Reasoning about Quality and Fuzziness of Strategic Behaviors

Patricia Bouyer, Orna Kupferman, Nicolas Markey, Bastien Maubert, Aniello Murano, Giuseppe Perelli

Reasoning about Quality and Fuzziness of Strategic Behaviors

PATRICIA BOUYER, Université Paris-Saclay, CNRS, ENS Paris-Saclay, LMF, France
ORNA KUPFERMAN, Hebrew University, Israel
NICOLAS MARKEY, Irisa, CNRS & Inria & Université de Rennes, France
BASTIEN MAUBERT, Università degli Studi di Napoli "Federico II", Italy
ANIELLO MURANO, Università degli Studi di Napoli "Federico II", Italy
GIUSEPPE PERELLI, Sapienza University of Rome, Italy

Temporal logics are extensively used for the specification of on-going behaviors of computer systems. Two significant developments in this area are the extension of traditional temporal logics with modalities that enable the specification of on-going *strategic* behaviors in multi-agent systems, and the transition of temporal logics to a *quantitative* setting, where different satisfaction values enable the specifier to formalize concepts such as certainty or quality. In the first class, SL (*Strategy Logic*) is one of the most natural and expressive logics describing strategic behaviors. In the second class, a notable logic is LTL[$\mathcal{F}$], which extends LTL with *quality operators*.

In this work we introduce and study SL[$\mathcal{F}$], which enables the specification of quantitative strategic behaviors. The satisfaction value of an SL[$\mathcal{F}$] formula is a real value in $[0, 1]$, reflecting "how much" or "how well" the strategic on-going objectives of the underlying agents are satisfied. We demonstrate the applications of SL[$\mathcal{F}$] in quantitative reasoning about multi-agent systems, showing how it can express and measure concepts like stability in multi-agent systems, and how it generalizes some fuzzy temporal logics. We also provide a model-checking algorithm for SL[$\mathcal{F}$], based on a quantitative extension of Quantified CTL*. Our algorithm provides the first decidability result for a quantitative extension of Strategy Logic. In addition, it can be used for synthesizing strategies that maximize the quality of the systems' behavior.

CCS Concepts: • **Theory of computation → Logic and verification**; **Modal and temporal logics**; **Verification by model checking**; *Automata over infinite objects*; *Tree languages.*

Additional Key Words and Phrases: Synthesis, quantitative, fuzzy

ACM Reference Format:
Patricia Bouyer, Orna Kupferman, Nicolas Markey, Bastien Maubert, Aniello Murano, and Giuseppe Perelli. 2021. Reasoning about Quality and Fuzziness of Strategic Behaviors. *ACM Trans. Comput. Logic* 1, 1 (January 2021), 38 pages. https://doi.org/10.1145/--

1 INTRODUCTION

1.1 Temporal logics

One of the significant developments in formal reasoning has been the use of *temporal logics* for the specification of on-going behaviors of reactive systems [34, 42, 72]. Traditional temporal

Authors' addresses: Patricia Bouyer, Université Paris-Saclay, CNRS, ENS Paris-Saclay, LMF, Gif-sur-Yvette, France; Orna Kupferman, Hebrew University, Jerusalem, Israel; Nicolas Markey, Irisa, CNRS & Inria & Université de Rennes, Rennes, France; Bastien Maubert, Università degli Studi di Napoli "Federico II", Naples, Italy; Aniello Murano, Università degli Studi di Napoli "Federico II", Naples, Italy; Giuseppe Perelli, Sapienza University of Rome, Rome, Italy.

1529-3785/2021/1-ART $15.00
https://doi.org/10.1145/--

logics are interpreted over Kripke structures, modeling closed systems, and can quantify over the computations of the systems in a universal and existential manner. The need to reason about multi-agent systems has led to the development of specification formalisms that enable the specification of on-going strategic behaviors in multi-agent systems [8, 31, 67]. These formalisms, most notably ATL, ATL*, and Strategy Logic (SL), include quantification over strategies of the different agents and of the computations they may force the system into, making it possible to specify concepts that have been traditionally studied in game theory, such as Nash equilibria.

The duration of games in classical game theory is finite and the outcome of the game depends on its final position [69, 70]. In contrast, agents in multi-agent systems maintain an *on-going interaction* with each other [50], and reasoning about their behavior refers not to their final state (in fact, we consider non-terminating systems, with no final state) but rather to the language of infinite computations that they generate. The logics ATL* and SL both extend Linear-time Temporal Logic (LTL) [72], and thus can express rich properties of on-going strategic behaviors. However, their semantics are Boolean: either a system satisfies a formula, or it fails to satisfy it. The Boolean nature of traditional temporal logic is a real obstacle in the context of strategic reasoning. Indeed, while many strategies may attain a desired objective, they may do so at different levels of quality or certainty. Consequently, designers would be willing to give up manual design and adopt automatic procedures only after being convinced that the latter generate systems of comparable quality and certainty. For this to happen, one should first extend the specification formalism to one that supports quantitative aspects of the systems and strategies. We propose such an extension in this work: we merge the most natural and expressive logic for strategic reasoning with a recent, very powerful quantitative extension of LTL, called LTL$[\mathcal{F}]$.

The logic LTL$[\mathcal{F}]$ is a fuzzy logic that augments LTL with quality operators [3]. The satisfaction value of an LTL$[\mathcal{F}]$ formula is a real value in $[0,1]$, intended to measure the quality in which the computation satisfies the specification. For example, the set $\mathcal{F}$ may contain the $\min\{x,y\}$, $\max\{x,y\}$, and $1-x$ functions, which are the standard quantitative analogues of the $\wedge$, $\vee$, and $\neg$ operators, and are known in fuzzy logics as the Zadeh operators. The novelty of LTL$[\mathcal{F}]$ is the ability to manipulate values by arbitrary functions. These can prioritize different scenarios or reduce the satisfaction value of computations that exhibit less desirable behaviors. For example, consider a "carrier" drone c that tries to bring an artifact to a rescue point, while keeping it as far as possible from a "villain" adversarial drone v. They evolve in a three-dimensional unit cube, in which coordinates are triples $\vec{\gamma} = (\gamma_1, \gamma_2, \gamma_3) \in [0,1]^3$. We use the triples of atomic propositions $p_{\vec{\gamma}} = (p_{\gamma_1}, p_{\gamma_2}, p_{\gamma_3})$ and $q_{\vec{\gamma}} = (q_{\gamma_1}, q_{\gamma_2}, q_{\gamma_3})$ to denote the coordinates of c and v, respectively. Assume that $\mathcal{F}$ contains the function $\mathrm{dist}\colon [0,1]^3 \times [0,1]^3 \to [0,1]$, which maps two points in the cube to the (normalized) distance between them, and let the (Boolean) atomic proposition "safe" characterize positions where the artifact has reached the rescue point. In this scenario, the quality of an execution, or path, can be formalized with the following LTL$[\mathcal{F}]$ formula:

$$\psi_{\text{rescue}} = \mathrm{dist}(p_{\vec{\gamma}}, q_{\vec{\gamma}})\ \mathbf{U}\ \text{safe}$$

where $\mathbf{U}$ is the classic "until" operator. Indeed, the satisfaction value of ψ_{rescue} is 0 on every path in which the artifact is never rescued, and otherwise it is the minimum distance between the carrier and the villain along the trajectory from the beginning until the rescue point is reached.

1.2 Our contributions

We introduce and study the logic SL$[\mathcal{F}]$, which can be viewed both as an extension of LTL$[\mathcal{F}]$ to the strategic setting and as an extension of SL to the quantitative setting. Thus, while both LTL$[\mathcal{F}]$ and SL extend LTL, each in a different direction, SL$[\mathcal{F}]$ merges both extensions. The result is a strong yet clean logic that enables the specification of quantitative strategic behaviors. Syntax-wise,

SL[$\mathcal{F}$] lifts LTL[$\mathcal{F}$] to the strategic setting by introducing a strategy quantifier $\langle\langle x \rangle\rangle$, which returns the maximum satisfaction value of the formula in computations that are possible outcomes of the strategy x, and binding operator (a, x), which assigns strategy x to agent a. Then, SL[$\mathcal{F}$] also lifts SL to the quantitative setting by introducing quality operators as in LTL[$\mathcal{F}$]. The semantics of SL[$\mathcal{F}$] is defined with respect to *weighted* multi-agent systems, namely ones where atomic propositions have truth values in $[0, 1]$, reflecting quality or certainty. In addition to introducing the logic, our main contribution is a model-checking procedure for SL[$\mathcal{F}$], which enables formal reasoning about both quality and fuzziness of strategic behaviors. In addition, the model-checking procedure can be used as a synthesis algorithm that produces witness strategies that maximize the value of the formula. As we shall elaborate below, the merging of the two extensions in one logic poses new technical challenges. In particular, the external quantifiers on the strategies makes it impossible to evaluate the formulas in a bottom-up manner.

1.2.1 Specifying strategies' quality. In the example from above, with the carrier drone trying to rescue an artifact and the villain drone trying to steal it, we can specify the quality of a strategy x for the carrier as the minimal quality of the behaviors resulting from x against any strategy y of the villain. If the quality of a behavior is specified by formula ψ_{rescue} as above, then the quality of a strategy x is 0 if the villain has a counter-strategy y to steal the artifact (in which case it never reaches the rescue point); otherwise, it is the minimum distance between the carrier and the villain until the artifact is rescued over all possible counter-strategies y of the villain. We are interested in maximizing the quality of carrier's strategies, which is formalized with the following formula (recall that c and v are agents corresponding to the carrier and the villain):

$$\varphi_{\text{rescue}} = \langle\langle x \rangle\rangle [[y]] (c, x)(v, y)\ \mathbf{A}(\text{dist}(p_{\vec{\gamma}}, q_{\vec{\gamma}})\ \mathbf{U}\ \text{safe})$$

where $[[y]]$ is the dual of $\langle\langle y \rangle\rangle$, which returns the minimum value of the subformula in its scope over all strategies y. Thus, given a strategy x, formula $\varphi_{\text{quality}}(x) = [[y]](c, x)(v, y)\mathbf{A}(\text{dist}(p_{\vec{\gamma}}, q_{\vec{\gamma}})\ \mathbf{U}\ \text{safe})$ minimizes the quality over all possible strategies y of the villain. It is therefore the minimal quality that strategy x is guaranteed to enforce. In the end, formula φ_{rescue} maximizes this value over all possible strategies x of the carrier.

Note that in this example, the formula $\varphi_{\text{quality}}(x)$ does not simply specify the ability of the carrier to behave in some desired manner. Rather, it associates a satisfaction value in $[0, 1]$ with strategy x. This suggests that SL[$\mathcal{F}$] can be used not only to specify strategic behaviors with quantitative objectives, but also for quantizing notions from game theory that are traditionally Boolean. For example. beyond specifying that a strategy profile is a Nash Equilibrium, we can specify how far it is from being such an equilibrium, namely how much an agent may gain by a deviation that witnesses the instability. As we explain later in Section 2.4, we can actually express concepts such as ε-Nash Equilibria [70].

1.2.2 Synthesizing optimal strategies. We consider the following generalisation of the classic model-checking problem: given a weighted concurrent game structure $\mathcal{G}$, an SL[$\mathcal{F}$] formula φ and a predicate $P \subseteq [0, 1]$, does the satisfaction value of φ on $\mathcal{G}$ belong to P? To solve this problem, we employ an approach that recently proved handy in the study of a number of logics for strategic reasoning [13, 45, 61], and which consists in reducing the problem to the model checking of (some appropriate extension of) Quantified CTL* (the extension of CTL* with second-order quantifiers on atomic propositions, QCTL* for short [48, 54, 55, 60, 77]). This is, however, the first time this approach is used in a quantitative setting.

To do so, we first need to define a suitable quantitative extension of Quantified CTL*. We define QCTL*[$\mathcal{F}$] as the extension of CTL*[$\mathcal{F}$] [3] with quantifiers over atomic propositions where, similarly to strategy quantifiers in SL[$\mathcal{F}$], existential/universal quantifications on atomic propositions

are seen as maximizations/minimizations over possible valuations of the quantified propositions on the models. We show that the ability to assign the quantified atomic propositions with arbitrary values in $[0, 1]$ enables the specification of legal tiling of grids of unbounded dimensions, making the model-checking problem for QCTL$^\star[\mathcal{F}]$ undecidable. For our purpose of SL$[\mathcal{F}]$ model checking, however, where quantification over atomic propositions encodes quantification over strategies, we show that one can restrict attention to Booleanly-Quantified CTL$^\star[\mathcal{F}]$ (BQCTL$^\star[\mathcal{F}]$ for short), where the quantified atomic propositions are assigned Boolean values in $\{0, 1\}$. For BQCTL$^\star[\mathcal{F}]$, we are able to solve the model-checking problem, providing a solution also to SL$[\mathcal{F}]$ model checking. We also study additional properties of QCTL$^\star[\mathcal{F}]$ and BQCTL$^\star[\mathcal{F}]$, focusing on their *Lipschitz continuity*, namely the effect of perturbing the values of atomic propositions on the satisfaction value of formulas.

A general approach to CTL* model checking is to repeatedly evaluate the innermost state subformulas by viewing them as (existentially or universally quantified) LTL formulas, and replace them with fresh atomic propositions [44]. This directly extends to CTL$^\star[\mathcal{F}]$, using weighted fresh atomic propositions [3]. However this technique does not work for BQCTL$^\star[\mathcal{F}]$: indeed, the truth values of the innermost formulas depend on the values of the externally quantified atomic propositions. Therefore, we build upon both the automata-theoretic approach to CTL* model checking [57] and the word-automata construction developed for LTL$[\mathcal{F}]$ [3], extending the latter from infinite words to infinite trees. More precisely, given a BQCTL$^\star[\mathcal{F}]$ formula φ and a predicate $P \subseteq [0, 1]$, we construct an alternating parity tree automaton that accepts exactly all the labeled trees t such that the satisfaction value of φ on t is in P. The translation, and hence the complexity of our model-checking algorithm, is non-elementary: we show that the problem is actually $(k + 1)$-Exptime-complete for formulas involving up to k nested quantifications on atomic propositions; we show a similar complexity result for SL$[\mathcal{F}]$, in terms of nesting of strategy quantifiers. Similarly to LTL$[\mathcal{F}]$ [3], our complexity results hold as long as the quality operators in $\mathcal{F}$ can be computed in the complexity class considered. Otherwise, they are the computational bottleneck.

Finally we observe that, as is often the case for this sort of algorithms based on tree automata [80], whenever the answer to the model-checking problem is positive, we can synthesize (outermost existentially-quantified) witness strategies. More precisely, if a formula $\varphi = \langle\langle x_1 \rangle\rangle \dots \langle\langle x_n \rangle\rangle \psi$ starts with a sequence of existentially-quantified strategies, and it holds that the satisfaction value of φ in some weighted game $\mathcal{G}$ belongs to $P \subseteq [0, 1]$, then our algorithm can be used to synthesize strategies $x_1, \dots, x_n$ that maximize within P the quality as specified by ψ.

1.3 Related work

There have been long lines of works about games with quantitative objectives (in a broad sense), *e.g.* stochastic games [46, 76], timed games [11], or weighted games with various kinds of objectives (parity [43], mean-payoff [41] or energy [17, 28]). This does not limit to zero-sum games, but also includes the study of various solution concepts (see for instance [6, 22, 24, 26, 79]). Similarly, extensions of the classical temporal logics LTL and CTL with quantitative semantics have been studied in different contexts, with discounting [2, 36], averaging [16, 20], or richer constructs [3, 15]. In contrast, the study of quantitative temporal logics for strategic reasoning has remained rather limited: works on LTL$[\mathcal{F}]$ include algorithms for synthesis and rational synthesis [3, 4, 6, 7], but no logics combine the quantitative aspect of LTL$[\mathcal{F}]$ with the strategic reasoning offered by SL and, to the best of our knowledge, our model-checking algorithm for SL$[\mathcal{F}]$ is the first decidability result for a quantitative extension of a strategic specification formalism (unless restricting to bounded-memory strategies).

Baier and others have focused on a variant of SL in a stochastic setting [12]; model checking was proven decidable for memoryless strategies, and undecidable in the general case. A quantitative version of SL with Boolean goals over one-counter games has been considered in [18]; only a periodicity property was proven, and no model-checking algorithm is known in that setting as well. Finally, Graded SL [10] extends SL by quantifying on the number of strategies witnessing a given strategy quantifier, and is decidable.

The other quantitative extensions we know of concern ATL /ATL*, and most of the results are actually adaptations of similar (decidability) results for the corresponding extensions of CTL and CTL*; this includes probabilistic ATL [32], timed ATL [25, 52], multi-valued ATL [53], and weighted versions of ATL [27, 62, 81]. Finally, some works have considered non-quantitative ATL with quantitative constraints on the set of allowed strategies [1, 38], proving decidability of the model-checking problem.

2 QUANTITATIVE STRATEGY LOGIC

Let Σ be an alphabet. A *finite* (resp. *infinite*) *word* over Σ is an element of Σ^* (resp. Σ^ω). The *length* of a finite word $w = w_0 w_1 \dots w_n$ is $|w| := n+1$, and $\text{last}(w) := w_n$ is its last letter. Given a finite (resp. infinite) word w and $0 \leq i < |w|$ (resp. $i \in \mathbb{N}$), we let w_i be the letter at position i in w, $w_{\leq i} = w_0 \dots w_i$ is the (nonempty) prefix of w that ends at position i and $w_{\geq i} = w_i w_{i+1} \dots$ is the suffix of w that starts at position i. As usual, for any partial function f, we write $\text{dom}(f)$ for the domain of f.

Strategy logic with functions, denoted SL[$\mathcal{F}$], generalizes both SL [31, 67] and LTL[$\mathcal{F}$] [3] by replacing the Boolean operators of SL with arbitrary functions over $[0,1]$. The logic is actually a family of logics, each parameterised by a set $\mathcal{F}$ of functions.

2.1 Syntax

We build on the branching-time variant of SL [45], which does not add expressiveness with respect to the classic semantics [67] but presents several benefits (see [45] for more details), one of them being that it makes the connection with Quantified CTL tighter.

Definition 2.1. Let $\mathcal{F} \subseteq \{f\colon [0,1]^m \to [0,1] \mid m \in \mathbb{N}\}$ be a set of functions over $[0,1]$ of possibly different arities. The syntax of SL[$\mathcal{F}$] is defined with respect to a finite set of *atomic propositions* AP, a finite set of agents Agt and a set of strategy variables Var. The set of SL[$\mathcal{F}$] formulas is defined by the following grammar:

$$\varphi ::= p \mid \langle\!\langle x \rangle\!\rangle \varphi \mid (a,x)\varphi \mid \mathbf{E}\psi \mid f(\varphi, \dots, \varphi)$$
$$\psi ::= \varphi \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi \mid f(\psi, \dots, \psi)$$

where $p \in \text{AP}$, $x \in \text{Var}$, $a \in \text{Agt}$, and $f \in \mathcal{F}$.

Formulas of type φ are called *state formulas*, those of type ψ are called *path formulas*. Formulas $\langle\!\langle x \rangle\!\rangle \varphi$ are called *strategy quantifications* whereas formulas $(a,x)\varphi$ are called *bindings*. Modalities $\mathbf{X}$ and $\mathbf{U}$ are temporal modalities, which take a specific quantitative semantics as we see below.

We may use $\top$, $\vee$, and $\neg$ to denote functions 1, max and $1-x$, respectively. We can then define the following classic abbreviations: $\bot := \neg\top$, $\varphi \wedge \varphi' := \neg(\neg\varphi \vee \neg\varphi')$, $\varphi \to \varphi' := \neg\varphi \vee \varphi'$, $\mathbf{F}\psi := \top \mathbf{U} \psi$, $\mathbf{G}\psi := \neg\mathbf{F}\neg\psi$, $\mathbf{A}\psi := \neg\mathbf{E}\neg\psi$ and $[\![x]\!]\varphi := \neg\langle\!\langle x \rangle\!\rangle \neg\varphi$.

Intuitively, the value of formula $\varphi \vee \varphi'$ is the maximal value of the two formulas φ and φ', $\varphi \wedge \varphi'$ takes the minimal value of the two formulas, and the value of $\neg\varphi$ is one minus that of φ. The implication $\varphi \to \varphi'$ thus takes the maximal value between that of φ' and one minus that of φ.

In a Boolean setting, we assume that the values of the atomic propositions are in $\{0,1\}$ (0 represents false whereas 1 represents true), and so are the output values of functions in $\mathcal{F}$. One can then check that $\varphi \vee \varphi'$, $\varphi \wedge \varphi'$, $\neg\varphi$ and $\varphi \rightarrow \varphi'$ take their usual Boolean meaning.

We will come back later to temporal modalities, strategy quantifications and bindings.

2.2 Semantics

While SL is evaluated on classic concurrent game structures with Boolean valuations for atomic propositions, SL[$\mathcal{F}$] formulas are interpreted on weighted concurrent game structures, in which atomic propositions have values in $[0,1]$. We first define classic concurrent game structures, and then extend them to the quantitative setting.

