Parameterized Linear Temporal Logics Meet Costs: Still not Costlier than LTL (full version)⋆

Martin Zimmermann

Reactive Systems Group, Saarland University, 66123 Saarbrücken, Germany
{zimmermann}@react.uni-saarland.de

Abstract. We continue the investigation of parameterized extensions of Linear Temporal Logic (LTL) that retain the attractive algorithmic properties of LTL: a polynomial space model checking algorithm and a doubly-exponential time algorithm for solving games. Alur et al. and Kupferman et al. showed that this is the case for Parametric LTL (PLTL) and PROMPT-LTL respectively, which have temporal operators equipped with variables that bound their scope in time. Later, this was also shown to be true for Parametric LDL (PLDL), which extends PLTL to be able to express all ω-regular properties.

Here, we generalize PLTL to systems with costs, i.e., we do not bound the scope of operators in time, but bound the scope in terms of the cost accumulated during time. Again, we show that model checking and solving games for specifications in PLTL with costs is not harder than the corresponding problems for LTL. Finally, we discuss PLDL with costs and extensions to multiple cost functions.

1 Introduction

Parameterized linear temporal logics address a serious shortcoming of Linear-temporal Logic (LTL) [26]: LTL is not able to express timing constraints, e.g., while $\mathbf{G}(q \rightarrow \mathbf{F}p)$ expresses that every request q is eventually answered by a response p, the waiting time between requests and responses might diverge. This is typically not the desired behavior, but cannot be ruled out by LTL.

To overcome this shortcoming, parameterized LTL (PLTL) [1] was introduced by Alur et al., which extends LTL with parameterized operators of the form $\mathbf{F}_{\leq x}$ and $\mathbf{G}_{\leq y}$, where x and y are variables. The formula $\mathbf{G}(q \rightarrow \mathbf{F}_{\leq x}p)$ expresses that every request q is answered by p within an arbitrary, but fixed number of steps $\alpha(x)$. Here, α is a variable valuation, a mapping of variables to natural numbers. Typically, one is interested in whether a PLTL formula is satisfied with respect to some variable valuation. For example, the model checking problem asks whether a given transition system satisfies a given PLTL specification φ with respect to some α, i.e., whether every path satisfies φ with respect to α.

Similarly, solving infinite games amounts to determining whether there is an α such that Player 0 has a strategy such that every play that is consistent with the strategy satisfies the winning condition with respect to α. Alur et al. showed that the PLTL model checking problem is PSpace-complete. Kupferman et al. later considered PROMPT–LTL [23], which can be seen as the fragment of PLTL without the parameterized always operator, and showed that PROMPT–LTL model checking is still PSpace-complete and that PROMPT–LTL realizability, an abstract notion of infinite game, is 2ExpTime-complete. While the results of Alur et al. relied on involved pumping arguments, the results of Kupferman et al. where all based on the so-called alternating-color technique, which basically allows to reduce PROMPT–LTL to LTL. Furthermore, the result on realizability was extended to infinite games on graphs [38], again using the alternating-color technique.

Another serious shortcoming of LTL (and its parameterized variants) is their expressiveness: LTL is equi-expressive to first-order logic with order [22] and thus not as expressive as ω-regular expressions. This shortcoming was addressed by a long line of temporal logics [18, 24, 34, 35, 37] with regular expressions, finite automata, or grammar operators to obtain the full expressivity of the ω-regular languages. One of these logics is Linear Dynamic Logic (LDL), which has temporal operators $\langle r \rangle$ and $[r]$, where r is a regular expression. For example, the formula $[r_0] (q \rightarrow \langle r_1 \rangle p)$ holds in a word w, if for every request q at a position n such that $w_0 \cdots w_{n-1}$ matches r_0, there is a position $n' \geq n$ such that p holds at n' and $w_n \cdots w_{n'-1}$ matches r_1. Intuitively, the diamond operator corresponds to the eventually operator of LTL, but is guarded by a regular expression. Dually, the box operator is a guarded always. Although LDL is more expressive than LTL, its algorithmic properties are similar: model checking is PSpace-complete and solving games is 2ExpTime-complete [34].

There are temporal logics whose expressiveness goes even beyond the ω-regular languages to capture properties of recursive programs, which are typically ω-contextfree. The visibly ω-contextfree languages [2] are an important class of languages located between the ω-regular ones and the ω-contextfree ones that enjoy desirable closure properties, which make it suitable to be employed in verification. Temporal logics that capture this class are visibly LTL [10], the fixed-point logic VP-μTL [9], and visibly LDL (VLDL) [36]. The logic visibly LTL enhances LTL with visibly rational expressions [11], and VP-μTL extends the linear-time μ-calculus [33] with non-local modalities. Finally, VLDL has the same temporal operators as LDL, but allows to use visibly pushdown automata instead of regular expressions as guards. For all these logics, model checking is ExpTime-complete, i.e., (under standard complexity theoretic assumptions) harder than the model checking problem for LTL. Furthermore, solving games with VLDL winning conditions is 3ExpTime-complete, again harder than solving LTL games. Thus, going beyond the ω-regular languages does increase the complexity of these problems at last.

All these logics tackle one shortcoming of LTL, but not both simultaneously. This was achieved for the first time by adding parameterized operators to LDL.

The logic, called parameterized LDL (PLDL), has additional operators $\langle r \rangle_{\leq x}$ and $[r]_{\leq y}$ with the expected semantics: the variables bound the scope of the operator. And even for this logic, which has parameters and is more expressive than LTL, model checking is still PSpace-complete and solving games is 2ExpTime-complete [19]. Again, these problems were solved by an application of the alternating-color technique. One has to overcome some technicalities, but the general proof technique is the same as for PROMPT–LTL.

The decision problems for the parameterized logics mentioned above are boundedness problems, e.g., one asks for an upper bound on the waiting times between requests and responses in case of the formula $\mathbf{G}(q \to \mathbf{F}_{\leq x}p)$. Recently, more general boundedness problems in logics and automata received a lot of attention to obtain decidable quantitative extensions of monadic second-order logic and better synthesis algorithms. In general, boundedness problems are undecidable for automata with counters, but become decidable if the acceptance conditions can refer to boundedness properties of the counters, but the transition relation has no access to counter values. Recent advances include logics and automata with bounds [4, 7], satisfiability algorithms for these logics [5, 6, 8, 32], and regular cost-functions [17]. However, these formalisms, while very expressive, are intractable and thus not suitable for verification and synthesis. Thus, less expressive formalisms were studied that appear more suitable for practical applications, e.g., finitary parity [15], parity with costs [21], energy-parity [14], mean-payoff-parity [16], consumption games [12], and the use of weighted automata for specifying quantitative properties [3, 13]. In particular, the parity condition with cost is defined in graphs whose edges are weighted by natural numbers (interpreted as costs) and requires the existence of a bound b such that almost every occurrence of an odd color is followed by an occurrence of a larger even color such that the cost between these positions is at most b. Although strictly stronger than the classical parity condition, solving parity games with costs is as hard as solving parity games [21, 25].

We investigate parameterized temporal logics in a weighted setting similar to the one of parity conditions with costs: our graphs are equipped with cost functions that label the edges with natural numbers and parameterized operators are now evaluated with respect to cost instead of time, i.e., the parameters bound the accumulated cost instead of the elapsed time. Thus, the formula $\mathbf{G}(q \to \mathbf{F}_{\leq x}p)$ requires that every request q is answered with cost at most $\alpha(x)$. We show the following results about PLTL with costs (cPLTL):

First, we refine the alternating-color technique to the cost-setting, which requires to tackle some technical problems induced by the fact that accumulated cost, unlike time, does not increase in every step, e.g., if an edge with cost zero is traversed. In particular, infinite paths with finite cost have to be taken care of appropriately.

Second, we show that Kupferman et al.'s proofs based on the alternating-color technique can be adapted to the cost-setting as well. For model checking, we again obtain PSpace-completeness while solving games is still 2ExpTime-complete.

Third, we consider PLDL with costs (cPLDL), which is defined as expected: the diamond and the box operator may be equipped with parameters bounding their scope. Again, the complexity does not increase: model checking is PSPACE-complete while solving games is 2EXPTIME-complete.

Fourth, we generalize both logics to a setting with multiple cost functions. Now, the parameterized temporal operators have another parameter that determines the cost function under which they are evaluated. Even these extensions do not increase complexity: model checking is again PSPACE-complete while solving games is still 2EXPTIME-complete.

Fifth, we also study the optimization variant of the model checking and the game problem for these logics: here, one is interested in finding the *optimal* variable valuation for which a given transition system satisfies the specification. For example, for the request-response condition one is interested in minimizing the waiting times between requests and responses. For cPLTL and cPLDL, we show that the model checking optimization problem can be solved in polynomial space while the optimization problem for infinite games can be solved in triply-exponential time. These results are similar to the ones obtained for PLTL [1, 38]. In particular, the exponential gap between the decision and the optimization variant of solving infinite games exists already for PLTL. Whether this gap can be closed is an open problem. A first step towards this direction was made by giving an approximation algorithm for this problem with doubly-exponential running time [31].

The paper is structured as follows: in Section 2, we introduce cPLTL and discuss basic properties. Then, in Section 3, we extend the alternating-color technique to the setting with costs, which we apply in Section 4 to the model checking problem and in Section 5 to solve infinite games. In Section 6, we extend these results to cPLDL and, in Section 7, to multiple cost functions. Finally, in Section 8, we investigate model checking and game-solving as optimization problems.

2 Parametric LTL with Costs

Let $\mathcal{V}$ be an infinite set of variables and let P be a set of atomic propositions. The formulas of cPLTL are given by the grammar

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

where $p \in P$ and $z \in \mathcal{V}$. We use the derived operators $\mathtt{tt} := p \vee \neg p$ and $\mathtt{ff} := p \wedge \neg p$ for some fixed $p \in P$, $\mathbf{F}\varphi := \mathtt{tt}\mathbf{U}\varphi$, and $\mathbf{G}\varphi := \mathtt{ff}\mathbf{R}\varphi$. Furthermore, we use $p \rightarrow \varphi$ and $\neg p \rightarrow \varphi$ as shorthand for $\neg p \vee \varphi$ and $p \vee \varphi$, respectively. Additional derived operators are introduced on page 6.

The set of subformulas of a cPLTL formula φ is denoted by $\mathrm{cl}(\varphi)$ and we define the size of φ to be the cardinality of $\mathrm{cl}(\varphi)$. Furthermore, we define

$$\mathrm{var}_{\mathbf{F}}(\varphi) = \{z \in \mathcal{V} \mid \mathbf{F}_{\leq z}\psi \in \mathrm{cl}(\varphi)\}$$

to be the set of variables parameterizing eventually operators in φ, and

$$\text{var}_{\mathbf{G}}(\varphi) = \{z \in \mathcal{V} \mid \mathbf{G}_{\leq z}\psi \in \text{cl}(\varphi)\}$$

to be the set of variables parameterizing always operators in φ. Finally, $\text{var}(\varphi) = \text{var}_{\mathbf{F}}(\varphi) \cup \text{var}_{\mathbf{G}}(\varphi)$ denotes the set of all variables appearing in φ.

cPLTL is evaluated on so-called cost-traces (traces for short) of the form

$$w = w_0\, c_0\, w_1\, c_1\, w_2\, c_2 \cdots \in \left(2^P \cdot \mathbb{N}\right)^{\omega},$$

which encode the evolution of the system in terms of the atomic propositions that hold true in each time instance, and in terms of the cost of changing the system state. The cost of the trace w is defined as $\text{cst}(w) = \sum_{j \geq 0} c_j$, which might be infinite. A finite cost-trace is required to begin and end with an element of 2^P. The cost $\text{cst}(w)$ of a finite cost-trace $w = w_0c_0w_1c_1\cdots c_{n-1}w_n$ is defined as $\text{cst}(w) = \sum_{j=0}^{n-1} c_j$.

Furthermore, we require the existence of a distinguished atomic proposition κ such that all cost-traces satisfy $c_j > 0$ if, and only if, $\kappa \in w_{j+1}$, i.e., κ indicates that the last step had non-zero cost. We use the proposition κ to reason about costs: for example, we are able to express whether a trace has cost zero and whether a trace has cost ∞. In the following, we will ensure that all our systems only allow traces that satisfy this assumption.

