RWTH Aachen

Department of Computer Science
Technical Report

Parametric LTL Games

Martin Zimmermann

ISSN 0935–3232 · Aachener Informatik-Berichte · AIB-2010-11

RWTH Aachen · Department of Computer Science · June 2010

The publications of the Department of Computer Science of *RWTH Aachen University* are in general accessible through the World Wide Web.

`http://aib.informatik.rwth-aachen.de/`

Parametric LTL Games

Martin Zimmermann*

Lehrstuhl Informatik 7, RWTH Aachen University, Germany
zimmermann@automata.rwth-aachen.de

Abstract. We consider graph games of infinite duration with winning conditions in parameterized linear temporal logic, where the temporal operators are equipped with variables for time bounds. In model checking such specifications were introduced as "PLTL" by Alur et al. and (in a different version called "PROMPT-LTL") by Kupferman et al..
Our work lifts their results on model checking for PLTL and PROMPT-LTL to the level of games: we present algorithms that determine whether a player wins a game with respect to some, infinitely many, or all variable valuations. All these algorithms run in doubly-exponential time; so, adding bounded temporal operators does not increase the complexity compared to solving plain LTL games. Furthermore, we show how to determine optimal valuations that allow a player to win a game.

1 Introduction

Two-player graph games of infinite duration are a tool to synthesize controllers for reactive systems, i.e., systems which have to interact with an (possibly antagonistic) environment. Requirements on the controlled system are typically given by a subset of the system's executions. The controller has to react to the moves of the environment in a way such that the execution satisfies the requirement. The requirements are typically given by acceptance conditions from the theory of automata on infinite words. However, in practice, it is often more convenient to work in a logic framework. A concise way to specify requirements on infinite executions is to use linear temporal logic (LTL). Its advantages include a compact, variable-free syntax and intuitive semantics which makes LTL suitable to be used in applications.

LTL synthesis was implicitly solved by Büchi and Landweber in their seminal work [3] on Church's problem [5], as LTL specifications can be translated into Muller automata. Pnueli and Rosner [10, 11] explicitly considered LTL synthesis for reactive systems and proved the problem to be **2EXPTIME**-complete. Despite this, there are many fragments of LTL for which the synthesis problem has lower complexity [2].

However, LTL lacks capabilities to express timing constraints, which are often desirable in applications. In LTL, a request-response requirement "every request q is followed by a response p" can be expressed by $\mathbf{G}(q \rightarrow \mathbf{F}p)$. Here, one is typically interested in quantitative information, i.e., how long does it take to answer the requests.

To this end, extensions of LTL with timing constraints were introduced. The simplest approach is to add the operator $\mathbf{F}_{\leq k}$ (for some fixed bound $k \in \mathbb{N}$) with the obvious semantics. The request-response requirement is then expressed

* The author's work was supported by the project *Games for Analysis and Synthesis of Interactive Computational Systems (GASICS) of the European Science Foundation.*

by $\mathbf{G}(q \rightarrow \mathbf{F}_{\leq k} p)$ for some suitable k. But finding the right bound k is not practicable: it is generally not known beforehand and depends on the granularity of the model of the system. On the other hand, adding $\mathbf{F}_{\leq k}$ does not increase the expressiveness of LTL, as it can be expressed by a disjunction of nested next-operators.

Therefore, extensions of LTL with variable bounds for model checking were considered. *Parametric Linear Temporal Logic* (PLTL), introduced by Alur et al. [1], adds the operators $\mathbf{F}_{\leq x}$ and $\mathbf{G}_{\leq y}$ to LTL, where x and y are free variables. Satisfaction is defined with respect to a variable valuation α mapping variables to natural numbers: $\mathbf{F}_{\leq x}\varphi$ holds, if φ is satisfied within the next $\alpha(x)$ steps, while $\mathbf{G}_{\leq y}\psi$ holds, if ψ is satisfied for the next $\alpha(y)$ steps. The request-response requirement is now expressed by the formula $\mathbf{G}(q \rightarrow \mathbf{F}_{\leq x} p)$ where the bound x is a free variable. Deciding whether a transition system satisfies a PLTL formula with respect to some, infinitely many, or all variable valuations is **PSPACE**-complete [1] (as is LTL model checking [12]). In the same work, it was shown how to determine optimal variable valuations.

The present paper lifts the results on PLTL to graph-based games: we present algorithms to determine whether a player wins a PLTL game with respect to some, infinitely many, or all variable valuations. For winning conditions with only parameterized eventualities $\mathbf{F}_{\leq x}$ or only parameterized always-operators $\mathbf{G}_{\leq y}$, solving games can be seen as an optimization problem: which is the best variable valuation such that a player wins a given game with respect to that valuation? For several notions of a "best valuation" we show how to find such an optimal valuation and corresponding winning strategies. This continues recent work on time-optimal winning strategies for infinite games with (extensions of) request-response winning conditions [7, 13] and on finitary games [4].

The correctness of the algorithms presented in [1] relies on elaborate pumping arguments which do not seem to be applicable to games. Also, Kupferman et al. [8] argued that the algorithms are too involved and therefore proposed PROMPT–LTL, which can be seen as the fragment of PLTL containing the formulae without parameterized always' and such that parameterized eventualities are all bounded by the same variable. Formally, they add the *prompt-eventually* operator $\mathbf{F_P}$ to LTL. The semantics is defined with respect to a free, but fixed bound k: $\mathbf{F_P}\varphi$ holds, if φ holds within the next k steps. This differs from the semantics of $\mathbf{F}_{\leq k}$, as k is a free variable. The request-response requirement is expressed by $\mathbf{G}(q \rightarrow \mathbf{F_P} p)$. Here, the bound is stated implicitly in the semantics of PROMPT–LTL.

The *alternating-color technique* presented in [8] allows a comprehensive treatment of PROMPT–LTL: it is used to solve the model checking and assume-guarantee model checking problem, as well as the realizability problem, an abstract game given only by a winning condition φ, but without an underlying game graph (similar to the game theoretic formulation of Church's problem [5]).

To obtain our results, we first apply the alternating-color technique to graph-based PROMPT–LTL games, thereby transferring the results on realizability of PROMPT–LTL specifications to this domain. Then, we are able to solve the problems for PLTL, employing the results on PROMPT–LTL games at several points. As model checking can be seen as a one-player game, our results also give a simpler proof of the results on PLTL model checking.

All our algorithms run in doubly-exponential time, which is asymptotically optimal, as solving classical LTL games is **2EXPTIME**-complete. Hence, adding bounded temporal operators to LTL does not increase the complexity of solving games with winning conditions in the extended logics. This confirms similar findings on PLTL model checking.

This paper is structured as follows: Section 2 contains the definitions of the logics and games discussed in the remainder. In Section 3, we describe how to adapt the alternating-color technique to graph-based games. Then, we use this to prove our main results: we show how to solve PLTL games in Section 4; in Section 5 we show how to find optimal strategies for certain games for which a notion of optimality exists. Finally, Section 6 gives a short conclusion and pointers to further research.

2 Definitions

The set of non-negative integers is denoted by $\mathbb{N}$. The powerset of a set S is denoted by 2^S. Throughout this paper let P be a set of *atomic propositions*.

Infinite Games. An *(initialized and labeled) arena* $\mathcal{A} = (V, V_0, V_1, E, v_0, l)$ consists of a finite directed graph (V, E), a partition $\{V_0, V_1\}$ of V denoting the positions of *Player* 0 and *Player* 1, an *initial vertex* $v_0 \in V$, and a *labeling function* $l\colon V \to 2^P$. It is assumed that every vertex has at least one outgoing edge. A *play* $\rho = \rho_0\rho_1\rho_2\ldots$ is an infinite path starting in v_0. The *trace* of ρ is $t(\rho) = l(\rho_0)l(\rho_1)l(\rho_2)\ldots$. A *strategy for Player* i is a mapping $\sigma\colon V^*V_i \to V$ such that $(\rho_n, \sigma(\rho_0\ldots\rho_n)) \in E$ for all play prefixes $\rho_0\ldots\rho_n \in V^*V_i$. A play ρ is *consistent with* σ if $\rho_{n+1} = \sigma(\rho_0,\ldots\rho_n)$ for all $\rho_0\ldots\rho_n \in V^*V_i$.

A *memory structure* $\mathfrak{M} = (M, m_0, \mathrm{upd})$ *for* $\mathcal{A}$ consists of a set M of *memory states*, an *initial memory state* $m_0 \in M$, and an *update function* $\mathrm{upd}\colon M \times V \to M$. This function can be extended to $\mathrm{upd}^*\colon V^* \to M$ by $\mathrm{upd}^*(v_0) = m_0$ and $\mathrm{upd}^*(\rho_0\ldots\rho_n\rho_{n+1}) = \mathrm{upd}(\mathrm{upd}^*(\rho_0\ldots\rho_n), \rho_{n+1})$. A *next-move function for Player* i is a function $\mathrm{nxt}\colon V_i \times M \to V$ which satisfies $(v, \mathrm{nxt}(v, m)) \in E$ for all $v \in V_i$ and all $m \in M$. It induces a *strategy* σ *with memory* $\mathfrak{M}$ via $\sigma(\rho_0\ldots\rho_n) = \mathrm{nxt}(\rho_n, \mathrm{upd}^*(\rho_0\ldots\rho_n))$. A strategy is called *finite-state* if it can be implemented with a finite memory structure. The *size* of $\mathfrak{M}$ (and, slightly abusive, σ) is $|M|$.