Definition 2.2. A *concurrent game structure* (CGS) is a tuple $\mathcal{G} = \langle \mathrm{Agt}, \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta \rangle$ where Agt is a finite set of agents, Act is a finite set of actions, V is a finite set of states, $\mathrm{v}_\iota \in \mathrm{V}$ is an initial state and $\Delta \colon \mathrm{V} \times \mathrm{Act}^{\mathrm{Agt}} \rightarrow \mathrm{V}$ is the transition function.

An element of $\mathrm{Act}^{\mathrm{Agt}}$ is a *joint action*, and we let $\mathrm{succ}(\mathrm{v}) = \{\mathrm{v}' \in \mathrm{V} \mid \exists \vec{c} \in \mathrm{Act}^{\mathrm{Agt}}.\ \mathrm{v}' = \Delta(\mathrm{v}, \vec{c})\}$ for each $\mathrm{v} \in \mathrm{V}$. For the sake of simplicity, we assume in the sequel that $\mathrm{succ}(\mathrm{v}) \neq \varnothing$ for all $\mathrm{v} \in \mathrm{V}$.

A *play* in $\mathcal{G}$ is an infinite sequence $\pi = (\mathrm{v}_i)_{i \in \mathbb{N}}$ of states in V such that $\mathrm{v}_0 = \mathrm{v}_\iota$ and $\mathrm{v}_i \in \mathrm{succ}(\mathrm{v}_{i-1})$ for all $i > 0$. We write $\mathrm{Play}_{\mathcal{G}}$ for the set of plays in $\mathcal{G}$, and $\mathrm{Play}_{\mathcal{G}}(v)$ for the set of plays in $\mathcal{G}$ starting from v. In this and all similar notations, we might omit to mention $\mathcal{G}$ when it is clear from the context. A (strict) prefix of a play π is a finite sequence $\rho = (\mathrm{v}_i)_{0 \leq i \leq L}$, for some $L \in \mathbb{N}$, which we denote $\pi_{\leq L}$. We write $\mathrm{Prefx}(\pi)$ for the set of strict prefixes of play π. Such finite prefixes are called *histories*, and we let $\mathrm{Hist}_{\mathcal{G}}(\mathrm{v}) = \mathrm{Prefx}(\mathrm{Play}_{\mathcal{G}}(\mathrm{v}))$ and $\mathrm{Hist}_{\mathcal{G}} = \bigcup_{\mathrm{v} \in \mathrm{V}} \mathrm{Hist}_{\mathcal{G}}(\mathrm{v})$. We extend the notion of strict prefixes and the notation Prefx to histories in the natural way, requiring in particular that $\rho \notin \mathrm{Prefx}(\rho)$. A (finite) extension of a history ρ is any history ρ' such that $\rho \in \mathrm{Prefx}(\rho')$.

A *strategy* is a mapping $\sigma \colon \mathrm{Hist}_{\mathcal{G}} \rightarrow \mathrm{Act}$, and we write $\mathrm{Str}_{\mathcal{G}}$ for the set of strategies in $\mathcal{G}$. An *assignment* is a partial function $\chi \colon \mathrm{Var} \cup \mathrm{Agt} \rightharpoonup \mathrm{Str}_{\mathcal{G}}$, that assigns strategies to variables and agents. The assignment $\chi[a \mapsto \sigma]$ maps a to σ and is equal to χ otherwise. Let χ be an assignment and ρ a history. We define the set of *outcomes* of χ from ρ as the set $\mathrm{Out}(\chi, \rho)$ of plays $\pi = \rho \cdot \mathrm{v}_1\mathrm{v}_2 \ldots$ such that for every $i \in \mathbb{N}$, there exists a joint action $\vec{c} \in \mathrm{Act}^{\mathrm{Agt}}$ such that for each agent $a \in \mathrm{dom}(\chi)$, $\vec{c}(a) = \chi(a)(\pi_{\leq |\rho|+i-1})$ and $\mathrm{v}_{i+1} = \Delta(\mathrm{v}_i, \vec{c})$, where $\mathrm{v}_0 = \mathrm{last}(\rho)$.

Definition 2.3. A *weighted concurrent game structure* (WCGS) is a tuple $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Agt}, \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta, \mathrm{w} \rangle$ where AP is a finite set of atomic propositions, $\langle \mathrm{Agt}, \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta \rangle$ is a CGS, and $\mathrm{w} \colon \mathrm{V} \rightarrow [0,1]^{\mathrm{AP}}$ is a weight function.

In a WCGS, for each position $\mathrm{v} \in \mathrm{V}$ and atomic proposition $p \in \mathrm{AP}$, the value $\mathrm{w}(\mathrm{v})(p)$ indicates the degree to which p holds in v. We now define the semantics of SL[$\mathcal{F}$].

Definition 2.4. Consider a WCGS $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Agt}, \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta, \mathrm{w} \rangle$, a set of variables Var, and a partial assignment χ of strategies for Agt and Var. Given an SL[$\mathcal{F}$] state formula φ and a history ρ, we use $\{\!\{\varphi\}\!\}^{\mathcal{G}}_{\chi}(\rho)$ to denote the satisfaction value of φ in the last state of ρ under the assignment χ. Given an SL[$\mathcal{F}$] path formula ψ, a play π, and a point in time $i \in \mathbb{N}$, we use $\{\!\{\psi\}\!\}^{\mathcal{G}}_{\chi}(\pi, i)$ to denote the satisfaction value of ψ in the suffix of π that starts in position i. The satisfaction value is defined inductively as follows:

$$\{\!\{p\}\!\}^{\mathcal{G}}_{\chi}(\rho) = \mathrm{w}(\mathrm{last}(\rho))(p)$$

$$\{\!\{\langle\!\langle x \rangle\!\rangle \varphi\}\!\}^{\mathcal{G}}_{\chi}(\rho) = \sup_{\sigma \in \mathrm{Str}_{\mathcal{G}}} \{\!\{\varphi\}\!\}^{\mathcal{G}}_{\chi[x \mapsto \sigma]}(\rho)$$

$$\{\!\{(a, x)\varphi\}\!\}^{\mathcal{G}}_{\chi}(\rho) = \{\!\{\varphi\}\!\}^{\mathcal{G}}_{\chi[a \mapsto \chi(x)]}(\rho)$$

$$\begin{aligned}
\{\!|\mathbf{E}\psi|\!\}^{\mathcal{G}}_{\chi}(\rho) &= \sup_{\pi \in \mathrm{Out}(\chi,\rho)} \{\!|\psi|\!\}^{\mathcal{G}}_{\chi}(\pi, |\rho| - 1) \\
\{\!|f(\varphi_1, \ldots, \varphi_m)|\!\}^{\mathcal{G}}_{\chi}(\rho) &= f(\{\!|\varphi_1|\!\}^{\mathcal{G}}_{\chi}(\rho), \ldots, \{\!|\varphi_m|\!\}^{\mathcal{G}}_{\chi}(\rho)) \\
\{\!|\varphi|\!\}^{\mathcal{G}}_{\chi}(\pi, i) &= \{\!|\varphi|\!\}^{\mathcal{G}}_{\chi}(\pi_{\leq i}) \\
\{\!|\mathbf{X}\psi|\!\}^{\mathcal{G}}_{\chi}(\pi, i) &= \{\!|\psi|\!\}^{\mathcal{G}}_{\chi}(\pi, i+1) \\
\{\!|\psi_1 \mathbf{U} \psi_2|\!\}^{\mathcal{G}}_{\chi}(\pi, i) &= \sup_{j \geq i} \min\left(\{\!|\varphi_2|\!\}^{\mathcal{G}}_{\chi}(\pi, j), \min_{k \in [i, j-1]} \{\!|\varphi_1|\!\}^{\mathcal{G}}_{\chi}(\pi, k)\right) \\
\{\!|f(\psi_1, \ldots, \psi_m)|\!\}^{\mathcal{G}}_{\chi}(\pi, i) &= f(\{\!|\psi_1|\!\}^{\mathcal{G}}_{\chi}(\pi, i), \ldots, \{\!|\psi_m|\!\}^{\mathcal{G}}_{\chi}(\pi, i))
\end{aligned}$$

Strategy quantification $\langle\!\langle x \rangle\!\rangle \varphi$ computes the maximal value a choice of strategy for variable x can give to formula φ. Dually, $[\![x]\!]\varphi$ computes the minimal value a choice of strategy for variable x can give to formula φ. Binding $(a, x)\varphi$ just assigns strategy given by x to agent a. The branching quantifier $\mathbf{E}\psi$ computes the supremum value on all possible outcomes of the strategies currently assigned. Temporal modality $\mathbf{X}\psi$ takes the value of ψ at the next step of a given outcome, while $\psi_1 \mathbf{U} \psi_2$ maximizes, over all positions along the play, the minimum between the value of ψ_2 at that position and the minimal value of ψ_1 before this position.

In a Boolean setting, we recover the standard semantics of SL. Also the fragment of SL$[\mathcal{F}]$ with only temporal operators and functions $\vee$ and $\neg$ corresponds to Fuzzy Linear-time Temporal Logic [49, 59]. Note that by equipping $\mathcal{F}$ with adequate functions, we can capture various classic fuzzy interpretations of Boolean operators, such as the Zadeh, Gödel-Dummett or Łukasiewicz interpretations (see for instance [49]). However the interpretation of the temporal operators is fixed in our logic.

Remark 1. As we shall see, once we fix a finite set of possible satisfaction values for the atomic propositions in a formula φ, as is the case when a model is chosen, the set of possible satisfaction values for subformulas of φ becomes finite. Therefore, the suprema in the above definition are in fact maxima.

For a state formula φ and a weighted game structure $\mathcal{G}$, we write $\{\!|\varphi|\!\}^{\mathcal{G}}$ for $\{\!|\varphi|\!\}^{\mathcal{G}}_{\emptyset}(\mathrm{v}_\iota)$.

2.3 Model checking

The problem we are interested in is the following generalisation of the model checking problem, which is solved in [3] for LTL$[\mathcal{F}]$ and CTL$^\star[\mathcal{F}]$.

Definition 2.5 (Model-checking problem). Given an SL$[\mathcal{F}]$ state formula φ, a WCGS $\mathcal{G}$ and a predicate $P \subseteq [0, 1]$, decide whether $\{\!|\varphi|\!\}^{\mathcal{G}} \in P$.

In the sequel, we require that deciding whether a rational value v is in P can be done in time polynomial in the size of the representation of v and P. As a typical example, if P is a finite union of intervals with rational bounds (given explicitly), then membership in P can be decided in linear time.

The precise complexity of the model-checking problem will be stated in terms of *block nesting depth* of formulas, which we introduce as a relaxation of the usual *nesting depth*. While the nesting depth of a formula is the maximal number of nested quantifiers it contains, the block nesting depth of a formula φ, written $\mathrm{bnd}(\varphi)$, counts blocks of quantifiers of the same polarity as one. This notion is quite close to the usual *alternation depth*, except that for block nesting depth each different nested block of quantifiers counts as one, even if they have the same polarity.

Definition 2.6. Formally, $\mathrm{bnd}(\varphi)$ is defined inductively as follows:

$$\begin{aligned}
\mathrm{bnd}(p) &= 0\\
\mathrm{bnd}(\langle\!\langle x\rangle\!\rangle\varphi) &= \begin{cases}\mathrm{bnd}(\varphi) & \text{if } \varphi = \langle\!\langle y\rangle\!\rangle\varphi' \text{ for some } y \in \mathrm{Var}\\ 1+\mathrm{bnd}(\varphi) & \text{otherwise}\end{cases}\\
\mathrm{bnd}((a,x)\varphi) &= \mathrm{bnd}(\varphi)\\
\mathrm{bnd}(f(\varphi_1,\cdots,\varphi_k)) &= \max\{\mathrm{bnd}(\varphi_i) \mid 1 \le i \le k\}\\
\mathrm{bnd}(\mathbf{E}\psi) &= \mathrm{bnd}(\psi)\\
\mathrm{bnd}(\mathbf{X}\psi) &= \mathrm{bnd}(\psi)\\
\mathrm{bnd}(\psi\mathbf{U}\psi') &= \max\{\mathrm{bnd}(\psi),\mathrm{bnd}(\psi')\}
\end{aligned}$$

Example 2.7. Here are a few examples of block-nesting depth of some formulas.

$$\begin{aligned}
\mathrm{bnd}(\langle\!\langle x\rangle\!\rangle\langle\!\langle y\rangle\!\rangle(a,x)(b,y)\ \mathbf{EF}p) &= 1\\
\mathrm{bnd}(\langle\!\langle x\rangle\!\rangle\langle\!\langle y\rangle\!\rangle\langle\!\langle z\rangle\!\rangle(a,x)(b,y)(c,z)\ \mathbf{EF}p) &= 1\\
\mathrm{bnd}(\langle\!\langle x\rangle\!\rangle\langle\!\langle y\rangle\!\rangle[\![z]\!](a,x)(b,y)(c,z)\ \mathbf{EF}p) &= 2\\
\mathrm{bnd}(\langle\!\langle x\rangle\!\rangle\langle\!\langle y\rangle\!\rangle(a,x)(b,y)\ \mathbf{EF}\ \langle\!\langle z\rangle\!\rangle(c,z)\ \mathbf{AG}q) &= 2
\end{aligned}$$

Theorem 2.8. *The model-checking problem for* $\mathrm{SL}[\mathcal{F}]$ *is decidable. It is* $(k+1)$-Exptime*-complete for formulas of block nesting depth at most* k.

2.4 What can SL[$\mathcal{F}$] express?

$\mathrm{SL}[\mathcal{F}]$ naturally embeds SL. Indeed, if the values of the atomic propositions are in $\{0,1\}$ and the only allowed functions in $\mathcal{F}$ are $\vee$, $\wedge$, and $\neg$, then the satisfaction value of the formula is in $\{0,1\}$ and coincides with the value of the corresponding SL formula. Below we illustrate how quantities enable the specification of rich strategic properties.

2.4.1 Drone battle. A "carrier" drone c helped by a "guard" drone g try to bring an artifact to a rescue point and keep it away from the "villain" adversarial drone v. They evolve in a three-dimensional unit cube, in which coordinates are triples $\vec{\gamma} = (\gamma_1,\gamma_2,\gamma_3) \in [0,1]^3$. We use the triples of atomic propositions $p_{\vec{\gamma}} = (p_{\gamma_1},p_{\gamma_2},p_{\gamma_3})$ and $q_{\vec{\gamma}} = (q_{\gamma_1},q_{\gamma_2},q_{\gamma_3})$ to denote the coordinates of c and v, respectively. Write $\mathrm{dist}\colon [0,1]^3 \times [0,1]^3 \to [0,1]$ for the (normalized) distance between two points in the cube. Let the (Boolean) atomic proposition "safe" denote that the artifact has reached the rescue point. In $\mathrm{SL}[\mathcal{F}]$, we can capture the level of safety for the artifact, defined as the minimum distance between the carrier and the villain along a trajectory to reach the rescue point. Indeed, the formula

$$\varphi_{\text{rescue}} = \langle\!\langle x\rangle\!\rangle\langle\!\langle y\rangle\!\rangle(c,x)(g,y)\ \mathbf{A}(\mathrm{dist}(p_{\vec{\gamma}},q_{\vec{\gamma}})\mathbf{U}\ \text{safe})$$

states that the carrier and guard drones cooperate to keep the villain as far as possible from the artifact, until it is rescued. Note that the satisfaction value of the $\mathrm{LTL}[\mathcal{F}]$ specification is 0 along any path in which the artifact is never rescued.

Notice that formula φ_{rescue} above can be expressed in the extension $\mathrm{ATL}^\star[\mathcal{F}]$ of $\mathrm{ATL}^\star$ with functions. $\mathrm{ATL}^\star$ is a syntactic fragment of SL in which strategy quantifications are constrained to be of the following form:

$$\langle\!\langle x_1\rangle\!\rangle\cdots\langle\!\langle x_k\rangle\!\rangle[\![x_{k+1}]\!]\cdots[\![x_n]\!](a_{\nu(1)},x_1)\cdots(a_{\nu(n)},x_n)\ \varphi$$

where $\mathrm{Agt} = \{x_i \mid 1 \le i \le n\}$ and $\nu\colon [1,n] \to [1,n]$ is a permutation. In other terms, existential strategy quantification in $\mathrm{ATL}^\star$ (and $\mathrm{ATL}^\star[\mathcal{F}]$) assigns a new strategy to a subset of the players,

and drops the strategies previously assigned to the other players. Notice that in φ_{rescue}, the universal strategy quantification and assignment for Agent v is implicit, and is actually hidden in the path quantifier $\mathbf{A}$: φ_{rescue} is equivalent to

$$\langle\langle x\rangle\rangle\langle\langle y\rangle\rangle[[z]](c,x)(g,y)(v,z)\ \mathbf{A}(\text{dist}(p_{\vec{\gamma}}, q_{\vec{\gamma}})\mathbf{U}\ \text{safe})$$

The strategies of the carrier and the guard being quantified before that of the villain implies that they are unaware of the villain's future moves. Now assume the guard is a double agent to whom the villain communicates its plan. Then its strategy can depend on the villain's strategy, which is captured by the following formula:

$$\varphi_{\text{spy}} = \langle\langle x\rangle\rangle[[z]]\langle\langle y\rangle\rangle(c,x)(g,y)(v,z)\ \mathbf{A}(\text{dist}(p_{\vec{\gamma}}, q_{\vec{\gamma}})\ \mathbf{U}\ \text{safe})$$

Note that this formula φ_{spy} cannot be expressed in ATL$^\star[\mathcal{F}]$, that cannot capture alternation of strategy quantification because each strategic quantifier resets previously assigned strategies, while SL$[\mathcal{F}]$ inherits from SL the possibility to freely alternate existential and universal quantifications on strategies in a first-order fashion. In fact φ_{spy} actually belongs to the fragment SL$_{1G}[\mathcal{F}]$, which we study in Section 6.

2.4.2 Synthesis with quantitative objectives. The problem of synthesis for LTL specifications dates back to [73]. The setting is simple: two agents, a controller and an environment, operate on two disjoint sets of variables in the system. The controller wants a given LTL specification ψ to be satisfied in the infinite execution, while the environment wants to prevent it. The problem consists in synthesizing a strategy for the controller such that, no matter the behavior of the environment, the resulting execution satisfies ψ. Recently, this problem has been addressed in the context of LTL$[\mathcal{F}]$, where the controller aims at maximizing the value of an LTL$[\mathcal{F}]$ formula ψ, while the environment acts as minimizer. Both problems can be easily represented in SL and SL$[\mathcal{F}]$ respectively, with the formula

$$\varphi_{\text{synt}} = \langle\langle x\rangle\rangle[[y]]\ (c,x)(e,y)\ \mathbf{A}\psi$$

where c and e are the controller and environment agent, respectively, and ψ the temporal specification expressed in either LTL or LTL$[\mathcal{F}]$.

Assume now that controller and environment are both composed of more than one agent, namely $c_1, \ldots, c_n$ and $e_1, \ldots, e_n$, and each controller component has the power to adjust its strategic choice based on the strategies selected by the environmental agents of lower rank. That is, the strategy selected by agent c_k depends on the strategies selected by agents e_j, for every $j < k$. We can write an SL$[\mathcal{F}]$ formula to represent this generalized synthesis problem as follows:

$$\varphi_{\text{synt}} = \langle\langle x_1\rangle\rangle[[y_1]] \cdots \langle\langle x_n\rangle\rangle[[y_n]]\ (c_1,x_1)(e_1,y_1)\ldots(c_n,x_n)(e_n,y_n)\ \mathbf{A}\psi.$$

Notice that every controller agent is bound to an existentially quantified variable, that makes it try to maximize the satisfaction value of the formula in its scope. On the other hand, environmental agents are bound to universally quantified variables, which makes them try to minimize the satisfaction value.

In general in SL, each alternation between existential and universal quantification on strategies yields an additional exponential in the complexity of the model-checking problem. In Section 6 we show that, for the special case of formulas of the form φ_{synt}, such alternation does not affect the computational complexity of the model-checking problem.

2.4.3 Nash equilibria in weighted games. An important feature of SL in terms of expressiveness is that it captures Nash equilibria (NEs, for short) and other common solution concepts. This extends to SL$[\mathcal{F}]$, but in a much stronger sense: first, objectives in SL$[\mathcal{F}]$ are quantitative, so that *profitable deviation* is not a simple Boolean statement; second, the semantics of the logic is quantitative, so

that *being a NE* is a quantitative property, and we can actually express *how far* a strategy profile is from being a NE.