Also, to evaluate formulas we need to instantiate the variables parameterizing the temporal operators. To this end, we define a variable valuation to be a mapping $\alpha\colon \mathcal{V} \to \mathbb{N}$. Now, we can define the model relation between a cost-trace $w = w_0\, c_0\, w_1\, c_1\, w_2\, c_2 \cdots$, a position n of w, a variable valuation α, and a cPLTL formula as follows:

- $(w, n, \alpha) \models p$ if, and only if, $p \in w_n$,
- $(w, n, \alpha) \models \neg p$ if, and only if, $p \notin w_n$,
- $(w, n, \alpha) \models \varphi \wedge \psi$ if and only if $(w, n, \alpha) \models \varphi$ and $(w, n, \alpha) \models \psi$,
- $(w, n, \alpha) \models \varphi \vee \psi$ if and only if $(w, n, \alpha) \models \varphi$ or $(w, n, \alpha) \models \psi$,
- $(w, n, \alpha) \models \mathbf{X}\varphi$ if, and only if, $(w, n + 1, \alpha) \models \varphi$,
- $(w, n, \alpha) \models \varphi\mathbf{U}\psi$ if, and only if, there exists a $j \geq 0$ such that $(w, n+j, \alpha) \models \psi$ and $(w, n + k, \alpha) \models \varphi$ for every k in the range $0 \leq k < j$,
- $(w, n, \alpha) \models \varphi\mathbf{R}\psi$ if, and only if, for every $j \geq 0$: either $(w, n + j, \alpha) \models \psi$ or there exists a k in the range $0 \leq k < j$ such that $(w, n + k, \alpha) \models \varphi$,
- $(w, n, \alpha) \models \mathbf{F}_{\leq z}\varphi$ if, and only if, there exists a $j \geq 0$ with $\text{cst}(w_nc_n \cdots c_{n+j-1}w_{n+j}) \leq \alpha(z)$ such that $(w, n + j, \alpha) \models \varphi$, and
- $(w, n, \alpha) \models \mathbf{G}_{\leq z}\varphi$ if, and only if, for every $j \geq 0$ with $\text{cst}(w_nc_n \cdots c_{n+j-1}w_{n+j}) \leq \alpha(z)$: $(w, n + j, \alpha) \models \varphi$.

Note that we recover the semantics of PLTL as the special case where every c_n is equal to one.

For the sake of brevity, we write $(w, \alpha) \models \varphi$ instead of $(w, 0, \alpha) \models \varphi$ and say that w is a model of φ with respect to α. For variable-free formulas, we even drop the α and write $w \models \varphi$.

As usual for parameterized temporal logics, the use of variables has to be restricted: bounding eventually and always operators by the same variable leads to an undecidable satisfiability problem [1].

Definition 1. *A cPLTL formula φ is well-formed, if* $\mathrm{var}_{\mathbf{F}}(\varphi) \cap \mathrm{var}_{\mathbf{G}}(\varphi) = \emptyset$.

In the following, we only consider well-formed formulas and omit the qualifier "well-formed". Also, we will denote variables in $\mathrm{var}_{\mathbf{F}}(\varphi)$ by x and variables in $\mathrm{var}_{\mathbf{G}}(\varphi)$ by y, if the formula φ is clear from context.

We consider the following fragments of cPLTL. Let φ be a cPLTL formula:

- φ is an LTL formula, if $\mathrm{var}(\varphi) = \emptyset$.
- φ is a cPLTL$_{\mathbf{F}}$ formula, if $\mathrm{var}_{\mathbf{G}}(\varphi) = \emptyset$.
- φ is a cPLTL$_{\mathbf{G}}$ formula, if $\mathrm{var}_{\mathbf{F}}(\varphi) = \emptyset$.

Every LTL, cPLTL$_{\mathbf{F}}$, and every cPLTL$_{\mathbf{G}}$ formula is well-formed by definition.

Example 1.

1. The formula $\mathbf{G}(q \to \mathbf{F}_{\leq x}p)$ is satisfied with respect to α, if every request (a position where q holds) is followed by a response (a position where p holds) such that the cost of the infix between the request and the response is at most $\alpha(x)$.
2. The (max-) parity condition with costs [21] can be expressed[1] in cPLTL via

$$\mathbf{FG}\left(\bigwedge_{c \in \{1,3,\ldots,d-1\}}\left(c \to \mathbf{F}_{\leq x}\bigvee_{c' \in \{c+1,c+3,\ldots,d\}} c'\right)\right),$$

where d is the maximal color, which we assume w.l.o.g. to be even.
However, the Streett condition with costs [21] cannot be expressed in cPLTL, as it is defined with respect to multiple cost functions, one for each Streett pair. We extend cPLTL to multiple cost functions in Section 7.

As for PLTL, one can also parameterize the until and the release operator and consider bounds of the form "$> z$". However, this does not increase expressiveness of the logic. Formally, we define

- $(w,n,\alpha) \models \varphi\mathbf{U}_{\leq z}\psi$ if, and only if, there exists a $j \geq 0$ with $\mathrm{cst}(w_nc_n\cdots c_{n+j-1}w_{n+j}) \leq \alpha(z)$ such that $(w,n+j,\alpha) \models \psi$ and $(w,n+k,\alpha) \models \varphi$ for every k in the range $0 \leq k < j$,
- $(w,n,\alpha) \models \varphi\mathbf{R}_{\leq z}\psi$ if, and only if, for every $j \geq 0$ with $\mathrm{cst}(w_nc_n\cdots c_{n+j-1}w_{n+j}) \leq \alpha(z)$: $(w,n+j,\alpha) \models \psi$ or there exists a k in the range $0 \leq k < j$ such that $(w,n+k,\alpha) \models \varphi$,
- $(w,n,\alpha) \models \mathbf{F}_{>z}\varphi$ if, and only if, there exists a $j \geq 0$ with $\mathrm{cst}(w_nc_n\cdots c_{n+j-1}w_{n+j}) > \alpha(z)$ such that $(w,n+j,\alpha) \models \varphi$, and

---

[1] Note that the bound in the parity condition with costs may depend on the trace while one typically uses global bounds for cPLTL (see, e.g., Section 4 and Section 5). However, for games in finite arenas (and thus also for model checking) these two variants coincide [21].

- $(w, n, \alpha) \models \mathbf{G}_{>z}\varphi$ if, and only if,for every $j \geq 0$ with $\mathrm{cst}(w_n c_n \cdots c_{n+j-1} w_{n+j}) > \alpha(z)$ satisfies $(w, n+j, \alpha) \models \varphi$.
- $(w, n, \alpha) \models \varphi\mathbf{U}_{>z}\psi$ if, and only if,there exists a $j \geq 0$ with $\mathrm{cst}(w_n c_n \cdots c_{n+j-1} w_{n+j}) > \alpha(z)$ such that $(w, n+j, \alpha) \models \psi$ and $(w, n+k, \alpha) \models \varphi$ for every k in the range $0 \leq k < j$, and
- $(w, n, \alpha) \models \varphi\mathbf{R}_{>z}\psi$ if, and only if,for every $j \geq 0$ with $\mathrm{cst}(w_n c_n \cdots c_{n+j-1} w_{n+j}) > \alpha(z)$: $(w, n+j, \alpha) \models \psi$ or there exists a k in the range $0 \leq k < j$ such that $(w, n+k, \alpha) \models \varphi$.

Let $\varphi \equiv \psi$ denote equivalence of the formulas φ and ψ, i.e., for every w, every n, and every α, we have $(w, n, \alpha) \models \varphi$ if, and only if,$(w, n, \alpha) \models \psi$. Then, we have the following equivalences (which also restrict the use of variables as defined in Definition 1):

- $\varphi\mathbf{U}_{\leq x}\psi \equiv \varphi\mathbf{U}\psi \wedge \mathbf{F}_{\leq x}\psi$ $-$ $\mathbf{G}_{>x}\varphi \equiv \mathbf{F}_{\leq x}\mathbf{GX}(\neg\kappa \vee \mathbf{G}\varphi)$
- $\varphi\mathbf{R}_{\leq y}\psi \equiv \varphi\mathbf{R}\psi \vee \mathbf{G}_{\leq y}\psi$ $-$ $\varphi\mathbf{U}_{>y}\psi \equiv \mathbf{G}_{\leq y}(\varphi \wedge \mathbf{FX}(\kappa \wedge \varphi\mathbf{U}\psi))$
- $\mathbf{F}_{>y}\varphi \equiv \mathbf{G}_{\leq y}\mathbf{FX}(\kappa \wedge \mathbf{F}\varphi)$ $-$ $\varphi\mathbf{R}_{>x}\psi \equiv \mathbf{F}_{\leq x}(\varphi \vee \mathbf{GX}(\neg\kappa \vee \varphi\mathbf{R}\psi))$

Before we begin proving some useful lemmas about cPLTL we first show that the new logic is indeed more expressive than PLTL. Here, we use $\models_{\mathrm{PLTL}}$ to denote the model relation for PLTL, which is obtained by assigning cost one to every edge, i.e., the scope of the parameterized operators is bounded in the length of infixes. To simplify notion, let $\pi(w_0\, c_0\, w_1\, c_1\, w_2\, c_2 \cdots) = w_0 w_1 w_2 \cdots$ denote the projection of a cost-trace to a trace in $(2^P)^\omega$.

Lemma 1. *Let $\varphi = \mathbf{F}_{\leq x}p$. There is no PLTL formula φ' such that for every variable valuation α there is an α' such that the following holds for every cost-trace w and every position n: $(w, n, \alpha) \models \varphi$ if, and only if, $(\pi(w), n, \alpha') \models_{PLTL} \varphi'$.*

Proof. We exploit the fact that φ' has no access to the magnitude of the costs c_n, only to their signs via the proposition κ.

Towards a contradiction, assume there is such a φ', let α be a variable valuation and let α' be the corresponding valuation such that $(w, n, \alpha) \models \varphi$ if, and only if, $(\pi(w), n, \alpha') \models_{\mathrm{PLTL}} \varphi'$ for every cost-trace w and every n. Let

$$w_0 = \emptyset\, (\alpha(x) + 1)\, \{p\}\, 0\, (\emptyset\, 0\,)^\omega,$$

i.e., p holds with cost $\alpha(x) + 1$ for the first time, and let

$$w_1 = \emptyset\, \alpha(x)\, \{p\}\, 0\, (\emptyset\, 0\,)^\omega,$$

i.e., p holds with cost $\alpha(x)$ for the first time. Thus, $(w_0, 0, \alpha) \not\models \varphi$ and $(w_1, 0, \alpha) \models \varphi$. However, we have $\pi(w_0) = \pi(w_1)$, i.e., either both of the projections satisfy φ' with respect to α' or both do not. This yields the desired contradiction. $\square$

Note that φ' in Lemma 1 may not depend on α. If it could, then we could hardcode α into φ, provided there are propositions reflecting the exact costs

incurred along an edge. Thus, the current statement of Lemma 1 is the most general variant.

Note that we defined cPLTL formulas to be in negation normal form. Nevertheless, a negation can be pushed to the atomic propositions using the duality of the operators. Thus, we can define the negation of a cPLTL formula.

Lemma 2. *For every cPLTL formula φ there exists an efficiently constructible cPLTL formula $\neg\varphi$ s.t.*

1. *$(w, n, \alpha) \models \varphi$ if, and only if,$(w, n, \alpha) \not\models \neg\varphi$ for every w, every n, and every α,*
2. *$|\neg\varphi| = |\varphi|$.*
3. *If φ is well-formed, then so is $\neg\varphi$.*
4. *If φ is an LTL formula, then so is $\neg\varphi$.*
5. *If φ is a cPLTL$_\mathbf{F}$ formula, then $\neg\varphi$ is a cPLTL$_\mathbf{G}$ formula and vice versa.*

Proof. We construct $\neg\varphi$ by induction over the construction of φ using the dualities of the operators:

- $\neg(p) = \neg p$ - $\neg(\neg p) = p$
- $\neg(\varphi \wedge \psi) = (\neg\varphi) \vee (\neg\psi)$ - $\neg(\varphi \vee \psi) = (\neg\varphi) \wedge (\neg\psi)$
- $\neg(\varphi \mathbf{U} \psi) = \neg\varphi \mathbf{R} \neg\psi$ - $\neg(\varphi \mathbf{R} \psi) = \neg\varphi \mathbf{U} \neg\psi$
- $\neg(\mathbf{F}_{\leq x}\varphi) = \mathbf{G}_{\leq x}\neg\varphi$ - $\neg(\mathbf{G}_{\leq y}\varphi) = \mathbf{F}_{\leq y}\neg\varphi$

The latter four claims of Lemma 2 follow from the definition of $\neg\varphi$ while the first one can be shown by a straightforward induction over φ's construction. □

Another important property of parameterized logics is monotonicity: increasing (decreasing) the values for parameterized eventually operators (parameterized always operators) preserves satisfaction.