Linear Temporal Logics. The formulae of *Linear Temporal Logic* (LTL) are given by the grammar $\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi \mid \varphi\mathbf{R}\varphi$, where $p \in P$. Also, we use the derived operators $\mathbf{tt} := p \vee \neg p$ and $\mathbf{ff} := p \wedge \neg p$ for some fixed $p \in P$, $\mathbf{F}\varphi := \mathbf{tt}\mathbf{U}\varphi$, and $\mathbf{G}\varphi := \mathbf{ff}\mathbf{R}\varphi$. The semantics of LTL are defined in the standard way; for an ω-word $w = w_0w_1w_2\ldots \in \left(2^P\right)^\omega$ and a position $i \in \mathbb{N}$ we write $(w, i) \models \varphi$, if $w_iw_{i+1}w_{i+2}\ldots$ is a model of φ.

Prompt Linear Temporal Logic (PROMPT–LTL) [8] adds the unary operator $\mathbf{F_P}$ to the LTL operators. Here, satisfaction is defined with respect to an ω-word $w \in \left(2^P\right)^\omega$, a position $i \in \mathbb{N}$, and a bound $k \in \mathbb{N}$. For LTL operators, the semantics is independent of k and defined as above. For $\mathbf{F_P}$ we define

- $(w, i, k) \models \mathbf{F_P}\varphi$ iff there exists $j \leq k$ such that $(w, i + j, k) \models \varphi$.

Let $\mathcal{X}$ and $\mathcal{Y}$ be two disjoint sets of *variables*. *Parametric Linear Temporal Logic* (PLTL) [1] adds the unary operators $\mathbf{F}_{\leq x}$ for $x \in \mathcal{X}$ and $\mathbf{G}_{\leq y}$ for $y \in \mathcal{Y}$ to the LTL operators[1]. The set of variables occurring in φ is denoted by $\mathrm{var}(\varphi)$ and defined in the obvious way. The fragments $\mathrm{PLTL}_{\mathbf{F}}$ and $\mathrm{PLTL}_{\mathbf{G}}$ contain the formulae φ with $\mathrm{var}(\varphi) \subseteq \mathcal{X}$ respectively $\mathrm{var}(\varphi) \subseteq \mathcal{Y}$. These formulae are called *unipolar*. The semantics of PLTL is defined with respect to an ω-word $w \in \left(2^P\right)^\omega$, a position $i \in \mathbb{N}$, and a *variable valuation* $\alpha\colon \mathcal{X} \cup \mathcal{Y} \to \mathbb{N}$. For the LTL operators, the semantics is again standard and independent of α. For the parameterized operators, we define

- $(w, i, \alpha) \models \mathbf{F}_{\leq x}\varphi$ iff there exists $j \leq \alpha(x)$ such that $(w, i + j, \alpha) \models \varphi$, and
- $(w, i, \alpha) \models \mathbf{G}_{\leq y}\varphi$ iff for all $j \leq \alpha(y)$: $(w, i + j, \alpha) \models \varphi$.

For all logics considered here, the size $|\varphi|$ of a formula φ is measured by counting the distinct subformulae of φ. This is due to the fact, that the translation of LTL into automata does not distinguish between equal subformulae.

The logics LTL and PLTL (but not the fragments $\mathrm{PLTL}_{\mathbf{F}}$ and $\mathrm{PLTL}_{\mathbf{G}}$) are closed under negation, although we only allow formulae in negation normal form. This is due to the duality of $\mathbf{U}$ and $\mathbf{R}$, and $\mathbf{F}_{\leq x}$ and $\mathbf{G}_{\leq y}$. Thus, we will use $\neg\varphi$ as shorthand for the equivalent formula obtained by pushing the negation to the atomic propositions. Note also that $\varphi \in \mathrm{PLTL}_{\mathbf{F}}$ implies $\neg\varphi \in \mathrm{PLTL}_{\mathbf{G}}$ and $\varphi \in \mathrm{PLTL}_{\mathbf{G}}$ implies $\neg\varphi \in \mathrm{PLTL}_{\mathbf{F}}$.

Remark 1.

(i) For every PROMPT–LTL formula φ and every $k \in \mathbb{N}$, there exists an LTL formula φ_k such that $(w, i, k) \models \varphi$ iff $(w, i) \models \varphi_k$.
(ii) For every PLTL formula φ and every valuation α, there exists an LTL formula φ_α such that $(w, i, \alpha) \models \varphi$ iff $(w, i) \models \varphi_\alpha$.

This can be shown by replacing the bounded operators by disjunctions or conjunctions of nested next-operators. Hence, the size of the formulae φ_k and φ_α is linear in k respectively in $\sum_{z \in \mathrm{var}(\varphi)} \alpha(z)$.

Games with Winning Conditions in Linear Temporal Logics. In Remark 1, we have seen that PROMPT–LTL with respect to a fixed bound k and PLTL with respect to a fixed valuation α are no more expressive than LTL (albeit more succinct). Hence, we will treat k and α as free variables and require Player 0 to play in a way such that there exists a bound k or a valuation α satisfying the winning condition φ. We will introduce three types of games with winning conditions in linear temporal logics, one for each logic we introduced above.

An LTL *game* $\mathcal{G} = (\mathcal{A}, \varphi)$ consists of an arena $\mathcal{A}$ and an LTL formula φ. A play ρ is *won* by Player 0, if $(t(\rho), 0) \models \varphi$, otherwise it is *won* by Player 1. A strategy σ for Player i is a *winning strategy* for her, if every play that is consistent with σ is won by Player i. If Player i has a winning strategy then we say she *wins* $\mathcal{G}$ (and Player $1 - i$ *loses* $\mathcal{G}$).

[1] In [1], the authors also introduced the operators $\mathbf{U}_{\leq x}$, $\mathbf{R}_{\leq y}$, $\mathbf{F}_{>y}$, $\mathbf{G}_{>x}$, $\mathbf{U}_{>y}$, and $\mathbf{R}_{>x}$. However, they showed that all these operators can be expressed using $\mathbf{F}_{\leq x}$ and $\mathbf{G}_{\leq y}$ only, at the cost of a linear increase of the formula's size. Also, we ignore constant bounds as they do not add expressiveness.

Moreover, a PROMPT–LTL *game* $\mathcal{G} = (\mathcal{A}, \varphi)$ consists of an arena $\mathcal{A}$ and a PROMPT–LTL formula φ. Player 0 *wins* $\mathcal{G}$ if there exists a bound k and a strategy σ for her such that $(t(\rho), 0, k) \models \varphi$ for every play ρ that is consistent with σ. Player 1 *wins* $\mathcal{G}$ if for all bounds k he has a strategy τ_k such that $(t(\rho), 0, k) \not\models \varphi$ for every play ρ that is consistent with τ_k. Note that τ_k may depend on k, hence Player 1 does not have a single winning strategy (as Player 0 does), but a family $(\tau_k)_{k\in\mathbb{N}}$ of strategies.

Finally, a PLTL *game* $\mathcal{G} = (\mathcal{A}, \varphi)$ consists of an arena $\mathcal{A}$ and a PLTL formula φ. Player 0 *wins* a play ρ *with respect to a variable valuation* α if $(t(\rho), 0, \alpha) \models \varphi$, otherwise Player 1 *wins* ρ with *respect to* α. A strategy for Player i is a *winning strategy* for her *with respect to* α if every play that is consistent with σ is won by Player i with respect to α. Then, we say that Player i *wins* $\mathcal{G}$ *with respect to* α (and Player $1-i$ *loses* $\mathcal{G}$ *with respect to* α). We define the set $\mathcal{W}^i_\mathcal{G}$ of *winning valuations* for Player i in $\mathcal{G} = (\mathcal{A}, \varphi)$ by

$$\mathcal{W}^i_\mathcal{G} = \{\alpha \mid \text{Player } i \text{ wins } \mathcal{G} \text{ with respect to } \alpha\} \ .$$

Here (and from now on) we assume that α's domain is restricted to the variables occurring in φ. Unipolar, $\text{PLTL}_\mathbf{G}$, and $\text{PLTL}_\mathbf{F}$ games are defined by restricting the winning conditions to unipolar, $\text{PLTL}_\mathbf{G}$, and $\text{PLTL}_\mathbf{F}$ formulae.

While we require Player 0 in a PROMPT–LTL game to play in a way such that there exists a bound on the prompt-eventualities, we define PLTL games with respect to a given valuation. This is done to extend the definitions in [1] respectively [8].

Note that a $\text{PLTL}_\mathbf{F}$ game $\mathcal{G}$ can be translated into a PROMPT–LTL game $\mathcal{G}'$ by replacing every subformula $\mathbf{F}_{\leq x}\psi$ of $\mathcal{G}$ by $\mathbf{F}_\mathbf{P}\psi$. Then, $\alpha \in \mathcal{W}^0_\mathcal{G}$ implies that Player 0 wins $\mathcal{G}'$ with bound $\max_{x\in\text{var}(\varphi)} \alpha(x)$. Dually, if Player 0 wins $\mathcal{G}'$ with bound k, then $\mathcal{W}^0_\mathcal{G}$ contains the valuation that maps every variable to k. Conversely, every PROMPT–LTL game $\mathcal{G}$ can be translated into a $\text{PLTL}_\mathbf{F}$ game $\mathcal{G}'$ such that Player 0 wins $\mathcal{G}$ iff $\mathcal{W}^0_{\mathcal{G}'} \neq \emptyset$.