Assume that the objective of each agent $a_i \in \mathrm{Agt}$ is given as an LTL[$\mathcal{F}$] formula ψ_i. Define function $\preceq\colon [0,1]^2 \to [0,1]$ such that (using infix notation) $\alpha \preceq \beta$ equals 1 if $\alpha \leq \beta$, and equals 0 otherwise. We define the following formula, for a family $(x_i)_{a_i \in \mathrm{Agt}}$ of variables:

$$\varphi_{\mathrm{NE}}((x_i)_{a_i\in\mathrm{Agt}}) = (a_1,x_1)\dots(a_n,x_n) \bigwedge_{a_i\in\mathrm{Agt}} [\![y_i]\!]\ ((a_i,y_i)\ \mathbf{A}\psi_i) \preceq \mathbf{A}\psi_i$$

If assignment χ maps each x_i to some strategy σ_i, then the strategy profile $(\sigma_i)_{a_i\in\mathrm{Agt}}$ is a NE if, and only if, φ_{NE} evaluates to 1 under assignment χ. Indeed, because $[\![y_i]\!]$ minimizes its subformula over all strategies y_i, formula $[\![y_i]\!]\ ((a_i,y_i)\ \mathbf{A}\psi_i) \preceq \mathbf{A}\psi_i$ evaluates to 1 if, and only if, for all strategies y_i, $((a_i,y_i)\ \mathbf{A}\psi_i) \preceq \mathbf{A}\psi_i$ evaluates to 1, i.e., y_i is not a profitable deviation for agent a_i. Notice that, since $\langle\!\langle y_i \rangle\!\rangle$ maximizes over y_i, formula[1] $(\langle\!\langle y_i \rangle\!\rangle\ (a_i,y_i)\ \mathbf{A}\psi_i) \preceq \mathbf{A}\psi_i$ evaluates to 1 if, and only if, the best deviation for a_i is not profitable, and thus no deviations are profitable. Thus we could equivalently characterise NEs with

$$\varphi'_{\mathrm{NE}}((x_i)_{a_i\in\mathrm{Agt}}) = (a_1,x_1)\dots(a_n,x_n) \bigwedge_{a_i\in\mathrm{Agt}} (\langle\!\langle y_i \rangle\!\rangle\ (a_i,y_i)\ \mathbf{A}\psi_i) \preceq \mathbf{A}\psi_i$$

2.4.4 ε-equilibria in weighted games. Adopting a more quantitative point of view, we can measure how much agent i can benefit from a selfish deviation using formula $\langle\!\langle y_i \rangle\!\rangle \mathrm{diff}((a_i,y_i)\psi_i,\psi_i)$, where $\mathrm{diff}(x,y) = \max\{0, x-y\}$[2]. The maximal benefit that some agent may get is then captured by the following formula:

$$\varphi_{\overline{\mathrm{NE}}}((x_i)_{a_i\in\mathrm{Agt}}) = \langle\!\langle y \rangle\!\rangle\ (a_1,x_1)\dots(a_n,x_n) \bigvee_{a_i\in\mathrm{Agt}} \mathrm{diff}((a_i,y)\mathbf{A}\psi_i, \mathbf{A}\psi_i).$$

As previously, a strategy profile $(\sigma_i)_{a_i\in\mathrm{Agt}}$ is a NE if, and only if, under the assignment χ that maps each x_i to σ_i, formula $\varphi_{\overline{\mathrm{NE}}}$ evaluates to 0. Interestingly, formula $\varphi_{\overline{\mathrm{NE}}}$ can be used to characterise ε-NE: a strategy profile in χ is an ε-NE if, and only if, $\varphi_{\overline{\mathrm{NE}}}$ takes value less than or equal to ε under assignment χ.

2.4.5 Secure equilibria in weighted games. Secure equilibria [30] are special kinds of NEs in two-player games, where besides improving their objectives, the agents also try to harm their opponent. Following the ideas above, we characterise secure equilibria in SL[$\mathcal{F}$] as follows:

$$\varphi_{\mathrm{SE}}(x_1,x_2) = (a_1,x_1)(a_2,x_2) \bigwedge_{i\in\{1,2\}} [\![y]\!]((a_i,y)\mathbf{A}\psi_1, (a_i,y)\mathbf{A}\psi_2) \preceq_i (\mathbf{A}\psi_1, \mathbf{A}\psi_2)$$

where $(\alpha_1,\alpha_2) \preceq_i (\beta_1,\beta_2)$ is 1 when $(\alpha_i \leq \beta_i) \vee (\alpha_i = \beta_i \wedge \alpha_{3-i} \leq \beta_{3-i})$, and 0 otherwise.

Secure equilibria have also been studied in $\mathbb{Q}$-weighted games [26]: in that setting, the objective of the agents is to optimize e.g. the (limit) infimum or supremum of the sequence of weights encountered along the play. We can characterise secure equilibria in such setting (after first applying an affine transformation to have all weights in $[0,1]$): indeed, assuming that weights are encoded as the value of atomic proposition w, the value of formula $\mathbf{G}w$ is the infimum of the weights, while the value of $\mathbf{FG}w$ is the limit infimum. We can then characterise secure equilibria with (limit) infimum and supremum objectives by using those formulas as the objectives for the agents in formula φ_{SE}.

[1] Notice how parentheses are placed differently here, including $\langle\langle\, y_i \,\rangle\rangle$ in the first argument of $\preceq$.

[2] We use max here only because our functions are required to return values in $[0,1]$. In the way we use diff in $\varphi_{\overline{\mathrm{NE}}}$, the first argument will always be larger than or equal to the second one.

Other classical properties of games can be expressed, such as doomsday equilibria (which generalise winning secure equilibria in n-player games) [29], robust Nash equilibria (considering profitable deviations of coalitions of agents) [21], or strategy dominance and admissibility [14, 23], to cite a few.

2.4.6 Rational synthesis. Weak rational synthesis [6, 47, 56] aims at synthesizing a strategy profile for a controller c_0 and the n components $(c_i)_{1\leq i\leq n}$ constituting the environment, in such a way that (1) the whole system satisfies some objective ψ_0, and (2) under the strategy of the controller, the strategies of the n components form a Nash equilibrium (or any given solution concept) for their own individual objectives $(\psi_i)_{1\leq i\leq n}$.

That a given strategy profile $(x_i)_{c_i\in \mathrm{Agt}}$ satisfies the two conditions above can be expressed as follows:

$$\varphi_{\mathrm{wRS}}((x_i)_{0\leq i\leq n}) = (c_0, x_0)(c_1, x_1)\dots(c_n, x_n)[\mathbf{A}\psi_0 \wedge \varphi_{\mathrm{NE}}((x_i)_{1\leq i\leq n})]$$

The formula returns the minimum between the satisfaction value of ψ_0 and that of $\varphi_{\mathrm{NE}}((x_i)_{1\leq i\leq n})$. Thus, the satisfaction value of φ_{wRS} is zero if the strategy profile $(x_i)_{1\leq i\leq n}$ is not a NE under strategy x_0 assigned to c_0, and it returns the satisfaction value of ψ_0 under the whole strategy profile otherwise. Then the value of

$$\langle\langle x_0\rangle\rangle\langle\langle x_1\rangle\rangle\dots\langle\langle x_n\rangle\rangle\ \varphi_{\mathrm{wRS}}((x_i)_{0\leq i\leq n})$$

is the best value that the system can collectively achieve for ψ_0 under the condition that the components in the environment follow an NE. Obviously, we can go beyond NE and use any other solution concept that can be expressed in SL[$\mathcal{F}$].

The counterpart of weak rational synthesis is strong rational synthesis, which aims at synthesizing a strategy only for controller c_0 in such a way that the objective ψ_0 is maximized over the worst NE that can be played by the environment component over the strategy of c_0 itself. This can be expressed as follows:

$$\varphi_{\mathrm{sRS}}(x_0) = [[x_1]]\dots[[x_n]]\ (c_0, x_0)(c_1, x_1)\dots(c_n, x_n)\ \mathbf{A}[\neg\varphi_{\mathrm{NE}}((x_i)_{1\leq i\leq n}) \vee \psi_0]$$

The disjunction in this formula returns the maximum value between $\neg\varphi_{\mathrm{NE}}((x_i)_{1\leq i\leq n})$ and ψ_0: if $(x_i)_{1\leq i\leq n}$ is a NE, it returns the value of ψ_0, and it returns 1 otherwise. The value of$\varphi_{\mathrm{sRS}}(x_0)$ then is the smallest value that ψ_0 may take when players $(a_i)_{1\leq i\leq n}$ play according to a NE (if any). Finally, the value of

$$\langle\langle x_0\rangle\rangle\varphi_{\mathrm{sRS}}(x_0)$$

is the best value of ψ_0 that the controller can achieve under the condition that the components in the environment are playing the NE that worsens it (or it is 1 if the controller can enforce that no NEs exist).

2.4.7 Social-welfare reasoning. As shown in [6], the objective formula ψ_0 above can be used to capture well-studied social-welfare functions [70]. Typical examples of that are the *utilitarian social-welfare function*

$$\psi_0^{\mathrm{util}} = \frac{1}{n}\sum_{i=1}^{n}\psi_i$$

taking the average payoff of the agents, and the *egalitarian social-welfare function*

$$\psi_0^{\mathrm{egal}} = \min_{i=1,\dots,n}\{\psi_i\}$$

taking the minimum among the payoffs of the agents. Generally, utilitarian and egalitarian functions reflects the two ends of the political spectrum in terms of economic view. Therefore, a combination of the two functions can be used to represent what lies in the middle. Observe that SL[$\mathcal{F}$] can also

evaluate such combinations of the two functions. For instance, a linear combination is represented as

$$\psi_0^{\text{linear}} = \lambda \cdot \psi_0^{\text{util}} + (1-\lambda) \cdot \psi_0^{\text{egal}}$$

where the parameter λ ranges between 0 and 1. As noted in [6], when ψ_0 represents a social-welfare function, the weak and strong Rational-Synthesis problems amount to maximizing the welfare in the collaborative and non-collaborative scenarios, respectively.

The expressiveness of SL[$\mathcal{F}$] allows us to move even further. For a given social-welfare function represented by a formula ψ_0 and a strategy x_0 for the controller, the formula

$$\text{OPT}(x_0) = \langle\langle x_1 \rangle\rangle \dots \langle\langle x_n \rangle\rangle (c_1, x_1) \dots (c_n, x_n) \psi_0$$

denotes the *parameterised social optimum*, that is, the best value of the social-welfare function, when the strategy x_0 of the controller c_0 is fixed.

For a given game, the *Price of Stability* (PoS) is defined as the ratio between the (parameterised) social optimum and the best value of the social-welfare function over the strategy profiles that constitute a Nash Equilibrium. On the other hand, the *Price of Anarchy* (PoA) is the ratio between the social optimum and the worst value of the social welfare over all Nash Equilibria of the game. Observe that, since the social optimum is the best value achievable, both ratios are greater than 1 [3]. Thus, the inverses of PoS and PoA are between 0 and 1 and can then be represented in SL[$\mathcal{F}$] as follows:

$$\text{PoS}^{-1}(x_0) = \frac{\langle\langle x_1 \rangle\rangle \dots \langle\langle x_n \rangle\rangle (\varphi_{\text{NE}}(x_1, \dots, x_n) \wedge \psi_0)}{\text{OPT}(x_0)}$$

$$\text{PoA}^{-1}(x_0) = \frac{[[x_1]] \dots [[x_n]] (\neg\varphi_{\text{NE}}(x_1, \dots, x_n) \vee \psi_0)}{\text{OPT}(x_0)}$$

Therefore, equivalently to minimizing such prices, the controller is interested in maximizing their inverses. We have the following

$$\text{BestPoS} = \langle\langle x_0 \rangle\rangle PoS(x_0) \qquad\qquad \text{BestPoA} = \langle\langle x_0 \rangle\rangle PoA(x_0)$$

2.4.8 Core equilibria. In cooperative game theory, *core equilibrium* is probably the best-known solution concept and sometimes related to the one of Nash Equilibrium for non-cooperative games. Differently from NEs (but similarly to Strong NEs) it accounts multilateral deviations (also called coalition deviations) that, in order to be beneficial, must improve the payoff of the deviating agents *no matter* what is the reaction of the opposite coalition. More formally, for a given strategy profile $(x_i)_{a_i \in \text{Agt}}$, a coalition $C \subseteq \text{Agt}$ has a *beneficial deviation* $(y_i)_{a_i \in C}$ if, for all strategy profiles $(z_i)_{a_i \in \text{Agt} \setminus C}$ and for all $a_i \in C$, it holds that $(x_i)_{a_i \in \text{Agt}} \psi_i \prec (y_i)_{a_i \in C} (z_i)_{a_i \in \text{Agt} \setminus C} \psi_i$. We say that a strategy profile $(x_i)_{a_i \in \text{Agt}}$ is a *core equilibrium* if, for every coalition, there is no beneficial deviation. This can be written in SL[$\mathcal{F}$] as follows:

$$\varphi_{\text{core}}(x_i)_{a_i \in \text{Agt}} = \bigwedge_{C \subseteq \text{Agt}} [[y_i]]_{a_i \in C} \langle\langle y_i \rangle\rangle_{a_j \in \text{Agt} \setminus C} \left(\bigwedge_{a_j \in C} (a_i, y_i)_{a_i \in \text{Agt}} \psi_j \preceq (a_i, x_i)_{a_i \in \text{Agt}} \psi_j \right)$$

The strategy profile $(x_i)_{a_i \in \text{Agt}}$ is a core equilibrium if, and only if, the formula $\varphi_{\text{core}}(x_i)_{a_i \in \text{Agt}}$ evaluates to 1. The existence of a core equilibrium could then be expressed with the formula $\langle\langle x_1 \rangle\rangle \dots \langle\langle x_n \rangle\rangle \varphi_{\text{core}}(x_i)_{a_i \in \text{Agt}}$, which takes value 1 if, and only if, there exists a core equilibrium.

[3] This is assuming that payoffs are always positive, which is the case in this paper.

2.4.9 Accumulated payoffs. One class of problems that cannot be captured by SL[$\mathcal{F}$] are those involving payoffs that are accumulated along time, such as mean payoff or discounted payoff games [83]. To capture them we would need a way to express sums of payoffs over infinitely many time steps. While we could assume that $\mathcal{F}$ contains necessary arithmetic operations such as sum and product, finitary formulas could only express sums over finitely many time steps. On the other hand, the semantics of the until operator involves payoffs in infinitely many time steps, but its semantics does not allow computing any kind of sum over these.

3 QUANTIFIED CTL*[$\mathcal{F}$]

In this section we introduce Quantified CTL*[$\mathcal{F}$] (QCTL*[$\mathcal{F}$], for short) which extends both CTL*[$\mathcal{F}$] and QCTL* [60]. On the one hand, it extends CTL*[$\mathcal{F}$] with second order quantification over atomic propositions, on the other hand it extends QCTL* to the quantitative setting of CTL*[$\mathcal{F}$]. In the setting with quantitative atomic propositions, the natural semantics for propositional quantification is to range over the whole interval $[0,1]$; however, it is also meaningful to have propositional quantification restricted to $\{0,1\}$. This leads to two logics, QCTL*[$\mathcal{F}$] and BQCTL*[$\mathcal{F}$] respectively, that have different properties. In Section 4, we show that the model-checking problem for BQCTL*[$\mathcal{F}$] is decidable. Then, in Section 5, we show that Boolean quantification is sufficient in order to capture quantification on strategies, and so we obtain the decidability of SL[$\mathcal{F}$] model checking by a reduction to BQCTL*[$\mathcal{F}$]. In order, however, to capture extensions of SL[$\mathcal{F}$] with quantitative strategies such as probabilistic strategies (see [9] for a probabilistic extension of SL), one needs the full power of QCTL*[$\mathcal{F}$]. In Section 7.2, we show that the model-checking problem for QCTL*[$\mathcal{F}$] is undecidable when considering the tree semantics, which corresponds to strategies with perfect recall (see Section 3.3).

3.1 Syntax

Let $\mathcal{F} \subseteq \{f\colon [0,1]^m \to [0,1] \mid m \in \mathbb{N}\}$ be a set of functions over $[0,1]$.

Definition 3.1. The syntax of QCTL*[$\mathcal{F}$] is defined with respect to a finite set AP of atomic propositions, using the following grammar:

$$\varphi ::= p \mid \exists p.\, \varphi \mid \mathbf{E}\psi \mid f(\varphi, \dots, \varphi)$$
$$\psi ::= \varphi \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi \mid f(\psi, \dots, \psi)$$

where p ranges over AP and f over $\mathcal{F}$.

Formulas of type φ are called *state formulas*, those of type ψ are called *path formulas*, and QCTL*[$\mathcal{F}$] consists of all the state formulas defined by the grammar. An atomic proposition which is not under the scope of a quantification is called *free*. If no atomic proposition is free in a formula φ, then we say that φ is *closed*. We again use $\top$, $\vee$, and $\neg$ to denote functions 1, max and $1 - x$, as well as classic abbreviations already introduced for SL[$\mathcal{F}$].

3.2 Semantics

QCTL*[$\mathcal{F}$] formulas are evaluated on unfoldings of weighted Kripke structures.

Definition 3.2. A *weighted Kripke structure* (WKS) is a tuple $\mathcal{K} = \langle \mathrm{AP}, \mathrm{S}, \mathrm{s}_\iota, R, \mathrm{w}\rangle$ where AP is a finite set of atomic propositions, S is a finite set of states, $\mathrm{s}_\iota \in \mathrm{S}$ is an initial state, $R \subseteq \mathrm{S} \times \mathrm{S}$ is a left-total[4] transition relation, and $\mathrm{w}\colon \mathrm{S} \to [0,1]^{\mathrm{AP}}$ is a weight function.

[4] *i.e.*, for all $\mathrm{s} \in \mathrm{S}$, there exists s' such that $(\mathrm{s}, \mathrm{s}') \in R$.

A *path* in $\mathcal{K}$ is an infinite word $\pi = \pi_0\pi_1\dots$ over S such that $\pi_0 = \mathrm{s}_\iota$ and $(\pi_i, \pi_{i+1}) \in R$ for all i. By analogy with concurrent game structures we call finite prefixes of paths *histories*, and write $\mathrm{Hist}_\mathcal{K}$ for the set of all histories in $\mathcal{K}$. We also let $\mathrm{Val}_\mathcal{K} = \{\mathrm{w}(\mathrm{s})(p) \mid \mathrm{s} \in \mathrm{S} \text{ and } p \in \mathrm{AP}\}$ be the finite set of values appearing in $\mathcal{K}$.

Given finite nonempty sets D of directions, AP of atomic propositions, and $\mathcal{V} \subseteq [0,1]$ of possible values, a $\mathcal{V}^{\mathrm{AP}}$-labeled D-tree, (or $(\mathcal{V}^{\mathrm{AP}}, D)$-tree for short, or $\mathcal{V}^{\mathrm{AP}}$-tree when directions are understood), is a pair $t = (\tau, \mathrm{w})$ where $\tau \subseteq D^+$ is closed under non-empty prefixes, all *nodes* $u \in \tau$ start with the same direction r, called the *root*, and have at least one *child* $u \cdot d \in \tau$, and $\mathrm{w} : \tau \to \mathcal{V}^{\mathrm{AP}}$ is a *weight function*. We let $\mathrm{Val}_t \subseteq \mathcal{V}$ be the image of w on τ. A *branch* $\lambda = u_0u_1\dots$ is an infinite sequence of nodes such that for all $i \geq 0$, u_{i+1} is a child of u_i. We let $\mathrm{Br}(u)$ be the set of branches that start in node u.

A tree $t = (\tau, \mathrm{w})$ is *complete* if for all node $u \in \tau$ and direction $d \in D$, we have $u \cdot d \in \tau$. Given a $\mathcal{V}^{\mathrm{AP}}$-labeled D-tree $t = (\tau, \mathrm{w})$, we let $\overline{t} = (\overline{\tau}, \overline{\mathrm{w}})$ be the only complete $\mathcal{V}^{\mathrm{AP}} \cup \{\bullet\}$-labeled D-tree such that for all $u \in \tau$, $\overline{\mathrm{w}}(u) = \mathrm{w}(u)$, and for all $u \in \overline{\tau} \setminus \tau$, $\overline{\mathrm{w}}(u) = \{\bullet\}$, where $\bullet$ is a fresh symbol that labels artificial nodes added to make the tree complete. The reason for this definition is that it is more convenient to define tree automata on complete trees. A *pointed tree* is a pair (t, u) where u is a node in t. We say that a tree $t = (\tau, \mathrm{w})$ is *Boolean in* p if for all $u \in \tau$ we have $\mathrm{w}(u)(p) \in \{0,1\}$. As with weighted Kripke structures, we let $\mathrm{Val}_t = \{\mathrm{w}(u)(p) \mid u \in \tau \text{ and } p \in \mathrm{AP}\}$.

Let $p \in \mathrm{AP}$. A *p-labeling* for a $(\mathcal{V}^{\mathrm{AP}}, D)$-tree $t = (\tau, \mathrm{w})$ is a mapping $\ell_p : \tau \to [0,1]$. Letting $\mathcal{V}' = \ell_p(\tau)$, the composition of t with ℓ_p is the $((\mathcal{V} \cup \mathcal{V}')^{\mathrm{AP}}, D)$-tree defined as $t \otimes \ell_p := (\tau, \mathrm{w}')$, where $\mathrm{w}'(u)(p) = \ell_p(u)$ and $\mathrm{w}'(u)(q) = \mathrm{w}(u)(q)$ for $q \neq p$. If $\ell_p(\tau) \subseteq \{0,1\}$, we call ℓ_p a *Boolean p-labeling*.

Finally, the *tree unfolding of a weighted Kripke structure* $\mathcal{K}$ over atomic propositions AP and states S is the $\mathrm{Val}_\mathcal{K}^{\mathrm{AP}}$-labeled S-tree $t_\mathcal{K} = (\mathrm{Hist}_\mathcal{K}, \mathrm{w}')$, where $\mathrm{w}'(u) = \mathrm{w}(\mathrm{last}(u))$ for every $u \in \mathrm{Hist}_\mathcal{K}$.