Lemma 3. *Let φ be a cPLTL formula and let α and β be variable valuations satisfying $\alpha(x) \leq \beta(x)$ for every $x \in \mathrm{var}_\mathbf{F}(\varphi)$ and $\alpha(y) \geq \beta(y)$ for every $y \in \mathrm{var}_\mathbf{G}(\varphi)$. If $(w, \alpha) \models \varphi$, then $(w, \beta) \models \varphi$.*

Especially, if we are interested in checking whether a formula is satisfied with respect to some α, we can always recursively replace every subformula $\mathbf{G}_{\leq y}\psi$ by $\psi \vee \mathbf{X}(\neg\kappa \mathbf{U}(\neg\kappa \wedge \psi))$, as this is equivalent to $\mathbf{G}_{\leq y}\psi$ with respect to every variable valuation mapping y to zero, which is the smallest possible value for y. Note that we have to ignore the current truth value of κ, as it indicates the cost of the last transition, not the cost of the next one.

3   The Alternating-Color Technique for Costs

In the following, we investigate model checking and infinite games for cPLTL and present algorithms solving these problems. The tool of choice to solve these problems for parameterized temporal logics is the so-called alternating-color

technique. It was originally introduced for PROMPT–LTL by Kupferman et al. [23], but has been shown to be applicable to extensions as well [19, 38].

In this section, we introduce the alternating-color technique and extend it again, this time to a setting with costs. To apply the technique, one introduces a fresh proposition and interprets its truth values as colors. Then, a changepoint is a position where the color differs from the one at the previous position. Now, one replaces every parameterized eventually operator $\mathbf{F}_{\leq x}\psi$ by an LTL formula φ requiring ψ to be satisfied within at most one changepoint. Thus, if the distance between changepoints is bounded by $k \in \mathbb{N}$ and φ is satisfied, then the parameterized eventually is satisfied with respect to $\alpha(x) = 2k$, as every longer infix contains at least two changepoints. Conversely, if $\mathbf{F}_{\leq x}\psi$ is satisfied with respect to $\alpha(x)$ and if the distance between changepoints is always at least $\alpha(x)$, then φ is satisfied as well, as there is at most one changepoint in every infix of length $\alpha(x)$.

Thus, one can inductively replace parameterized eventually operators by LTL formulas that are equivalent on all traces where the distance between changepoints is bounded from above and from below. These bounds arise from properties of the problem we consider. In particular, in model-checking one always has an ultimately periodic error trace, if an LTL formula does not hold. By requiring that the color changes infinitely often (an LTL-definable property) one obtains an ultimately periodic error trace with the desired bounds on the distance. A similar argument works for infinite games, too. Here, one relies on the existence of finite-state winning strategies, which again bound the distance between changepoints.

Here, we extend the alternating-color technique to the setting with costs. The main technical difficulty we have to overcome are cost-traces of finite cost, i.e., those that have a suffix of cost zero. On these, the parameterized eventually operator degenerates to a classical eventually operator.

Fix a fresh atomic proposition $p \notin P$. We say that a cost-trace

$$w' = w_0' c_0' w_1' c_1' w_2' c_2' \cdots \in \left(2^{P \cup \{p\}} \cdot \mathbb{N} \right)^\omega$$

is a coloring of a cost trace

$$w = w_0 c_0 w_1 c_1 w_2 c_2 \cdots \in \left(2^P \cdot \mathbb{N} \right)^\omega,$$

if $w_n' \cap P = w_n$ and $c_n' = c_n$ for every n, i.e., w' and w only differ in the truth values of the new proposition p. A position n is a changepoint of w', if $n = 0$ or if the truth value of p in w_{n-1}' and w_n' differs. A block of w' is an infix $w_n' c_n' \cdots w_{n+j}'$ of w' such that n and $n + j + 1$ are successive changepoints. If a coloring has only finitely many changepoints, then we refer to its suffix starting at the last changepoint as its tail, i.e., the coloring is the concatenation of a finite number of blocks and its tail.

Let $k \in \mathbb{N}$. We say that w' is k-bounded if every block and its tail (if it has one) has cost at most k. Dually, we say that w' is k-spaced, if every block has cost at least k. Note that we do not have a requirement on the cost of the tail in this case.

Given a cPLTL$_\mathbf{F}$ formula φ, let rel(φ) denote the LTL formula obtained from φ by recursively replacing every subformula $\mathbf{F}_{\leq x}\psi$ by

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

Intuitively, the relativized formula requires rel(ψ) to be satisfied within at most one changepoint. On bounded and spaced colorings, φ and rel(φ) are "equivalent".

Lemma 4 (cp. Lemma 2.1 of [23]). *Let w be a cost-trace and let φ be a cPLTL$_\mathbf{F}$ formula.*

1. *Let $(w, \alpha) \models \varphi$ for some variable valuation α. Then, $w' \models \mathrm{rel}(\varphi)$ for every $(k+1)$-spaced coloring w' of w, where $k = \max_{x \in \mathrm{var}(\varphi)} \alpha(x)$.*
2. *Let $w' \models \mathrm{rel}(\varphi)$ for some k-bounded coloring w' of w. Then, $(w, \alpha) \models \varphi$, where $\alpha(x) = 2k$ for every x.*

Proof. Note that w and its colorings coincide on their cost. Hence, when speaking about the cost of an infix or suffix, we do not have to specify whether we refer to w or to a coloring of w.

1.) Fix a $(k+1)$-spaced coloring w' of w, where $k = \max_{x \in \mathrm{var}(\varphi)} \alpha(x)$. We show that $(w, n, \alpha) \models \varphi$ implies $(w', n) \models \mathrm{rel}(\varphi)$ by induction over the construction of φ.

The only non-trivial case is the one of a parameterized eventually: thus, assume $(w, n, \alpha) \models \mathbf{F}_{\leq x}\psi$, i.e., there is a j with $\mathrm{cst}(w_n c_n \cdots c_{n+j-1} w_{n+j}) \leq \alpha(x)$ and $(w, n+j, \alpha) \models \psi$. By induction hypothesis, we have $(w', n+j) \models \mathrm{rel}(\psi)$. As w' is $(k+1)$-spaced, i.e., the cost of each block is at least $k+1$, there is at most one changepoint between (and including) the positions n and $n+j-1$ in w'. Hence, $(w', n) \models p\mathbf{U}(\neg p\mathbf{U}\mathrm{rel}(\psi)))$, if $p \in w'_n$, and $(w', n) \models \neg p\mathbf{U}(p\mathbf{U}\mathrm{rel}(\psi)))$ otherwise. Thus, $(w', n) \models \mathrm{rel}(\mathbf{F}_{\leq x}\psi)$.

2.) Dually, fix a k-bounded coloring w' of w and define the variable valuation α with $\alpha(x) = 2k$ for every x. We show that $(w', n) \models \mathrm{rel}(\varphi)$ implies $(w, n, \alpha) \models \varphi$ by induction over the construction of φ.

Again, the only non-trivial case is the one of a parameterized eventually: thus, let $(w', n) \models \mathrm{rel}(\mathbf{F}_{\leq x}\psi)$. We assume $(w', n) \models p$ (the other case is dual). Then, we have $(w', n) \models p\mathbf{U}(\neg p\mathbf{U}\mathrm{rel}(\psi))$, i.e., rel($\psi$) is satisfied at some position $n+j$ such that there is at most one changepoint between (and including) the positions n and $n+j-1$ in w'. As w' is k-bounded, this implies that the cost of the infix $w_n c_n \cdots w_{n+j}$ is bounded by $2k$. Furthermore, applying the induction hypothesis yields $(w, n+j, \alpha) \models \psi$. Hence, $(w, n, \alpha) \models \mathbf{F}_{\leq x}\psi$. $\qquad\square$

4 Model Checking

In this section, we solve the model checking problem for cPLTL via an application of the alternating-color technique. Our approach is similar to the one of Kupferman et al. for PROMPT–LTL model checking [23], but we have to overcome some technical difficulties arising from cost-traces of cost zero. To this end,

we ensure that the parameterized eventually operators are treated as classical eventually operators on tails of cost-traces of cost zero.

A transition system $\mathcal{S}$ is a tuple $\mathcal{S} = (S, s_I, E, \ell, \text{cst})$ consisting of a finite directed graph (S, E), an initial state $s_I \in S$, a labeling function $\ell \colon S \to 2^P$, and a cost function $\text{cst} \colon E \to \mathbb{N}$. We encode the weights in binary, although the algorithms we present in this section and their running times and space requirements are oblivious to the exact weights. Furthermore, we assume that every state has at least one successor to spare us from dealing with finite paths. Recall our requirement on cost-traces having a distinguished atomic property κ indicating the sign of the cost of the previous transition. Thus, we require $\mathcal{S}$ to satisfy the following property: if $\kappa \in \ell(v')$, then $\text{cst}(v, v') > 0$ for every edge $(v, v') \in E$ leading to v'. Dually, if $\kappa \notin \ell(v')$, then $\text{cst}(v, v') = 0$ for every edge $(v, v') \in E$.

A path through $\mathcal{S}$ is a sequence $\pi = s_0 s_1 s_2 \cdots$ with $s_0 = s_I$ and $(s_n, s_{n+1}) \in E$ for every n. Its cost-trace $\text{tr}(\pi)$ is defined as

$$\text{tr}(\pi) = \ell(s_0)\text{cst}(s_0, s_1)\ell(s_1)\text{cst}(s_1, s_2)\ell(s_2)\text{cst}(s_2, s_3) \cdots ,$$

which satisfies our assumption on the proposition κ.

The transition system $\mathcal{S}$ satisfies a cPLTL formula φ with respect to a variable valuation α, if the trace of every path through $\mathcal{S}$ satisfies φ with respect to α. The cPLTL model checking problem asks, given a transition system $\mathcal{S}$ and a cPLTL formula φ, whether $\mathcal{S}$ satisfies φ with respect to some α.

Theorem 1. *The cPLTL model checking problem is* PSPACE-*complete.*

We begin by showing PSPACE-membership. First note that we can restrict ourselves to cPLTL$_\mathbf{F}$ formulas: given a cPLTL formula φ, let φ' denote the formula obtained by recursively replacing every subformula $\mathbf{G}_{\leq y}\psi$ by $\psi \vee \mathbf{X}(\neg\kappa\mathbf{U}(\neg\kappa \wedge \psi))$. Due to Lemma 3 and the discussion below it, every transition system $\mathcal{S}$ satisfies φ with respect to some α if, and only if, $\mathcal{S}$ satisfies φ' with respect to some α'.

Next, we show how to apply the alternating-color: recall that the classical algorithm for LTL model checking searches for a fair path, i.e., one that visits infinitely many accepting states, in the product of $\mathcal{S}$ with a Büchi automaton recognizing the models of the negated specification. If such a path exists, then $\mathcal{S}$ does not satisfy the specification, as the fair path contains a path π through $\mathcal{S}$ and an accepting run of the automaton on its trace, i.e., the trace does not satisfy the specification. If there is no such fair path, then $\mathcal{S}$ satisfies the specification.

For cPLTL we have to find such a path for every α in order to show that $\mathcal{S}$ does not satisfy the specification with respect to any α. To this end, one relativizes the cPLTL$_\mathbf{F}$ specification as described in Section 3 and builds an automaton for the negation of the relativized formula in conjunction with a formula that ensures that every ultimately periodic model is both k-bounded and k'-spaced for some appropriate k and k'. Then, we search for a pumpable fair path in the product of the system and the Büchi automaton recognizing the models of the negated specification, which is non-deterministically labeled by p.

Applying Lemma 4 and pumping a fair path through the product appropriately yields a counterexample for every α. Thus, model checking is reduced to finding a pumpable fair path. Let us stress again that this algorithm is similar to the one for PROMPT–LTL, we just have to pay attention to some intricacies stemming from the fact that we want to bound the cost, not the waiting time: there might be paths with finite cost, which have to be dealt with appropriately.

