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 $(w, i + j, k) \models \varphi$.