Definition 3.3 (Semantics). Consider finite sets D of directions, AP of atomic propositions, and $\mathcal{V} \subseteq [0,1]$ of possible values. We fix a $\mathcal{V}^{\mathrm{AP}}$-labeled D-tree $t = (\tau, \mathrm{w})$. Given a QCTL$^\star[\mathcal{F}]$ state formula φ and a node u of t, we use $\{\!\{\varphi\}\!\}^t(u)$ to denote the satisfaction value of φ in node u. Given a QCTL$^\star[\mathcal{F}]$ path formula ψ and a branch λ of τ, we use $\{\!\{\psi\}\!\}^t(\lambda)$ to denote the satisfaction value of ψ along λ. The satisfaction value is defined inductively as follows:

$$
\begin{aligned}
\{\!\{p\}\!\}^t(u) &= \mathrm{w}(u)(p)\\
\{\!\{\exists p.\,\varphi\}\!\}^t(u) &= \sup_{\ell_p : \tau \to [0,1]} \{\!\{\varphi\}\!\}^{t \otimes \ell_p}(u)\\
\{\!\{\mathbf{E}\psi\}\!\}^t(u) &= \sup_{\lambda \in \mathrm{Br}(u)} \{\!\{\psi\}\!\}^t(\lambda)\\
\{\!\{f(\varphi_1, \dots, \varphi_n)\}\!\}^t(u) &= f(\{\!\{\varphi_1\}\!\}^t(u), \dots, \{\!\{\varphi_n\}\!\}^t(u))\\
\{\!\{\varphi\}\!\}^t(\lambda) &= \{\!\{\varphi\}\!\}^t(\lambda_0) \text{ where } \lambda_0 \text{ is the first node of } \lambda\\
\{\!\{\mathbf{X}\psi\}\!\}^t(\lambda) &= \{\!\{\psi\}\!\}^t(\lambda_{\geq 1})\\
\{\!\{\psi_1 \mathbf{U} \psi_2\}\!\}^t(\lambda) &= \sup_{i \geq 0} \min(\{\!\{\psi_2\}\!\}^t(\lambda_{\geq i}), \min_{0 \leq j < i} \{\!\{\psi_1\}\!\}^t(\lambda_{\geq j}))\\
\{\!\{f(\psi_1, \dots, \psi_n)\}\!\}^t(\lambda) &= f(\{\!\{\psi_1\}\!\}^t(\lambda), \dots, \{\!\{\psi_n\}\!\}^t(\lambda))
\end{aligned}
$$

BQCTL* has the same syntax as QCTL*. Its semantics, which we write $\{\!\{.\}\!\}^t_{\mathrm{B}}(.)$, only differs with $\{\!\{.\}\!\}^t(.)$ in the case of quantification on propositions, which is defined as follows:

$$
\{\!\{\exists p.\,\varphi\}\!\}^t_{\mathrm{B}}(u) = \sup_{\ell_p : \tau \to \{0,1\}} \{\!\{\varphi\}\!\}^{t \otimes \ell_p}_{\mathrm{B}}(u)
$$

For a tree t with root r we write $\{\!\{\varphi\}\!\}^t$ for $\{\!\{\varphi\}\!\}^t(r)$, for a weighted Kripke structure $\mathcal{K}$ we write $\{\!\{\varphi\}\!\}^{\mathcal{K}}$ for $\{\!\{\varphi\}\!\}^{t_{\mathcal{K}}}$, and similarly for $\{\!\{\varphi\}\!\}^t_{\mathrm{B}}$ and $\{\!\{\varphi\}\!\}^{\mathcal{K}}_{\mathrm{B}}$.

We will also consider the logic QLTL$[\mathcal{F}]$ and its fragment LTL$[\mathcal{F}]$, which we define now.

Definition 3.4. The syntax of QLTL$[\mathcal{F}]$ is defined with respect to a finite set AP of atomic propositions, using the following grammar:

$$\psi ::= p \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi \mid \exists p.\, \varphi \mid f(\psi, \ldots, \psi)$$

where p ranges over AP and f over $\mathcal{F}$.

The semantics of a QLTL$[\mathcal{F}]$ formula is defined on an infinite sequence of weight assignments for atoms in AP: given a path $\pi \in ([0,1]^{\mathrm{AP}})^\omega$, the satisfaction value of a QLTL formula ψ over π is defined as expected and written $\{\!\{\psi\}\!\}(\pi)$. We write $\psi \equiv \psi'$ if $\{\!\{\psi\}\!\}(\pi) = \{\!\{\psi'\}\!\}(\pi)$ for all $\pi \in ([0,1]^{\mathrm{AP}})^\omega$, and we define the satisfaction value $\{\!\{\psi\}\!\}(\mathcal{K})$ of a QLTL$[\mathcal{F}]$ formula ψ on a Kripke structure $\mathcal{K}$ as the infimum of its satisfaction value over all paths of $\mathcal{K}$.

The logic LTL$[\mathcal{F}]$ is the fragment of QLTL$[\mathcal{F}]$ obtained by removing quantification on atomic propositions.

3.3 Discussion of the semantics

Several semantics for QCTL* exist, among which the *structure semantics*, in which valuations for quantified atomic propositions are chosen on the states of the Kripke structure, and the *tree semantics*, where the Kripke structure is first unfolded in an infinite tree and valuations are chosen independently on each node. The former allows reasoning about *memoryless* strategies, which only depend on the current position, while the latter can be used to reason about *perfect recall* strategies that depend on the entire history (see [60] for more detail). Since in this work we consider prefect-recall strategies, we based the semantics of QCTL$^\star[\mathcal{F}]$ (and thus BQCTL$^\star[\mathcal{F}]$) on the tree semantics for QCTL*.

Actually, if $\mathcal{F} = \{\top, \vee, \neg\}$, then BQCTL$^\star[\mathcal{F}]$ evaluated on Boolean Kripke structures corresponds precisely to QCTL* with tree semantics [60].

Note that even with quantification restricted to Boolean valuations, BQCTL$^\star[\mathcal{F}]$ is still quantitative: instead of merely stating the existence of a satisfying Boolean p-labeling, $\exists p.\, \varphi$ maximizes the satisfaction value of φ over all possible Boolean p-labelings.

BQCTL$^\star[\mathcal{F}]$ can be seen as a fragment of QCTL$^\star[\mathcal{F}]$, if $\mathcal{F}$ contains a function to test whether atomic propositions have only Boolean values. More precisely, let Bool : $[0,1] \to [0,1]$ be the function that maps x to 1 if $x \in \{0,1\}$, and to 0 otherwise. One can express Boolean second-order quantification using fuzzy second-order quantification and the Bool function as follows:

$$\exists^{\mathrm{B}} p.\varphi := \exists p.(\mathbf{AG}\ \mathrm{Bool}(p) \wedge \varphi)$$

We then have that for all $\varphi \in$ QCTL$^\star[\mathcal{F}]$ such that φ does not contain a propositional quantifier, for all $(\mathcal{V}^{\mathrm{AP}}, D)$-tree t and node $u \in t$, $\{\!\{\exists^{\mathrm{B}} p.\varphi\}\!\}^t(u) = \{\!\{\exists p.\varphi\}\!\}^t_{\mathrm{B}}(u)$. It follows that:

Lemma 3.5. *For every* BQCTL* *formula* φ*, letting* φ_B *be the* QCTL* *formula obtained from* φ *by replacing inductively each* $\exists p.\varphi'$ *in* φ *with* $\exists^B p.\varphi'_B$*, we have that for all* $(\mathcal{V}^{\mathrm{AP}}, D)$*-tree* t *and node* $u \in t$*,*

$$\{\!\{\varphi\}\!\}^t_B(u) = \{\!\{\varphi_B\}\!\}^t(u)$$

Therefore BQCTL$^\star[\mathcal{F}] \subseteq$ QCTL$^\star[\mathcal{F} \cup \{\mathrm{Bool}\}]$, in the sense that every formula in BQCTL$^\star[\mathcal{F}]$ has an equivalent in QCTL$^\star[\mathcal{F} \cup \{\mathrm{Bool}\}]$. We now announce our main results on the model-checking problem for these logics. Together, they imply that the latter inequality is strict, in the sense that there is no computable translation from QCTL$^\star[\mathcal{F}]$ to BQCTL$^\star[\mathcal{F}]$.

3.4 Model-checking problem

Definition 3.6. The quantitative model-checking problem for QCTL$^\star[\mathcal{F}]$ (resp. BQCTL$^\star[\mathcal{F}]$) consists in, given a QCTL$^\star[\mathcal{F}]$ state formula φ, a weighted Kripke structure $\mathcal{K}$, and a predicate $P \subseteq [0,1]$, decide whether $\{\!\{\varphi\}\!\}^{\mathcal{K}} \in P$ (resp. $\{\!\{\varphi\}\!\}^{\mathcal{K}}_{\mathrm{B}} \in P$).

Similarly to SL$[\mathcal{F}]$, the precise complexity of the model-checking problem will be stated in terms of *block nesting depth* of formulas, which counts the maximal number of nested blocks of propositional quantifiers of same polarity in a formula φ, and is written $\mathrm{bnd}(\varphi)$.

We show that when quantification on propositions is restricted to Boolean values, then model checking is decidable for arbitrary quality operators (Theorem 3.7), but allowing quantification over arbitrary values leads to undecidability already for a very small set of simple quality operators (Theorem 3.8). The restriction to Boolean values allows us to establish a finiteness result about the possible satisfaction values of a formula (Lemma 4.3), which is central in our automata construction to solve the model-checking problem. In fact, restricting quantification to any fixed finite set of values would be enough to obtain the finiteness result, and thus Theorem 3.7 could be generalized to that case. The proof that we present in the next section can be extended easily, only by generalizing slightly the projection operation on tree automata (see Proposition 4.1) and adapting Lemma 4.3.

Theorem 3.7. *The quantitative model-checking problem for* BQCTL$^\star[\mathcal{F}]$ *is decidable. It is* $(k+1)$-Exptime*-complete for formulas of block nesting depth at most* k.

Theorem 3.8. *The quantitative model-checking problem for* QCTL$^\star[\mathcal{F}]$ *is undecidable if* $\mathcal{F}$ *contains the functions* $(x,y) \mapsto x = y$ *and* $(x,y) \mapsto y = x/2$.

Theorem 3.7 is proved in the next section. Together with a reduction from SL$[\mathcal{F}]$ to BQCTL$^\star[\mathcal{F}]$ that we present in Section 5, it entails the decidability of model checking SL$[\mathcal{F}]$ announced in Theorem 2.8. We prove Theorem 3.8 in Section 7.1, where we also establish some additional results on the expressivity and continuity of QCTL$^\star[\mathcal{F}]$.

4 MODEL CHECKING BQCTL$^\star[\mathcal{F}]$

In this section we first set up usual definitions for automata on infinite trees. We then prove a technical lemma on the finiteness of possible satisfaction values for BQCTL$^\star[\mathcal{F}]$, which is crucial in the automata construction that we then present to solve the model-checking problem.

4.1 Alternating parity tree automata

We recall alternating parity tree automata. We start with basic definitions for two-player turn-based parity games, or simply parity games, that we use to define the semantics of automata.

Parity games. A *parity game* is a structure $G = (V, E, \mathrm{v}_\iota, C)$, where the set of states $V = V_E \uplus V_A$ is partitioned between states of Eve (V_E) and those of Adam (V_A), $E \subseteq V \times V$ is a set of *moves*, v_ι is an initial state and $C : V \to \mathbb{N}$ is a *coloring function* of finite codomain. In states V_E, Eve chooses the next state, while Adam chooses in states V_A. A play is an infinite sequence of states $\mathrm{v}_0\mathrm{v}_1\mathrm{v}_2\ldots$ such that $\mathrm{v}_0 = \mathrm{v}_\iota$ and for all $i \geq 0$, $(\mathrm{v}_i, \mathrm{v}_{i+1}) \in E$ (written $\mathrm{v}_i \to \mathrm{v}_{i+1}$). We assume that for every $\mathrm{v} \in V$ there exists $\mathrm{v}' \in V$ such that $\mathrm{v} \to \mathrm{v}'$. A strategy for Eve is a partial function $V^* \rightharpoonup V$ that maps each finite prefix of a play ending in a state $\mathrm{v} \in V_E$ to a next state v' such that $\mathrm{v} \to \mathrm{v}'$. A play $\mathrm{v}_0\mathrm{v}_1\mathrm{v}_2\ldots$ *follows* a strategy σ of Eve if for every $i \geq 0$ such that $\mathrm{v}_i \in V_E$, $\mathrm{v}_{i+1} = \sigma(\mathrm{v}_0\ldots\mathrm{v}_i)$. A strategy σ is winning if every play that follows it satisfies the parity condition, i.e., the least color seen infinitely often along the play is even.

Parity tree automata. For a set Z, $\mathbb{B}^+(Z)$ is the set of formulas built from the elements of Z as atomic propositions using the connectives $\vee$ and $\wedge$, and with $\top, \bot \in \mathbb{B}^+(Z)$. For convenience, we only define automata on complete trees.

Fix a set $\mathcal{V} \subseteq [0,1]$. An *alternating parity tree automaton (APT)* on $(\mathcal{V}^{\mathrm{AP}}, D)$-trees is a tuple $\mathcal{A} = (Q, \delta, q_\iota, C)$ where Q is a finite set of states, $q_\iota \in Q$ is an initial state, $\delta : Q \times \mathcal{V}^{\mathrm{AP}} \to \mathbb{B}^+(D \times Q)$ is a transition function, and $C : Q \to \mathbb{N}$ is a coloring function. A *nondeterministic (resp. universal) parity tree automaton (NPT, resp. UPT)* is an APT $\mathcal{A} = (Q, \delta, q_\iota, C)$ such that for every $q \in Q$ and $a \in \mathcal{V}^{\mathrm{AP}}$, $\delta(q, a)$ is written in disjunctive (resp. conjunctive) normal form and for every direction $s \in D$ each disjunct (resp. conjunct) contains exactly one element of $\{s\} \times Q$.[5] We may use $\mathcal{N}$ and $\mathcal{U}$ to denote, respectively, nondeterministic and universal automata. An NPT is *deterministic* if for each $q \in Q$ and $a \in 2^{\mathrm{AP}}$, $\delta(q, a)$ consists of a single disjunct.

Acceptance of a pointed $(\mathcal{V}^{\mathrm{AP}}, D)$-tree (t, u_ι), where $t = (\tau, \mathrm{w})$, by an APT $\mathcal{A} = (Q, \delta, q_\iota, C)$ is defined via the parity game $G(\mathcal{A}, t, u_\iota) = (V, E, \mathrm{v}_\iota, C')$ where $V = \tau \times Q \times \mathbb{B}^+(D \times Q)$, state (u, q, α) belongs to Eve if α is of the form $\alpha_1 \vee \alpha_2$ or $[s, q']$, and to Adam otherwise, $\mathrm{v}_\iota = (u_\iota, q_\iota, \delta(q_\iota, u_\iota))$, and $C'(u, q, \alpha) = C(q)$. Moves in $G(\mathcal{A}, t, u_\iota)$ are defined by the following rules:

$$(u, q, \alpha_1 \dagger \alpha_2) \to (u, q, \alpha_i) \quad \text{where } \dagger \in \{\vee, \wedge\} \text{ and } i \in \{1, 2\},$$
$$(u, q, [s, q']) \to (u \cdot s, q', \delta(q', \mathrm{w}(u \cdot s)))$$

States of the form $(u, q, \top)$ and $(u, q, \bot)$ are sinks, winning for Eve and Adam respectively.

A pointed tree (t, u) is *accepted* by $\mathcal{A}$ if Eve has a winning strategy in $G(\mathcal{A}, t, u)$, and the *language* of $\mathcal{A}$ is the set of pointed trees accepted by $\mathcal{A}$, written $\mathcal{L}(\mathcal{A})$. We write $t \in \mathcal{L}(\mathcal{A})$ if $(t, r) \in \mathcal{L}(\mathcal{A})$, where r is the root of t. We also write $\overline{\mathcal{L}}$ for the complement of a language $\mathcal{L}$.

The *size* $|\mathcal{A}|$ of an APT $\mathcal{A}$ is its number of states plus the sum of the sizes of all formulas appearing in the transition function. We also call *index* of an APT the number of priorities in its parity condition. Given two APT $\mathcal{A}$ and $\mathcal{A}'$ we denote $\mathcal{A} \wedge \mathcal{A}'$ (resp. $\mathcal{A} \vee \mathcal{A}'$) the APT of size linear in $|\mathcal{A}|$ and $|\mathcal{A}'|$ that accepts the intersection (resp. union) of the languages of $\mathcal{A}$ and $\mathcal{A}'$.

Word automata. We also consider word automata that we run on branches of trees. We assimilate infinite words with infinite trees over the singleton set of directions $\{\mathrm{next}\}$. A *parity word automaton* is a parity tree automaton on $(\mathcal{V}^{\mathrm{AP}}, \{\mathrm{next}\})$-trees. In the case of a nondeterministic parity word automaton, transitions are represented as a mapping $\Delta : Q \times \mathcal{V}^{\mathrm{AP}} \to 2^Q$ which, in a state $q \in Q$, reading the label $a \in \mathcal{V}^{\mathrm{AP}}$ of the current state in the word, indicates a set of states $\Delta(q, a)$ from which Eve can choose to send in the next position of the word.

4.2 Operations on automata

We recall the following operations on automata.

Projections. The first two constructions are existential and universal projection. The classic projection of a nondeterministic tree automaton over an atomic proposition p, introduced by Rabin to deal with second-order monadic quantification [74], consists in letting the nondeterministic tree automaton guess in each node a valuation for p. We obtain an automaton that accepts an input tree if and only if there exists a p-labeling for that tree that makes it accepted by the original automaton. Similarly, one can easily define universal projection of universal tree automata.

In our setting, we have the following:

[5] What we call UPT here is sometimes referred to as comp{NPT} in the literature.

Proposition 4.1. *Given an NPT $\mathcal{N}$ on $\mathcal{V}^{\mathrm{AP}}$-labeled trees and an atomic proposition $p \in \mathrm{AP}$, one can build in linear time an NPT $\mathcal{N}\Downarrow_p^{\exists}$ such that*

$$(t,u) \in \mathcal{L}(\mathcal{N}\Downarrow_p^{\exists}) \text{ iff there exists a Boolean } p\text{-labeling } \ell_p \text{ for } t \text{ s.t. } (t \otimes \ell_p, u) \in \mathcal{L}(\mathcal{N}).$$

Given a UPT $\mathcal{N}$ on $\mathcal{V}^{\mathrm{AP}}$-labeled trees and an atomic proposition $p \in \mathrm{AP}$, one can build in linear time a UPT $\mathcal{U}\Downarrow_p^{\forall}$ such that

$$(t,u) \in \mathcal{L}(\mathcal{U}\Downarrow_p^{\forall}) \text{ iff for all Boolean } p\text{-labelings } \ell_p \text{ for } t,\ (t \otimes \ell_p, u) \in \mathcal{L}(\mathcal{U}).$$

Simulation. Before performing projections, we will need to turn alternating automata into nondeterministic or universal ones, via the classic simulation theorem.

Theorem 4.2 (Simulation [68]). *Given an APT $\mathcal{A}$, one can build in exponential time an NPT $\mathcal{N}$ and a UPT $\mathcal{U}$, both of exponential size and linear index, such that $\mathcal{L}(\mathcal{N}) = \mathcal{L}(\mathcal{U}) = \mathcal{L}(\mathcal{A})$.*

Remark 2. Note that the simulation theorem in [68] is only stated for nondeterministic automata. However we can obtain easily the result for universal ones as follows: first, notice that universal automata are dual to nondeterministic automata, and that we can turn an NPT into a UPT of linear size that accepts the complement language of the original NPT. Now to build a UPT accepting the same language as an APT $\mathcal{A}$, one first builds in linear time an APT $\mathcal{A}'$ that accepts $\overline{\mathcal{L}(\mathcal{A})}$. Then, by the simulation theorem, one can build an NPT $\mathcal{N}$ of exponential size equivalent to $\mathcal{A}'$. Finally one builds the linear-size UPT $\mathcal{U}$ accepting $\overline{\mathcal{L}(\mathcal{N})} = \mathcal{L}(\mathcal{A})$.

4.3 Finiteness of possible satisfaction values

As in the case of LTL$[\mathcal{F}]$ we show that since the set of possible satisfaction values of an atomic proposition is finite, so is the set of possible satisfaction values of each BQCTL$^\star[\mathcal{F}]$ formula. This property allows using max instead of sup in Definition 3.3. The set of possible satisfaction values does not depend on the set of directions D, which we thus omit in the following lemma.