It is easy to verify that for all types of games introduced above, there is at most one player who wins a given game. A game is *determined*, if one of the players wins it.

Proposition 1. LTL *games (and therefore also* PLTL *games with respect to a fixed variable valuation and also* PROMPT–LTL*-games) are determined with finite-state strategies, i.e., for every* LTL *game* $\mathcal{G}$*, one of the players has a finite-state winning strategy for* $\mathcal{G}$*. Determining the winner is* **2EXPTIME***-complete [11] and finite-state winning strategies can be computed in doubly-exponential time.*

Note that Player 1 has a family $(\tau_k)_{k\in\mathbb{N}}$ of finite-state strategies τ_k, if he wins a PROMPT–LTL-game.

3 Solving PROMPT–LTL Games

In this section, we discuss PROMPT–LTL games. In [8], a solution to the realizability problem for specifications in PROMPT–LTL is presented. Realizability

asks whether Player 1 has a winning strategy in an abstract game without underlying arena. The players alternatingly pick propositions, Player 0 from a set I of inputs and Player 1 from a set O of outputs. Hence, by picking $i_n \subseteq I$ respectively $o_n \subseteq O$ they form an infinite word $(i_0 \cup o_0)(i_1 \cup o_1)(i_2 \cup o_2)\ldots$, which is winning for Player 1 if it satisfies a given PROMPT–LTL formula φ over $I \cup O$. A strategy for Player 1 is a mapping $\left(2^I\right)^* \to 2^O$. In [8], it is shown that the problem of deciding whether Player 1 has a winning strategy for a PROMPT–LTL specification can be reduced to the same problem for LTL specifications. As the reduction increases the size of the formula only linearly, PROMPT–LTL realizability is **2EXPTIME**-complete, as is LTL realizability [11].

In the following, we will rephrase this result in the setting of graph-based games, which is conceptually simpler than reducing graph-based games to the realizability problem. Furthermore, we will see that the size of a finite-state winning strategy for the LTL game will induce a bound on the waiting times for the prompt-eventualities in the original game. This will be used to solve PLTL games and to find optimal winning strategies for them.

In [8], the realizability problem for PROMPT–LTL specifications (among other problems) is solved by the *alternating-color technique*: let $p \notin P$ be a fixed proposition. An ω-word $w' = w'_0 w'_1 w'_2 \ldots \in \left(2^{P \cup \{p\}}\right)^\omega$ is a *p-coloring* of $w = w_0 w_1 w_2 \ldots \in \left(2^P\right)^\omega$ if $w'_n \cap P = w_n$, i.e., w_n and w'_n coincide on all propositions in P. The additional proposition p can be thought of as the color of w'_n: we say that position n is *green* if $p \in w'_n$, and say that it is *red* if $p \notin w'_n$. Given $k \in \mathbb{N}$ we say that w' is *k-spaced*, if the colors in w' change infinitely often, but not twice in any infix of length k. Dually, w' is *k-bounded*, if the colors change at least once in every infix of length $k+1$. See [8] for more details.

Let $alt_p := \mathbf{GF}p \wedge \mathbf{GF}\neg p$. It is satisfied if the colors change infinitely often. Given a PROMPT–LTL formula φ let $rel(\varphi)$ denote the LTL formula obtained by inductively replacing every subformula $\mathbf{F_P}\psi$ by $(p \rightarrow (p\mathbf{U}(\neg p\mathbf{U}\psi))) \wedge (\neg p \rightarrow (\neg p\mathbf{U}(p\mathbf{U}\psi)))$. Finally, given a PROMPT–LTL formula φ, define $c(\varphi) := alt_p \wedge rel(\varphi)$. It forces a coloring to have infinitely many blocks and every subformula $\mathbf{F_P}\psi$ to be satisfied within two consecutive blocks. We have $|c(\varphi)| \leq 7(|\varphi| + 1)$, as we count the number of distinct subformulae.

Lemma 1 ([8]). *Let φ be a* PROMPT–LTL *formula, $w \in \left(2^P\right)^\omega$, and $k \in \mathbb{N}$.*

(i) If $(w, 0, k) \models \varphi$, then $(w', 0) \models c(\varphi)$ for every k-spaced p-coloring w' of w.
(ii) If w' is a k-bounded p-coloring of w with $(w', 0) \models c(\varphi)$, then $(w, 0, 2k) \models \varphi$.

The previous lemma is the key to solving PROMPT–LTL games: we will transform the original arena $\mathcal{A}$ into an arena $\mathcal{A}'$ in which Player 0 produces p-colorings of the plays of the original arena, i.e., $\mathcal{A}'$ will consist of two disjoint copies of $\mathcal{A}$, one labeled with p, the other one not. Assume a play is in vertex v in one component. Then, the player whose turn it is at v chooses a successor v' of v and Player 0 picks a component. The play then continues in this component's vertex v'. We split this into two sequential moves: first, the player whose turn it is chooses a successor and then Player 0 chooses the component. Thus, we have to introduce a new vertex for every edge of $\mathcal{A}$ which allows Player 0 to choose the component.

Formally, given an arena $\mathcal{A} = (V, V_0, V_1, E, v_0, l)$, define the expanded arena $\mathcal{A}' := (V', V'_0, V'_1, E', v'_0, l')$ by

- $V' = V \times \{0,1\} \cup E$,
- $V'_0 = V_0 \times \{0,1\} \cup E$,
- $V'_1 = V_1 \times \{0,1\}$,
- $E' = \{((v,0),e),((v,1),e),(e,(v',0)),(e,(v',1)) \mid e = (v,v') \in E\}$,
- $v'_0 = (v_0,0)$,
- $l'(e) = \emptyset$ for all $e \in E$ and $l'(v,b) = \begin{cases} l(v) \cup \{p\} & \text{if } b = 0, \\ l(v) & \text{if } b = 1. \end{cases}$

Note that $\mathcal{A}'$ is bipartite with partition $\{V \times \{0,1\}, E\}$ and every play has the form $(\rho_0,b_0)e_0(\rho_1,b_1)e_1(\rho_2,b_2)\ldots$ where $\rho_0\rho_1\rho_2\ldots$ is a play in $\mathcal{A}$, $e_n = (\rho_n,\rho_{n+1})$, and the b_n are in $\{0,1\}$. Also, we have $|\mathcal{A}'| \leq 2|\mathcal{A}| + |\mathcal{A}|^2$.

This construction necessitates a modification of the semantics of the game: only every other vertex is significant when it comes to determining the winner of a play in $\mathcal{A}'$, the choice vertices have to be ignored. This motivates *blinking semantics* for LTL games. Let $\mathcal{G} = (\mathcal{A},\varphi)$ be an LTL game and $\rho = \rho_0\rho_1\rho_2\ldots$ be a play. Player 0 wins ρ with blinking semantics if $(t(\rho_0\rho_2\rho_4\ldots),0) \models \varphi$. Analogously, Player 1 wins ρ with blinking semantics if $(t(\rho_0\rho_2\rho_4\ldots),0) \not\models \varphi$. Winning strategies and winning $\mathcal{G}$ with blinking semantics is defined in the obvious way. The standard automata theoretic construction for solving LTL games can be easily adapted to account for blinking semantics: let $f(n) := 2((n^2+n)2^n)^{(n^2+n)2^n}((n^2+n)2^n)!$.

Lemma 2. LTL *games* $\mathcal{G} = (\mathcal{A},\varphi)$ *with blinking semantics are determined with finite-state strategies of size* $f(|\varphi|)$.

Proof. From φ, we construct a non-deterministic Büchi Automaton $\mathfrak{A}_\varphi$ that accepts exactly those words over 2^P which satisfy φ. Then, every transition $t = (q,a,q')$ in $\mathfrak{A}_\varphi$ is split into two transitions (q,a,q_t) and $(q_t,\bot,q')$ using a new state q_t and a dummy symbol $\bot$. The new automaton $\mathfrak{A}^b_\varphi$ accepts exactly those words $a_0\bot a_1\bot a_2\ldots$ such that $a_0a_1a_2\ldots$ is accepted by $\mathfrak{A}_\varphi$. Finally, we determinize $\mathfrak{A}^b_\varphi$ into a deterministic parity automaton, which can be turned into a memory structure of same size for $\mathcal{G}$ with blinking semantics. If the constructions presented in [6] and [9] are used, then $f(|\varphi|)$ bounds the size of the memory. □

Now, we are able to solve PROMPT–LTL games by a reduction to LTL games with blinking semantics using the alternating-color technique.

