahead: if Player O wins $\Gamma_f(\mathrm{rel}(\varphi))$, which has an ω-regular winning condition, then also $\Gamma_{f'}(\mathrm{rel}(\varphi))$ for some triply-exponential constant f' [16]. We can model $\Gamma_{f'}(\mathrm{rel}(\varphi))$ as a delay-free parity game of quadruply-exponential size by storing the lookahead explicitly in the state space of the parity game. A positional winning strategy in this parity game only produces k-bounded plays, where k is the size of the delay-free game, as the color has to change infinitely often. Hence, such a strategy can be turned into a winning strategy for Player O in $\Gamma_{f'}(\varphi)$ with respect to some quadruply-exponential bound k. However, this naive approach is not optimal: we present a more involved construction that achieves a triply-exponential bound k. The problem with the aforementioned approach is that the decision to produce a change point depends on the complete lookahead. We show how to base this decision on an exponentially smaller abstraction of the lookaheads, which yields an asymptotically optimal bound k. To this end, we extend the construction underlying the algorithm for ω-regular delay games [16] by integrating the alternating-color technique.

Intuitively, we assign to each $w \in \Sigma_I^*$, i.e., to each potential additional information Player O has access to due to the lookahead, the behavior w induces in a deterministic automaton $\mathcal{A}$ accepting $L(\mathrm{rel}(\varphi))$, namely the state changes induced by w and the most important color on these runs. We construct $\mathcal{A}$ such that it keeps track of change points in its state space, which implies that they are part of the behavior of w. Then, we construct a parity game in which Player I picks such behaviors instead of concrete words over Σ_I and Player O constructs a run on suitable representatives. The resulting game is of triply-exponential size and a positional winning strategy for this game can be turned back into a winning strategy for $\Gamma_f(\varphi)$ satisfying asymptotically optimal bounds on the initial lookahead and the bound k. Thus, we save one exponent by not explicitly considering the lookahead, but only its effects.

We first extend the construction of a delay-free parity game $\mathcal{G}$ that has the same winner as $\Gamma_f(\mathrm{rel}(\varphi))$ from [16]. The extension is necessary to obtain a "small" bound k when applying the alternating-color technique, which turns a positional winning strategy for $\mathcal{G}$ into a winning strategy for Player O in $\Gamma_{f'}(\varphi)$ for some f'.

To this end, let $\mathcal{A} = (Q, \Sigma_I \times \Sigma_O, q_I, \delta, \Omega)$ be a deterministic max-parity automaton[1]

[1] Recall that $\Omega\colon Q \to \mathbb{N}$ is a coloring of the states and a run $q_0q_1q_2\cdots$ is accepting, if the maximal color occurring infinitely often in $\Omega(q_0)\Omega(q_1)\Omega(q_2)\cdots$ is even. See, e.g., [11] for details.

recognizing $L(\mathrm{rel}(\varphi))$. First, as in the original construction, we add a deterministic monitoring automaton to keep track of certain information of runs. In the ω-regular case [16], this information is the maximal priority encountered during a run. Here, we additionally need to remember whether the input word contains a change point. Let $T_0 = 2^{\{p\}}$ and $T = T_0 \times \{0, 1\}$. Furthermore, for $(t, s) \in T$ and $t' \in T_0$, we define the update $\mathrm{upd}((t, s), t') \in T$ of (t, s) by t' to be (t', s'), where $s' = 0$ if, and only if, $s = 0$ and $t = t'$. Intuitively, the first component of a tuple in T stores the last truth value of p and the second component is equal to one if, and only if, there was a change point.

Now, we define the deterministic parity automaton $\mathcal{T} = (Q_\mathcal{T}, \Sigma_I \times \Sigma_O, q_I^\mathcal{T}, \delta_\mathcal{T}, \Omega_\mathcal{T})$ with $Q_\mathcal{T} = Q \times \Omega(Q) \times T$, $q_I^\mathcal{T} = (q_I, \Omega(q_I), (t', 0))$ for some arbitrary $t' \in T_0$, $\Omega(q, m, t) = m$, and $\delta_\mathcal{T}((q, m, t), \binom{a}{b}) = (q', \max\{m, \Omega(q')\}, \mathrm{upd}(t, b \cap \{p\}))$ with $q' = \delta(q, \binom{a}{b})$.