Lemma 4.3. *Let $\mathcal{V} \subset [0,1]$ be a finite set of values with $\{0,1\} \subseteq \mathcal{V}$, let φ be a* BQCTL$^\star[\mathcal{F}]$ *state formula and ψ a* BQCTL$^\star[\mathcal{F}]$ *path formula, with respect to* AP*. Define*

$$\mathrm{Val}_{\varphi,\mathcal{V}} = \{\{\!\{\varphi\}\!\}^t(u) \mid t \text{ is a } \mathcal{V}^{\mathrm{AP}}\text{-labeled tree and } u \in t\}$$

be the set of values taken by φ in nodes of $\mathcal{V}^{\mathrm{AP}}$-labeled trees. Similarly, define

$$\mathrm{Val}_{\psi,\mathcal{V}} = \{\{\!\{\psi\}\!\}^t(\lambda) \mid t \text{ is a } \mathcal{V}^{\mathrm{AP}}\text{-labeled tree, } u \in t \text{ and } \lambda \in Br(u)\}$$

Then, $|\mathrm{Val}_{\varphi,\mathcal{V}}| \leq |\mathcal{V}|^{|\varphi|}$ and $|\mathrm{Val}_{\psi,\mathcal{V}}| \leq |\mathcal{V}|^{|\psi|}$, and one can compute sets $\widetilde{\mathrm{Val}}_{\varphi,\mathcal{V}}$ and $\widetilde{\mathrm{Val}}_{\psi,\mathcal{V}}$ of size at most $|\mathcal{V}|^{|\varphi|}$ and $|\mathcal{V}|^{|\psi|}$ respectively, and such that $\mathrm{Val}_{\varphi,\mathcal{V}} \subseteq \widetilde{\mathrm{Val}}_{\varphi,\mathcal{V}}$ and $\mathrm{Val}_{\psi,\mathcal{V}} \subseteq \widetilde{\mathrm{Val}}_{\psi,\mathcal{V}}$.

Proof. We prove the result by mutual induction on φ and ψ. Clearly, $\mathrm{Val}_p = \mathcal{V}$.

For $\varphi = \exists p.\, \varphi'$: let t be a $\mathcal{V}^{\mathrm{AP}}$-labeled tree and $u \in t$. Since $\mathrm{Val}_t \subseteq \mathcal{V}$ and by assumption $\mathcal{V}$ contains 0 and 1, for all Boolean p-valuation ℓ_p for t, $\mathrm{Val}_{t\otimes\ell_p} \subseteq \mathcal{V}$. It follows by definition of $\mathrm{Val}_{\varphi',\mathcal{V}}$ that for all such ℓ_p we have $\{\!\{\varphi'\}\!\}^{t\otimes\ell_p}(u) \in \mathrm{Val}_{\varphi',\mathcal{V}}$. Therefore $\{\!\{\exists p.\, \varphi'\}\!\}^t(u)$ is the supremum of a subset of $\mathrm{Val}_{\varphi',\mathcal{V}}$, which by induction hypothesis is of size at most $|\mathcal{V}|^{|\varphi'|}$. It follows that the supremum is indeed a maximum, and that $\{\!\{\exists p.\, \varphi'\}\!\}^t(u) \in \mathrm{Val}_{\varphi',\mathcal{V}}$. Hence, $\mathrm{Val}_{\exists p.\,\varphi',\mathcal{V}} \subseteq \mathrm{Val}_{\varphi',\mathcal{V}}$, and thus $|\mathrm{Val}_{\exists p.\,\varphi',\mathcal{V}}| \leq |\mathrm{Val}_{\varphi',\mathcal{V}}| \leq |\mathcal{V}|^{|\varphi'|} \leq |\mathcal{V}|^{|\exists p.\,\varphi'|}$.

For $\varphi = \mathbf{E}\psi$, again $\{\!\{\mathbf{E}\psi\}\!\}^t(u)$ is a supremum over a subset of $\mathrm{Val}_{\psi,\mathcal{V}}$, which by induction hypothesis is of size at most $\mathcal{V}^{|\psi|}$. The supremum is thus reached, hence $\mathrm{Val}_{\mathbf{E}\psi,\mathcal{V}} \subseteq \mathrm{Val}_{\psi,\mathcal{V}}$ and $|\mathrm{Val}_{\mathbf{E}\psi,\mathcal{V}}| \leq |\mathrm{Val}_{\psi,\mathcal{V}}| \leq |\mathcal{V}|^{|\psi|} \leq |\mathcal{V}|^{|\mathbf{E}\psi|}$.

For $\varphi = f(\varphi_1, \dots, \varphi_n)$, we have $\mathrm{Val}_{\varphi,\mathcal{V}} = \{f(v_1, \dots, v_n) \mid v_i \in \mathrm{Val}_{\varphi_i,\mathcal{V}}\}$, hence $|\mathrm{Val}_{\varphi,\mathcal{V}}|$ is at most $\prod_{i=1}^{n} |\mathrm{Val}_{\varphi_i,\mathcal{V}}|$. By induction hypothesis, we get that $|\mathrm{Val}_{\varphi,\mathcal{V}}| \leq \prod_{i=1}^{n} |\mathcal{V}|^{|\varphi_i|}$, which in turn is no greater than $|\mathcal{V}|^{|\varphi_1|+\dots+|\varphi_n|}$, which is no greater than $|\mathcal{V}|^{|\varphi|}$.

For $\psi = \varphi$, the result follows by hypothesis of mutual induction.

For $\psi = \mathbf{X}\psi'$, we have that $\mathrm{Val}_{\psi,\mathcal{V}} = \mathrm{Val}_{\psi',\mathcal{V}}$, and the result follows.

For $\psi = \psi_1 \mathbf{U} \psi_2$, the value of ψ is defined via suprema and infima over possible values for ψ_1 and ψ_2, which are finitely many by the induction hypothesis. The suprema and infima are thus maxima and minima, and $\mathrm{Val}_{\psi,\mathcal{V}} \subseteq \mathrm{Val}_{\psi_1,\mathcal{V}} \cup \mathrm{Val}_{\psi_2,\mathcal{V}}$. Hence $|\mathrm{Val}_{\psi,\mathcal{V}}| \leq |\mathrm{Val}_{\psi_1,\mathcal{V}}| + |\mathrm{Val}_{\psi_2,\mathcal{V}}|$, by induction hypothesis $|\mathrm{Val}_{\psi_1,\mathcal{V}}| + |\mathrm{Val}_{\psi_2,\mathcal{V}}| \leq |\mathcal{V}|^{|\psi_1|} + |\mathcal{V}|^{|\psi_2|}$, since $\mathcal{V}$ contains at least 0 and 1, $|\mathcal{V}| \geq 2$ and thus $|\mathcal{V}|^{|\psi_1|} + |\mathcal{V}|^{|\psi_2|} \leq |\mathcal{V}|^{|\psi_1|+|\psi_2|}$, which is no greater than $|\mathcal{V}|^{|\psi|}$.

In all cases, the claim for over-approximations follows by the same reasoning as above. □

The finite over-approximation of the set of possible satisfaction values induces a finite alphabet for the automata used in our model-checking procedure, which we now describe.

4.4 Model-checking procedure

We extend the automata-based model-checking procedure for CTL* from [57]. Note that since the quantified atomic propositions may appear in different subformulas, we cannot extend the algorithm for CTL$^\star[\mathcal{F}]$ from [3], as the latter applies the technique of [44], where the evaluation of each subformula is independent.

Proposition 4.4. *Let $\mathcal{V} \subset [0,1]$ be a finite set of values such that $\{0,1\} \subseteq \mathcal{V}$, and let D be a finite set of directions. For every* BQCTL$^\star[\mathcal{F}]$ *state formula φ and predicate $P \subseteq [0,1]$, one can construct an APT $\mathcal{A}_\varphi^{\mathcal{V},P}$ over $(\mathcal{V}^{\mathrm{AP}} \cup \{\bullet\})$-trees such that for every $\mathcal{V}^{\mathrm{AP}}$-labeled D-tree t,*

$$\mathcal{A}_\varphi^{\mathcal{V},P} \text{ accepts } \bar{t} \text{ if and only if } \{\!|\varphi|\!\}^t \in P.$$

The APT $\mathcal{A}_\varphi^{\mathcal{V},P}$ is of size at most $(\mathrm{bnd}(\varphi)+1)$-exponential in $|\varphi|$, and its index is at most $\mathrm{nd}(\varphi)$-exponential in $|\varphi|$.

Proof. The proof proceeds by an induction on the structure of the formula φ and strengthens the induction statement as follows: one can construct an APT $\mathcal{A}_\varphi^{\mathcal{V},P}$ such that for every $\mathcal{V}^{\mathrm{AP}}$-labeled D-tree t, for every node $u \in t$, we have that $\mathcal{A}_\varphi^{\mathcal{V},P}$ accepts $\bar{t}$ from node u if and only if $\{\!|\varphi|\!\}^t(u) \in P$ (recall that $\bar{t}$ is the complete tree obtained from t by adding dummy nodes labeled with $\bullet$).

If $\varphi = p$, the automaton has one state and accepts a tree $t = (\tau, \mathrm{w})$ in node $u \in \tau$ if $\mathrm{w}(u)(p) \in P$, and rejects otherwise.

If $\varphi = \exists p.\, \varphi'$, we want to check whether the maximal satisfaction value of φ', over all possible p-labelings of the input tree, is in P. To do so we first compute a finite set $\widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}}$ of exponential size such that $\mathrm{Val}_{\varphi',\mathcal{V}} \subseteq \widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}}$, which we can do as established by Lemma 4.3. For each possible value $v \in \widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}} \cap P$, we check whether this value is reached for some p-labeling, and if the value of φ' is less than or equal to v for all p-labelings. For each $v \in \widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}} \cap P$, inductively build the APTs $\mathcal{A}_{\varphi'}^{\mathcal{V},\{v\}}$ and $\mathcal{A}_{\varphi'}^{\mathcal{V},[0,v]\cap P}$. Turn the first one into an NPT $\mathcal{N}_{=v}$ and the second one into a UPT $\mathcal{U}_{\leq v}$ (using Theorem 4.2). Project $\mathcal{N}_{=v}$ existentially on p, and call the result $\mathcal{N}'_{=v}$. Project $\mathcal{U}_{\leq v}$ universally on p, call the result $\mathcal{U}'_{\leq v}$. Finally, we can define the APT $\mathcal{A}_{\exists p.\, \varphi'}^{\mathcal{V},P} := \bigvee_{v \in \widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}} \cap P} \mathcal{N}'_{=v} \wedge \mathcal{U}'_{\leq v}$. It is then easy to see that this automaton accepts a tree if and only if there exists a value in P that is the maximum of the possible values taken by φ' for all p-labelings.

A block of existential quantifiers $\exists p_1 \dots \exists p_k . \varphi'$ can be treated similarly: compute $\widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}}$, for each $v \in \widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}} \cap P$ build the NPT $\mathcal{N}_{=v}$ and the UPT $\mathcal{U}_{\leq v}$ (incurring one exponential blowup),

project $\mathcal{N}_{=v}$ existentially on $p_1, \ldots, p_k$ and project $\mathcal{U}_{\leq v}$ universally on $p_1, \ldots, p_k$ (these operations are polynomial), and combine the resulting automata into the final automaton.

If $\varphi = \mathbf{E}\psi$: as in the classic automata construction for CTL* [58], we first let atoms(ψ) be the set of maximal state sub-formulas of ψ (that we call *atoms* thereafter – which have to be distinguished from atomic propositions of the formula). In a first step we see elements of atoms(ψ) as atomic propositions, and ψ as an LTL[$\mathcal{F}$] formula over atoms(ψ). According to Lemma 4.3 we can compute over-approximations $\widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}}$ for each $\varphi' \in \mathrm{atoms}(\psi)$, and we thus let $\widetilde{\mathrm{Val}} = \bigcup_{\varphi' \in \mathrm{atoms}(\psi)} \widetilde{\mathrm{Val}}_{\varphi',\mathcal{V}}$ be a finite over-approximation of the set of possible values for atoms. It is proven in [3, Theorem 2.9] that for every $P \subseteq [0,1]$, one can build a nondeterministic parity word automaton $\mathcal{W}^\psi_P$ of size exponential in $|\psi|^2$ such that $\mathcal{W}^\psi_P$ accepts a word $w \in (\widetilde{\mathrm{Val}}^{\mathrm{atoms}(\psi)})^\omega$ if and only if $\{\!\{\psi\}\!\}(w) \in P$. Now let us compute $\widetilde{\mathrm{Val}}_{\mathbf{E}\psi,\mathcal{V}}$ (again using Lemma 4.3), and for each $v \in \widetilde{\mathrm{Val}}_{\mathbf{E}\psi,\mathcal{V}} \cap P$, construct an NPT $\mathcal{N}^{E=v}$ that guesses a branch in its input and simulates $\mathcal{W}^\psi_{\{v\}}$ on it (it rejects if it leaves the "real" tree, i.e., if it sees a node labeled with $\bullet$). To obtain a universal word automaton of single exponential size that checks whether $\{\!\{\psi\}\!\}(w) \in [0,v]$, we first build a nondeterministic word automaton $\mathcal{W}^\psi_{(v,1]}$ following [3], and dualize it in linear time into a universal word automaton $\mathcal{W}^\psi_{[0,v]}$. Then we construct a UPT $\mathcal{U}^{A\leq v}$ that executes $\mathcal{W}^\psi_{[0,v]}$ on all branches of its input.[6] We now define the APT $\mathcal{A}^P$ on $\widetilde{\mathrm{Val}}^{\mathrm{atoms}(\psi)}$-trees as

$$\mathcal{A}^P = \bigvee_{v \in \widetilde{\mathrm{Val}}_{\mathbf{E}\psi,\mathcal{V}} \cap P} \mathcal{N}^{E=v} \wedge \mathcal{U}^{A\leq v}.$$

Now to go from atoms to standard atomic propositions, we define an APT $\mathcal{A}^{\mathcal{V},P}_{\mathbf{E}\psi}$ that simulates $\mathcal{A}^P$ by, in each state and each node of its input, guessing a value v_i in $\widetilde{\mathrm{Val}}_{\varphi_i}$ for each formula $\varphi_i \in \mathrm{atoms}(\psi)$, simulating $\mathcal{A}^P$ on the resulting weight label, and launching a copy of $\mathcal{A}^{\mathcal{V},\{v_i\}}_{\varphi_i}$ for each $\varphi_i \in \mathrm{atoms}(\psi)$. Note that the automaton is alternating and thus may have to guess several times the satisfaction value of a formula φ_i in a same node, but launching the $\mathcal{A}^{\mathcal{V},\{v_i\}}_{\varphi_i}$ forces it to always guess the same, correct value.

Finally, if $\varphi = f(\varphi_1, ..., \varphi_n)$, we list all combinations $(v_1, \ldots, v_n)$ of the possible satisfaction values for the subformulas φ_i such that $f(v_1, \ldots, v_n) \in P$, and we build automaton $\mathcal{A}^{\mathcal{V},P}_\varphi$ as the disjunction over such $(v_1, \ldots, v_n)$ of the conjunction of automata $\mathcal{A}^{\mathcal{V},\{v_i\}}_{\varphi_i}$.

The complexity of this procedure is non-elementary. More precisely, we claim that $\mathcal{A}^{\mathcal{V},P}_\varphi$ has size at most $(\mathrm{nd}(\varphi)+1)$-exponential and index at most $\mathrm{nd}(\varphi)$-exponential.

The case where φ is an atomic proposition is trivial.

For $\varphi = \exists p.\, \varphi'$, we transform an exponential number of APTs into NPTs or UPTs. This entails an exponential blowup in the size of each automaton, while the index remains linear. By induction hypothesis, the resulting automaton $\mathcal{A}^{\mathcal{V},P}_{\exists p.\, \varphi'}$ has at most $(\mathrm{nd}(\varphi')+2)$-exponentially many states and index at most $(\mathrm{nd}(\varphi')+1)$-exponential. Since $\mathrm{nd}(\varphi) = \mathrm{nd}(\varphi')+1$, the property holds.

If φ has the form $f(\varphi_1, ..., \varphi_n)$, then the automaton for φ is a combination of the automata for all φ_i, and for the various values those subformulas may take. By Lemma 4.3 there are at most $|\mathcal{V}|^{|\varphi_1|+...+|\varphi_n|} \leq |\mathcal{V}|^{|\varphi|}$ different combinations, so assuming (from the induction hypothesis) that

[6] We take $\mathcal{W}^\psi_{[0,v]}$ universal because it is not possible to simulate a nondeterministic word automaton on all branches of a tree, but it is possible for a universal one. Note that we could also determinise $\mathcal{W}^\psi_{[0,v]}$, but it would cost one more exponential.

the automata for φ_i have at most $(\mathrm{nd}(\varphi_i)+1)$-exponentially many states and index at most $\mathrm{nd}(\varphi_i)$-exponential, the automaton for φ has at most $(\mathrm{nd}(\varphi)+1)$-exponentially many states and index at most $\mathrm{nd}(\varphi)$-exponential (note that $\mathrm{nd}(\varphi_i) = \mathrm{nd}(\varphi)$).

Finally for $\varphi = \mathbf{E}\psi$: in [3], the automaton $\mathcal{W}_P^{\psi}$ is a generalized Büchi automaton of size exponential in $|\psi|^2$, and with at most $|\psi|$ Büchi acceptance conditions. One can turn this automaton into an equivalent Büchi automaton still exponential in $|\psi|^2$, which can be seen as a parity automaton with index 2. Then $\mathcal{N}^{E=v}$ and $\mathcal{U}^{A\leq v}$ both also have sizes exponential in $|\psi|^2$, and index 2. Finally, $\mathcal{A}^P$, which combines an exponential number of the automata above, has size exponential in $|\psi|^2$ and index 2. The final automaton $\mathcal{A}^{\mathcal{V},P}_{\mathbf{E}\psi}$ is obtained from that automaton by plugging the automata for $\mathcal{A}^{\mathcal{V},\bullet}_{\varphi_i}$, whose sizes and indices are respectively $(\mathrm{nd}(\varphi_i)+1)$- and $\mathrm{nd}(\varphi_i)$-exponential, and dominate the size and index of $\mathcal{A}^P$. Since $\mathrm{nd}(\varphi) = \max_i \mathrm{nd}(\varphi_i)$, it follows that for $\varphi = \mathbf{E}\psi$ the size of $\mathcal{A}^{\mathcal{V},P}_{\varphi}$ also is $(\mathrm{nd}(\varphi)+1)$-exponential, and its index is $\mathrm{nd}(\varphi)$-exponential. □

To see that Theorem 3.8 follows from Proposition 4.4, recall that by definition $\{\!\{\varphi\}\!\}^{\mathcal{K}} = \{\!\{\varphi\}\!\}^{t_{\mathcal{K}}}$. Thus to check whether $\{\!\{\varphi\}\!\}^{\mathcal{K}} \in P$, where atoms in $\mathcal{K}$ take values in $\mathcal{V}$, it is enough to build $\mathcal{A}^{\mathcal{V},P}_{\varphi}$ as in Proposition 4.4, take its product with a deterministic tree automaton that accepts only $\overline{t_{\mathcal{K}}}$, and check for emptiness of the product automaton. The formula complexity is $(\mathrm{nd}(\varphi)+1)$-exponential, but the structure complexity is polynomial.

For the lower bounds, consider the following fragment of QCTL*: $\mathsf{EQ}^k\mathsf{CTL}^\star_{\mathsf{alt}}$ is the fragment containing all formulas of QCTL* that are in prenex normal form (i.e. with all quantifications on atomic propositions at the beginning), have at most k alternations of propositional quantifiers, and start with an existential quantifier (see [60, page 8] for a formal definition); in $\mathsf{EQ}^k\mathsf{CTL}^\star_{\mathsf{alt}}$, quantifiers can be grouped together into at most k blocks of quantifiers, each block containing only one type of quantifiers. The model-checking problem for $\mathsf{EQ}^k\mathsf{CTL}^\star_{\mathsf{alt}}$ is proven to be $(k+1)$-Exptime-hard in [60]. Now, assuming $\mathcal{F}$ contains disjunction $\max\{x,y\}$ and negation $1-x$, $\mathsf{EQ}^k\mathsf{CTL}^\star_{\mathsf{alt}}$ can be translated in BQCTL$^\star[\mathcal{F}]$ with formulas of linear size and block nesting depth at most k, which gives us the lower-bounds of Theorem 3.7.

5 MODEL CHECKING SL$[\mathcal{F}]$

In this section we show how to reduce the model-checking problem for SL$[\mathcal{F}]$ to that of QCTL$^\star[\mathcal{F}]$. This reduction is a rather straightforward adaptation of the usual one for qualitative variants of SL (see, e.g., [13, 45, 61]). We essentially observe that it can be lifted to the quantitative setting.

We let Agt be a finite set of agents, and AP be a finite set of atomic propositions.