Theorem 1. *Let* $\mathcal{G} = (\mathcal{A},\varphi)$ *be a* PROMPT–LTL *game. Player* 0 *wins* $\mathcal{G}$ *iff she wins* $(\mathcal{A}', c(\varphi))$ *with blinking semantics.*

Proof. Let Player 0 win $\mathcal{G}$ with strategy σ and bound k. Define the strategy σ' for $\mathcal{A}'$ as follows:

- $\sigma'((\rho_0,b_0)e_0(\rho_1,b_1)e_1\ldots(\rho_n,b_n)) := (\rho_n,\sigma(\rho_0\ldots\rho_n))$, if $\rho_n \in V_0$, and
- $\sigma'((\rho_0,b_0)e_0(\rho_1,b_1)e_1\ldots(\rho_n,b_n)e_n) := \begin{cases} (\rho_{n+1},0) & \text{if } n \mod 2k < k, \\ (\rho_{n+1},1) & \text{if } n \mod 2k \geq k, \end{cases}$

where $e_n = (\rho_n,\rho_{n+1})$. Let $\rho = \rho_0\rho_1\rho_2\ldots$ be a play that is consistent with σ' and let $\rho' := \rho_0\rho_2\rho_4\ldots = (v_0,b_0)(v_1,b_1)(v_2,b_2)\ldots$. The sequence $v_0v_2v_4\ldots$ is a play of $\mathcal{A}$ that is consistent with σ. Hence, $(t(v_0v_2v_4\ldots),0,k) \models \varphi$. Also, $t(\rho')$

is a k-tight p-coloring of $t(v_0v_2v_4\dots)$. Hence, $(t(\rho'),0) \models c(\varphi)$ due to Lemma 1. Thus, σ' is a winning strategy for $\mathcal{G}'$ with blinking semantics.

For the other direction assume that Player 0 wins $(\mathcal{A}', c(\varphi))$ with blinking semantics. Then, she also has a finite-state winning strategy σ induced by some memory structure $\mathfrak{M} = (M, m_0, \mathrm{upd})$ and some next-move function nxt. We will construct a strategy for $\mathcal{A}$ by simulating a play in $\mathcal{A}'$. Therefore, we need to keep track of the component of $\mathcal{A}'$ the simulated play is in and the current memory state of $\mathfrak{M}$. Then, a move in $\mathcal{A}$ is simulated by two moves in $\mathcal{A}'$: the choice of the new vertex and the choice of the component. Hence, we transform $\mathfrak{M}$ into $\mathfrak{M}' := (M', m_0', \mathrm{upd}')$ for $\mathcal{A}$ where

- $M' = (V \times \{0,1\}) \times M$,
- $m_0' = ((v_0, 0), m_0)$,
- $\mathrm{upd}'(((v,b),m),v') = (\mathrm{nxt}(e, \mathrm{upd}(m,e)), \mathrm{upd}(\mathrm{upd}(m,e), \mathrm{nxt}(e, \mathrm{upd}(m,e))))$ where $e = (v, v')$.

Finally, we have to define a next-move function $\mathrm{nxt}' \colon V_0 \times M' \to V$ for Player 0: if $\mathrm{nxt}((v',b),m) = (v', v'')$, then $\mathrm{nxt}'(v, ((v',b),m)) := v''$. It remains to show that σ' induced by $\mathfrak{M}'$ and nxt' is a winning strategy for Player 0 for $\mathcal{G}$. We begin by relating plays consistent with σ' to plays that are consistent with σ, i.e., proving that the simulation is working correctly.

Lemma 3. *Let $\rho_0\rho_1\rho_2\dots$ be a play in $\mathcal{A}$ that is consistent with σ'. Then, there exist bits $b_0, b_1, b_2, \dots \in \{0,1\}$ such that*

(i) *$(\rho_0,b_0)(\rho_0,\rho_1)(\rho_1,b_1)(\rho_1,\rho_2)(\rho_2,b_2)\dots$ is a play in $\mathcal{A}'$ that is consistent with σ, and*

(ii) *if $\mathrm{upd}^*((\rho_0,b_0)(\rho_0,\rho_1)(\rho_1,b_1)\dots(\rho_{n-1},\rho_n)(\rho_n,b_n)) = m$, then $\mathrm{upd}'^*(\rho_0\dots\rho_n) = ((\rho_n,b_n),m)$.*

Proof. By induction over $\rho_0\dots\rho_n$: as a play in $\mathcal{A}$ always starts in v_0, both claims are true for $\rho_0 = v_0$ with $b_0 = 0$, since $v_0' = (v_0,0)$ and $\mathrm{upd}'^*(v_0) = m_0' = ((v_0,0),m_0)$. Now, let $\rho_0\dots\rho_n\rho_{n+1}$ be a play prefix in $\mathcal{A}$. The induction hypothesis gives us bits $b_0,\dots,b_n$ such that $(\rho_0,b_0)(\rho_0,\rho_1)(\rho_1,b_1)\dots(\rho_{n-1},\rho_n)(\rho_n,b_n)$ is a play in $\mathcal{A}'$ that is consistent with σ and

$$\mathrm{upd}'^*(\rho_0\dots\rho_n) = ((\rho_n,b_n),m) \ , \tag{1}$$

where $m := \mathrm{upd}^*((\rho_0,b_0)(\rho_0,\rho_1)(\rho_1,b_1)\dots(\rho_{n-1},\rho_n)(\rho_n,b_n))$. We consider two cases depending on whose turn it is at ρ_n:

If $\rho_n \in V_0$ let

$$\mathrm{nxt}((\rho_n,b_n),m) = (\rho_n,v) =: e \ , \tag{2}$$

$m' := \mathrm{upd}(m,e)$, and $(v,b) := \mathrm{nxt}(e,m')$. Then,

$$(\rho_0,b_0)(\rho_0,\rho_1)(\rho_1,b_1)\dots(\rho_{n-1},\rho_n)(\rho_n,b_n)e(v,b)$$

is consistent with σ, that is $b_{n+1} := b$.

We have to show $v = \rho_{n+1}$: as $\rho_0\dots\rho_n\rho_{n+1}$ is consistent with σ, we have

$$\rho_{n+1} = \mathrm{nxt}'(\rho_n, \mathrm{upd}'^*(\rho_0\dots\rho_n)) = \mathrm{nxt}'(\rho_n, ((\rho_n,b_n),m))$$

by 1. Also, $\mathrm{nxt}'(\rho_n, ((\rho_n, b_n), m)) = \overline{v}$, if $\mathrm{nxt}((\rho_n, b_n), m) = (\rho_n, \overline{v})$. Applying 2 we obtain $v = \overline{v} = \rho_{n+1}$. Hence, it remains to prove $\mathrm{upd}'(((\rho_n, b_n), m), \rho_{n+1}) = ((\rho_{n+1}, b_{n+1}), m'')$ where $m'' := \mathrm{upd}(m', (\rho_{n+1}, b_{n+1}))$. We have

$$\begin{aligned}
&\mathrm{upd}'(((\rho_n, b_n), m), \rho_{n+1}) \\
=&(\mathrm{nxt}(e, \mathrm{upd}(m, e)), \mathrm{upd}(\mathrm{upd}(m, e), \mathrm{nxt}(e, \mathrm{upd}(m, e)))) \\
=&(\mathrm{nxt}(e, m'), \mathrm{upd}(m', \mathrm{nxt}(e, m'))) \\
=&((v, b), \mathrm{upd}(m', (v, b))) \\
=&((\rho_{n+1}, b_{n+1}), m'') \ .
\end{aligned}$$

If $\rho_n \in V_1$, let $e := (\rho_n, \rho_{n+1})$, $m' := \mathrm{upd}(m, e)$, and $(\rho_{n+1}, b) := \mathrm{nxt}(e, m')$. Then,

$$(\rho_0, b_0)(\rho_0, \rho_1)(\rho_1, b_1) \dots (\rho_{n-1}, \rho_n)(\rho_n, b_n)e(v, b)$$

is consistent with σ, that is $b_{n+1} := b$.

Now, it remains to show $\mathrm{upd}'(((\rho_n, b_n), m), \rho_{n+1}) = ((\rho_{n+1}, b_{n+1}), m'')$ where $m'' := \mathrm{upd}(m', (\rho_{n+1}, b_{n+1}))$. However, the reasoning is analogous to the one for $\rho_n \in V_0$. □

Let $\rho = \rho_0\rho_1\rho_2 \dots$ be a play consistent with σ'. Due to Lemma 3 there exist bits $b_0, b_1, b_2 \dots \in \{0, 1\}$ such that $\rho' = (\rho_0, b_0)(\rho_0, \rho_1)(\rho_1, b_1)(\rho_1, \rho_2)(\rho_2, b_2) \dots$ is a play consistent with σ. Hence, the trace of $\rho'' = (\rho_0, b_0)(\rho_1, b_1)(\rho_2, b_2) \dots$ satisfies $c(\varphi)$. We claim that $t(\rho'')$ is k-bounded, where $k := |V| \cdot |M| + 1$. Then, we apply Lemma 1 and obtain that $(t(\rho), 0, 2k) \models \varphi$, as $t(\rho'')$ is a k-bounded p-coloring of $t(\rho)$.

Suppose ρ'' is not k-bounded. Then, there exist adjacent change-points i and j such that $j - i > k + 1$. Then, there also exist $i \leq i' < j' \leq j$ such that $\rho_{i'} = \rho_{j'}$ and $\mathrm{upd}^*((\rho_0, b_0)(\rho_0, \rho_1) \dots (\rho_{i'}, b_{i'})) = \mathrm{upd}^*((\rho_0, b_0)(\rho_0, \rho_1) \dots (\rho_{j'}, b_{j'}))$, i.e., the last vertices of both play prefixes are equal and the memory states after both play prefixes are equal, too. Hence, the play

$$\rho^* := (\rho_0, b_0)(\rho_0, \rho_1) \dots (\rho_{i'-1}, b_{i'-1})\big[(\rho_{i'}, b_{i'}) \dots (\rho_{j'-1}, b_{j-1'})\big]^\omega$$

is consistent with σ. Remember that the bits do not change between i and j. Thus, the colors do not change infinitely often in ρ^*. Hence, it does not satisfy $c(\varphi)$, which contradicts the fact that σ is a winning strategy for $(\mathcal{A}', c(\varphi))$. □

Corollary 1. *If Player* 0 *wins a* PROMPT–LTL *game* $(\mathcal{A}, \varphi)$*, then she also has a finite-state winning strategy of size* $2|\mathcal{A}|f(|c(\varphi)|)$ *which is winning for* $k = 2(|\mathcal{A}|f(|c(\varphi)|) + 1)$.