First, let us note that $\mathcal{T}$ does indeed keep track of the information described above.

▶ Remark 3. Let $w \in (\Sigma_I \times \Sigma_O)^+$ and let $(q_0, m_0, t_0) \cdots (q_{|w|}, m_{|w|}, t_{|w|})$ be the run of $\mathcal{T}$ on w starting in (q_0, m_0, t_0) such that $m_0 = \Omega(q_0)$ and $t_0 = (t'_0, 0)$ for some $t'_0 \in T_0$. Then, $q_0 q_1 \cdots q_{|w|}$ is the run of $\mathcal{A}$ on w starting in q_0, $m_{|w|} = \max\{\Omega(q_j) \mid 0 \le j \le |w|\}$, and $t_{|w|} = (t'_{|w|}, s_{|w|})$ such that $t'_{|w|}$ is the color of the last letter of w, and such that $s_{|w|} = 0$ if, and only if, all letters of w have color t'_0. In particular, if w is preceded by a word whose last letter has color t'_0, then there is a change point in w if, and only if, $s_{|w|} = 1$.

Next, we classify possible moves $w \in \Sigma_I^*$ according to the behavior they induce on $\mathcal{T}$. Let $\delta_\mathcal{P} \colon 2^{Q_\mathcal{T}} \times \Sigma_I \to 2^{Q_\mathcal{T}}$ denote the transition function of the power set automaton of the projection of $\mathcal{T}$ to Σ_I, i.e., $\delta_\mathcal{P}(S, a) = \{\delta_\mathcal{T}(q, \binom{a}{b}) \mid q \in S \text{ and } b \in \Sigma_O\}$. As usual, we define $\delta^*_\mathcal{P} \colon 2^{Q_\mathcal{T}} \times \Sigma_I^* \to 2^{Q_\mathcal{T}}$ inductively via $\delta^*_\mathcal{P}(S, \varepsilon) = S$ and $\delta^*_\mathcal{P}(S, wa) = \delta_\mathcal{P}(\delta^*_\mathcal{P}(S, w), a)$.

Let $D \subseteq Q_\mathcal{T}$ be non-empty and let $w \in \Sigma_I^+$. We define the function $r_w^D \colon D \to 2^{Q_\mathcal{T}}$ via

$$r_w^D(q, m, (t, s)) = \delta^*_\mathcal{P}(\{\, (q, \Omega(q), (t, 0)) \,\}, w)$$

for every $(q, m, (t, s)) \in D$. Note that we use $\Omega(q)$ and $(t, 0)$ as the second and third component in the input for $\delta^*_\mathcal{P}$, not m and (t, s) from the input to r_w^D. This resets the tracking components of $\mathcal{T}$. If we have $(q', m', (t', s')) \in r_w^D(q, m, (t, s))$, then there is a word w' over $\Sigma_I \times \Sigma_O$ whose projection to Σ_I is w and such that the run of $\mathcal{A}$ processing w' from q has the maximal priority m', t' is the color of the last letter of w, and s' encodes the existence of change points in w', as explained in Remark 3. Thus, this function captures the behavior induced by w on $\mathcal{T}$. We allow to restrict the domain of such a function, as we do not have to consider every possible state, only those that are reachable by the play prefix constructed thus far.

Let $r \colon Q_\mathcal{T} \to 2^{Q_\mathcal{T}}$ be a partial function. We say that w is a witness for r, if $r_w^{\mathrm{dom}(r)} = r$. Thus, we can assign a language $W_r \subseteq \Sigma_I^*$ of witnesses to each such r. Let $\mathfrak{R}$ denote the set of such functions r with infinite witness language W_r. If w is a witness of $r \in \mathfrak{R}$, then r encodes the state transformations induced by w in the projection of $\mathcal{A}$ to Σ_I as well as the maximal color occurring on these runs and the existence of change points on these. The latter is determined by the letters projected away, but still stored explicitly in the state space of the automaton. Furthermore, as we require $r \in \mathfrak{R}$ to have infinitely many witnesses, there are arbitrarily long words with the same behavior. On the other hand, the language W_r of witnesses of r is recognizable by a DFA of size 2^{n^2} [16], where n is the size of $\mathcal{T}$. Hence, every r also has a witness of length at most 2^{n^2}. This allows to replace *long* words $w \in \Sigma_I^*$ by equivalent ones that are bounded exponentially in n.