Models transformation. We first define for every WCGS $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Agt}, \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta, \mathrm{w}\rangle$ a WKS $\mathcal{K}_{\mathcal{G}} = \langle \mathrm{S}, \mathrm{s}_\iota, R, \mathrm{w}\rangle$ over some set AP$'$ and a bijection $\rho \mapsto u_\rho$ between the set of histories starting in the initial state v_ι of $\mathcal{G}$ and the set of nodes in $t_{\mathcal{K}_{\mathcal{G}}}$. We consider propositions $\mathrm{AP}_\mathrm{V} = \{p_\mathrm{v} \mid \mathrm{v} \in \mathrm{V}\}$, that we assume to be disjoint from AP. We let $\mathrm{AP}' = \mathrm{AP} \cup \mathrm{AP}_\mathrm{V}$. Define the Kripke structure $\mathcal{K}_{\mathcal{G}} = (\mathrm{S}, \mathrm{s}_\iota, R, \mathrm{w})$ where

- $\mathrm{S} = \{\mathrm{s}_\mathrm{v} \mid \mathrm{v} \in \mathrm{V}\}$,
- $\mathrm{s}_\iota = \mathrm{s}_{\mathrm{v}_\iota}$,
- $R = \{(\mathrm{s}_\mathrm{v}, \mathrm{s}_{\mathrm{v}'}) \in \mathrm{S}^2 \mid \exists \vec{c} \in \mathrm{Act}^{\mathrm{Agt}} \text{ s.t. } \Delta(\mathrm{v}, \vec{c}) = \mathrm{v}'\}$, and
- $\mathrm{w}(p)(\mathrm{s}_\mathrm{v}) = \begin{cases} 1 & \text{if } p \in \mathrm{AP}_\mathrm{V} \text{ and } p = p_\mathrm{v} \\ 0 & \text{if } p \in \mathrm{AP}_\mathrm{V} \text{ and } p \neq p_\mathrm{v} \\ \mathrm{w}(p)(\mathrm{v}) & \text{otherwise} \end{cases}$.

For every history $\rho = \mathrm{v}_0 \dots \mathrm{v}_k$, define the node $u_\rho = \mathrm{s}_{\mathrm{v}_0} \dots \mathrm{s}_{\mathrm{v}_k}$ in $t_{\mathcal{K}_\mathcal{G}}$ (which exists, by definition of $\mathcal{K}_\mathcal{G}$ and of tree unfoldings). Note that the mapping $\rho \mapsto u_\rho$ defines a bijection between the set of histories from v_ι and the set of nodes in $t_{\mathcal{K}_\mathcal{G}}$.

Formulas translation. Given a game $\mathcal{G} = \langle \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta, \mathrm{w} \rangle$ and a formula $\varphi \in \mathrm{SL}[\mathcal{F}]$, we define a $\mathrm{QCTL}^\star[\mathcal{F}]$ formula φ' such that $\{\!\{\varphi\}\!\}^\mathcal{G} = \{\!\{\varphi'\}\!\}^{\mathcal{K}_\mathcal{G}}$. More precisely, this translation is parameterised with a partial function $g : \mathrm{Agt} \rightharpoonup \mathrm{Var}$ which records bindings of agents to strategy variables. Suppose that $\mathrm{Act} = \{c_1, \dots, c_m\}$. We define the translation $(\cdot)^g$ by induction on state formulas φ and path formulas ψ. Here is the definition of $(\cdot)^g$ for state formulas:

$$(p)^g = p \qquad ((a,x)\varphi)^g = (\varphi)^{g[a \mapsto x]}$$

$$(\langle\!\langle x \rangle\!\rangle \varphi)^g = \exists p^x_{c_1} \dots \exists p^x_{c_m} . \Big(\varphi_{\mathrm{str}}(x) \wedge (\varphi)^g \Big),$$

$$\text{where } \varphi_{\mathrm{str}}(x) = \mathbf{AG}\Big(\bigvee\nolimits_{c \in \mathrm{Act}} (p^x_c \wedge \bigwedge\nolimits_{c' \neq c} \neg p^x_{c'}) \Big)$$

$$(\mathbf{A}\psi)^g = \mathbf{A}(\psi_{\mathrm{out}}(g) \rightarrow (\psi)^g)$$

$$\text{where } \psi_{\mathrm{out}}(g) = \mathbf{G}\left(\bigwedge\nolimits_{\mathrm{v} \in \mathrm{V}} \left(p_\mathrm{v} \rightarrow \bigvee\nolimits_{\vec{c} \in \mathrm{Act}^{\mathrm{Agt}}} \left(\bigwedge\nolimits_{a \in \mathrm{dom}(f)} p^{f(a)}_{\vec{c}(a)} \wedge \mathbf{X} p_{\Delta(\mathrm{v},\vec{c})} \right) \right) \right)$$

$$(f(\varphi_1, \dots, \varphi_n))^g = f((\varphi_1)^g, \dots, (\varphi_n)^g)$$

and for path formulas:

$$(\varphi)^g = (\varphi)^g \qquad (\mathbf{X}\psi)^g = \mathbf{X}(\psi)^g$$

$$(\psi \mathbf{U} \psi')^g = (\psi)^g \mathbf{U} (\psi')^g \qquad (f(\psi_1, \dots, \psi_n))^g = f((\psi_1)^g, \dots, (\psi_n)^g)$$

This translation is identical to that from branching-time SL to $\mathrm{QCTL}^\star$ in all cases, except for the case of functions which is straightforward. To see that it can be safely lifted to the quantitative setting, it suffices to observe the following: since quantification on atomic propositions is restricted in $\mathrm{BQCTL}^\star[\mathcal{F}]$ to Boolean values, and atoms in AP_V also have Boolean values, $\varphi_{\mathrm{str}}(x)$ and $\psi_{\mathrm{out}}(\chi)$ always have value 0 or 1 and thus they can play exactly the same role as in the qualitative setting: $\varphi_{\mathrm{str}}(x)$ holds if and only if the atomic propositions $p^x_{c_1}, \dots, p^x_{c_m}$ indeed code a strategy from the current state, and $\psi_{\mathrm{out}}(\chi)$ holds on a branch of $t_{\mathcal{K}_\mathcal{G}}$ if and only if in this branch each agent $a \in \mathrm{dom}(g)$ follows the strategies coded by atoms p^x_c. As a result $\exists p^x_{c_1} \dots \exists p^x_{c_m} . \Big(\varphi_{\mathrm{str}}(x) \wedge (\varphi)^g \Big)$ maximizes over those valuations for the $p^x_{c_i}$ that code for strategies, other valuations yielding value 0. Similarly, formula $\mathbf{A}(\psi_{\mathrm{out}}(g) \rightarrow (\psi)^g)$ minimizes over branches that represent outcomes of strategies in g, as others yield value 1.

One can now see that the following holds, where φ is an $\mathrm{SL}[\mathcal{F}]$ formula.

Lemma 5.1. *Let χ be an assignment and $g : \mathrm{Agt} \rightharpoonup \mathrm{Var}$ such that $\mathrm{dom}(g) = \mathrm{dom}(\chi) \cap \mathrm{Agt}$ and for all $a \in \mathrm{dom}(g)$, $g(a) = x$ implies $\chi(a) = \chi(x)$. Then*

$$\{\!\{\varphi\}\!\}^\mathcal{G}_\chi(\rho) = \{\!\{(\varphi)^g\}\!\}^{t_{\mathcal{K}_\mathcal{G}}}(u_\rho)$$

As a result, the quantitative model-checking problem for an $\mathrm{SL}[\mathcal{F}]$ formula φ, a weighted CGS $\mathcal{G}$ and a predicate $P \subseteq [0,1]$ can be solved by computing the $\mathrm{BQCTL}^\star[\mathcal{F}]$ formula $\varphi' = (\varphi)^\emptyset$, which is of size polynomial in $|\phi$, and the weighted Kripke structure $\mathcal{K}_\mathcal{G}$, of size polynomial in $|\mathcal{G}|$, and deciding whether $\{\!\{\varphi'\}\!\}^{\mathcal{K}_\mathcal{G}} \in P$, which can be done by Theorem 3.8. To preserve block nesting depth, we observe that the above translation can be massaged so that each block of existential (resp. universal) strategy quantifiers results in a single block of existential (resp. universal) proposition

quantifiers. We translate a block of existential strategy quantifiers as follows:

$$(\langle\langle x_1\rangle\rangle \dots \langle\langle x_k\rangle\rangle\varphi)^g = \exists p_{c_1}^{x_1} \dots \exists p_{c_m}^{x_1} \dots \exists p_{c_1}^{x_k} \dots \exists p_{c_m}^{x_k}.\Big(\bigwedge_{i=1}^{k} \varphi_{\mathrm{str}}(x_i) \wedge (\varphi)^g\Big)$$

This establishes the upper-bounds in Theorem 2.8. The reduction presented above is polynomial and preserves block nesting depth, so that we inherit the lower-bounds from BQCTL$^\star[\mathcal{F}]$ established in Theorem 3.8.

Remark 3. Lemma 5.1 together with Lemma 4.3 imply that SL$[\mathcal{F}]$ formulas also can take only exponentially many values when a finite domain is fixed for atomic propositions. This justifies the observation of Remark 1 that supremum and infimum in the semantics of SL$[\mathcal{F}]$ can be replaced with maximum and minimum.

6 THE CASE OF SL$_{1G}[\mathcal{F}]$

We now study the fragment SL$_{1G}[\mathcal{F}]$, which is the extension to the quantitative setting of the one-goal fragment SL$_{1G}$ of SL [66]. In the Boolean setting, SL$_{1G}$ is of interest because it is strictly more expressive than ATL* but enjoys the same complexity. Indeed, both its model-checking and satisfiability are 2-Exptime-complete. In addition to ATL* it allows for arbitrary alternation of existential and universal strategy quantifications, and thus can express complex dependencies between strategies (see the last example in Section 2.4.1, which is similar to the *Escape from Alcatraz* example in [66]). We show that in the quantitative setting, the one-goal fragment SL$_{1G}[\mathcal{F}]$ also enjoys this better computational complexity.

In order to define the syntax, we need to introduce the notions of *quantification prefix* and *binding prefix*. A quantification prefix is a sequence $\wp = \langle\langle x_1]] \dots \langle\langle x_n]]$, where $\langle\langle x_i]] \in \{\langle\langle x_i\rangle\rangle, [[x_i]]\}$ is either an existential or universal quantification. For a fixed set of agents Agt $= \{a_1, \dots, a_n\}$ a binding prefix is a sequence $\flat = (a_1, x_1), \dots (a_n, x_n)$, where every agent in Agt occurs exactly once. A combination $\wp\flat$ is *closed* if every variable occurring in $\flat$ occurs in some quantifier of $\wp$. We can now give the definition of SL$_{1G}[\mathcal{F}]$ syntax.

*Definition 6.1 (*SL$_{1G}[\mathcal{F}]$ *Syntax).* Let AP be a set of Boolean atomic propositions, and let $\mathcal{F} \subseteq \{f : [0,1]^m \to [0,1] \mid m \in \mathbb{N}\}$ be a set of functions over $[0,1]$. The set of SL$_{1G}[\mathcal{F}]$ formulas is defined by the following grammar:

$$\varphi := p \mid f(\varphi, \dots, \varphi) \mid \varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi \mid \wp\flat\varphi,$$

where $p \in$ AP, $f \in \mathcal{F}$, and $\wp\flat$ is a closed combination of a quantification prefix and of a binding prefix.

Note that all SL$[\mathcal{F}]$ formulas are sentences, as all strategy variables are quantified immediately before being bound to some agent. The *sentence nesting depth* of an SL$_{1G}[\mathcal{F}]$ formula is defined as follows:

- $\mathrm{snd}(p) = 0$ for every $p \in$ AP;
- $\mathrm{snd}(f(\varphi_1, \dots, \varphi_n)) = \max_{1 \le i \le n}\{\mathrm{snd}(\varphi_i)\}$;
- $\mathrm{snd}(\mathbf{X}\psi) = \mathrm{snd}(\psi)$;
- $\mathrm{snd}(\psi_1\mathbf{U}\psi_2) = \max\{\mathrm{snd}(\psi_1), \mathrm{snd}(\psi_2)\}$.
- $\mathrm{snd}(\wp\flat\varphi) = \mathrm{snd}(\varphi) + 1$;

Intuitively, the sentence nesting depth measures the number of sentences, i.e., formulas with no free agent or variable, that are nested into each other in the formula.

In order to solve the model-checking problem for SL$_{1G}[\mathcal{F}]$, we need the technical notion of *concurrent multi-player parity game* introduced in [65].

Definition 6.2. A *concurrent multi-player parity game* (CMPG) is a tuple $\mathrm{P} = \langle \mathrm{Agt}, \mathrm{Act}, V, \mathrm{v}_\iota, \Delta, \mathrm{p}\rangle$, where $\mathrm{Agt} = 0, \dots, n$ is a set of agents indexed with natural numbers, $\langle \mathrm{Agt}, \mathrm{Act}, V, \mathrm{v}_\iota, \Delta\rangle$ is a CGS, and $\mathrm{p} : V \to \mathbb{N}$ is a priority function.

In a CMPG, agents are split in two teams: each player i with $i \bmod 2 = 0$ is part of the existential (even) team, the other players are part of the universal (odd) team. The goal in a CMPG is to check whether there exists a strategy for 0 such that, for each strategy for 1, there exists a strategy for 2, and so forth, such that the induced plays satisfy the parity condition. Then, we say that the existential team wins the game. Otherwise the universal team wins the game.

As shown in [65, Theorem 4.1, Corollary 4.1], one can decide the winners of a CMPG $\mathrm{P} = \langle \mathrm{Agt}, \mathrm{Act}, V, \mathrm{v}_\iota, \Delta, \mathrm{p}\rangle$ in time polynomial w.r.t. $|V|$ and $|\mathrm{Act}|$, and exponential w.r.t. $|\mathrm{Agt}|$ and $k = \max \mathrm{p}$ (the maximal priority).

Theorem 6.3. *The model-checking problem for closed formulas of* $\mathrm{SL}_{1G}[\mathcal{F}]$ *is decidable, and* 2-Exptime*-complete.*

Proof. We let $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Agt}, \mathrm{Act}, \mathrm{V}, \mathrm{v}_\iota, \Delta, \mathrm{w}\rangle$ be a WCGS and we consider a formula of the form $\wp \flat \varphi$. We also assume, for simplicity, that $\wp = \langle\!\langle x_0 \rangle\!\rangle [\![x_1]\!], \dots, \langle\!\langle x_k \rangle\!\rangle [\![x_1]\!]$, that is, quantifiers perfectly alternate between existential and universal.[7] Note that the formula $\wp \flat \varphi$ is a sentence, therefore the choice of an assignment is useless. Moreover, recall that, by Lemma 5.1 and in particular Remark 3, the set $V(\wp \flat \varphi)$ of possible values is bounded by $2^{|\wp \flat \varphi|}$.

We proceed by induction on the sentence nesting depth. As base case let $\mathrm{snd}(\wp \flat \varphi) = 1$, i.e., there is no occurrence of neither quantifiers nor bindings in φ. Then, φ can be regarded as an $\mathrm{LTL}[\mathcal{F}]$ formula that can be interpreted over paths of the WKS $\mathcal{K} = \langle \mathrm{AP}, \mathrm{V}, \mathrm{v}_\iota, R, \mathrm{w}\rangle$ where

$$R = \{(\mathrm{v}_1, \mathrm{v}_2) \mid \exists \vec{c} \in \mathrm{Act}^{\mathrm{Agt}}.\ \mathrm{v}_2 = \Delta(\mathrm{v}_1, \vec{c})\}$$

Now, thanks to [3, Theorem 2.9], for every value $v \in V(\wp \flat \varphi)$, we can build a parity word automaton $\mathcal{B}_{\mathcal{K},\varphi,P_v}$ of size exponential in $|\varphi|^2$ such that $\mathcal{B}_{\mathcal{K},\varphi,P_v}$ accepts a word $w \in (\widetilde{\mathrm{Val}}^{\mathrm{atoms}(\psi)})^\omega$ if, and only if, $\{\!\{\psi\}\!\}(w) \in P_v = [v, 1]$. Following [71], we can convert $\mathcal{B}_{\mathcal{K},\varphi,P_v}$ into a deterministic parity word automaton $\mathcal{A}_{\mathcal{K},\varphi,P_v} = \langle \mathrm{V}, Q, q_\iota, \delta, \mathrm{p}\rangle$ of size doubly-exponential in the size of φ and index bounded by $2^{|\varphi|}$.

At this point, define the following CMPG $\mathrm{P} = \langle \mathrm{Agt}', \mathrm{Act}, V', \mathrm{v}'_\iota, \Delta', \mathrm{p}'\rangle$ such that

- $\mathrm{Agt}' = \{0, \dots, k\}$ is a set of agents, one for every variable occurring in $\wp$, ordered in the same way as in $\wp$ itself;
- Act is the set of actions in $\mathcal{G}$;
- $V' = \mathrm{V} \times Q$ is the product of the states of $\mathcal{G}$ and the automaton $\mathcal{A}_{\mathcal{K},\varphi,P_v}$;
- $\mathrm{v}'_\iota = (\mathrm{v}_\iota, q_\iota)$ is the pair given by the initial states of $\mathcal{G}$ and $\mathcal{A}_{\mathcal{K},\varphi,\iota}$, respectively;
- $\mathrm{p}'(\mathrm{v}, q) = \mathrm{p}(q)$ mimics the parity function of $\mathcal{A}_{\mathcal{K},\varphi,P_v}$;
- if $\vec{c} \in \mathrm{Act}^{\mathrm{Agt}'}$, $\Delta'((\mathrm{v}, q), \vec{c}) = (\Delta(\mathrm{v}, \flat(\vec{c})), \delta(q, \mathrm{w}(\mathrm{v})))$ mimics the execution of both $\mathcal{G}$ and $\mathcal{A}_{\mathcal{K},\varphi,P_v}$.

The game emulates two things, one per each component of its state-space. In the first, it emulates a path π generated in $\mathcal{G}$. In the second, it emulates the execution of the automaton $\mathcal{A}_{\mathcal{K},\varphi,P_v}$ when it reads the path π generated in the first component. By construction, it results that every execution $(\pi, \eta) \in \mathrm{V}^\omega \times Q^\omega$ in P satisfies the parity condition determined by p' if, and only if, $\{\!\{\varphi\}\!\}^{\mathcal{K}}(\pi) \in P_v$. Moreover, observe that, since $\mathcal{A}_{\mathcal{K},\varphi,P_v}$ is deterministic, for every possible history ρ in $\mathcal{G}$, there is a unique partial run η_ρ that makes the partial execution (ρ, η_ρ) possible in P. This makes the sets of possible strategies $\mathrm{Str}_{\mathcal{G}}(\mathrm{v}_\iota)$ and $\mathrm{Str}_{\mathrm{P}}(\mathrm{v}'_\iota)$ in bijection. P has a winning strategy if and only if

[7] To reduce to this case, one can add quantifications on dummy variables.

$\{\!\{\wp \flat \varphi\}\!\}^{\mathcal{G}}(\mathrm{v}_\iota) \in P_v$. In order to compute the exact value of $\{\!\{\wp \flat \varphi\}\!\}^{\mathcal{G}}(\mathrm{v}_\iota)$, we repeat the procedure described above for every $v \in V(\wp \flat \varphi)$ and take the maximum v of those for which $\{\!\{\wp \flat \varphi\}\!\}^{\mathcal{G}}(\mathrm{v}_\iota) \in P_v$.

For the induction case, assume we can compute the satisfaction value of every $\mathrm{SL}_{1G}[\mathcal{F}]$ formula with sentence nesting depth at most n, and let $\mathrm{snd}(\wp \flat \varphi) = n+1$. Observe that, for every subsentence $\wp' \flat' \varphi'$ of $\wp \flat \varphi$, we have that $\mathrm{snd}(\wp' \flat' \varphi') \leq n$ and so, by induction hypothesis, we can compute $\{\!\{\wp' \flat' \varphi'\}\!\}^{\mathcal{G}}(\mathrm{v})$ for every $\mathrm{v} \in \mathrm{V}$. Now, introduce a fresh atomic proposition $p_{(\wp' \flat' \varphi')}$ whose weight in $\mathcal{G}$ is defined as $\mathrm{w}(v)(p_{(\wp' \flat' \varphi')}) = \{\!\{\wp' \flat' \varphi'\}\!\}^{\mathcal{G}}(v)$ and a set of fresh atomic propositions p_{v}, one for every $\mathrm{v} \in \mathrm{V}$, whose weights in $\mathcal{G}$ are defined as $\mathrm{w}(\mathrm{v})(p_{\mathrm{v}}) = 1$ and $\mathrm{w}(\mathrm{v})(p_{\mathrm{v}'}) = 0$ if $\mathrm{v} \neq \mathrm{v}'$. Now, consider the Boolean formula

$$\Phi(\wp' \flat' \varphi') = \bigvee_{\mathrm{v} \in \mathrm{V}} (p_{\mathrm{v}} \wedge p_{(\wp' \flat' \varphi')})$$

Observe that every disjunct is a conjunction of the form $p_{\mathrm{v}} \wedge p_{(\wp' \flat' \varphi')}$, whose satisfaction value on a state v' is the minimum among the weights of p_{v} and $p_{(\wp' \flat' \varphi')}$. This can be either 0, if $\mathrm{v} \neq \mathrm{v}'$ or $\{\!\{\wp' \flat' \varphi'\}\!\}^{\mathcal{G}}(\mathrm{v})$, if $\mathrm{v} = \mathrm{v}'$. Now, the big disjunction takes the maximum among them. Therefore, we obtain that $\{\!\{\Phi(\wp' \flat' \varphi')\}\!\}^{\mathcal{G}}(\mathrm{v}) = \{\!\{\wp' \flat' \varphi'\}\!\}^{\mathcal{G}}(\mathrm{v})$. This allows us to replace every occurrence of $\wp' \flat' \varphi'$ in φ with the Boolean combination $\Phi(\wp' \flat' \varphi')$, making the resulting formula to be of sentence nesting depth 1. Thus, we can apply the procedure described in the base case, to compute $\{\!\{\wp \flat \varphi\}\!\}^{\mathcal{G}}(\mathrm{v})$.