Recall that p is the distinguished atomic proposition used to relativize cPLTL formulas. A colored Büchi graph with costs $(V, v_I, E, \ell, \mathrm{cst}, F)$ consists of a finite direct graph (V, E), an initial vertex v_I, a labeling function $\ell\colon V \to 2^{\{p\}}$, a cost function $\mathrm{cst}\colon E \to \mathbb{N}$, and a set $F \subseteq V$ of accepting vertices. A path $v_0 v_1 v_2 \cdots$ is pumpable, if each of its blocks induced by p contains a vertex repetition such that the cycle formed by the repetition has non-zero cost[2]. Note that we do not have a requirement on the cost of the tail, if the path has one. The path is fair, if it visits F infinitely often. The pumpable non-emptiness problem asks, given a colored Büchi graph with costs, whether it has an initial pumpable fair path.

Lemma 5. *If a colored Büchi graph with costs has an initial pumpable fair path, then also one of the form $\pi_0 \pi_1^\omega$ with $|\pi_0 \pi_1| \in \mathcal{O}(n^2)$, where n is the number of vertices of the graph.*

Proof. Let π be an arbitrary initial pumpable fair path. First, assume it has only finitely many changepoints. If there are two blocks that start with the same vertex, then we can remove all blocks in between and obtain another initial pumpable fair path. Thus, we can assume that π has at most n blocks. Furthermore, the length of each block can be bounded by $\mathcal{O}(n)$ by removing cycles while retaining the state repetition with non-zero cost and at least one accepting vertex (provided the block has one). Now, consider the tail: by removing infixes one can find a cycle of length at most n containing an accepting vertex and a path of length at most n leading from the last changepoint to a vertex on the cycle. Hence, we define π_0 to be the prefix containing all blocks and the path leading to the cycle and define π_1 to be the cycle. Then, we have $|\pi_0 \pi_1| \in \mathcal{O}(n^2)$ and $\pi_0 \pi_1^\omega$ is an initial pumpable fair path.

On the other hand, if π contains infinitely many changepoints, then we can remove blocks and shorten other blocks as described above until we have constructed a prefix $\pi_0 \pi_1$ such that $\pi_0 \pi_1^\omega$ has the desired properties. In this case, we can assume that the first position of π_1 is a changepoint by "rotating" π_1 appropriately and appending a suitable prefix of it to π_0. $\qquad\square$

Let $\mathcal{S} = (S, s_I, E, \ell, \mathrm{cst})$ be a transition system and let φ be a cPLTL$_\mathbf{F}$ formula. Furthermore, consider the LTL formula

$$\chi = (\mathbf{GF}p \wedge \mathbf{GF}\neg p) \leftrightarrow \mathbf{GF}\kappa,$$

which is satisfied by a cost-trace, if the trace has infinitely many changepoints if, and only if,[3] it has cost ∞. Now, let $\mathfrak{A} = (Q, 2^{P \cup \{p\}}, q_I, \delta, F)$ be a non-

---

[2] Note that our definition is more involved than the one of Kupferman et al., since we require a cycle with non-zero cost instead of any circle.

[3] Here, we use our assumption on κ indicating the sign of the costs.

deterministic Büchi automaton recognizing the models of the LTL formula $\neg\mathrm{rel}(\varphi)\wedge$ χ, which we can pick such that its number of states is bounded exponentially in $|\varphi|$. Now, define the colored Büchi graph with costs

$$\mathcal{S}\times\mathfrak{A}=(S\times Q\times 2^{\{p\}},(s_I,q_I,\emptyset),E',\ell',\mathrm{cst}',F')$$

where

- $((s,q,C),(s',q',C'))\in E'$ if, and only if,$(s,s')\in E$ and $q'\in\delta(q,\ell(s)\cup C)$,
- $\ell(s,q,C)=C$,
- $\mathrm{cst}'((s,q,C),(s',q',C'))=\mathrm{cst}(s,s')$, and
- $F'=S\times F\times 2^{\{p\}}$.

Lemma 6. *[cp. Lemma 4.2 of [23]] $\mathcal{S}$ does not satisfy φ with respect to any α if, and only if,$\mathcal{S}\times\mathfrak{A}$ has an initial pumpable fair path.*

Proof. Let $\mathcal{S}$ not satisfy φ with respect to any variable valuation. Fix $k=(|S|\cdot|Q|+3)\cdot W$, where W is the largest cost in $\mathcal{S}$, and define the valuation α by $\alpha(x)=2k$ for every x. As $\mathcal{S}$ does not satisfy φ with respect to α, there is a path π through $\mathcal{S}$ with $(\mathrm{tr}(\pi),\alpha)\not\models\varphi$. Thus, due to Lemma 4.2, every k-bounded coloring of w does not satisfy $\mathrm{rel}(\varphi)$.

Now, let w' be a k-bounded and $(k-W)$-spaced coloring of $\mathrm{tr}(\pi)$ which starts with p not holding true. Such a coloring can always be constructed, as W is the largest cost appearing in $\mathcal{S}$. Note that w' satisfies χ by construction. Thus, we have $w'\models\neg\mathrm{rel}(\varphi)\wedge\chi$, i.e., there is an accepting run $q_0q_1q_2\cdots$ of $\mathfrak{A}$ on w'. Consider the path

$$(s_0,q_0,w'_0\cap\{p\})(s_1,q_1,w'_1\cap\{p\})(s_2,q_2,w'_2\cap\{p\})\cdots$$

where $s_0s_1s_2\cdots=\pi$, which is fair by construction. We claim that it is pumpable: consider a block, which is $(k-W)$-spaced. Thus, it contains at least $|S|\cdot|Q|+2$ many edges with non-zero cost, enough to enforce a vertex repetition with non-zero cost in between. To this end, one takes the sets V_j of vertices visited between the j-th and the $(j+1)$-th edge with non-zero cost (including the j-th edge). This yields $|S|\cdot|Q|+1$ non-empty sets of vertices of $\mathcal{S}\times\mathfrak{A}$ that coincide in their third component, as we are within one block. However, there are only $|S|\cdot|Q|$ many such vertices, which yields the desired repetition.

Now, consider the converse implication and let α' be an arbitrary variable valuation. We show that $\mathcal{S}$ does not satisfy φ with respect to α'. Due to Lemma 3, it is sufficient to show that $\mathcal{S}$ does not satisfy φ with respect to the valuation α mapping every variable to $k=\min_{x\in\mathrm{var}(\varphi)}\alpha'(x)$.

Fix an initial pumpable fair path of $\mathcal{S}\times\mathfrak{A}$. It has a vertex repetition in every block such that the induced cycle has non-zero cost. We pump each such cycle $k+1$ times to obtain the path

$$(s_0,q_0,C_0)(s_1,q_1,C_1)(s_2,q_2,C_2)\cdots.$$

By construction, $\pi=s_0s_1s_2\cdots$ is a path through $\mathcal{S}$ and

$$w'=(\ell(s_0)\cup C_0)(\ell(s_1)\cup C_1)(\ell(s_2)\cup C_2)\cdots$$

is a coloring of its trace $\mathrm{tr}(\pi)$. Also, $q_0 q_1 q_2 \cdots$ is an accepting run of $\mathfrak{A}$ on w', i.e., $w' \models \neg\mathrm{rel}(\varphi) \wedge \chi$. Lastly, w' is $(k+1)$-spaced by construction, as the cost function of $\mathcal{S} \times \mathfrak{A}$ is induced by the one of $\mathcal{S}$.

Assume towards a contradiction that $\mathcal{S}$ satisfies φ with respect to α, which implies $(\mathrm{tr}(\pi), \alpha) \models \varphi$. Applying Lemma 4.1 yields that every $(k+1)$-spaced coloring of $\mathrm{tr}(\pi)$ satisfies $\mathrm{rel}(\varphi)$. However, w' is a $(k+1)$-spaced coloring which satisfies $\neg\mathrm{rel}(\varphi)$, i.e., we have derived the desired contradiction. $\square$

Now, we are ready to prove Theorem 1.

Proof. PSPACE-hardness holds already for LTL model checking [30], which is a special case of cPLTL model checking. Membership is witnessed by the following algorithm: check whether the colored Büchi graph $\mathcal{S} \times \mathfrak{A}$ has an initial pumpable fair path, which is correct due to Lemma 6. But as the graph is of exponential size, it has to be constructed and tested for non-emptiness on-the-fly.

Due to Lemma 5, it suffices to check for the existence of an ultimately periodic path $\pi_0 \pi_1^\omega$ such that $|\pi_0 \pi_1| \leq n \in \mathcal{O}(|\mathcal{S} \times \mathfrak{A}|^2)$, i.e., n is exponential in the size of φ and quadratic in the size of $\mathcal{S}$. To this end, one guesses a vertex v (the first vertex of π_1) and checks the following reachability properties:

1. Is v reachable from v_I via a path where each block contains a cycle with non-zero cost?
2. Is v reachable from v via a non-empty path that visits an accepting vertex and which either has no changepoint or where each block contains a cycle with non-zero cost? In this case, we also require that v and the last vertex on the path from v_I to v guessed in item 1.) differ on their third component in order to make v a changepoint. This spares us from having a block that spans π_0 and π_1.

All these reachability problems can be solved in non-deterministic polynomial space, as a successor of a vertex of $\mathcal{S} \times \mathfrak{A}$ can be guessed and verified in polymonial time and the length of the paths to be guessed is bounded by n, which can be represented with polynomially many bits. $\square$

Furthermore, by applying both directions of the proof of Lemma 6, we obtain an exponential upper bound on the values of a satisfying variable valuation, if one exists. This is asymptotically tight, as one can already show exponential lower bounds for PROMPT–LTL [23].

Corollary 1. *Fix a transition system $\mathcal{S}$ and a cPLTL-formula φ such that $\mathcal{S}$ satisfies φ with respect to some α. Then, $\mathcal{S}$ satisfies φ with respect to a valuation that is bounded exponentially in the size of φ and linearly in the number of states of $\mathcal{S}$ and in the maximal cost in $\mathcal{S}$.*

Dually, one can show the existence of an exponential variable valuation that witnesses whether a given cPLTL$_\mathbf{G}$ specification is satisfied with respect to every variable valuation. The following lemma states the contrapositive of this statement, which we prove using pumping arguments that are similar to the ones for the analogous results for PLTL$_\mathbf{G}$ and PLDL$_\mathbf{G}$ [20].

Lemma 7. *Fix a transition system $\mathcal{S}$ and a cPLTL$_{\mathbf{G}}$-formula φ such that $\mathcal{S}$ does not satisfy φ with respect to every α. Then, $\mathcal{S}$ does not satisfy φ with respect to a valuation that is bounded exponentially in the size of φ and linearly in the number of states of $\mathcal{S}$ and in the maximal cost in $\mathcal{S}$.*

Proof. Let $\mathfrak{A}$ be a Büchi automaton recognizing the models of $\mathrm{rel}(\neg\varphi) \wedge \chi$, which is of exponential size in $|\varphi|$. Define $k^* = (4 \cdot |\mathfrak{A}| \cdot |\mathcal{S}| + 2) \cdot W$, where W is the largest cost in $\mathcal{S}$, and let α^* be the variable valuation mapping every variable to k^*. We consider the contrapositive and show: if there is an α such that $\mathcal{S}$ does not satisfy φ with respect to α, then $\mathcal{S}$ does not satisfy φ with respect to α^*.

Thus, assume there is an α and a path π such that $(\mathrm{tr}(\pi), \alpha) \models \neg\varphi$. Due to upwards-monotonicity we can assume w.l.o.g. that α maps all variables to the same value, call it k.

Let $\mathrm{tr}(\pi)'$ be a $(k^* + W + 1)$-bounded and $(k^* + 1)$-spaced p-coloring of $\mathrm{tr}(\pi)$ that starts with p not holding true in the first position, which can always be constructed as W is the largest cost. Applying Lemma 4.1 shows that $\mathrm{tr}(\pi)'$ satisfies $\mathrm{rel}(\neg\varphi)$. Furthermore, it satisfies χ by construction. Fix some accepting run of $\mathfrak{A}$ on $\mathrm{tr}(\pi)'$ and consider an arbitrary block of $\mathrm{tr}(\pi)'$: if the run does not visit an accepting state during the block, we can remove (if necessary) infixes of the block where the run reaches the same state before and after the infix and where the state of $\mathcal{S}$ at the beginning and the end of the infix are the same, until the block has length at most $|\mathfrak{A}| \cdot |\mathcal{S}|$ and thus cost at most $|\mathfrak{A}| \cdot |\mathcal{S}| \cdot W$.