Next, we define a delay-free parity game in which Player I picks functions $r_i \in \mathfrak{R}$ while Player O picks states q_i such that there is a word w'_i in $(\Sigma_I \times \Sigma_O)^*$ whose projection to Σ_I is a witness of r_i and such that w'_i leads $\mathcal{T}$ from q_i to q_{i+1}. By construction, this property is independent of the choice of the witness. Furthermore, to account for the delay, Player I is always two moves ahead. Thus, instead of picking explicit words over their respective alphabets, the players pick abstractions, Player I explicitly and Player O implicitly by constructing the run.

Formally we define the parity game $\mathcal{G} = ((V, V_O, V_I, E), \Omega')$ where $V = V_I \cup V_O$, $V_I = \{v_I\} \cup \mathfrak{R} \times Q_\mathcal{T}$ with the designated initial vertex v_I of the game, and $V_O = \mathfrak{R}$. Further, E is the union of the following sets of edges: initial moves $\{(v_I, r) \mid \mathrm{dom}(r) = \{q_I^\mathcal{T}\}\}$ for Player I, regular moves $\{((r, q), r') \mid \mathrm{dom}(r') = r(q)\}$ for Player I, and moves $\{(r, (r, q)) \mid q \in \mathrm{dom}(r)\}$ for Player O. Finally, $\Omega'(v) = m$, if $v = (r, (q, m, s)) \in \mathfrak{R} \times Q_\mathcal{T}$, and zero otherwise.

This finishes the construction of the game $\mathcal{G}$. The following lemma states the relation between $\mathcal{G}$ and the delay games with winning conditions φ and $\mathrm{rel}(\varphi)$ and implies the equivalence of the delay games with winning conditions φ and $\mathrm{rel}(\varphi)$.

▶ **Lemma 4.** *Let $n = |Q_\mathcal{T}|$, where $Q_\mathcal{T}$ is the set of states of $\mathcal{T}$ as defined above.*
1. *If Player O wins $\Gamma_f(\varphi)$ for some delay function f, then also $\Gamma_f(\mathrm{rel}(\varphi))$ for the same f.*
2. *If Player O wins $\Gamma_f(\mathrm{rel}(\varphi))$ for some delay function f, then also $\mathcal{G}$.*
3. *If Player O wins $\mathcal{G}$, then also $\Gamma_f(\varphi)$ for the constant delay function f with $f(0) = 2^{n^2+1}$ and some bound $k \leq 2^{2n^2+2}$.*

The automaton $\mathcal{A}$ recognizing $L(\mathrm{rel}(\varphi))$ can be constructed such that $|\mathcal{A}| \in 2^{2^{\mathcal{O}(|\varphi|)}}$, which implies $n \in 2^{2^{\mathcal{O}(|\varphi|)}}$, using a standard construction for translating LTL into non-deterministic Büchi automata and then Schewe's determinization construction [24]. Applying all implications of Lemma 4 yields upper bounds on the neccessary constant lookahead and on the neccessary bound k on the scope of the prompt eventually operator.

▶ **Corollary 5.** *If Player O wins $\Gamma_f(\varphi)$ for some delay function f and some k, then also for some constant delay function f with $f(0) \in 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ and some $k \in 2^{2^{2^{\mathcal{O}(|\varphi|)}}}$ simultaneously.*

4 Lower Bounds for LTL and Prompt-LTL Delay Games

We complement the upper bounds on the complexity of solving Prompt-LTL delay games, on the necessary lookahead, and on the necessary bound k by proving tight lower bounds in all three cases. The former two bounds already hold for LTL.