The next corollary is concerned with the computational complexity of determining the winner of a PROMPT–LTL game.

Corollary 2. *Solving* PROMPT–LTL *games is* **2EXPTIME**-*complete.*

Proof. Hardness follows directly from the **2EXPTIME**-completeness of solving LTL games. Membership in **2EXPTIME** is witnessed by the reduction presented above (which increases the size of the arena and the formula only polynomially) and the fact that solving LTL games with blinking semantics can be done in doubly-exponential time. □

4 Solving PLTL Games

Remember that $\mathcal{W}^i_\mathcal{G}$ contains the variable valuations α (restricted to the variables occurring in the winning condition of $\mathcal{G}$) that allow Player i to win $\mathcal{G}$ with respect to α. We are interested in the following problems:

Membership: Given a PLTL game $\mathcal{G}$ and a valuation α, does $\alpha \in \mathcal{W}^i_\mathcal{G}$ hold?
Emptiness: Given a PLTL game $\mathcal{G}$, is $\mathcal{W}^i_\mathcal{G}$ empty?
Finiteness: Given a PLTL game $\mathcal{G}$, is $\mathcal{W}^i_\mathcal{G}$ finite?
Universality: Given a PLTL game $\mathcal{G}$, does $\mathcal{W}^i_\mathcal{G}$ contain all variable valuations?

Let $\mathcal{G} = (\mathcal{A}, \varphi)$ be a PLTL game and α a variable valuation. Applying Remark 1 to games, we obtain: Player i wins $\mathcal{G}$ with respect to α iff she wins the LTL game $(\mathcal{A}, \varphi_\alpha)$. The winner and a finite-state winning strategy can be effectively computed. Hence, we obtain our first result about PLTL games.

Theorem 2. *The membership problem for $\mathcal{W}^i_\mathcal{G}$ is decidable.*

To solve the other problems, we make use of the duality of unipolar games and the duality of the emptiness and universality problem. For an arena $\mathcal{A} = (V, V_0, V_1, E, v_0, l)$, let $\overline{\mathcal{A}} := (V, V_1, V_0, E, v_0, l)$ be its *dual arena*, where the two players swap their positions. Given a PLTL game $\mathcal{G} = (\mathcal{A}, \varphi)$, the *dual game* is $\overline{\mathcal{G}} := (\overline{\mathcal{A}}, \neg\varphi)$. The dual game of a $\text{PLTL}_\mathbf{G}$ game is a $\text{PLTL}_\mathbf{F}$ game and vice versa.

Remark 2. Let α be a valuation and $\mathcal{G}$ a PLTL game. Player i wins $\mathcal{G}$ with respect to α iff Player $1-i$ wins $\overline{\mathcal{G}}$ with respect to α.

The sets $\mathcal{W}^i_\mathcal{G}$ enjoy two types of dualities, which we rely on in the following. The first one is due to determinacy of LTL games, the second one due to Remark 2.

Lemma 4. *Let $\mathcal{G}$ be a* PLTL *game. Then*

(i) $\mathcal{W}^0_\mathcal{G}$ is the complement of $\mathcal{W}^1_\mathcal{G}$.
(ii) $\mathcal{W}^i_\mathcal{G} = \mathcal{W}^{1-i}_{\overline{\mathcal{G}}}$.

Another property that is used in the following is the monotonicity of the parameterized operators: let $\alpha(x) \leq \beta(x)$ and $\alpha(y) \geq \beta(y)$. Then, $(w, i, \alpha) \models \mathbf{F}_{\leq x}\varphi$ implies $(w, i, \beta) \models \mathbf{F}_{\leq x}\varphi$ and $(w, i, \alpha) \models \mathbf{G}_{\leq y}\varphi$ implies $(w, i, \beta) \models \mathbf{G}_{\leq y}\varphi$. Hence, the set $\mathcal{W}^0_\mathcal{G}$ is upwards-closed if $\mathcal{G}$ is a $\text{PLTL}_\mathbf{F}$ game, and downwards-closed if $\mathcal{G}$ is a $\text{PLTL}_\mathbf{G}$ game (where valuations are compared componentwise).

In the remainder, we solve the emptiness, finiteness, and universality problem for PLTL games. We begin by considering unipolar games and then reduce the problems for PLTL games to problems for unipolar games.

Lemma 5. *Let $\mathcal{G}_\mathbf{F} = (\mathcal{A}_\mathbf{F}, \varphi_\mathbf{F})$ be a $\text{PLTL}_\mathbf{F}$ game, let $\mathcal{G}_\mathbf{G} = (\mathcal{A}_\mathbf{G}, \varphi_\mathbf{G})$ be a $\text{PLTL}_\mathbf{G}$ game, and let $i \in \{0, 1\}$. The emptiness, finiteness, and universality problems for $\mathcal{W}^i_{\mathcal{G}_\mathbf{F}}$ and $\mathcal{W}^i_{\mathcal{G}_\mathbf{G}}$ are decidable.*

Proof. We have already established in Lemma 4 (ii) that it suffices to consider $i = 0$. In the following, let α_0 be the valuation that maps every variable to zero.

Emptiness of $\mathcal{W}^0_{\mathcal{G}_\mathbf{F}}$: The emptiness problem can be decided by a reduction to PROMPT–LTL games. Let φ' be the PROMPT–LTL formula obtained from $\varphi_\mathbf{F}$

by recursively replacing every parameterized subformula $\mathbf{F}_{\leq x}\psi$ by $\mathbf{F}_{\mathbf{P}}\psi$. Player 0 wins $(\mathcal{A}_{\mathbf{F}}, \varphi')$ iff $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}} \neq \emptyset$. The former can be decided by Theorem 1.

Finiteness of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$: $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$ is finite iff it is empty, due to upwards-closure of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$.

Universality of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$: $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$ is universal iff Player 0 wins $\mathcal{G}_{\mathbf{F}}$ with respect to α_0, again due to upwards-closure of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$. Now, apply Theorem 2.

Emptiness of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$: $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is non-empty iff Player 0 wins $\mathcal{G}_{\mathbf{G}}$ with respect to α_0, due to downwards-closure of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$. Now, apply Theorem 2.

Universality of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$: The universality problem can be reduced to the emptiness problem for $\mathrm{PLTL}_{\mathbf{F}}$ games. Applying Lemma 4 yields: $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is universal iff $\mathcal{W}^1_{\mathcal{G}_{\mathbf{G}}}$ is universal iff $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is empty. The latter problem is decidable, as shown above.

Finiteness of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$: The finiteness problem can be reduced to the universality problem for a (simpler) $\mathrm{PLTL}_{\mathbf{G}}$ game. We assume that $\varphi_{\mathbf{G}}$ has at least one temporal operator parameterized with a variable, since the problem is trivial otherwise. The set $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is infinite iff there is a variable $y \in \mathrm{var}(\varphi_{\mathbf{G}})$ such that y is mapped to infinitely many values by the valuations in $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$. By downwards-closure we can assume that all other variables are mapped to zero. Furthermore, y is mapped to infinitely many values iff it is mapped to all possible values, again by downwards-closure. To combine this, we define φ_y to be the formula obtained from $\varphi_{\mathbf{G}}$ by inductively replacing every subformula $\mathbf{G}_{\leq z}\psi$ for $z \neq y$ by ψ and define $\mathcal{G}_y := (\mathcal{A}_{\mathbf{G}}, \varphi_y)$. Then, $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is infinite, iff there exists some variable $y \in \mathrm{var}(\varphi_{\mathbf{G}})$ such that $\mathcal{W}^0_{\mathcal{G}_y}$ is universal. So, deciding whether $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is infinite can be done by solving $|var(\varphi_{\mathbf{G}})|$ many universality problems for $\mathrm{PLTL}_{\mathbf{G}}$ games, which were discussed above. □

We are now able to state and prove the main result of this section.

Theorem 3. *Let $\mathcal{G} = (\mathcal{A}, \varphi)$ be a* PLTL *game and $i \in \{0, 1\}$. The emptiness, finiteness, and universality problems for $\mathcal{W}^i_{\mathcal{G}}$ are decidable.*

Proof. Again, due to Lemma 4 (ii) it suffices to consider $i = 0$.

Emptiness of $\mathcal{W}^0_{\mathcal{G}}$: Let $\varphi_{\mathbf{F}}$ be the formula obtained from φ by inductively replacing every subformula $\mathbf{G}_{\leq y}\psi$ by ψ, and let $\mathcal{G}_{\mathbf{F}} := (\mathcal{A}, \varphi_{\mathbf{F}})$. Applying downwards-closure, we obtain that $\mathcal{W}^0_{\mathcal{G}}$ is empty iff $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$ is empty. As $\mathcal{G}_{\mathbf{F}}$ is a $\mathrm{PLTL}_{\mathbf{F}}$ game, we can decide the emptiness of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$ by applying Lemma 5.