Regarding the complexity, note that the size of P is $|\mathrm{V}| \cdot |Q|$ that is in turn linear with respect to the size of $\mathcal{G}$ and doubly exponential in the size of φ. This is due to the fact that the automaton $\mathcal{A}_{\mathcal{K},\varphi,P_v}$ results from the construction of the NGBW $\mathcal{B}_{\mathcal{K},\varphi,P_v}$, of size singly exponential in $|\varphi|$ and the transformation to a DPW, that adds up another exponential to the construction. On the other hand, the number of priorities in P is only singly exponential in $|\varphi|$, and it is due to the fact that the transformation from NGBW to DPW requires a singly exponential number of priorities. Therefore, the CMPG P can be solved in time polynomial w.r.t. the size $\mathcal{G}$ and double exponential in $|\varphi|$. Such a 2-Exptime procedure is executed a number of time exponential in φ, which is still doubly exponential.

Hardness follows from that of SL_{1G} [66]. □

7 ON QCTL$^\star[\mathcal{F}]$ AND MSO$[\mathcal{F}]$

In this section we first study the expressiveness of QCTL$^\star[\mathcal{F}]$. It is shown in [60] that QCTL* is equivalent to MSO on trees. We extend this result by augmenting MSO with quality operators and showing that the resulting logic corresponds to QCTL$^\star[\mathcal{F}]$ when second-order quantification is over fuzzy sets (in which case the logic is called MSO$[\mathcal{F}]$). If second-order quantification is over classic sets instead, then the logic (called BMSO$[\mathcal{F}]$) corresponds to BQCTL$^\star[\mathcal{F}]$. We then prove that, unlike BQCTL$^\star[\mathcal{F}]$, the model-checking problem for QCTL$^\star[\mathcal{F}]$ is undecidable even when $\mathcal{F}$ contains only two simple functions, namely $(x, y) \mapsto x = y$ and $(x, y) \mapsto y = x/2$. Finally we show that when all functions in $\mathcal{F}$ are Lipschitz-continuous, then so is the semantics of QCTL$^\star[\mathcal{F}]$.

7.1 Equivalence with MSO$[\mathcal{F}]$

We extend MSO on trees with quality operators, and compare the expressiveness of the resulting logic with that of QCTL$^\star[\mathcal{F}]$. Multi-valued semantics of MSO have been proposed in the litterature, both on words and on trees [39, 40, 75], and links with fuzzy automata have been established. In [39] the authors introduce a multi-valued version of MSO, in which the Boolean connectives are replaced by their fuzzy versions, as an extension of Łukasiewicz logic [63] to MSO. From such extension, they develop a corresponding automata theory, providing a multi-valued version of the Büchi theorem. Differently from our approach, however, the set of functions of this multi-valued MSO is

limited to those deriving from the Boolean connectives. Instead, here we consider an arbibrary set of function, producing a more general setting than the one presented in [39].

In [40], the authors start from a 4-valued logic (MK-valued logic), taking also undefined and error values, together with classic true and false. Yhey define a fuzzy version of it, namely MK-fuzzy MSO, with truth values of propositions given by 4-tuples in $[0,1]^4$ such that the four components always sum to 1. This is used to measure the degree of truth of a property that accounts for the levels of truth, falsity, undefinedness, and error. They introduce MK-fuzzy automata and solve some of the related problems. We conjecture that the classification of thruth values given there can be encoded in our formalism, and that we can embed reasoning in MK-fuzzy MSO in our setting. We leave this problem open for future investigation.

Let $\mathcal{V}ar_1$ and $\mathcal{V}ar_2$ be countable sets of first-order and second-order variables, respectively.

Definition 7.1. The syntax of MSO$[\mathcal{F}]$ is given by the following grammar:

$$\varphi ::= P_a(x) \mid x = y \mid x \in X \mid S(x,y) \mid \exists x\varphi \mid \exists X\varphi \mid f(\varphi,\dots,\varphi)$$

where $a \in \mathrm{AP}$, $x, y \in \mathcal{V}ar_1$, $X \in \mathcal{V}ar_2$ and $f \in \mathcal{F}$.

Formulas of MSO$[\mathcal{F}]$ are interpreted on infinite $(\mathcal{V}^{\mathrm{AP}}, D)$-trees and first-order variables represent nodes of the tree. But instead of usual sets, second-order variables represent *fuzzy sets*, i.e., mappings $X\colon \tau \to [0,1]$ indicating to which degree each node of the tree τ belongs to the set.

To evaluate formulas with free variables, we need *interpretations*: an interpretation $\mathcal{I}$ for a formula $\varphi(x_1,\dots,x_k,X_1,\dots,X_l)$ with free first-order variables $x_1,\dots,x_k$ and free second-order variables $X_1,\dots,X_l$ maps each free first-order variable x_i to some node $u \in \tau$ and each free second-order variable X_i to some fuzzy set $X\colon \tau \to [0,1]$. The domain of $\mathcal{I}$, denoted with $\mathrm{dom}(\mathcal{I})$, is the set $\{x_1,\dots,x_k,X_1,\dots,X_l\}$ of variables where $\mathcal{I}$ is defined.

Definition 7.2 (Semantics). Given a $(\mathcal{V}^{\mathrm{AP}}, D)$-tree $t = (\tau, \mathrm{w})$, an MSO$[\mathcal{F}]$ formula φ and an interpretation $\mathcal{I}$ for the free variables of φ, the semantics of φ is defined as follows:

$$\begin{aligned}
\{\!\{P_a(x)\}\!\}^t(\mathcal{I}) &= \mathrm{w}(\mathcal{I}(x))(a)\\
\{\!\{x = y\}\!\}^t(\mathcal{I}) &= \begin{cases} 1 & \text{if } \mathcal{I}(x) = \mathcal{I}(y)\\ 0 & \text{otherwise}\end{cases}\\
\{\!\{x \in X\}\!\}^t(\mathcal{I}) &= \mathcal{I}(X)(\mathcal{I}(x))\\
\{\!\{S(x,y)\}\!\}^t(\mathcal{I}) &= \begin{cases} 1 & \text{if } \mathcal{I}(y) \text{ is a child of } \mathcal{I}(x)\\ 0 & \text{otherwise}\end{cases}\\
\{\!\{\exists x\varphi\}\!\}^t(\mathcal{I}) &= \sup_{u\in\tau} \{\!\{\varphi\}\!\}^t(\mathcal{I}[x \mapsto u])\\
\{\!\{\exists X\varphi\}\!\}^t(\mathcal{I}) &= \sup_{Y\colon \tau\to[0,1]} \{\!\{\varphi\}\!\}^t(\mathcal{I}[X \mapsto Y])\\
\{\!\{f(\varphi_1,\dots,\varphi_n)\}\!\}^t(\mathcal{I}) &= f(\{\!\{\varphi_1\}\!\}^t(\mathcal{I}),\dots,\{\!\{\varphi_n\}\!\}^t(\mathcal{I}))
\end{aligned}$$

where $\mathcal{I}[\alpha \mapsto \beta]$ is the interpretation $\mathcal{J}$ such that $\mathrm{dom}(\mathcal{J}) = \mathrm{dom}(\mathcal{I}) \cup \{\alpha\}$, and $\mathcal{J}(\alpha) = \beta$ and $\mathcal{J}(\gamma) = \mathcal{I}(\gamma)$ for all $\gamma \in \mathrm{dom}(\mathcal{I}) \setminus \{\alpha\}$.

We also consider the case where set quantification is over classic sets instead of fuzzy ones. We call this logic BMSO$[\mathcal{F}]$. Its syntax is the same as MSO$[\mathcal{F}]$ and the semantics, which we write $\{\!\{\cdot\}\!\}^t_{\mathrm{B}}$, differs only in the case of second-order quantification, which is as follows:

$$\{\!\{\exists X\varphi\}\!\}^t_{\mathrm{B}}(\mathcal{I}) = \sup_{Y\colon \tau\to\{0,1\}} \{\!\{\varphi\}\!\}^t_{\mathrm{B}}(\mathcal{I}[X \mapsto Y])$$

Recall that, if Bool is the function that maps x to 1 if $x \in \{0,1\}$, and to 0 otherwise, then BQCTL$^\star[\mathcal{F}]$ is a fragment of QCTL$^\star[\mathcal{F} \cup \{\text{Bool}\}]$ (Lemma 3.5), and Boolean quantification on atoms can be expressed in QCTL$^\star[\mathcal{F}]$ as $\exists p.(\mathbf{AG}\ \text{Bool}(p) \wedge \varphi)$. Similarly, BMSO$[\mathcal{F}]$ is a fragment of MSO$[\mathcal{F} \cup \{\text{Bool}\}]$: define

$$\text{Bool}(X) := \forall x.\, \text{Bool}(x \in X) \qquad\qquad \exists^{\text{B}} X.\varphi := \exists X.\, \text{Bool}(X) \wedge \varphi$$

It holds that:

Lemma 7.3. *For every* BMSO$[\mathcal{F}]$ *formula* φ*, letting* φ_B *be the* MSO$[\mathcal{F}]$ *formula obtained from* φ *by replacing inductively each* $\exists X.\varphi'$ *in* φ *with* $\exists^B X.\varphi'_B$*, we have that for all tree* t *and interpretation* $\mathcal{I}$*,* $\{\!\{\exists^B X.\varphi\}\!\}^\tau(\mathcal{I}) = \{\!\{\exists X.\varphi\}\!\}^\tau_B(\mathcal{I})$.

Below we prove that MSO$[\mathcal{F}]$ and QCTL$^\star[\mathcal{F}]$ have the same expressive power. As noticed in [60], one important conceptual difference is that quantification in MSO$[\mathcal{F}]$ refers to the whole tree, while QCTL$^\star[\mathcal{F}]$ can only refer to the reachable part of the tree. As a consequence, MSO$[\mathcal{F}]$ can express the fact that the current node is the root of the tree, while QCTL$^\star[\mathcal{F}]$ cannot.

Formally, that both logics are equally expressive means that

(1) for any (state-)formula $\varphi \in$ QCTL$^\star[\mathcal{F}]$, there is a formula $\tilde{\varphi}(x) \in$ MSO$[\mathcal{F}]$ with one free first-order variable x such that for any tree t with root r, it holds $\{\!\{\varphi\}\!\}^t(r) = \{\!\{\tilde{\varphi}\}\!\}^t(x \mapsto r)$, and symmetrically,
(2) for any formula $\varphi(x) \in$ MSO$[\mathcal{F}]$ with one free first-order variable x, there is a formula $\hat{\varphi} \in$ QCTL$^\star[\mathcal{F}]$ such that for any tree t with root r, it holds $\{\!\{\hat{\varphi}\}\!\}^t(r) = \{\!\{\varphi\}\!\}^t(x \mapsto r)$.

We use the possibility to express Boolean second-order quantification to capture first-order quantification in elements in QCTL$^\star[\mathcal{F}]$, and also to express path quantification in MSO$[\mathcal{F}]$.

Theorem 7.4. *If* $\{\neg, \wedge, \mathit{Bool}\} \in \mathcal{F}$ *then* MSO$[\mathcal{F}]$ *has the same expressive power as* QCTL$^\star[\mathcal{F}]$.

Proof. Translating QCTL$^\star[\mathcal{F}]$ to MSO$[\mathcal{F}]$ can be done as usual, using first-order quantifiers to express temporal operators, and second-order quantifiers to express quantification on atomic propositions. The only subtlety here is to express path quantification in MSO$[\mathcal{F}]$. The usual translation consists in using second-order quantification and express that the quantified set is a path, with the following formula:

$$\text{Path}(X) := \forall x, y.\, (x \in X \wedge y \in X) \rightarrow (x \preceq y \vee y \preceq x)$$

where

$$x \preceq y := \forall X.\, (x \in X \wedge S\text{-Closed}(X) \rightarrow y \in X)$$

and

$$S\text{-Closed}(X) := \forall x, y.\, (x \in X \wedge S(x,y) \rightarrow y \in X)$$

The translation of $\mathbf{E}\psi$ is then defined as follows:

$$(\mathbf{E}\psi)' := \exists X.\, \text{Path}(X) \wedge \psi'(X)$$

where $\psi'(X)$ is the translation of ψ in MSO$[\mathcal{F}]$. However in our context, for this translation to have the intended value which is the supremum of the satisfaction value of ψ over all paths, formulas Path(X) and $x \preceq y$ must have a Boolean value. We obtain this by using quantification on Boolean sets $\exists^{\text{B}} X$ defined above, letting

$$x \preceq y := \forall^{\text{B}} X.\, (x \in X \wedge S\text{-Closed}(X) \rightarrow y \in X)$$

and

$$(\mathbf{E}\psi)' := \exists^{\text{B}} X.\, \text{Path}(X) \wedge \psi'$$

For the direction from MSO[$\mathcal{F}$] to QCTL*[$\mathcal{F}$], we adapt the translation from MSO to QCTL* presented in [60]: for $\varphi \in$ MSO[$\mathcal{F}$], where free variable x represents the root of the tree, we inductively define the QCTL*[$\mathcal{F}$] formula $\widehat{\varphi}$ as follows:

$$\begin{array}{rcl@{\qquad}rcl}
\widehat{x = y} & := & p_y & \widehat{y = z} & := & \mathbf{EF}(p_y \wedge p_z) \\
\widehat{x \in X} & := & p_X & \widehat{y \in X} & := & \mathbf{EF}(p_y \wedge p_X) \\
\widehat{\exists y.\varphi'} & := & \exists^{\mathrm{B}} p_y.\, \mathrm{uniq}(p_y) \wedge \widehat{\varphi'} & & & \\
\widehat{\exists X.\varphi'} & := & \exists p_X.\, \widehat{\varphi'} & & & \\
\widehat{S(x,y)} & := & \mathbf{EX} p_y & \widehat{S(y,x)} & := & 0 \\
\widehat{S(y,z)} & := & \mathbf{EF}(p_y \wedge \mathbf{EX} p_z) & & & \\
\widehat{f(\varphi_1,\ldots,\varphi_n)} & := & f(\widehat{\varphi_1},\ldots,\widehat{\varphi_n}) & & &
\end{array}$$

where

$$\mathrm{uniq}(p) := \mathbf{EF}p \wedge \forall^{\mathrm{B}} q.\ (\mathbf{EF}(p \wedge q) \rightarrow \mathbf{AG}(p \rightarrow q))$$

To see that the above translation is correct, note that if p is Boolean in $t = (\tau, \mathrm{w})$, then

$$\{\!\{\mathrm{uniq}(p)\}\!\}^t = \begin{cases} 1 & \text{if } \mathrm{w}^{-1}(1) \text{ is a singleton} \\ 0 & \text{otherwise} \end{cases}$$

For an interpretation $\mathcal{I}$ such that $\mathcal{I}(x)$ is the root of τ we have that $\{\!\{\varphi\}\!\}^\tau(\mathcal{I}) = \{\!\{\widehat{\varphi}\}\!\}^t$, where $t = (\tau, \mathrm{w})$ and $\mathrm{w}(u)(p_y) = 1$ if $\mathcal{I}(y) = u$, 0 otherwise, and $\mathrm{w}(u)(p_X) = \mathcal{I}(X)(u)$.

For the other direction, the usual translation from CTL* to MSO can be extended to a translation from CTL*[$\mathcal{F}$] to MSO[$\mathcal{F}$]. The only thing that we need to take care of is that quantification over paths must be translated using Boolean second-order quantification, which is expressible in MSO[$\mathcal{F}$] with formula $\exists^{\mathrm{B}} X.\varphi$. □

The above proof can be easily adapted to obtain the following result:

Theorem 7.5. *If* $\{\neg, \wedge, \mathit{Bool}\} \in \mathcal{F}$ *then* BMSO[$\mathcal{F}$] *has same expressive power as* BQCTL*[$\mathcal{F}$].

As explained in the proof of Theorem 7.4, it seems hard to express quantification on paths in MSO[$\mathcal{F}$], or first-order quantification in QCTL*[$\mathcal{F}$], without the possibility to quantify on Boolean sets. We have seen that this quantification on Boolean sets can be expressed with fuzzy second-order quantification if we can use function Bool, but we do not know whether the converse is true, and we conjecture that it is not, meaning that MSO[$\{\neg, \wedge, \mathrm{Bool}\}$] is strictly more expressive than MSO[$\{\neg, \wedge\}$] with $\exists^{\mathrm{B}} X$ added as a basic construct. Note that if $\exists^{\mathrm{B}} X$ and $\exists^{\mathrm{B}} p$ are added as basic constructs of MSO[$\mathcal{F}$] and QCTL*[$\mathcal{F}$] respectively, then Theorem 7.4 holds as soon as $\mathcal{F}$ contains $\neg$ and $\wedge$.

7.2 Undecidability of QCTL*[$\mathcal{F}$] model checking

In this section we show an undecidability result for the model checking of QCTL*[$\mathcal{F}$]. In fact we first prove it for the logic QLTL[$\mathcal{F}$] and then show that it entails undecidability of QCTL*[$\mathcal{F}$]. While in QCTL*[$\mathcal{F}$] second-order quantification is made at the level of state formulas, in QLTL[$\mathcal{F}$] such quantifications can depend on the path on which the formula is evaluated, which makes the proof easier.

Theorem 7.6. *Model checking* QLTL[$\mathcal{F}$] *(on a fixed Kripke structure) is undecidable, even when* $\mathcal{F}$ *only contains* $\neg$, $\wedge$ *and the following two functions:*

$$\begin{array}{lll} \textit{equal}\colon [0,1]^2 \to \{0,1\} & & \\ \quad (x,y) \mapsto 1 & \textit{if } y = x & \\ \quad (x,y) \mapsto 0 & \textit{otherwise} & \end{array} \qquad \begin{array}{lll} \textit{half}\colon [0,1]^2 \to \{0,1\} & & \\ \quad (x,y) \mapsto 1 & \textit{if } y = x/2 & \\ \quad (x,y) \mapsto 0 & \textit{otherwise} & \end{array}$$

Proof. The proof consists in expressing a tiling problem as a QLTL[$\mathcal{F}$] model checking problem. In this setting, given a finite set C of colors, a *tile* is a 4-tuple of colors of C, seen as the four colors of the edges of a square. We write $u = \langle \mathit{top}(u), \mathit{bottom}(u), \mathit{left}(u), \mathit{right}(u)\rangle$ for such a tile. Given a set B of tiles, a tiling of a subset Z of $\mathbb{N}^2$ with tiles of B is a mapping $t\colon Z \mapsto B$. Such a tiling is valid whenever the colors of neighboring tiles agree; formally, a tiling t is valid if for any two positions $p = (x, y)$ and $p' = (x', y')$ in Z, it holds

- if $x' = x$ and $y' = y + 1$, then $\mathit{top}(t(p)) = \mathit{bottom}(t(p'))$;
- if $y' = y$ and $x' = x + 1$, then $\mathit{right}(t(p)) = \mathit{left}(t(p'))$.

We encode the following tiling problem:

> Given a finite set of tiles $B = \{u_i \mid 1 \le i \le n\}$, does any rectangular grid admit a valid tiling?

This problem is undecidable [82, pages 213-214].

In our encoding, we consider the Kripke structure $\mathcal{K}$ depicted in Figure 1 (which does not depend on the set of tiles given as input). In that structure, there are three (Boolean) atomic propositions, and the name of each state indicates the only atomic proposition having value 1 in that state.

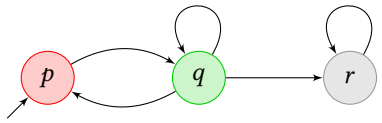

Fig. 1. Kripke structure $\mathcal{K}$

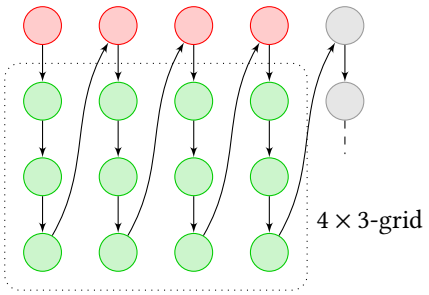

Fig. 2. A run of $\mathcal{K}$

We build two QLTL[$\mathcal{F}$] formulas in the sequel:

- the first one will be true along any path that corresponds to some rectangular grid, as depicted on Figure 2; additionally, if the path corresponds to such a grid, it will "number" all lines using via the value of an atomic proposition t;
- the second formula will rely on the numbering of lines to check the existence of a valid tiling.

For the first phase, we characterize the paths of $\mathcal{K}$ that encode a grid as shown on Figure 2. This is enforced using the conjunction of the following formulas:

- we label all states with a quantitative atomic proposition t in such a way that the value of t is divided by 2 as long as we only visit q-states, and it restarts from 1 at p- or r-states:

$$\exists t.\, \mathbf{G}((\neg q) \to t) \wedge \mathbf{G}((\mathbf{X}q) \to \mathrm{half}(t, \mathbf{X}t)).$$

  For this formula to take value 1, alll p- and r-states must have $t = 1$, and the value of t in a q-states must be half the value of t in the previous state;

- we enforce that we have a grid: all columns (of q-states) must have the same length:

$$\mathbf{G}((\mathbf{X}p) \rightarrow \mathsf{equal}(t, \mathbf{X}((\mathbf{X}q)\mathbf{U}(t \wedge \mathbf{X}\neg q)))).$$

  This takes value 1 if, and only if, the value of t in a state where $\mathbf{X}p$ holds (i.e., in the last state of all but the last column) equals the value of t in the last state of the next column.