All proofs share some similarities which we discuss first. In particular, they all rely on standard encodings of doubly-exponentially large numbers using *small* LTL formulas and the interaction between the players. Assume AP contains the propositions $b_0, \ldots, b_{n-1}, b_I, b_O$ and let $w \in (2^{\mathrm{AP}})^\omega$ and $i \in \mathbb{N}$. We interpret $w(i) \cap \{b_0, \ldots, b_{n-1}\}$ as binary encoding of a number in $[0, 2^n - 1]$, which we refer to as the address of position i. There is a formula ψ_{inc} of quadratic size in n such that $(w, i) \models \psi_{\mathrm{inc}}$ if, and only if, $m + 1 \bmod 2^n = m'$, where m is the address of position i and m' is the address of position $i+1$. Now, let $\psi_0 = \bigwedge_{j=0}^{n-1} \neg b_j \wedge \mathbf{G}\, \psi_{\mathrm{inc}}$. If $w \models \psi_0$, then the b_j form a cyclic addressing of the positions starting at zero, i.e., the address of position i is $i \bmod 2^n$. If this is the case, we define a block of w to be an infix that starts at a position with address zero and ends at the next position with address $2^n - 1$. We interpret the 2^n bits b_I of a block as a number x in $R = [0, 2^{2^n} - 1]$. Similarly, we interpret the 2^n bits b_O of a block as a number y from the same range R. Furthermore, there are *small* formulas that are satisfied at the start of the i-th block if, and only if, $x_i = y_i$ ($x_i < y_i$, respectively). However, we cannot compare numbers from different blocks for equality with *small* formulas. Nevertheless, if x_i is unequal to $x_{i'}$, then there is a single bit that witnesses this, i.e., the bit is one in x_i if, and only if, it is zero in $x_{i'}$. We will check this by letting one of the players specify the address of such a witness (but not the witness itself). The correctness of this claim is then verifiable by a *small* formula.

4.1 Lower Bounds on Lookahead

Our first result concerns a triply-exponential lower bound on the necessary lookahead in LTL delay games, which matches the upper bound proven in the previous section. The exponential lower bound 2^n on the necessary lookahead for ω-regular delay games is witnessed by winning conditions over the alphabet $1, \ldots, n$. These conditions require to remember letters and to compare them for equality and order [16]. Here, we show how to adapt the winning condition to the alphabet R, which yields a triply-exponential lower bound $2^{|R|}$. The main difficulty of the proof is the inability of small LTL formulas to compare letters from R. To overcome this, we exploit the interaction between the players of the game.

▶ **Theorem 6.** *For every $n > 0$, there is an* LTL *formula φ_n of size $\mathcal{O}(n^2)$ such that*

- *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. Fix some $n > 0$. In the following, we measure all formula sizes in n. Furthermore, let $I = \{b_0, \ldots, b_{n-1}, b_I, \#\}$ and $O = \{b_O, \rightarrow, \leftarrow\}$. Assume $\binom{\alpha}{\beta} \in (\Sigma_I \times \Sigma_O)^\omega$ satisfies ψ_0 from above. Then, α induces a sequence $x_0x_1x_2 \cdots \in R^\omega$ of numbers encoded by the bits b_I in each block. Similarly, β induces a sequence $y_0y_1y_2 \cdots \in R^\omega$.

The winning condition is intuitively described as follows: x_i and $x_{i'}$ with $i < i'$ constitute a bad j-pair, if $x_i = x_{i'} = j$ and $x_{i''} < j$ for all $i < i'' < i'$. Every sequence $x_0x_1x_2 \cdots$ contains a bad j-pair, e.g., pick j to be the maximal number occurring infinitely often. In order to win, Player O has to pick y_0 such that $x_0x_1x_2 \cdots$ contains a bad y_0-pair. It is known that this winning condition requires lookahead of length 2^m for Player O to win, where m is the largest number that can be picked [16].