On the other hand, assume the run visits at least one accepting state during the block. Fix one such position. Then, we can remove infixes as above between the beginning of the block and the position before the accepting state is visited and between the position after the accepting state is reached and before the end of the block. What remains is a block whose length is at most $2 \cdot |\mathfrak{A}| \cdot |\mathcal{S}| + 1$, at it has most $|\mathfrak{A}| \cdot |\mathcal{S}|$ many positions before the designated position, this position itself, and at most $|\mathfrak{A}| \cdot |\mathcal{S}|$ many after the designated position. Hence, the block has cost at most $(2 \cdot |\mathfrak{A}| \cdot |\mathcal{S}| + 1) \cdot W$.

Thus, we have constructed a $(2 \cdot |\mathfrak{A}| \cdot |\mathcal{S}| + 1) \cdot W$-bounded p-coloring $\mathrm{tr}(\hat{\pi})'$ of a trace $\mathrm{tr}(\hat{\pi})$ for some path $\hat{\pi}$ of $\mathcal{S}$, as well as an accepting run of $\mathfrak{A}$ on $\mathrm{tr}(\hat{\pi})'$. Hence, $\mathrm{tr}(\hat{\pi})'$ is a model of $\mathrm{rel}(\neg\varphi)$ and applying Lemma 4.2 shows that $\mathrm{tr}(\hat{\pi})$ is a model of $\neg\varphi$ with respect to the variable valuation mapping every variable to $2 \cdot (2 \cdot |\mathfrak{A}| \cdot |\mathcal{S}| + 1) \cdot W = k^*$. Therefore, $\mathcal{S}$ does not satisfy φ with respect to α^*. $\qquad\square$

5   Infinite Games

After having solved the model checking problem for cPLTL, we now turn our attention to solving infinite games with winning conditions in cPLTL. Our approach follows the one for the case of PLTL winning conditions [38], which in turn generalized the proof for PROMPT–LTL realizability [23]. Again, we have to deal with the existence of cost-traces with finite costs, which is done as in the case of model checking.

An arena $\mathcal{A} = (V, V_0, V_1, v_I, E, \ell, \mathrm{cst})$ consists of a finite directed graph (V, E), a partition (V_0, V_1) of V, an initial vertex $v_I \in V$, a labeling $\ell \colon V \to 2^P$, and a cost function $\mathrm{cst} \colon E \to \mathbb{N}$. Again, we encode the weights in binary, although the algorithms we present here and their running times and space requirements are oblivious to the exact weights. Also, we again assume that every vertex has at least one successor to avoid dealing with finite paths. Finally, we ensure our requirement on the proposition κ to indicate the sign of the costs in a cost-trace: if $\kappa \in \ell(v')$, then we require $\mathrm{cst}(v, v') > 0$ for every edge $(v, v') \in E$ leading to v'. Dually, if $\kappa \notin \ell(v')$, then $\mathrm{cst}(v, v') = 0$ for every edge $(v, v') \in E$.

A play $\rho = \rho_0 \rho_1 \rho_2 \cdots$ is a path through $\mathcal{A}$ starting in v_I and its cost-trace $\mathrm{tr}(\rho)$ is defined as

$$\mathrm{tr}(\rho) = \ell(\rho_0)\, \mathrm{cst}(\rho_0, \rho_1)\, \ell(\rho_1)\, \mathrm{cst}(\rho_1, \rho_2)\, \ell(\rho_2)\, \mathrm{cst}(\rho_2, \rho_3) \cdots .$$

A strategy for Player $i \in \{0, 1\}$ is a mapping $\sigma \colon V^* V_i \to V$ with $(v, \sigma(wv)) \in E$ for every $w \in V^*$ and $v \in V_i$. A play ρ is consistent with σ if $\rho_{n+1} = \sigma(\rho_0 \cdots \rho_n)$ for every n with $\rho_n \in V_i$.

A cPLTL game $\mathcal{G} = (\mathcal{A}, \varphi)$ consists of an arena $\mathcal{A}$ and a winning condition φ, which is a cPLTL formula. A strategy σ for Player 0 is winning with respect to some variable valuation α, if the trace of every play that is consistent with σ satisfies the winning condition φ with respect to α.

We are interested in determining whether Player 0 has a winning strategy for a given cPLTL game, and in determining a winning strategy for her if this is the case, which we refer to as solving the game.

Theorem 2. *Determining whether Player 0 has a winning strategy in a given cPLTL game is* 2ExpTime*-complete. Furthermore, a winning strategy (if one exists) can be computed in doubly-exponential time.*

First, we note that it is again sufficient to consider cPLTL$_\mathbf{F}$ formulas, as we are interested in the existence of a variable valuation (see the discussion below Lemma 3). Next, we apply the alternating-color technique: to this end, we modify the arena to allow Player 0 to produce colorings of plays of the original arena and use the relativized winning condition, i.e., we reduce the problem to a game with LTL winning condition. The winner (and a winning strategy) of such a game can be computed in doubly-exponential time [27, 28].

To allow for the coloring, we double the vertices of the arena, additionally label one copy with p and the other not, and split every move into two: first, the player whose turn it is picks an outgoing edge, then Player 0 decides in which copy she wants to visit the target, thereby picking the truth value of p.

Formally, given an arena $\mathcal{A} = (V, V_0, V_1, v_I, E, \ell, \mathrm{cst})$, the extended arena $\mathcal{A}' = (V', V_0', V_1', v_I', E', \ell', \mathrm{cst}')$ consists of

- $V' = V \times \{0, 1\} \cup E$,
- $V_0' = V_0 \times \{0, 1\} \cup E$ and $V_1' = V_1 \times \{0, 1\}$,
- $v_I' = (v_I, 0)$,
- $E' = \{((v, 0), e), ((v, 1), e), (e, (v', 0)), (e, (v', 1)) \mid e = (v, v') \in E\}$,

- $\ell'(e) = \emptyset$ for every $e \in E$ and $\ell'(v, b) = \begin{cases} \ell(v) & \text{if } b = 0, \\ \ell(v) \cup \{p\} & \text{if } b = 1, \end{cases}$ and

- $\text{cst}'((v, b), (v, v')) = \text{cst}(v, v')$ and $\text{cst}'((v, v'), (v', b')) = 0$.

A path through the new arena $\mathcal{A}'$ has the form $(\rho_0, b_0)e_0(\rho_1, b_1)e_1(\rho_2, b_2)\cdots$ for some path $\rho_0\rho_1\rho_2\cdots$ through $\mathcal{A}$, where $e_n = (\rho_n, \rho_{n+1})$ and $b_n \in \{0, 1\}$. Also, we have $|\mathcal{A}'| \in \mathcal{O}(|\mathcal{A}|^2)$. Note that we use the costs in $\mathcal{A}'$ only to argue the correctness of our construction, not to define the winning condition for the game in $\mathcal{A}'$.

Also, note that the additional choice vertices of the form $e \in E$ have to be ignored when it comes to evaluating the winning condition on the trace of a play. Thus, we consider games with LTL winning conditions under so-called *blinking semantics*: Player 0 wins a play $\rho = \rho_0\rho_1\rho_2\cdots$ under blinking semantics, if $\ell(\rho_0)\ell(\rho_2)\ell(\rho_4)\cdots$ satisfies the winning condition φ; otherwise, Player 1 wins. Winning strategies under blinking semantics are defined as expected. Determining whether Player 0 has a winning strategy for a given game with LTL winning condition under blinking semantics is 2ExpTime-complete, which can be shown by a slight variation of the proof for LTL games under classical semantics [27, 28]. Furthermore, if Player 0 has a winning strategy for such a game, then also a finite-state one of at most doubly-exponential size in $|\varphi|$.

Such a strategy for an arena $(V, V_0, V_1, v_I, E, \ell, \text{cst})$ is given by a memory structure $\mathcal{M} = (M, m_I, \text{upd})$ with a finite set M of memory states, an initial memory state $m_I \in M$, and an update function $\text{upd}: M \times V \to M$, and by a next-move function $\text{nxt}: V_0 \times M \to V$ satisfying $(v, \text{nxt}(v, m)) \in E$ for every m and every v. The function $\text{upd}^*: V^+ \to M$ is defined via $\text{upd}^*(v) = m_I$ and $\text{upd}^*(wv) = \text{upd}(\text{upd}^*(w), v)$. Then, the strategy σ implemented by $\mathcal{M}$ and nxt is defined by $\sigma(wv) = \text{nxt}(v, \text{upd}^*(wv))$. The size of σ is (slightly abusively) defined as $|M|$.

Given a game $(\mathcal{A}, \varphi)$ with cPLTL$_\mathbf{F}$ winning condition φ, define $\mathcal{A}'$ as above and let $\varphi' = \text{rel}(\varphi) \wedge \chi$, where $\chi = (\mathbf{GF}p \wedge \mathbf{GF}\neg p) \leftrightarrow \mathbf{GF}\kappa$. Recall that χ is satisfied by a cost-trace, if the trace has infinitely many changepoints if, and only if, it has cost ∞.

Lemma 8. *[cp. Lemma 3.1 of [23]] Player 0 has a winning strategy for $(\mathcal{A}, \varphi)$ with respect to some α if, and only if, she has a winning strategy for $(\mathcal{A}', \varphi')$ under blinking semantics.*

Proof. Let σ be a winning strategy for Player 0 in $(\mathcal{A}, \varphi)$ with respect to some fixed α and define $k = \max_{x \in \text{var}(\varphi)} \alpha(x)$. We define a strategy σ' for $\mathcal{A}'$ as follows:

$$\sigma'((\rho_0, b_0)(\rho_0, \rho_1)\cdots(\rho_{n-1}, \rho_n)(\rho_n, b_n)) = (\rho_n, \sigma(\rho_0 \cdots \rho_n))$$

if $(\rho_n, b_n) \in V_0'$, which implies $\rho_n \in V_0$. Thus, at a non-choice vertex, Player 0 mimics the behavior of σ. At choice vertices, she alternates between the two copies of the arena every time the cost has exceeded $k + 1$: let

$$w = (\rho_0, b_0)(\rho_0, \rho_1)\cdots(\rho_n, b_n)(\rho_n, \rho_{n+1})$$

be a play prefix ending in a choice vertex and let $n' \leq n$ be the last changepoint in $\ell'(\rho_0, b_0) \cdots \ell'(\rho_n, b_n)$. Now, we define

$$
\sigma'(w) = \begin{cases} (\rho_{n+1}, 0) & \text{if } (\mathrm{cst}(\rho_{n'} \cdots \rho_n) < k+1 \text{ and } b_n = 0) \text{ or} \\ & \quad (\mathrm{cst}(\rho_{n'} \cdots \rho_n) \geq k+1 \text{ and } b_n = 1), \\ (\rho_{n+1}, 1) & \text{if } (\mathrm{cst}(\rho_{n'} \cdots \rho_n) < k+1 \text{ and } b_n = 1) \text{ or} \\ & \quad (\mathrm{cst}(\rho_{n'} \cdots \rho_n) \geq k+1 \text{ and } b_n = 0). \end{cases}
$$

Let $\rho = \rho_0 \rho_1 \rho_2 \cdots$ be a play in $\mathcal{A}'$ that is consistent with σ' and let

$$
\rho' = \rho_0 \rho_2 \rho_4 \cdots = (v_0, b_0)(v_1, b_1)(v_2, b_2) \cdots .
$$

By definition of σ', the sequence $v_0 v_1 v_2 \cdots$ is a play in $\mathcal{A}$ that is consistent with σ and thus winning for Player 0 with respect to α, i.e., $(\mathrm{tr}(v_0 v_1 v_2 \cdots), \alpha) \models \varphi$. Also, $w' = \ell'(v_0, b_0) \ell'(v_1, b_1) \ell'(v_2, b_2) \cdots$ is a $(k+1)$-spaced coloring of the trace $\mathrm{tr}(v_0 v_1 v_2 \cdots)$. Hence, $w' \models \varphi'$ due to Lemma 4.1. Finally, w' satisfies χ by construction. Thus, σ' is a winning strategy for $(\mathcal{A}', \varphi')$ under blinking semantics.