Finiteness of $\mathcal{W}^0_{\mathcal{G}}$: If φ contains at least one $\mathbf{F}_{\leq x}$, then $\mathcal{W}^0_{\mathcal{G}}$ is infinite, iff it is non-empty, due to monotonicity of $\mathbf{F}_{\leq x}$. The emptiness of $\mathcal{W}^0_{\mathcal{G}}$ can be decided as discussed above. Otherwise, $\mathcal{G}$ is a $\mathrm{PLTL}_{\mathbf{G}}$ game whose finiteness problem can be decided by Lemma 5.

Universality of $\mathcal{W}^0_{\mathcal{G}}$: Let $\varphi_{\mathbf{G}}$ be the formula obtained from φ by inductively replacing every subformula $\mathbf{F}_{\leq x}\psi$ by ψ, and let $\mathcal{G}_{\mathbf{G}} := (\mathcal{A}, \varphi_{\mathbf{G}})$. Applying upwards-closure, we obtain that $\mathcal{W}^0_{\mathcal{G}}$ is universal iff $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ is universal. As $\mathcal{G}_{\mathbf{G}}$ is a $\mathrm{PLTL}_{\mathbf{G}}$ game, we can decide the universality of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$ due to Lemma 5. □

Corollary 3. *The emptiness, finiteness, and universality problem for* PLTL *games are* **2EXPTIME***-complete.*

Proof. Hardness follows directly from **2EXPTIME**-completeness of solving LTL games employing simple reductions to the problems considered here. Membership in **2EXPTIME** is witnessed by the linear-time reductions to LTL games discussed above.

5 Optimal Winning Strategies for unipolar PLTL Games

For unipolar games, it makes sense to view synthesis of winning strategies as an optimization problem: which is the *best* variable valuation α such that Player 0 can win with respect to α? We consider two quality measures for a valuation α for φ: the maximal parameter $\max_{z \in \mathrm{var}(\varphi)} \alpha(z)$ and the minimal parameter $\min_{z \in \mathrm{var}(\varphi)} \alpha(z)$. For a PLTL$_{\mathbf{F}}$ game, Player 0 tries to minimize the waiting times. Hence, we are interested in minimizing the minimal or maximal parameter. Dually, for PLTL$_{\mathbf{G}}$ games, we are interested in maximizing the quality measures. Again, we will only consider Player 0 as one can dualize the game to obtain similar results for Player 1. The main result of this section states that all these optimization problems can be solved effectively.

Theorem 4. *Let $\mathcal{G}_{\mathbf{F}} = (\mathcal{A}_{\mathbf{F}}, \varphi_{\mathbf{F}})$ be a* PLTL$_{\mathbf{F}}$ *game and $\mathcal{G}_{\mathbf{G}} = (\mathcal{A}_{\mathbf{G}}, \varphi_{\mathbf{G}})$ be a* PLTL$_{\mathbf{G}}$ *game. Then, the following values (and winning strategies realizing them) can be computed.*

(i) $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}} \min_{x \in \mathrm{var}(\varphi_{\mathbf{F}})} \alpha(x)$.
(ii) $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}} \max_{x \in \mathrm{var}(\varphi_{\mathbf{F}})} \alpha(x)$.
(iii) $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}} \max_{y \in \mathrm{var}(\varphi_{\mathbf{G}})} \alpha(y)$.
(iv) $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}} \min_{y \in \mathrm{var}(\varphi_{\mathbf{G}})} \alpha(y)$.

Proof. In the following, we assume that the formula under consideration contains at least one variable and that there is at least one variable valuation that lets Player 0 win the game under consideration.

We begin by considering formulae with a single variable: given a PLTL$_{\mathbf{F}}$ game $\mathcal{G} = (\mathcal{A}, \varphi)$ with $\mathrm{var}(\varphi) = \{x\}$, determine $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}}} \alpha(x)$. By replacing every subformula $\mathbf{F}_{\leq x}\psi$ of φ by $\mathbf{F}_{\mathbf{P}}\psi$, we obtain a PROMPT–LTL game $(\mathcal{A}, \varphi')$ which is won by Player 0 as we assume that $\mathcal{W}^0_{\mathcal{G}}$ is non-empty. Hence, applying Corollary 1 yields that Player 0 wins $(\mathcal{A}, \varphi')$ with bound $k := 2(|\mathcal{A}| f(|c(\varphi')|) + 1)$. Hence, Player 0 also wins the PLTL$_{\mathbf{F}}$ game $\mathcal{G}$ with respect to the valuation which maps x to k. Hence, $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}}} \alpha(x) \leq k$. Now, the minimal value can be found by binary search and Proposition 1.

Dually, given a PLTL$_{\mathbf{G}}$ game $\mathcal{G} = (\mathcal{A}, \varphi)$ with $\mathrm{var}(\varphi) = \{y\}$, determine $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}}} \alpha(y)$. If $\mathcal{W}^0_{\mathcal{G}}$ is universal, then $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}}} \alpha(y) = \infty$. Otherwise, consider the dual game $\overline{\mathcal{G}} = (\overline{\mathcal{A}}, \neg\varphi)$ which is a PLTL$_{\mathbf{F}}$ game with a single variable y as well. Replacing every subformula $\mathbf{F}_{\leq y}\psi$ of $\neg\varphi$ by $\mathbf{F}_{\mathbf{P}}\psi$, we obtain a PROMPT–LTL game $\mathcal{G}' := (\overline{\mathcal{A}}, \varphi')$ which is won by Player 0. Applying Corollary 1, we obtain that Player 0 wins $\mathcal{G}'$ with bound $k := 2(|\mathcal{A}| f(|c(\varphi')|) + 1)$, which implies that the valuation mapping y to k is in $\mathcal{W}^0_{\overline{\mathcal{G}}}$ and not in $\mathcal{W}^0_{\mathcal{G}} = \mathcal{W}^1_{\overline{\mathcal{G}}}$. As $\mathcal{W}^0_{\mathcal{G}}$ is downwards-closed, it cannot contain a variable valuation mapping y to a value larger than k. Hence, we have only a finite number of possible values

for $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}}} \alpha(y)$, and the maximal value can be found by binary search and Proposition 1.

Now, we are able to solve the optimization problems for winning conditions with more than one variable. The first case is exceptional, as it is the only one that does not involve a reduction to games with just one variable.

(i) Corollary 1 yields that Player 0 wins the PROMPT–LTL game $(\mathcal{A}_{\mathbf{F}}, \varphi')$ with bound $k := 2(|\mathcal{A}_{\mathbf{F}}| f(|c(\varphi')|) + 1)$, where φ' is obtained by replacing every subformula $\mathbf{F}_{\leq x}\psi$ of $\varphi_{\mathbf{F}}$ by $\mathbf{F}_{\mathbf{P}}\psi$. Thus, $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}} \min_{x \in \mathrm{var}(\varphi_{\mathbf{F}})} \alpha(x) \leq k$. Let $\mathcal{G}_{x,n}$ be obtained from $\mathcal{G}_{\mathbf{F}}$ by replacing every subformula $\mathbf{F}_{\leq x}\psi$ parameterized by x of the winning condition $\varphi_{\mathbf{F}}$ by $\mathbf{X}^n_{\vee}\psi$, where $\mathbf{X}^0_{\vee}\psi := \psi$ and $\mathbf{X}^{m+1}_{\vee}\psi := \psi \vee (\mathbf{X}^m_{\vee}\psi)$. To determine whether the minimum is even smaller than k, the smallest n such that $\mathcal{W}^0_{\mathcal{G}_{x,n}}$ is non-empty for some x, has to be found. It is equal to $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}} \min_{x \in \mathrm{var}(\varphi_{\mathbf{F}})} \alpha(x)$ and can be found by binary search in the interval $[0, k-1]$ for every variable $x \in \mathrm{var}(\varphi)$.

(ii) Let $\varphi'_{\mathbf{F}}$ be obtained from $\varphi_{\mathbf{F}}$ by renaming every variable in $\varphi_{\mathbf{F}}$ to z and let $\mathcal{G}' := (\mathcal{A}_{\mathbf{F}}, \varphi'_{\mathbf{F}})$. Then, we have $\min_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}} \max_{x \in \mathrm{var}(\varphi_{\mathbf{F}})} \alpha(x) = \min_{\alpha \in \mathcal{W}^0_{\mathcal{G}'}} \alpha(z)$, due to upwards-closure of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{F}}}$. The resulting minimization problem has a winning condition with a single variable, which can be solved effectively, as discussed above.

(iii) For every $y \in \mathrm{var}(\varphi_{\mathbf{G}})$ let φ_y be obtained from $\varphi_{\mathbf{G}}$ by replacing every subformula $\mathbf{G}_{\leq z}\psi$ for $z \neq y$ by ψ and let $\mathcal{G}_y := (\mathcal{A}_{\mathbf{G}}, \varphi_y)$. Then, we have $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}} \max_{y \in \mathrm{var}(\varphi_{\mathbf{G}})} \alpha(y) = \max_{y \in \mathrm{var}(\varphi_{\mathbf{G}})} \max_{\alpha \in \mathcal{W}^0_{\mathcal{G}_y}} \alpha(y)$, because of downwards-closure of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$. Hence, we have reduced the original problem to $|\mathrm{var}(\varphi_{\mathbf{G}})|$ maximization problems for winning conditions with a single variable, which can be solved as discussed above.