To specify this condition with a small LTL formula, we have to require Player O to copy y_0 ad infinitum, i.e., to pick $y_i = y_0$ for all i, and to mark the two positions constituting the bad y_0-pair. Furthermore, the winning condition allows Player I to mark one copy error introduced by Player O by specifying its address by a $\#$ (which may appear anywhere in α). This forces Player O to implement the copying correctly and thus allows a small formula to check that Player O indeed marks a bad y_0-pair. Consider the following properties:

1. $\#$ holds at most once. Player I uses $\#$ to specify the address where he claims an error.
2. $\rightarrow$ holds at exactly one position, which has to be the start of a block. Furthermore, we require the two numbers encoded by the propositions b_I and b_O within this block to be equal. Player O uses $\rightarrow$ to denote the first component of a claimed bad j-pair.
3. $\leftarrow$ holds at exactly one position, which has to be the start of a block and has to appear at a later position than $\rightarrow$. Again, we require the two numbers encoded by this block to be equal. Player O uses $\leftarrow$ to denote the second component of the claimed bad j-pair.
4. For every block between the two marked blocks, we require the number encoded by the b_I to be strictly smaller than the number encoded by the b_O.
5. If there is a position $i_\#$ marked by $\#$, then there are no two different positions $i \neq i'$ such that the following two conditions are satisfied: the addresses of i, i', and $i_\#$ are equal and b_O holds at i if, and only if, b_O does not hold at i'. Such positions witness an error in the copying process by Player O, which manifests itself in a single bit, whose address is marked by Player I at any time in the future.

Each of these properties $i \in \{1,2,3,4,5\}$ can be specified by an LTL formula ψ_i of at most quadratic size. Now, let $\varphi_n = (\psi_0 \wedge \psi_1) \rightarrow (\psi_2 \wedge \psi_3 \wedge \psi_4 \wedge \psi_5)$. We show that Player O wins $\Gamma_f(\varphi_n)$ for some triply-exponential constant delay function, but not for any smaller one.

Fix $n' = 2^{2^n}$. We begin by showing that Player O wins $\Gamma_f(\varphi_n)$ for the constant delay function with $f(0) = 2^n \cdot 2^{n'}$. A simple induction shows that every word $w \in R^*$ of length $2^{n'}$ contains a bad j-pair for some $j \in R$. Thus, a move $\Sigma_I^{f(0)}$ made by Player I in round 0 interpreted as sequence $x_0 x_1 \cdots x_{2^{n'}-1} \in R^*$ contains a bad j-pair for some fixed j. Hence, Player O's strategy τ_O produces the sequence j^ω and additionally marks the corresponding bad j-pair with ➜ and 🡐. Every outcome of a play that is consistent with τ_O and satisfies ψ_0 also satisfies $\psi_2 \wedge \psi_3 \wedge \psi_4 \wedge \psi_5$, as Player O correctly marks a bad j-pair and never introduces a copy-error. Hence, τ_O is a winning strategy for Player O.

It remains to show that Player I wins $\Gamma_f(\varphi_n)$, if $f(0) \leq 2^n \cdot (2^{n'} - 2) \geq 2^{n'} = 2^{2^{2^n}}$. Let $w_{n'} \in R^*$ be recursively defined via $w_0 = 0$ and $w_j = w_{j-1}\, j\, w_{j-1}$. A simple induction shows that $w_{n'}$ does not contain a bad j-pair, for every $j \in R$, and that $|w_{n'}| = 2^{n'} - 1$.

Consider the following strategy τ_I for Player I in $\Gamma_f(\varphi_n)$: τ ensures that ψ_0 is satisfied by the b_j, which fixes them uniquely to implement a cyclic addressing starting at zero. Furthermore, he picks the b_I's so that the sequence of numbers $x_0 x_1 \cdots x_\ell$ he generates during the first 2^n rounds is a prefix of $w_{n'}$. This is possible, as each x_i is encoded by 2^n bits and by the choice of $f(0)$. As a response during the first 2^n rounds, Player O determines some number $y \in R$. During the next rounds, Player I finishes $w_{n'}$ and then picks some fixed $x \neq y$ ad infinitum (while still implementing the cyclic addressing). In case Player O picks both markings ➜ and 🡐 in way that is consistent with properties 2, 3, and 4 as above, let $y_0 y_1 \cdots, y_i$ be the sequence of numbers picked by her up to and including the number marked by 🡐. If they are not all equal, then there is an address that witnesses the difference between two of these numbers. Player I then marks exactly one position with the same address using #. If this is not the case, he never marks a position with #.