Now, let σ' be a winning strategy for Player 0 in $(\mathcal{A}', \varphi')$ which we can assume (w.l.o.g.) to be implemented by $\mathcal{M}' = (M', m'_I, \mathrm{upd}')$ and some next-move function nxt' such that $|M'|$ is doubly-exponential in $|\varphi|$. We define a strategy σ for $\mathcal{A}$ by simulating a play in $\mathcal{A}'$ that is consistent with σ'.

To this end, define the memory structure $\mathcal{M} = (M, m_I, \mathrm{upd})$ for $\mathcal{A}$ with $M = (V \times \{0, 1\}) \times M'$, $m_I = ((v, 0), m'_I)$, and

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

where $e = (v, v')$ and $m' = \mathrm{upd}'(m, e)$. Intuitively, the update-function mimics two moves in $\mathcal{A}'$: first, the one from (v, b) to $e = (v, v')$ and then the move from this choice vertex determined by the strategy σ', which is given by $\mathrm{nxt}'(e, m')$, where m' is the updated memory state.

Let w be a play prefix of a play in $\mathcal{A}$. The memory state $\mathrm{upd}^*(w) = ((v, b), m)$ encodes the following information: the simulated play w' in $\mathcal{A}'$ ends in (v, b), where v is the last vertex of w, and we have $\mathrm{upd}'^*(w') = m$. Hence, it contains all information necessary to apply the next-move function nxt' to mimic σ'. Thus, we define the next-move function $\mathrm{nxt} \colon V_0 \times M \to V$ for Player 0 in $\mathcal{A}$ by

$$
\mathrm{nxt}(v, ((v', b), m)) = \begin{cases} v'' & \text{if } v = v' \text{ and } \mathrm{nxt}'((v', b), m) = (v', v''), \\ \overline{v} & \text{otherwise, for some } \overline{v} \in V \text{ with } (v, \overline{v}) \in E. \end{cases}
$$

By definition of $\mathcal{M}$, the second case of the definition is never invoked, since $\mathrm{upd}^*(wv) = ((v', b), m)$ always satisfies $v = v'$.

It remains to show that the strategy σ implemented by $\mathcal{M}$ and nxt is indeed a winning strategy for Player 0 for $(\mathcal{A}, \varphi)$ with respect to some α. To this end, let $k = (|V| \cdot |M| + 3) \cdot W$ and define $\alpha(x) = 2k$ for every x, where W is the largest weight in $\mathcal{A}$.

Let $\rho_0\rho_1\rho_2\cdots$ be a play in $\mathcal{A}$ that is consistent with σ. A straightforward induction shows that there exist bits $b_0, b_1, b_2, \cdots$ such that the play

$$(\rho_0, b_0)(\rho_0, \rho_1)(\rho_1, b_1)(\rho_1, \rho_2)(\rho_2, b_2)\cdots$$

in $\mathcal{A}'$ is consistent with σ'. Hence, $w'' = \ell'(\rho_0, b_0)\ell'(\rho_1, b_1)\ell'(\rho_2, b_2)\cdots$ satisfies φ'. We show that w'' is k-bounded. This suffices to finish the proof as we can apply Lemma 4.2 and obtain $(\mathrm{tr}(\rho), \alpha) \models \varphi$, as w'' is a k-bounded coloring of $\mathrm{tr}(\rho)$. Thus, σ is a winning strategy for Player 0 for $(\mathcal{A}, \varphi)$ with respect to α.

Towards a contradiction assume that w'' is not k-bounded. Then, there exist positions $i < j$ such that

- $\rho_i = \rho_j$,
- $\mathrm{upd}'^*((\rho_0, b_0) \cdots (\rho_i, b_i)) = \mathrm{upd}'^*((\rho_0, b_0) \cdots (\rho_j, b_j))$,
- the bits $b_i, \ldots, b_j$ are all equal, and
- $\mathrm{cst}(\rho_i \cdots \rho_j) > 0$.

To show this, one defines the sets V_j of vertices visited between the j-th and the $(j+1)$-th edge with non-zero cost (including the j-th edge). This yields $|V| \cdot |M| + 1$ non-empty sets of vertices of $(V \times \{0, 1\}) \times M$ that coincide on the bit stored in their second component. Hence, we have derived the desired vertex repetition, as there are only $|V| \cdot |M|$ such vertices.

Thus, the play

$$\rho^* = (\rho_0, b_0)(\rho_0, \rho_1) \cdots (\rho_{i-1}, \rho_i)(\rho_i, b_i)\big[(\rho_i, \rho_{i+1})(\rho_{i+1}, b_{i+1}) \cdots (\rho_{j-1}, \rho_j)(\rho_j, b_j)\big]^\omega,$$

obtained by traversing the cycle between (ρ_i, b_i) and (ρ_j, b_j) infinitely often, is consistent with σ', since the memory states reached at the beginning and the end of the loop are the same. Remember that the bits do not change between i and j. Thus, $\mathrm{tr}(\rho^*)$ has only finitely many change points, but infinitely many occurrences of κ and does therefore not satisfy χ under blinking semantics. This contradicts the fact that σ' is a winning strategy for $(\mathcal{A}', \mathrm{rel}(\varphi) \wedge \chi)$ under blinking semantics. $\qquad\square$

Now, we are able to prove Theorem 2.

Proof. Hardness follows immediately from the 2ExpTime-hardness of determining the winner of an LTL game [27, 28], which is a special case.

Membership in 2ExpTime follows from the reductions described above: first, we turn the winning condition into a cPLTL$_\mathbf{F}$ formula and construct the LTL game under blinking semantics obtained from expanding the arena and relativizing the winning condition. This game is only polynomially larger than the original one and its winner (and a winning strategy) is computable in doubly-exponential time. $\qquad\square$

By applying both directions of the proof of Lemma 8, we obtain a doubly-exponential upper bound on the values of a satisfying variable valuation, if one exists. This is asymptotically tight, as one can already show doubly-exponential lower bounds for PROMPT–LTL [38].

Corollary 2. *Fix a cPLTL game $\mathcal{G} = (\mathcal{A}, \varphi)$ such that Player 0 has a winning strategy for $\mathcal{G}$ with respect to some α. Then, Player 0 has a winning strategy for $\mathcal{G}$ with respect to a valuation that is bounded doubly-exponentially in the size of φ and linearly in the number of vertices of $\mathcal{A}$ and in the maximal cost in $\mathcal{A}$.*

6 Parametric LDL with Costs

Recall the two main shortcomings of LTL, the lack of expressiveness and impossibility to impose time- and cost-bounds. Here, we have dealt with the second shortcoming by introducing cPLTL. However, we have not dealt with the first shortcoming. In this section, we address both simultaneously by extending PLDL to a cost-setting.

Linear Dynamic logic (LDL) [18, 34] extends LTL by temporal operators that are guarded by regular expressions, e.g., $\langle r \rangle \varphi$ holds at position n, if there is a j such that φ holds at position $n + j$ and the infix between positions n and $n + j - 1$ matches r. The resulting logic has the full expressiveness of the ω-regular languages while retaining many of LTL's desirable properties like a simple syntax, intuitive semantics, a polynomial space algorithm for model checking, and a doubly-exponential time algorithm for solving games. Parametric LDL (PLDL) [19] allows to parameterize such operators, i.e., $\langle r \rangle_{\leq x} \varphi$ holds at position n with respect to a variable valuation α, if there is a $j \leq \alpha(x)$ such that φ holds at position $n + j$ and the infix between positions n and $n + j - 1$ matches r. Model checking and solving games with PLDL specifications is not harder than for LTL, although PLDL is more expressive and has parameterized operators. In this section, we consider cPLDL where the parameters bound the cost of the infix instead of the length. This logic has the full expressiveness of the ω-regular languages and parameterized operators whose scope is bounded by costs instead of by time. Thus, it can express modulo-counting properties not expressible in LTL, PLTL, and cPLTL, as well as cost-bounds, which are not expressible in PLTL and PLDL.

Formally, formulas of cPLDL are given by the grammar

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle r \rangle \varphi \mid [r] \varphi \mid \langle r \rangle_{\leq z} \varphi \mid [r]_{\leq z} \varphi$$
$$r ::= \phi \mid \varphi? \mid r + r \mid r \, ; r \mid r^*$$

where $p \in P$, $z \in \mathcal{V}$, and where ϕ ranges over propositional formulas over P. As for cPLTL, cPLDL formulas are evaluated on cost-traces with respect to variable valuations. Satisfaction of atomic formulas and of conjunctions and disjunctions is defined as usual, and for the four temporal operators, we define

- $(w, n, \alpha) \models \langle r \rangle \varphi$ if there exists $j \geq 0$ such that $(n, n + j) \in \mathcal{R}(r, w, \alpha)$ and $(w, n + j, \alpha) \models \varphi$,
- $(w, n, \alpha) \models [r] \varphi$ if for all $j \geq 0$ with $(n, n + j) \in \mathcal{R}(r, w, \alpha)$ we have $(w, n + j, \alpha) \models \varphi$,
- $(w, n, \alpha) \models \langle r \rangle_{\leq z} \varphi$ if there exists $j \geq 0$ with $\mathrm{cst}(w_n c_n \cdots c_{n+j-1} w_{n+j}) \leq \alpha(z)$ such that $(n, n + j) \in \mathcal{R}(r, w, \alpha)$ and $(w, n + j, \alpha) \models \varphi$, and

- $(w, n, \alpha) \models [r]_{\leq z} \varphi$ if for all $j \geq 0$ with $\text{cst}(w_n c_n \cdots c_{n+j-1} w_{n+j}) \leq \alpha(z)$ and with $(n, n+j) \in \mathcal{R}(r, w, \alpha)$ we have $(w, n+j, \alpha) \models \varphi$.

Here, the relation $\mathcal{R}(r, w, \alpha) \subseteq \mathbb{N} \times \mathbb{N}$ contains all pairs $(m, n) \in \mathbb{N} \times \mathbb{N}$ such that $w_m \cdots w_{n-1}$ matches r and is defined inductively by

- $\mathcal{R}(\phi, w, \alpha) = \{(n, n+1) \mid w_n \models \phi\}$ for propositional φ,
- $\mathcal{R}(\psi?, w, \alpha) = \{(n, n) \mid (w, n, \alpha) \models \psi\}$,
- $\mathcal{R}(r_0 + r_1, w, \alpha) = \mathcal{R}(r_0, w, \alpha) \cup \mathcal{R}(r_1, w, \alpha)$,
- $\mathcal{R}(r_0 ; r_1, w, \alpha) = \{(n_0, n_2) \mid \exists n_1 \text{ s.t. } (n_0, n_1) \in \mathcal{R}(r_0, w, \alpha) \text{ and } (n_1, n_2) \in \mathcal{R}(r_1, w, \alpha)\}$, and
- $\mathcal{R}(r^*, w, \alpha) = \{(n, n) \mid n \in \mathbb{N}\} \cup \{(n_0, n_{k+1}) \mid \exists n_1, \ldots, n_k \text{ s.t. } (n_j, n_{j+1}) \in \mathcal{R}(r, w, \alpha) \text{ for all } j \leq k\}$.

Again, we restrict ourselves to well-formed formulas, i.e., those whose set of variables parameterizing diamond operators and whose set of variables parameterizing box operators are disjoint.

Using the duality of the operators $\langle r \rangle_{\leq z}$ and $[r]_{\leq z}$ (note that r is left unchanged), one can prove an analogue of Lemma 2.

Lemma 9. *For every cPLDL formula φ there exists an efficiently constructible cPLDL formula $\neg\varphi$ s.t.*

1. *$(w, n, \alpha) \models \varphi$ if, and only if, $(w, n, \alpha) \not\models \neg\varphi$ for every w, every n, and every α, and*
2. *$|\neg\varphi| = |\varphi|$.*

Note that we do not claim that negation preserves well-formedness and that we have not (yet) defined unipolar fragments of cPLDL. This is because this statement is wrong: the negation of the well-formed cPLDL-formula $[([p]_{\leq x} p)?]_{\leq x} p$ is $\langle ([p]_{\leq x} p)? \rangle_{\leq x} \neg p$, which is not well-formed. The issue is that negation does not flip parameterized operators in tests, i.e., formulas of the form $\phi?$ in regular expressions, which also requires us to be careful when defining the unipolar fragments of cPLDL: let φ be a cPLDL formula.