(iv) Let $\varphi'_{\mathbf{G}}$ be obtained from $\varphi_{\mathbf{G}}$ by renaming every variable in $\varphi_{\mathbf{G}}$ to z and let $\mathcal{G}' = (\mathcal{A}_{\mathbf{G}}, \varphi'_{\mathbf{G}})$. Then, $\max_{\alpha \in \mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}} \min_{y \in \mathrm{var}(\varphi_{\mathbf{G}})} \alpha(y) = \max_{\alpha \in \mathcal{W}^0_{\mathcal{G}'}} \alpha(z)$, again due to downwards-closure of $\mathcal{W}^0_{\mathcal{G}_{\mathbf{G}}}$. The resulting maximization problem has a winning condition with a single variable, which can be solved effectively. □

A strategy for Player 0 for a PROMPT–LTL game $\mathcal{G}$ is *optimal*, if it lets her win with bound k, but there is no winning strategy for her with bound $k-1$. By translating a PROMPT–LTL game into a $\mathrm{PLTL}_{\mathbf{F}}$ game we are also able to compute optimal strategies for PROMPT–LTL games.

Corollary 4. *Optimal winning strategies (and their bounds) for* PROMPT–LTL *games are computable.*

6 Conclusion

We presented **2EXPTIME** algorithms that decide whether Player i wins a PLTL game with respect to some, infinitely many, or all variable valuations. Also, we presented algorithms for computing optimal strategies. All these problems are solved by a reduction to (in some cases several) LTL games. Table 1 lists the number of LTL games needed to solve in order to answer the problem under consideration. Note that in some cases, these are LTL games with blinking semantics, which are no harder to solve than classical LTL games.

Table 1. Complexity of decision and optimization problems for PLTL

φ	Problem	LTL games to solve
PLTL	$\alpha \in \mathcal{W}_{\mathcal{G}}^0$?	1
PLTL	$\mathcal{W}_{\mathcal{G}}^0$ empty?	1
PLTL	$\mathcal{W}_{\mathcal{G}}^0$ universal?	1
PLTL	$\mathcal{W}_{\mathcal{G}}^0$ finite?	$\max\{\lvert\mathrm{var}(\varphi) \cap \mathcal{X}\rvert, 1\}$
$\mathrm{PLTL}_{\mathbf{G}}$	$\max_{\alpha \in \mathcal{W}_{\mathcal{G}}^0} \max_{y \in \mathrm{var}(\varphi)} \alpha(y)$	$\lvert var(\varphi)\rvert(\log_2(2(\lvert\mathcal{A}\rvert f(7(\lvert\varphi\rvert+1)))+1)+1)$
$\mathrm{PLTL}_{\mathbf{G}}$	$\max_{\alpha \in \mathcal{W}_{\mathcal{G}}^0} \min_{y \in \mathrm{var}(\varphi)} \alpha(y)$	$\log_2(2(\lvert\mathcal{A}\rvert f(7(\lvert\varphi\rvert+1)))+1)+1$
$\mathrm{PLTL}_{\mathbf{F}}$	$\min_{\alpha \in \mathcal{W}_{\mathcal{G}}^0} \min_{x \in \mathrm{var}(\varphi)} \alpha(x)$	$\lvert var(\varphi)\rvert(\log_2(2(\lvert\mathcal{A}\rvert f(7(\lvert\varphi\rvert+1)))+1)+1)$
$\mathrm{PLTL}_{\mathbf{F}}$	$\min_{\alpha \in \mathcal{W}_{\mathcal{G}}^0} \max_{x \in \mathrm{var}(\varphi)} \alpha(x)$	$\log_2(2(\lvert\mathcal{A}\rvert f(7(\lvert\varphi\rvert+1)))+1)+1$

The decision problems for PROMPT–LTL and PLTL (with the exception of the finiteness problem for PLTL) are solvable by solving a single LTL game of the same size. Hence, adding the bounded operators does not increase the asymptotic computational complexity of solving these games. However, the optimization problems need (in the straight-forward approach) an exponential number of LTL games to solve, which are in some cases of doubly-exponential size, hence the algorithms are in **4EXPTIME**. It is open whether this can be improved.

Another interesting question concerns the tradeoff between the size of a finite-state strategy and the quality of the bounds it is winning for.

Acknowledgments I want to thank Christof Löding and Wolfgang Thomas for helpful discussions, and Roman Rabinovich for coming up with the name *blinking semantics*. Also, I want to thank anonymous referees for their valuable comments on an earlier version of this paper.

References

1. Alur, R., Etessami, K., La Torre, S., Peled, D.: Parametric Temporal Logic for "Model Measuring". In: Wiedermann, J., van Emde Boas, P., Nielsen, M. (eds) ICALP 1999. LNCS, vol. 1644, pp. 159–168. Springer, Heidelberg (1999)
2. Alur, R., La Torre, S.: Deterministic Generators and Games for LTL Fragments. ACM Trans. Comput. Log. 5(1), 1–25 (2004)
3. Büchi, J.R., Landweber, L.H.: Solving Sequential Conditions by Finite-state Strategies. Trans. Amer. Math. Soc. 138, 295–311 (1969)
4. Chatterjee, K., Henzinger, Thomas A., Horn, F.: Finitary winning in ω-regular games. ACM Trans. Comput. Log. 11(1), (2009)
5. Church, A.: Applications of Recursive Arithmetic to the Problem of Circuit Synthesis. In: Summaries of the Summer Institute of Symbolic Logic. vol. I, pp. 3–50. Cornell Univ., Ithaca (1957)
6. Gastin, P., Oddoux, D.: Fast LTL to Büchi Automata Translation. In: Berry, G., Comon, H., Finkel, A. (eds) CAV 2001. LNCS, vol. 2102, pp. 53–65. Springer, Heidelberg (2001)
7. Horn, F., Thomas, W., Wallmeier, N.: Optimal Strategy Synthesis in Request-Response Games. In: Cha, S.D., Choi, J.-Y., Kim, M., Lee, I., Viswanathan, M. (eds) ATVA 2009. LNCS, vol. 5311, pp. 361–373. Springer, Heidelberg (2008)
8. Kupferman, O., Piterman, N., Vardi, M.: From Liveness to Promptness. In: Damm, W., Hermanns, H. (eds) CAV 2007. LNCS, vol. 4590, pp. 406–419. Springer, Heidelberg (2007)
9. Piterman, N.: From Nondeterministic Büchi and Streett Automata to Deterministic Parity Automata. Logical Methods in Computer Science 3(3), 1–21 (2007)
10. Pnueli, A., Rosner, R.: On the Synthesis of a Reactive Module. In: POPL 1989. pp. 179–190. ACM Press, New York (1989)

11. Pnueli, A., Rosner, R.: On the Synthesis of an Asynchronous Reactive Module. In: Ausiello, G., Dezani-Ciancaglini, M., Ronchi Della Rocca, S. (eds) ICALP 1989. LNCS, vol. 372, pp. 652–671. Springer, Heidelberg (1989)
12. Sistla, A. P., Clarke, E. M.: The Complexity of Propositional Linear Temporal Logics. J. ACM 32(3), 733–749 (1985)
13. Zimmermann, M.: Time-optimal Winning Strategies for Poset Games. In: Maneth, S. (ed) CIAA 2009. LNCS, vol. 5642, pp. 217–226. Springer, Heidelberg (2009)

Aachener Informatik-Berichte

This list contains all technical reports published during the past five years. A complete list of reports dating back to 1987 is available from `http://aib.informatik.rwth-aachen.de/`. To obtain copies consult the above URL or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

2005-01 * Fachgruppe Informatik: Jahresbericht 2004

2005-02 Maximillian Dornseif, Felix C. Gärtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"

2005-03 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions

2005-04 Daniel Mölle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem

2005-05 Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots

2005-06 Simon Fischer, Berthold Vöcking: Adaptive Routing with Stale Information

2005-07 Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks

2005-08 Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut

2005-09 Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures

2005-10 Benedikt Bollig: Automata and Logics for Message Sequence Charts

2005-11 Simon Fischer, Berthold Vöcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture

2005-12 Neeraj Mittal, Felix Freiling, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed Systems

2005-13 Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments

2005-14 Felix C. Freiling, Sukumar Ghosh: Code Stabilization

2005-15 Uwe Naumann: The Complexity of Derivative Computation

2005-16 Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)

2005-17 Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)

2005-18 Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlußbericht des GI-Arbeitskreises "Features"

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht über den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Köln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Röglin, Berthold Vöcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-01 * Fachgruppe Informatik: Jahresbericht 2005

2006-02 Michael Weber: Parallel Algorithms for Verification of Large Systems

2006-03 Michael Maier, Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulation

2006-05 Uwe Naumann, Jean Utke, Patrick Heimbach, Chris Hill, Derya Ozyurt, Carl Wunsch, Mike Fagan, Nathan Tallent, Michelle Strout: Adjoint Code by Source Transformation with OpenAD/F

2006-06 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Divide-and-Color

2006-07 Thomas Colcombet, Christof Löding: Transforming structures by set interpretations

2006-08 Uwe Naumann, Yuxiao Hu: Optimal Vertex Elimination in Single-Expression-Use Graphs

2006-09 Tingting Han, Joost-Pieter Katoen: Counterexamples in Probabilistic Model Checking

2006-10 Mesut Günes, Alexander Zimmermann, Martin Wenig, Jan Ritzerfeld, Ulrich Meis: From Simulations to Testbeds - Architecture of the Hybrid MCG-Mesh Testbed

2006-11 Bastian Schlich, Michael Rohrbach, Michael Weber, Stefan Kowalewski: Model Checking Software for Microcontrollers