Consider an outcome of a play that is consistent with τ_I and let $x_0 x_1 x_2 \cdots \in R^\omega$ and $y_0 y_1 y_2 \cdots \in R^\omega$ be the sequences of numbers induced by the outcome. By definition of τ_I, the antecedent $\psi_0 \wedge \psi_1$ of φ_n is satisfied and $x_0 x_1 x_2 \cdots = w_{n'} \cdot x^\omega$ for some $x \neq y_0$.

If Player O never uses her markers ➜ and 🡐 in a way that satisfies $\psi_2 \wedge \psi_3 \wedge \psi_4$, then Player I wins the play, as it satisfies the antecedent of φ_n, but not the consequent. Thus, it remains to consider the case where the outcome satisfies $\psi_2 \wedge \psi_3 \wedge \psi_4$. Let $y_0 y_1 \cdots y_i$ be the sequence of numbers picked by her up to and including the number marked by 🡐. Assume we have $y_0 = y_1 = \cdots = y_i$. Then, ➜ and 🡐 specify a bad y_0-pair, as implied by $\psi_2 \wedge \psi_3 \wedge \psi_4$ and the equality of the y_j. As $w_{n'}$ does not contain a bad y_0-pair, we conclude $y_0 = x$. However, τ_I ensures $y_0 \neq x$. Hence, our assumption is false, i.e., the y_j are not all equal. In this situation, τ_I marks a position whose address witnesses this difference. This implies that ψ_5 is not satisfied, i.e., the play is winning for Player I. Hence, τ_I is winning for him. ◀

4.2 Lower Bounds on the Bound k

Our next result is a lower bound on the necessary bound k in a Prompt-LTL delay game, which is proven by a small adaption of the game constructed in the previous proof. The winning condition additionally requires Player O to use the mark 🡐 at least once and k measures the number of rounds before Player O does so. It turns out Player I can enforce a triply-exponential k, which again matches the upper bound proven in the previous section.

▶ **Theorem 7.** *For every $n > 0$, there is a Prompt LTL formula φ'_n of size $\mathcal{O}(n^2)$ such that*
- *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. Let $\varphi_n' = (\psi_0 \wedge \psi_1) \rightarrow (\psi_2 \wedge \psi_3 \wedge \psi_4 \wedge \psi_5 \wedge \mathbf{F_P} \twoheadleftarrow)$, where the alphabets and the formulas ψ_i are as in the proof of Theorem 6.

Let $k = f(0) = 2^n \cdot 2^{n'}$ with $n' = 2^{2^n}$ as above. Then, the strategy τ_O for Player O described in the proof of Theorem 6 is winning for $\Gamma_f(\varphi_n)$ with bound k: it places both markers within the first $f(0) = k$ positions, as it specifies a bad j-pair within this range.

Now, assume we have $k < 2^n \cdot 2^{n'}$, consider the strategy τ_I for Player I as defined in the proof of Theorem 6, and recall that every outcome that is consistent with τ_I starts with the sequence $w_{n'}$ in the first component. Satisfying $\psi_2 \wedge \psi_3 \wedge \psi_4 \wedge \psi_5$ against τ_I requires Player O to mark a bad j-pair and to produce the sequence j^ω. However, Player I does not produce a bad j-pair in the first k positions, i.e., the conjunct $\mathbf{F_P} \twoheadleftarrow$ is not satisfied with respect to k. Hence, τ_I is winning for Player I in $\Gamma_f(\varphi_n)$ with bound k. ◀

4.3 Lower Bounds on Complexity

Our final result settles the complexity of solving Prompt-LTL delay games. The triply-exponential algorithm presented in the previous section is complemented by proving the problem to be 3ExpTime-complete, which even holds for LTL. The proof is a combination of techniques developed for the lower bound on the lookahead presented above and of techniques from the ExpTime-hardness proof for solving delay games whose winning conditions are given by deterministic safety automata [16] and is presented in the full version [17].

▶ **Theorem 8.** *The following problem is* 3ExpTime*-complete: given an* LTL *formula φ, does Player O win $\Gamma_f(\varphi)$ for some delay function f?*

5 Delay Games on Non-deterministic, Universal, and Alternating Automata

Finally, we argue that the lower bounds just proven for LTL delay games can be modified to solve open problems about ω-regular delay games whose winning conditions are given by non-deterministic, universal, and alternating automata (note that non-determinism and universality are not dual here, as delay games are asymmetric).