- φ is a cPLDL$_\Diamond$ formula, if it does not contain a parameterized box operator.
- φ is a cPLDL$_\Box$ formula, if it does not contain a parameterized diamond operator *and* if its negation is a cPLDL$_\Diamond$ formula.

For cPLTL$_\mathbf{G}$ formulas, the second conjunct in the second item above is trivial, but, as we have seen in the example above, this is no longer true for cPLDL$_\Box$. The second conjunct is necessary to be able to solve problems for cPLDL$_\Box$ by dualizing the formula into an cPLDL$_\Diamond$ formula. This becomes crucial when we consider the optimization problems in Setion 8, the only place where we deal with cPLDL$_\Box$ formulas.

Finally, Lemma 3 holds for cPLDL, too.

Lemma 10. *Let φ be a cPLDL formula and let α and β be variable valuations satisfying $\alpha(x) \leq \beta(x)$ for every $x \in \text{var}_\mathbf{F}(\varphi)$ and $\alpha(y) \geq \beta(y)$ for every $y \in \text{var}_\mathbf{G}(\varphi)$. If $(w, \alpha) \models \varphi$, then $(w, \beta) \models \varphi$.*

The alternating-color technique is applicable to PLDL [19]: to this end, one introduces changepoint-bounded variants of the unparameterized diamond operator and of the unparameterized box operator whose semantics only quantify over infixes with at most one changepoint. LDL formulas with changepoint-bounded operators can be translated into Büchi automata of exponential size. As usual, the parameterized box operators can again be disregard due to monotonicity. Thus, given a $\text{PLDL}_\Diamond$ formula, one replaces every diamond operator by a changepoint-bounded one and can then show that both formulas are equivalent, provided the distance between color-changes is appropriately bounded and spaced. This allows to extend the algorithms for model checking and realizability based on the alternating-color technique to PLDL. The detailed construction is described in [19].

In the setting with costs investigated here, the approach is similar: one has to replace the parameterized diamond operators by changepoint-bounded ones. Furthermore, the formula $\chi = (\mathbf{GF}p \wedge \mathbf{GF}\neg p) \leftrightarrow \mathbf{GF}\kappa$ used in the applications of the alternating-color technique in Sections 4 and 5 is replaced by an equivalent LDL formula, which is possible as LDL subsumes LTL. The resulting formula is again translatable into a Büchi automaton of exponential size. Thus, the constructions presented in the previous two sections solving the model checking and the game problem are again applicable.

Theorem 3. *The cPLDL model checking problem is* PSpace-*complete and solving infinite games with cPLDL winning conditions is* 2ExpTime-*complete.*

7 Multiple Cost Functions

In this section, we consider parameterized temporal logics with multiple cost functions. These extensions allow to express, e.g., the Streett condition with costs, which is not expressible in the logics we considered thus far. For the sake of simplicity, we restrict our attention to cPLTL, although all results hold for cPLDL, too.

Fix some dimension $d \in \mathbb{N}$. The syntax of mult-cPLTL is obtained by equipping the parameterized temporal operators by a coordinate $i \in \{1, \ldots, d\}$, denoted by $\mathbf{F}_{\leq_i x}$ and $\mathbf{G}_{\leq_i y}$. Here, a cost-trace is of the form $w_0 \, \overline{c}_0 \, w_1 \, \overline{c}_1 \, w_2 \, \overline{c}_2 \cdots$ with $w_n \in 2^P$ and $\overline{c}_n \in \mathbb{N}^d$. Thus, for every $i \in \{1, \ldots, d\}$, we can define

$$\text{cst}_i(w_0\overline{c}_0 \cdots \overline{c}_{n-1}w_n) = \sum_{j=0}^{n-1} (\overline{c}_j)_i$$

for every finite cost-trace $w_0\overline{c}_0 \cdots \overline{c}_{n-1}w_n$, where $(\overline{c}_j)_i$ denotes the i-th entry of the vector $\overline{c}_j$. Furthermore, we require for every coordinate i a proposition κ_i such that $\kappa_i \in w_{n+1}$ if, and only if, $(\overline{c}_n)_i > 0$.

The semantics of atomic formulas, boolean connectives, and unparameterized temporal operators are unchanged and for the parameterized operators, we define

- $(w, n, \alpha) \models \mathbf{F}_{\leq_i z} \varphi$ if, and only if, there exists a $j \geq 0$ with $\mathrm{cst}_i(w_n \bar{c}_n \cdots \bar{c}_{n+j-1} w_{n+j}) \leq \alpha(z)$ such that $(w, n+j, \alpha) \models \varphi$, and
- $(w, n, \alpha) \models \mathbf{G}_{\leq_i z} \varphi$ if, and only if, for every $j \geq 0$ with $\mathrm{cst}_i(w_n \bar{c}_n \cdots \bar{c}_{n+j-1} w_{n+j}) \leq \alpha(z)$: $(w, n+j, \alpha) \models \varphi$.

In this setting, we consider the model checking problem for transition systems with d cost functions and want to solve games in arenas with d cost functions.

Example 2. A Streett condition with costs $(Q_i, P_i)_{i \in \{1,...,d\}}$ [21] can be expressed[4] in mult-cPLTL via

$$\mathbf{FG}\left(\bigwedge\nolimits_{i \in \{1,...,d\}} (Q_i \to \mathbf{F}_{\leq_i x} P_i) \right).$$

This property is not expressible in the logics with a single cost function, which can again be shown by a pumping argument.

Again, we restrict ourselves to formulas where no variable parameterizes an eventually- and an always operator, but we allow a variable to parameterize operators with different coordinates. Furthermore, the fragments mult-cPLTL$_\mathbf{F}$ and mult-cPLTL$_\mathbf{G}$ are defined as for cPLTL, i.e., a formula is a mult-cPLTL$_\mathbf{F}$ formula (a mult-cPLTL$_\mathbf{G}$ formula), if it does not contain parameterized always operators (parameterized eventually operators).

Lemma 2 can be extended to mult-cPLTL by adding the rules $\neg(\mathbf{F}_{\leq_i x}\varphi) = \mathbf{G}_{\leq_i x}\neg\varphi$ and $\neg(\mathbf{G}_{\leq_i y}\varphi) = \mathbf{F}_{\leq_i y}\neg\varphi$ to the proof.

Lemma 11. *For every mult-cPLTL formula φ there exists an efficiently constructible mult-cPLTL formula $\neg\varphi$ s.t.*

1. *$(w, n, \alpha) \models \varphi$ if, and only if, $(w, n, \alpha) \not\models \neg\varphi$ for every w, every n, and every α,*
2. *$|\neg\varphi| = |\varphi|$.*
3. *If φ is well-formed, then so is $\neg\varphi$.*
4. *If φ is an LTL formula, then so is $\neg\varphi$.*
5. *If φ is a mult-cPLTL$_\mathbf{F}$ formula, then $\neg\varphi$ is a mult-cPLTL$_\mathbf{G}$ formula and vice versa.*

Furthermore, Lemma 3 holds for mult-cPLTL as well.

Lemma 12. *Let φ be a mult-cPLTL formula and let α and β be variable valuations satisfying $\alpha(x) \leq \beta(x)$ for every $x \in \mathrm{var}_\mathbf{F}(\varphi)$ and $\alpha(y) \geq \beta(y)$ for every $y \in \mathrm{var}_\mathbf{G}(\varphi)$. If $(w, \alpha) \models \varphi$, then $(w, \beta) \models \varphi$.*

The alternating-color technique is straightforwardly extendable to the new logic mult-cPLTL: one introduces a fresh proposition p_i for each coordinate i and defines $\chi = \bigwedge_{i=1}^{d}((\mathbf{GF}p_i \wedge \mathbf{GF}\neg p_i) \leftrightarrow \mathbf{GF}\kappa_i)$. Furthermore, the notions of

---

[4] The same disclaimer as for the parity condition with costs applies here. See Footnote 1.

i-blocks, k-boundedness in coordinate i, and k-spaceness in coordinate i are defined as expected. Then, the proofs presented in Section 4 and Section 5 can be extended to the setting with multiple cost functions.

In the model checking case, the third component of the set of states of the colored Büchi graph $\mathcal{S} \times \mathfrak{A}$ has the form $2^{\{p_1,\ldots,p_d\}}$, i.e., it is of exponential size. However, this is no problem, as the automaton $\mathfrak{A}$ is already of exponential size. Similarly, in the case of infinite games, each vertex of the original arena has 2^d copies in $\mathcal{A}'$, one for each element in $2^{\{p_1,\ldots,p_d\}}$ allowing Player 0 to produce appropriate colorings with the propositions p_i. The resulting game has an arena of exponential size (in the size of the original arena and of the original winning condition) and an LTL winning condition under blinking semantics. Such a game can still be solved in doubly-exponential time. To this end, one turns the winning condition into a deterministic parity automaton of doubly-exponential size with exponentially many colors, constructs the product of the arena and the parity automaton, which yields a parity game of doubly-exponential size with exponentially many colors. Such a game can be solved in doubly-exponential time [29].

Theorem 4. *The mult-cPLTL model checking problem is* PSpace-*complete and solving infinite games with mult-cPLTL winning conditions is* 2ExpTime-*complete.*

Again, the same results hold for mult-cPLDL, which is defined as expected.

8   Optimization Problems

It is natural to treat model checking and solving games with specifications in parameterized linear temporal logics as an optimization problem: determine the *optimal* variable valuation such that the system satisfies the specification with respect to it. For parameterized eventually operators, we are interested in minimizing the waiting times while for parameterized always', we are interested in maximizing the waiting times. Due to the undecidability results for not well-defined formulas one considers the optimization problems for the unipolar fragments, i.e., for formulas having either no parameterized eventuallies or no parameterized always'. In this section, we present algorithms for such optimization problems given by cPLTL specifications. In the following, we encode the weights of the transition system or arena under consideration in unary to obtain our results. Whether these results can also be shown for a binary encoding is an open question.

For model checking, we are interested in the following four problems: given a transition system $\mathcal{S}$ and a cPLTL$_\mathbf{F}$ formula $\varphi_\mathbf{F}$ and a cPLTL$_\mathbf{G}$ formula $\varphi_\mathbf{G}$, respectively, determine

1. $\min_{\{\alpha \mid \mathcal{S} \text{ satisfies } \varphi_\mathbf{F} \text{ w.r.t. } \alpha\}} \min_{x \in \mathrm{var}_\mathbf{F}(\varphi_\mathbf{F})} \alpha(x)$,
2. $\min_{\{\alpha \mid \mathcal{S} \text{ satisfies } \varphi_\mathbf{F} \text{ w.r.t. } \alpha\}} \max_{x \in \mathrm{var}_\mathbf{F}(\varphi_\mathbf{F})} \alpha(x)$,
3. $\max_{\{\alpha \mid \mathcal{S} \text{ satisfies } \varphi_\mathbf{G} \text{ w.r.t. } \alpha\}} \max_{y \in \mathrm{var}_\mathbf{G}(\varphi_\mathbf{G})} \alpha(y)$, and

4. $\max_{\{\alpha | \mathcal{S} \text{ satisfies } \varphi_{\mathbf{G}} \text{ w.r.t. } \alpha\}} \min_{y \in \text{var}_{\mathbf{G}}(\varphi_{\mathbf{G}})} \alpha(y)$.

Applying the monotonicity of the parameterized operators and (in the first case) the alternating-color technique to all but one variable reduces the four optimization problems to ones where the specification has a single variable (cp. [1]). Furthermore, the upper bounds presented in Corollary 1 and in Lemma 7 yield an exponential search space for an optimal valuation: if this space is empty, then there is no α such that $\mathcal{S}$ satisfies $\varphi_{\mathbf{F}}$ with respect to α in the first two cases. On the other hand, if the search space contains every such α, then $\mathcal{S}$ satisfies $\varphi_{\mathbf{G}}$ with respect to every α in the latter two cases.

Thus, it remains the check whether the specification is satisfied with respect to some valuation that is bounded exponentially. In this setting, one can construct an exponentially sized non-deterministic Büchi automaton recognizing the models of the specification with respect to the given valuation (using a slight adaption of the construction presented in [38] accounting for the fact that we keep track of cost instead of time). This automaton can be checked for non-emptiness in polynomial space using an on-the-fly construction. Thus, an optimal α can be found in polynomial space by binary search.