2006-12 Benedikt Bollig, Joost-Pieter Katoen, Carsten Kern, Martin Leucker: Replaying Play in and Play out: Synthesis of Design Models from Scenarios by Learning

2006-13 Wong Karianto, Christof Löding: Unranked Tree Automata with Sibling Equalities and Disequalities

2006-14 Danilo Beuche, Andreas Birk, Heinrich Dreier, Andreas Fleischmann, Heidi Galle, Gerald Heller, Dirk Janzen, Isabel John, Ramin Tavakoli Kolagari, Thomas von der Maßen, Andreas Wolfram: Report of the GI Work Group "Requirements Management Tools for Product Line Engineering"

2006-15 Sebastian Ullrich, Jakob T. Valvoda, Torsten Kuhlen: Utilizing optical sensors from mice for new input devices

2006-16 Rafael Ballagas, Jan Borchers: Selexels: a Conceptual Framework for Pointing Devices with Low Expressiveness

2006-17 Eric Lee, Henning Kiel, Jan Borchers: Scrolling Through Time: Improving Interfaces for Searching and Navigating Continuous Audio Timelines

2007-01 * Fachgruppe Informatik: Jahresbericht 2006

2007-02 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, and Harald Zankl: SAT Solving for Termination Analysis with Polynomial Interpretations

2007-03 Jürgen Giesl, René Thiemann, Stephan Swiderski, and Peter Schneider-Kamp: Proving Termination by Bounded Increase

2007-04 Jan Buchholz, Eric Lee, Jonathan Klein, and Jan Borchers: coJIVE: A System to Support Collaborative Jazz Improvisation

2007-05 Uwe Naumann: On Optimal DAG Reversal

2007-06 Joost-Pieter Katoen, Thomas Noll, and Stefan Rieger: Verifying Concurrent List-Manipulating Programs by LTL Model Checking

2007-07 Alexander Nyßen, Horst Lichter: MeDUSA - MethoD for UML2-based Design of Embedded Software Applications

2007-08 Falk Salewski and Stefan Kowalewski: Achieving Highly Reliable Embedded Software: An empirical evaluation of different approaches

2007-09 Tina Kraußer, Heiko Mantel, and Henning Sudbrock: A Probabilistic Justification of the Combining Calculus under the Uniform Scheduler Assumption

2007-10 Martin Neuhäußer, Joost-Pieter Katoen: Bisimulation and Logical Preservation for Continuous-Time Markov Decision Processes

2007-11 Klaus Wehrle (editor): 6. Fachgespräch Sensornetzwerke

2007-12 Uwe Naumann: An L-Attributed Grammar for Adjoint Code

2007-13 Uwe Naumann, Michael Maier, Jan Riehme, and Bruce Christianson: Second-Order Adjoints by Source Code Manipulation of Numerical Programs

2007-14 Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch: OpenAD/F: A Modular, Open-Source Tool for Automatic Differentiation of Fortran Codes

2007-15 Volker Stolz: Temporal assertions for sequential and concurrent programs

2007-16 Sadeq Ali Makram, Mesut Güneç, Martin Wenig, Alexander Zimmermann: Adaptive Channel Assignment to Support QoS and Load Balancing for Wireless Mesh Networks

2007-17 René Thiemann: The DP Framework for Proving Termination of Term Rewriting

2007-18 Uwe Naumann: Call Tree Reversal is NP-Complete

2007-19 Jan Riehme, Andrea Walther, Jörg Stiller, Uwe Naumann: Adjoints for Time-Dependent Optimal Control

2007-20 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf: Three-Valued Abstraction for Probabilistic Systems

2007-21 Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre: Compositional Modeling and Minimization of Time-Inhomogeneous Markov Chains

2007-22 Heiner Ackermann, Paul W. Goldberg, Vahab S. Mirrokni, Heiko Röglin, and Berthold Vöcking: Uncoordinated Two-Sided Markets

2008-01 * Fachgruppe Informatik: Jahresbericht 2007

2008-02 Henrik Bohnenkamp, Marielle Stoelinga: Quantitative Testing

2008-03 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, Harald Zankl: Maximal Termination

2008-04 Uwe Naumann, Jan Riehme: Sensitivity Analysis in Sisyphe with the AD-Enabled NAGWare Fortran Compiler

2008-05 Frank G. Radmacher: An Automata Theoretic Approach to the Theory of Rational Tree Relations

2008-06 Uwe Naumann, Laurent Hascoet, Chris Hill, Paul Hovland, Jan Riehme, Jean Utke: A Framework for Proving Correctness of Adjoint Message Passing Programs

2008-07 Alexander Nyßen, Horst Lichter: The MeDUSA Reference Manual, Second Edition

2008-08 George B. Mertzios, Stavros D. Nikolopoulos: The λ-cluster Problem on Parameterized Interval Graphs

2008-09 George B. Mertzios, Walter Unger: An optimal algorithm for the k-fixed-endpoint path cover on proper interval graphs

2008-10 George B. Mertzios, Walter Unger: Preemptive Scheduling of Equal-Length Jobs in Polynomial Time

2008-11 George B. Mertzios: Fast Convergence of Routing Games with Splittable Flows

2008-12 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, Verena Wolf: Abstraction for stochastic systems by Erlang's method of stages

2008-13 Beatriz Alarcón, Fabian Emmes, Carsten Fuhs, Jürgen Giesl, Raúl Gutiérrez, Salvador Lucas, Peter Schneider-Kamp, René Thiemann: Improving Context-Sensitive Dependency Pairs

2008-14 Bastian Schlich: Model Checking of Software for Microcontrollers

2008-15 Joachim Kneis, Alexander Langer, Peter Rossmanith: A New Algorithm for Finding Trees with Many Leaves

2008-16 Hendrik vom Lehn, Elias Weingärtner and Klaus Wehrle: Comparing recent network simulators: A performance evaluation study

2008-17 Peter Schneider-Kamp: Static Termination Analysis for Prolog using Term Rewriting and SAT Solving

2008-18 Falk Salewski: Empirical Evaluations of Safety-Critical Embedded Systems

2008-19 Dirk Wilking: Empirical Studies for the Application of Agile Methods to Embedded Systems

2009-02 Taolue Chen, Tingting Han, Joost-Pieter Katoen, Alexandru Mereacre: Quantitative Model Checking of Continuous-Time Markov Chains Against Timed Automata Specifications

2009-03 Alexander Nyßen: Model-Based Construction of Embedded Real-Time Software - A Methodology for Small Devices

2009-04 Daniel Klünder: Entwurf eingebetteter Software mit abstrakten Zustandsmaschinen und Business Object Notation

2009-05 George B. Mertzios, Ignasi Sau, Shmuel Zaks: A New Intersection Model and Improved Algorithms for Tolerance Graphs

2009-06 George B. Mertzios, Ignasi Sau, Shmuel Zaks: The Recognition of Tolerance and Bounded Tolerance Graphs is NP-complete

2009-07 Joachim Kneis, Alexander Langer, Peter Rossmanith: Derandomizing Non-uniform Color-Coding I

2009-08 Joachim Kneis, Alexander Langer: Satellites and Mirrors for Solving Independent Set on Sparse Graphs

2009-09 Michael Nett: Implementation of an Automated Proof for an Algorithm Solving the Maximum Independent Set Problem

2009-10 Felix Reidl, Fernando Sánchez Villaamil: Automatic Verification of the Correctness of the Upper Bound of a Maximum Independent Set Algorithm

2009-11 Kyriaki Ioannidou, George B. Mertzios, Stavros D. Nikolopoulos: The Longest Path Problem is Polynomial on Interval Graphs

2009-12 Martin Neuhäußer, Lijun Zhang: Time-Bounded Reachability in Continuous-Time Markov Decision Processes

2009-13 Martin Zimmermann: Time-optimal Winning Strategies for Poset Games

2009-14 Ralf Huuck, Gerwin Klein, Bastian Schlich (eds.): Doctoral Symposium on Systems Software Verification (DS SSV'09)

2009-15 Joost-Pieter Katoen, Daniel Klink, Martin Neuhäußer: Compositional Abstraction for Stochastic Systems

2009-16 George B. Mertzios, Derek G. Corneil: Vertex Splitting and the Recognition of Trapezoid Graphs

2009-17 Carsten Kern: Learning Communicating and Nondeterministic Automata

2009-18 Paul Hänsch, Michaela Slaats, Wolfgang Thomas: Parametrized Regular Infinite Games and Higher-Order Pushdown Strategies

2010-02 Daniel Neider, Christof Löding: Learning Visibly One-Counter Automata in Polynomial Time

2010-03 Holger Krahn: MontiCore: Agile Entwicklung von domänenspezifischen Sprachen im Software-Engineering

2010-04 René Wörzberger: Management dynamischer Geschäftsprozesse auf Basis statischer Prozessmanagementsysteme

2010-07 George B. Mertzios: A New Intersection Model for Multitolerance Graphs, Hierarchy, and Efficient Algorithms

2010-08 Carsten Otto, Marc Brockschmidt, Christian von Essen, Jürgen Giesl: Automated Termination Analysis of Java Bytecode by Term Rewriting

2010-09 George B. Mertzios, Shmuel Zaks: The Structure of the Intersection of Tolerance and Cocomparability Graphs

2010-10 Peter Schneider-Kamp, Jürgen Giesl, Thomas Ströder, Alexander Serebrenik, René Thiemann: Automated Termination Analysis for Logic Programs with Cut

* These reports are only available as a printed version.

Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.