Recall that solving delay games with winning conditions given by deterministic parity automata is ExpTime-complete and that exponential constant lookahead is sufficient and in general necessary. These upper bounds yield doubly-exponential upper bounds on both complexity and lookahead for non-deterministic and universal parity automata via determinization, which incurs an exponential blowup. Similarly, we obtain triply-exponential upper bounds on both complexity and lookahead for alternating parity automata, as determinization incurs a doubly-exponential blowup in this case.

For alternating automata, these upper bounds are tight, as LTL can be translated into linearly-sized alternating automata (even with very weak acceptance conditions). Hence, the triply-exponential lower bounds proven in the previous section hold here as well.

To prove doubly-exponential lower bounds for the case of non-deterministic and universal automata, one has to modify the constructions presented in the previous section. Let us first consider the case of non-deterministic automata: to obtain a matching doubly-exponential lower bound on the necessary lookahead, we require Player I to produce an input sequence in $\{0,1\}^\omega$, where we interpret every block of n bits as the binary encoding of a number in $\{0, 1, \ldots, 2^n - 1\}$. In order to win, Player O also has to pick an encoding of a number j with her first n moves such that the sequence of numbers picked by Player I contains a bad j-pair. To allow the automaton to check the correctness of this pick, we require Player O to

Automaton type	complexity	lookahead
deterministic parity	ExpTime-complete	exponential
non-deterministic parity	2ExpTime-complete	doubly-exponential
universal parity	2ExpTime-complete	doubly-exponential
alternating parity	3ExpTime-complete	triply-exponential

Figure 1 Overview of results for the ω-regular case.

repeat the encoding of the number ad infinitum. Then, the automaton can guess and verify the two positions comprising the bad j-pair. Finally, to prevent Player O from incorrectly copying the encoding of j (which manifests itself in a single bit), we use the same marking construction as in the previous section: Player I can mark one position i by a **#** to claim an error in some bit at position $i \bmod n$. The automaton can guess the value $i \bmod n$ and verify that there is no such error (and that the guess was correct). Using similar ideas one can encode an alternating exponential space Turing machine proving the 2ExpTime lower bound on the complexity for non-deterministic automata.

For universal automata, the constructions are even simpler, since we do not need the marking of Player I. Instead, we use the universality to check that Player O copies her pick j correctly. Altogether, we obtain the results presented in Figure 1, where careful analysis shows that the lower bounds already hold for weaker acceptance conditions than parity, e.g., safety and weak parity (the case of reachability acceptance is exceptional, as such games are PSpace-complete for non-deterministic automata [16]).

6 Conclusion

We identified Prompt-LTL as the first quantitative winning condition for delay games that retains the desirable qualities of ω-regular delay games: in particular, bounded lookahead is sufficient to win Prompt-LTL delay games and to determine the winner of such games is 3ExpTime-complete. This complexity should be contrasted to that of delay-free LTL and Prompt-LTL games, which are already 2ExpTime-complete. We complemented the complexity result by giving tight triply-exponential bounds on the necessary lookahead and on the necessary bound k for the prompt eventually operator.

All our lower bounds already hold for LTL and therefore also for (very-weak) alternating Büchi automata, since LTL can be translated into such automata of linear size [10]. On the other hand, we obtained tight matching upper bounds: solving delay games on alternating automata is 3ExpTime-complete and triply-exponential lookahead is in general necessary and always sufficient. Furthermore, our lower bounds can be modified to complete the picture in the ω-regular case with regard to the branching mode of the specification automaton: solving delay games with winning conditions given by non-deterministic or universal automata is 2ExpTime-complete and doubly-exponential lookahead is sufficient and in general necessary.

Finally, as usual for results based on the alternating-color technique, our results on Prompt-LTL hold for the stronger logics PLTL [1], PLDL [6], and their variants with costs [26] as well.

References

1 Rajeev Alur, Kousha Etessami, Salvatore La Torre, and Doron Peled. Parametric temporal logic for "model measuring". *ACM Trans. Comput. Log.*, 2(3):388–407, 2001.

2 Rajeev Alur and Salvatore La Torre. Deterministic generators and games for LTL fragments. *ACM Trans. Comput. Log.*, 5(1):1–25, 2004.