Theorem 5. *The cPLTL model checking optimization problems can be solved in polynomial space, if the weights are encoded in unary.*

A similar approach works for infinite games as well. Here, we are interested in computing

1. $\min_{\{\alpha | \text{Pl. 0 has winning strategy for } \mathcal{G}_{\mathbf{F}} \text{ w.r.t. } \alpha\}} \min_{x \in \text{var}_{\mathbf{F}}(\varphi_{\mathbf{F}})} \alpha(x)$,
2. $\min_{\{\alpha | \text{Pl. 0 has winning strategy for } \mathcal{G}_{\mathbf{F}} \text{ w.r.t. } \alpha\}} \max_{x \in \text{var}_{\mathbf{F}}(\varphi_{\mathbf{F}})} \alpha(x)$,
3. $\min_{\{\alpha | \text{Pl. 0 has winning strategy for } \mathcal{G}_{\mathbf{G}} \text{ w.r.t. } \alpha\}} \min_{x \in \text{var}_{\mathbf{G}}(\varphi_{\mathbf{G}})} \alpha(x)$, and
4. $\min_{\{\alpha | \text{Pl. 0 has winning strategy for } \mathcal{G}_{\mathbf{G}} \text{ w.r.t. } \alpha\}} \max_{x \in \text{var}_{\mathbf{G}}(\varphi_{\mathbf{G}})} \alpha(x)$.

and witnessing winning strategies for given cPLTL games $\mathcal{G}_{\mathbf{F}}$ with cPLTL$_{\mathbf{F}}$ winning condition $\varphi_{\mathbf{F}}$ and $\mathcal{G}_{\mathbf{G}}$ with cPLTL$_{\mathbf{G}}$ winning condition $\varphi_{\mathbf{G}}$.

Again, one can reduce these problems to the case of winning conditions with a single variable and by applying determinacy of games with respect to a fixed valuation. It even suffices to consider the case of cPLTL$_{\mathbf{F}}$ winning conditions with a single variable, due to duality of games: swapping the players and negating the winning condition in a game with cPLTL$_{\mathbf{G}}$ winning condition yields an equivalent -game with cPLTL$_{\mathbf{F}}$ winning condition. Corollary 2 gives a doubly-exponential upper bound on an optimal variable valuation. Hence, one can construct a deterministic parity automaton of triply-exponential size with exponentially many colors recognizing the models of the specification with respect to a fixed variable valuation α that is below the upper bound (again, see [38] for the detailed construction). Player 0 wins the parity game played in the original arena but using the language of the automaton as winning condition if, and only if, she has a winning strategy for the cPLTL$_{\mathbf{F}}$ game with respect to α. Such a parity game can be solved in triply-exponential time [29].

Theorem 6. *The cPLTL optimization problems for infinite games can be solved in triply-exponential time, if the weights are encoded in unary.*

Furthermore, the same results hold for cPLDL using appropriate adaptions of the automata constructions presented in [19, 20]. Here, we apply the requirement on negations of cPLDL$_\square$ formulas being cPLDL$_\lozenge$ formulas when dualizing a game with cPLDL$_\square$ winning condition into a game with cPLDL$_\lozenge$ winning condition by swapping the players and negating the winning condition. Without this requirement, we would not necessarily end up with a cPLDL$_\lozenge$ winning condition, but possibly with a non-wellformed winning condition.

Theorem 7. *The cPLDL model checking optimization problems can be solved in polynomial space and the cPLDL optimization problems for infinite games can be solved in triply-exponential time, if the weights are encoded in unary.*

However, for parameterized logics with multiple cost functions, these results do not remain valid, as it does not suffice to reduce the optimization problems to ones with a single variable, as a variable may bound operators in different dimensions. Thus, one has to keep track multiple costs, which incurs an additional exponential blow-up when done naively. Whether this can be improved is an open question.

9   Conclusion

We introduced parameterized temporal logics whose operators bound the accumulated cost instead of time as usual: cPLTL and cPLDL extend PLTL and PLDL to the cost-setting while mult-cPLTL and mult-cPLDL extend them to the multi-dimensional cost-setting. The logics we considered here and their inclusions are depicted in Figure 1: the upper four logics were introduced in this work.

Fig. 1. Overview over the logics considered in this work.

All four new logics retain the attractive algorithmic properties of LTL like a polynomial space model checking algorithm and a doubly-exponential time algorithm for solving infinite games. For cPLTL and cPLDL, even the optimization variants of these problems are not harder than for PLTL: polynomial space for model checking and triply-exponential time for solving games, if the weights are encoded in unary.

However, it is open whether these problems are strictly harder for logics with multiple cost functions or if the weights are encoded in binary. Another open question concerns the complexity of the optimization problem for infinite games: can these problems be solved in doubly-exponential time, i.e., is finding optimal variable valuations as hard as solving games? Note that this question is already open for PLTL. Recently, a step towards this direction was made by giving an approximation algorithm for this problem with doubly-exponential running
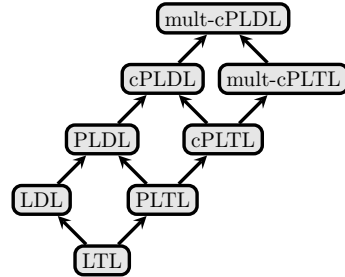
time [31]. Finally, one could consider weights from some arbitrary semiring and corresponding weighted parameterized temporal logics.

References

1. Alur, R., Etessami, K., Torre, S.L., Peled, D.: Parametric temporal logic for "model measuring". ACM Trans. Comput. Log. 2(3), 388–407 (2001)
2. Alur, R., Madhusudan, P.: Visibly pushdown languages. In: Babai, L. (ed.) STOC 04. pp. 202–211. ACM (2004), http://doi.acm.org/10.1145/1007352.1007390
3. Bloem, R., Chatterjee, K., Henzinger, T.A., Jobstmann, B.: Better quality in synthesis through quantitative objectives. In: Bouajjani, A., Maler, O. (eds.) CAV 2009. LNCS, vol. 5643, pp. 140–156. Springer (2009)
4. Bojańczyk, M.: A bounding quantifier. In: Marcinkowski, J., Tarlecki, A. (eds.) CSL 2004. LNCS, vol. 3210, pp. 41–55. Springer (2004)
5. Bojańczyk, M.: Weak MSO with the unbounding quantifier. Theory Comput. Syst. 48(3), 554–576 (2011)
6. Bojanczyk, M.: Weak MSO+U with path quantifiers over infinite trees. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) ICALP 2014 Part II. LNCS, vol. 8573, pp. 38–49. Springer (2014)
7. Bojańczyk, M., Colcombet, T.: Bounds in ω-regularity. In: LICS 2006. pp. 285–296. IEEE Computer Society (2006)
8. Bojańczyk, M., Toruńczyk, S.: Weak MSO+U over infinite trees. In: Dürr, C., Wilke, T. (eds.) STACS 2012. LIPIcs, vol. 14, pp. 648–660. Schloss Dagstuhl–Leibniz-Zentrum für Informatik (2012)
9. Bozzelli, L.: Alternating Automata and a Temporal Fixpoint Calculus for Visibly Pushdown Languages. In: Caires, L., Vasconcelos, V.T. (eds.) CONCUR 2007. LNCS, vol. 4703, pp. 476–491. Springer (2007)
10. Bozzelli, L., Sánchez, C.: Visibly linear temporal logic. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) IJCAR 2014. LNCS, vol. 8562, pp. 418–483. Springer (2014)
11. Bozzelli, L., Sánchez, C.: Visibly rational expressions. Act. Inf. 51(1), 25–49 (2014)
12. Brázdil, T., Chatterjee, K., Kucera, A., Novotný, P.: Efficient controller synthesis for consumption games with multiple resource types. In: Madhusudan, P., Seshia, S.A. (eds.) CAV 2012. LNCS, vol. 7358, pp. 23–38. Springer (2012)
13. Černý, P., Chatterjee, K., Henzinger, T.A., Radhakrishna, A., Singh, R.: Quantitative synthesis for concurrent programs. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 243–259. Springer (2011)
14. Chatterjee, K., Doyen, L.: Energy parity games. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010 Part II. LNCS, vol. 6199, pp. 599–610. Springer (2010)
15. Chatterjee, K., Henzinger, T.A., Horn, F.: Finitary winning in omega-regular games. ACM Trans. Comput. Log. 11(1) (2009)
16. Chatterjee, K., Henzinger, T.A., Jurdziński, M.: Mean-payoff parity games. In: LICS 2005. pp. 178–187. IEEE Computer Society (2005)
17. Colcombet, T.: The theory of stabilisation monoids and regular cost functions. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S.E., Thomas, W. (eds.) ICALP 2009 Part II. LNCS, vol. 5556, pp. 139–150. Springer (2009)
18. De Giacomo, G., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: Rossi, F. (ed.) IJCAI. IJCAI/AAAI (2013)

19. Faymonville, P., Zimmermann, M.: Parametric linear dynamic logic. In: Peron, A., Piazza, C. (eds.) GandALF 2014. EPTCS, vol. 161, pp. 60–73 (2014)
20. Faymonville, P., Zimmermann, M.: Parametric linear dynamic logic (full version). arXiv 1504.03880 (2015), http://arxiv.org/abs/1504.03880, accepted for publication in Information and Computation.
21. Fijalkow, N., Zimmermann, M.: Parity and Streett games with costs. LMCS 10(2) (2014)
22. Kamp, H.W.: Tense Logic and the Theory of Linear Order. Ph.D. thesis, Computer Science Department, University of California at Los Angeles, USA (1968)
23. Kupferman, O., Piterman, N., Vardi, M.Y.: From liveness to promptness. Formal Methods in System Design 34(2), 83–103 (2009)
24. Leucker, M., Sánchez, C.: Regular linear temporal logic. In: Jones, C., Liu, Z., Woodcock, J. (eds.) ICTAC 2007. LNCS, vol. 4711, pp. 291–305. Springer (2007)
25. Mogavero, F., Murano, A., Sorrentino, L.: On promptness in parity games. In: McMillan, K.L., Middeldorp, A., Voronkov, A. (eds.) LPAR 2013. LNCS, vol. 8312, pp. 601–618. Springer (2013)
26. Pnueli, A.: The temporal logic of programs. In: FOCS 1977. pp. 46–57. IEEE (1977)
27. Pnueli, A., Rosner, R.: On the synthesis of a reactive module. In: POPL 1989. pp. 179–190 (1989)
28. Pnueli, A., Rosner, R.: On the synthesis of an asynchronous reactive module. In: Ausiello, G., Dezani-Ciancaglini, M., Rocca, S.R.D. (eds.) ICALP 1989. LNCS, vol. 372, pp. 652–671. Springer (1989)
29. Schewe, S.: Solving parity games in big steps. In: Arvind, V., Prasad, S. (eds.) FSTTCS 2007. LNCS, vol. 4855, pp. 449–460. Springer (2007)
30. Sistla, A.P., Clarke, E.M.: The complexity of propositional linear temporal logics. J. ACM 32(3), 733–749 (1985)
31. Tentrup, L., Weinert, A., Zimmermann, M.: Approximating optimal bounds in Prompt-LTL realizability in doubly-exponential time (2015), arXiv:1511.09450
32. Vanden Boom, M.: Weak cost monadic logic over infinite trees. In: Murlak, F., Sankowski, P. (eds.) MFCS 2011. LNCS, vol. 6907, pp. 580–591. Springer (2011)
33. Vardi, M.Y.: A temporal fixpoint calculus. In: Ferrante, J., Mager, P. (eds.) POPL 88. pp. 250–259. ACM Press (1988), http://doi.acm.org/10.1145/73560.73582
34. Vardi, M.Y.: The rise and fall of LTL. In: D'Agostino, G., Torre, S.L. (eds.) GandALF 2011. EPTCS, vol. 54 (2011)
35. Vardi, M.Y., Wolper, P.: Reasoning about infinite computations. Inf. Comput. 115(1), 1–37 (1994)
36. Weinert, A., Zimmermann, M.: Visibly linear dynamic logic (2015), http://arxiv.org/abs/1512.05177, arXiv:1512.05177
37. Wolper, P.: Temporal logic can be more expressive. Information and Control 56(1–2), 72 – 99 (1983)
38. Zimmermann, M.: Optimal bounds in parametric LTL games. Theor. Comput. Sci. 493, 30–45 (2013)