3 Mikołaj Bojańczyk. A bounding quantifier. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *CSL 2004*, volume 3210 of *LNCS*, pages 41–55. Springer, 2004.

4 Rüdiger Ehlers. Symbolic bounded synthesis. *Form. Method. Syst. Des.*, 40(2):232–262, 2012.

5 E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *FOCS 1991*, pages 368–377. IEEE, 1991.

6 Peter Faymonville and Martin Zimmermann. Parametric linear dynamic logic. In Adriano Peron and Carla Piazza, editors, *GandALF 2014*, volume 161 of *EPTCS*, pages 60–73, 2014.

7 Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin. Antichains and compositional algorithms for LTL synthesis. *Form. Method. Syst. Des.*, 39(3):261–296, 2011.

8 Bernd Finkbeiner and Sven Schewe. Bounded synthesis. *STTT*, 15(5-6):519–539, 2013.

9 Wladimir Fridman, Christof Löding, and Martin Zimmermann. Degrees of lookahead in context-free infinite games. In Marc Bezem, editor, *CSL 2011*, volume 12 of *LIPIcs*, pages 264–276. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2011.

10 Paul Gastin and Denis Oddoux. Fast LTL to Büchi automata translation. In Gérard Berry, Hubert Comon, and Alain Finkel, editors, *CAV 2001*, volume 2102 of *LNCS*, pages 53–65. Springer, 2001.

11 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *LNCS*. Springer, 2002.

12 Michael Holtmann, Łukasz Kaiser, and Wolfgang Thomas. Degrees of lookahead in regular infinite games. *LMCS*, 8(3), 2012.

13 Florian Horn, Wolfgang Thomas, Nico Wallmeier, and Martin Zimmermann. Optimal strategy synthesis for request-response games. *RAIRO – Theor. Inf. and Applic.*, 49(3):179–203, 2015.

14 Frederick A. Hosch and Lawrence H. Landweber. Finite delay solutions for sequential conditions. In *ICALP 1972*, pages 45–60, 1972.

15 Felix Klein and Martin Zimmermann. What are strategies in delay games? Borel determinacy for games with lookahead. In Stephan Kreutzer, editor, *CSL 2015*, volume 41 of *LIPIcs*, pages 519–533. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015.

16 Felix Klein and Martin Zimmermann. How much lookahead is needed to win infinite games? *LMCS*, 12(3), 2016.

17 Felix Klein and Martin Zimmermann. Prompt delay. *arXiv*, 1602.05045, 2016.

18 Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. From liveness to promptness. *Form. Method. Syst. Des.*, 34(2):83–103, 2009.

19 Orna Kupferman and Moshe Y. Vardi. Safraless decision procedures. In *FOCS 2005*, pages 531–542. IEEE Computer Society, 2005.

20 Andrzej Mostowski. Games with forbidden positions. Technical Report 78, University of Gdańsk, 1991.

21 Amir Pnueli. The temporal logic of programs. In *FOCS 1977*, pages 46–57. IEEE, 1977.

22 Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *POPL 1989*, pages 179–190. ACM Press, 1989.

23 Amir Pnueli and Roni Rosner. On the synthesis of an asynchronous reactive module. In Giorgio Ausiello, Mariangiola Dezani-Ciancaglini, and Simona Ronchi Della Rocca, editors, *ICALP 1989*, volume 372 of *LNCS*, pages 652–671. Springer, 1989.

24 Sven Schewe. Tighter bounds for the determinisation of Büchi automata. In Luca de Alfaro, editor, *FOSSACS 2009*, volume 5504 of *LNCS*, pages 167–181. Springer, 2009.

25 Martin Zimmermann. Optimal bounds in parametric LTL games. *Theor. Comput. Sci.*, 493:30–45, 2013.

26 Martin Zimmermann. Parameterized linear temporal logics meet costs: Still not costlier than LTL. In Javier Esparza and Enrico Tronci, editors, *GandALF 2015*, volume 193 of *EPTCS*, pages 144–157, 2015.

27 Martin Zimmermann. Delay games with WMSO+U winning conditions. *RAIRO – Theor. Inf. and Applic.*, 2016. To appear.