We write Ψ_{grid} for the conjunction of the two formulas above, and $\mathbf{F}r$. Notice that Ψ_{grid} can be written as $\exists t.\ \psi_{\mathsf{grid}}$, with $\psi_{\mathsf{grid}} \in \mathrm{LTL}[\mathsf{equal}, \mathsf{half}]$. Figure 3 represents the path of Figure 2 when labeled with values for t witnessing the fact that Ψ_{grid} has value 1 along that path.

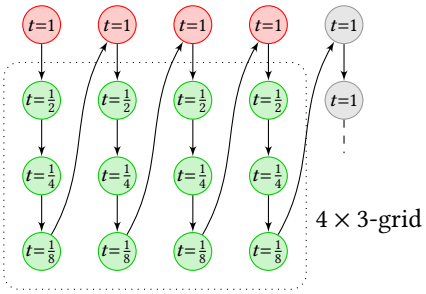

Fig. 3. A run of $\mathcal{K}$, labeled with values for t

The following lemma formalizes the meaning of Ψ_{grid}:

Lemma 7.7. *For any infinite path π in the Kripke structure $\mathcal{K}$ of Figure 1, it holds: $\{\!\{\Psi_{grid}\}\!\}(\pi) = 1$ if, and only if, π has the form $(p \cdot q^l)^m \cdot r^\omega$ for some $l, m \in \mathbb{N}$.*

We now encode the tiling problem on this grid: this is achieved by labeling all q-states with a (Boolean) atomic proposition from $(u_i)_{1 \leq i \leq n}$, and checking that neighboring tiles match:

- we first label q-states with the tiles u_i for all $1 \leq i \leq n$, in such a way that all cells of the grid receive a unique tile:

$$\exists (u_i)_{1\leq i\leq n}.\ \mathbf{G}\Big(q \rightarrow \Big(\bigvee_{1\leq i\leq n}\Big(\mathsf{equal}(u_i,\top) \wedge \bigwedge_{\substack{1\leq j\leq n\\ j\neq i}} \mathsf{equal}(u_j,\bot)\Big)\Big)\Big) \wedge \mathbf{G}\Big((\neg q) \rightarrow \bigwedge_{1\leq j\leq n} \mathsf{equal}(u_j,\bot)\Big)$$

- we check vertical matching:

$$\mathbf{G}((q \wedge \mathbf{X}q) \rightarrow \text{v-match})$$

  where v-match is a purely Boolean formula listing all combinations of formulas of the form $u_i \wedge \mathbf{X}u_j$ where the bottom-color of tile u_i and the top-color of u_j match;
- we check horizontal matching: for this we have to make the formula find the right-neighboring cell in the grid. For this, we add an extra labeling with another atomic proposition v; thanks to formula $(\mathsf{equal}(t, \mathbf{G}v) \wedge \mathsf{equal}(t, \mathbf{F}v))$, we force v to take the same value as the current t-value in all future states. The right-neighbor then is the (strict) next q-state where t and v have the same value; in that state, we check that the colors match:

$$\bigwedge_{1\leq i\leq n} \mathbf{G}\big(u_i \rightarrow \exists v.\ \mathsf{equal}(t, \mathbf{G}v) \wedge \mathsf{equal}(t, \mathbf{F}v) \wedge \mathbf{X}((\neg\mathsf{equal}(t, v))\mathbf{U}(\mathsf{equal}(t, v) \wedge \text{h-match}(u_i)))\big)$$

  where h-match(u_i) is a disjunction of all tiles whose left-color matches the right-color of u_i.

Write Ψ_{tiling} for the conjunction of the three formulas above. Notice that Ψ_{tiling} can be written as $\exists (u_i)_{1\leq i\leq n}.\ \psi_{\mathsf{tiling}}$, with $\psi_{\mathsf{tiling}} \in \mathrm{QLTL}[\{\mathsf{equal}\}]$. Then:

Lemma 7.8. *Let $\pi = (s_j)_{j\in\mathbb{N}}$ be an infinite path of the form $(p \cdot q^l)^m \cdot r^\omega$ for some $l, m \in \mathbb{N}$ in the Kripke structure $\mathcal{K}$ of Figure 1. Assume that π is labeled with atomic proposition t in such a way that for any i and j in $[0, m(l+1)-1]$, it holds that $\ell(s_i)(t) = \ell(s_j)(t)$ if, and only if, $i - j$ is a multiple of $l+1$. Then $\{\!\{\Psi_{tiling}\}\!\}(\pi) = 1$ if, and only if, the grid of size $l \times m$ admits a valid tiling with tiles in $\{u_i \mid 1 \le i \le n\}$.*

To conclude the proof of Theorem 7.6, we let Ψ be the QLTL[{equal, half}] formula

$$\Psi = \forall t.\ \exists (u_i)_{1\le i\le n}.\ (\mathsf{equal}(\psi_{\mathsf{grid}}, \top) \rightarrow \mathsf{equal}(\psi_{\mathsf{tiling}}, \top)).$$

Then $\{\!\{\Psi\}\!\}(\mathcal{K}) = 1$ if, and only if, any rectangular grid can be tiled with tiles $\{u_i \mid 1 \le i \le n\}$. □

Now to obtain undecidability of QCTL$^\star[\mathcal{F}]$ we need to show that all quantifications on atomic propositions can be made at the beginning, as QCTL$^\star[\mathcal{F}]$ does not allow mixing second-order quantification with temporal operators without requantifying on paths. To this aim notice that formula Ψ is almost in prenex form, but ψ_{tiling} still contains an existential quantification under a temporal modality. However, Ψ can be turned into prenex form, by noticing that for all QLTL$[\mathcal{F}]$ formulas ψ and ψ',

$$\mathbf{G}(\psi \rightarrow \exists v.\ \psi') \equiv \forall w.\ \exists v.\ (\mathsf{uniq}(w) \rightarrow \mathbf{G}((\psi \wedge w) \rightarrow \psi'))$$

where w is a new atomic proposition, and

$$\mathsf{uniq}(w) = \mathsf{equal}(w, \bot)\ \mathbf{U}\ (\mathsf{equal}(w, \top) \wedge \mathbf{XG}\,\mathsf{equal}(w, \bot))$$

requires w to be Boolean and to only label one state of the path.

In the end, Ψ can be written as

$$\forall t.\ \exists (u_i)_{1\le i\le n}.\ \forall w.\ \exists v.\ (\mathsf{equal}(\psi_{\mathsf{grid}}, \top) \rightarrow \mathsf{equal}(\psi'_{\mathsf{tiling}}, \top))$$

where $\psi'_{\mathsf{tiling}} \in$ LTL[equal] is obtained by applying the transformation above.

Now define the QCTL$^\star[\mathcal{F}]$ formula

$$\Phi = \forall t.\ \exists (u_i)_{1\le i\le n}.\ \forall w.\ \exists v.\ \mathbf{A}(\mathsf{equal}(\psi_{\mathsf{grid}}, \top) \rightarrow \mathsf{equal}(\psi'_{\mathsf{tiling}}, \top))$$

It is clear that $\{\!\{\Psi\}\!\}(\mathcal{K}) = 1$ if and only if $\{\!\{\Phi\}\!\}(\mathcal{K}) = 1$, hence the following result:

Theorem 7.9. *Model checking* QCTL$^\star[\mathcal{F}]$ *(on a fixed Kripke structure) is undecidable, even when $\mathcal{F}$ only contains $\neg$, $\wedge$, equal and half.*

7.3 Lipschitz continuity

Fuzzy logic is used to reason about imprecise information. In that setting, Lipschitz continuity guarantees a certain level of precision of the value of the formula depending on the level of precision of the information we have about the model [5, 64, 78]. In this section we show that when the quality functions in $\mathcal{F}$ are Lipschitz continuous, then so is the satisfaction value of QCTL$^\star[\mathcal{F}]$ formulas.

Let $P \subseteq$ AP. A P-labeling for a tree $t = (\tau, \mathrm{w})$ is a mapping $\ell_P \colon P \rightarrow (\tau \rightarrow [0,1])$. For any two subsets P and Q of AP, any P-labeling ℓ_P and Q-labeling ℓ_Q of t, we define their composition as the $P \cup Q$-labeling defined as follows:

$$\begin{array}{rcl} \ell_P \otimes \ell_Q \colon P \cup Q & \rightarrow & [0,1] \\ p & \mapsto & \begin{cases} \ell_Q(p) & \text{if } p \in Q, \text{ and} \\ \ell_P(p) & \text{if } p \in P \setminus Q. \end{cases} \end{array}$$

The distance between two P-labelings of $t = (\tau, \mathrm{w})$ is defined as

$$d_{P,t}(\ell_P, \ell'_P) = \sup_{u \in \tau} \max_{q \in P} |\ell_P(q)(u) - \ell'_P(q)(u)|.$$

Notice that, given ℓ_P, ℓ'_P and ℓ_Q, it holds

$$d_{P \cup Q,t}(\ell_P \otimes \ell_Q, \ell'_P \otimes \ell_Q) \leq d_{P,t}(\ell_P, \ell'_P).$$

When t and P are clear from the context, we omit them from the notation.

Let $t = (\tau, \mathrm{w})$ be a $([0,1]^{\mathrm{AP}}, D)$-tree, and u be a node of t. Let $\varphi \in \mathrm{QCTL}^\star[\mathcal{F}]$ be a state formula over AP. We define

$$\begin{array}{rccl} F(t,u,P,\varphi)\colon & ([0,1]^\tau)^P & \to & [0,1] \\ & \ell_P & \mapsto & \{\!\{\varphi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(u). \end{array}$$

Similarly, given an infinite branch λ of t and a path formula $\psi \in \mathrm{QCTL}^\star[\mathcal{F}]$, we define

$$\begin{array}{rccl} F(t,\lambda,P,\psi)\colon & ([0,1]^\tau)^P & \to & [0,1] \\ & \ell_P & \mapsto & \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda). \end{array}$$

For two metric spaces (D, d_D) and (I, d_I), a function $f\colon D \to I$ is *k-Lipschitz continuous* when, for every two elements X and Y in D, it holds

$$d_I(f(X), f(Y)) \leq k \cdot d_D(X, Y).$$

A $\mathrm{QCTL}^\star[\mathcal{F}]$ state formula φ is then k-Lipschitz continuous if, for every tree t, node u of t and subset $P \subseteq \mathrm{AP}$, the function $F(t,u,P,\varphi)$ is k-Lipschitz continuous. The definition is extended to path formulas in the natural way.

Theorem 7.10. *If all functions in $\mathcal{F}$ are k-Lipschitz continuous, then every formula in $\mathrm{QCTL}^\star[\mathcal{F}]$ having at most n nested functions is $\max(1, k^n)$-Lipschitz continuous.*

Proof. Consider a $\mathrm{QCTL}^\star[\mathcal{F}]$ formula φ. The proof is by induction on the structure of φ. The base case where φ is an atomic proposition p is easy: pick a tree t, a node u and a subset $P \subseteq \mathrm{AP}$; if $p \notin P$, then $F(t,u,P,\varphi)$ is a constant function; if $p \in P$, then $F(t,u,P,\varphi)(\ell_P) = \ell_P(p)(u)$ for any P-labeling ℓ_P, so that

$$|F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P)| = |\ell_P(p)(u) - \ell'_P(p)(u)| \leq d(\ell_P, \ell'_P)$$

We now consider the other cases, assuming that the result holds for all subformulas of φ:

- if $\varphi = \exists q.\psi$, then for any tree $t = (\tau, \mathrm{w})$, any node u, any subset $P \subseteq \mathrm{AP}$, and any P-labeling ℓ_P, we have
$$F(t,u,P,\varphi)(\ell_P) = \sup_{\ell_q \ \{q\}\text{-labeling}} \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P\otimes\ell_q)}(u).$$
By definition of sup, for any $\varepsilon > 0$, there exists a q-labeling ℓ_q^ε such that
$$F(t,u,P,\varphi)(\ell_P) \leq \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P\otimes\ell_q^\varepsilon)}(u) + \varepsilon.$$
Moreover, for any P-labeling ℓ'_P, we have
$$F(t,u,P,\varphi)(\ell'_P) = \sup_{\ell_q \ \{q\}\text{-labeling}} \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P\otimes\ell_q)}(u) \geq \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P\otimes\ell_q^\varepsilon)}(u).$$
It follows
$$\begin{aligned} F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P) &\leq \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P\otimes\ell_q^\varepsilon)}(u) - \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P\otimes\ell_q^\varepsilon)}(u) + \varepsilon \\ &\leq \max(1,k^n) \cdot d_{P\cup\{q\},t}(\ell_P \otimes \ell_q^\varepsilon, \ell'_P \otimes \ell_q^\varepsilon) + \varepsilon \qquad \text{(by i.h.)} \\ &\leq \max(1,k^n) \cdot d_{P,t}(\ell_P, \ell'_P) + \varepsilon. \end{aligned}$$

By symmetry, we get

$$|F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P)| \leq \max(1,k^n)\cdot d(\ell_P,\ell'_P) + \varepsilon.$$

As this holds for any positive ε, we have

$$|F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P)| \leq \max(1,k^n)\cdot d(\ell_P,\ell'_P).$$

- if $\varphi = \mathbf{E}\psi$, we apply similar arguments:

$$F(t,u,P,\varphi)(\ell_P) = \sup_{\lambda\in \mathrm{Br}(u)} \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda).$$

For any $\varepsilon > 0$, there is a branch λ_ε such that

$$F(t,u,P,\varphi)(\ell_P) \leq \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_\varepsilon) + \varepsilon.$$

Then for any P-labeling ℓ'_P, we have

$$\begin{aligned} F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P) &\leq \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_\varepsilon) - \{\!\{\psi\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P)}(\lambda_\varepsilon) + \varepsilon \\ &\leq \max(1,k^n)\cdot d(\ell_P,\ell'_P) + \varepsilon. \end{aligned}$$

As previously, it follows that

$$|F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P)| \leq \max(1,k^n)\cdot d(\ell_P,\ell'_P).$$

- if $\varphi = f(\varphi_1,...,\varphi_m)$, then

$$F(t,u,P,\varphi)(\ell_P) = f(F(t,u,P,\varphi_1)(\ell_P),...,F(t,u,P,\varphi_m)(\ell_P)).$$

Because f is k-Lipschitz continuous, we have

$$\begin{aligned} |F(t,u,P,\varphi)(\ell_P) - F(t,u,P,\varphi)(\ell'_P)| &\leq k\cdot \max_i(|F(t,u,P,\varphi_i)(\ell_P) - F(t,u,P,\varphi_i)(\ell'_P)|) \\ &\leq k\cdot \max(1,k^{n-1}) d(\ell_P,\ell'_P). \end{aligned}$$

- if $\varphi = \psi_1 \mathbf{U} \psi_2$, then

$$F(t,\lambda,P,\varphi)(\ell_P) = \sup_{i\in\mathbb{N}} \left\{ \min(\{\!\{\psi_2\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq i}), \min_{j<i}(\{\!\{\psi_1\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq j}))) \right\}.$$

Then for any $\varepsilon > 0$, there exists i_ε such that

$$F(t,\lambda,P,\varphi)(\ell_P) \leq \min(\{\!\{\psi_2\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq i_\varepsilon}), \min_{j<i_\varepsilon}(\{\!\{\psi_1\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq j}))) + \varepsilon.$$

Then for any ℓ'_P,

$$\begin{aligned} F(t,\lambda,P,\varphi)(\ell_P) - F(t,\lambda,P,\varphi)(\ell'_P) \leq \min(\{\!\{\psi_2\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq i_\varepsilon}), \min_{j<i_\varepsilon}(\{\!\{\psi_1\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq j}))) - \\ \min(\{\!\{\psi_2\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P)}(\lambda_{\geq i_\varepsilon}), \min_{j<i_\varepsilon}(\{\!\{\psi_1\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P)}(\lambda_{\geq j}))) + \varepsilon \end{aligned}$$

If the latter minimum is obtained for ψ_2 along $\lambda_{\geq i_\varepsilon}$, then we get

$$F(t,\lambda,P,\varphi)(\ell_P) - F(t,\lambda,P,\varphi)(\ell'_P) \leq \{\!\{\psi_2\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq i_\varepsilon}) - \{\!\{\psi_2\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P)}(\lambda_{\geq i_\varepsilon}) + \varepsilon.$$

Otherwise, this minimum is reached at some index $j_0 < i_\varepsilon$, and we have

$$F(t,\lambda,P,\varphi)(\ell_P) - F(t,\lambda,P,\varphi)(\ell'_P) \leq \{\!\{\psi_1\}\!\}^{(\tau,\mathrm{w}\otimes\ell_P)}(\lambda_{\geq j_0}) - \{\!\{\psi_1\}\!\}^{(\tau,\mathrm{w}\otimes\ell'_P)}(\lambda_{\geq j_0}) + \varepsilon.$$

For both cases, thanks to the induction hypothesis and applying the same arguments as for previous cases, we end up with

$$|F(t,\lambda,P,\varphi)(\ell_P) - F(t,\lambda,P,\varphi)(\ell'_P)| \leq \max(1,k^n)\cdot d(\ell_P,\ell'_P).$$

- if $\varphi = \mathbf{X}\psi$, the result is straightforward. □

It is not hard to see that all the considerations in the proof of Theorem 7.10, in particular the case $\varphi = \exists q.\psi$, apply also when the quantification is over Boolean atomic propositions, hence we also have the following.

Theorem 7.11. *If all functions in $\mathcal{F}$ are k-Lipschitz continuous, then every formula in* BQCTL$^\star[\mathcal{F}]$ *having at most n nested functions is* $\max(1, k^n)$*-Lipschitz continuous.*

8 FUTURE WORK

We introduced and studied SL$[\mathcal{F}]$, a formalism for specifying quality and fuzziness of strategic on-going behavior. We demonstrated the expressiveness of SL$[\mathcal{F}]$ through a number of examples from multi-agent systems, solved its model-checking problem and established its precise complexity. While it is non-elementary in general, we also identified an expressive fragment (SL$_{1G}[\mathcal{F}]$) that, as in the Boolean case, retains an elementary model-checking problem (it is 2-Exptime-complete).

As a means of studying SL$[\mathcal{F}]$, we extended QCTL* to the quantitative setting, leading to the logics QCTL$^\star[\mathcal{F}]$ and BQCTL$^\star[\mathcal{F}]$ depending on whether quantified propositions range over $[0, 1]$ or only over Boolean values $\{0, 1\}$. We described how the latter can be seen as a fragment of the former, and how they relate to MSO and Strategy Logic, with BQCTL$^\star[\mathcal{F}]$ allowing to capture pure strategies, while QCTL$^\star[\mathcal{F}]$ could be used to capture mixed ones. We showed that their semantics is Lipschitz-continuous when they use only Lipschitz-continuous functions. Finally we studied their model-checking problem and showed that it is undecidable for QCTL$^\star[\mathcal{F}]$ but decidable for BQCTL$^\star[\mathcal{F}]$, which allows us to prove decidability of model checking SL$[\mathcal{F}]$.

Beyond the applications described in the paper, we highlight here some interesting directions for future research. In classical temporal-logic model checking, *coverage* and *vacuity* algorithms measure the sensitivity of the system and its specifications to mutations, revealing errors in the modeling of the system and lack of exhaustiveness of the specification [33]. When applied to SL$[\mathcal{F}]$, these algorithms can set the basis to a formal reasoning about classical notions in game theory, like the sensitivity of utilities to price changes, the effectiveness of burning money [37, 51] or tax increase [35], and more.

9 ACKNOWLEDGMENTS

This work is partially based on the paper [19], which appeared in IJCAI'19.

This work is partially supported by the ERC Advanced Grant WhiteMech (No. 834228), by the EU ICT-48 2020 project TAILOR (No. 952215), by the PRIN project RIPER (No. 20203FFYLK), and by the JPMorgan AI Faculty Research Award "Resilience-based Generalized Planning and Strategic Reasoning".

REFERENCES

[1] Natasha Alechina, Brian Logan, Nguyen Hoang Nga, and Abdur Rakib. 2010. Resource-bounded alternating-time temporal logic. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'10)*, Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandeep Sen (Eds.). International Foundation for Autonomous Agents and Multiagent Systems, 481–488.

[2] Shaull Almagor, Udi Boker, and Orna Kupferman. 2014. Discounting in LTL. In *Proceedings of the 20th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS'14) (Lecture Notes in Computer Science, Vol. 8413)*, Erikz Ábrahám and Klaus Havelund (Eds.). Springer-Verlag, 424–439. https://doi.org/10.1007/978-3-642-54862-8_37

[3] Shaull Almagor, Udi Boker, and Orna Kupferman. 2016. Formally Reasoning about Quality. *J. ACM* 63, 3 (Sept. 2016), 24:1–24:56. https://doi.org/10.1145/2875421

[4] Shaull Almagor and Orna Kupferman. 2016. High-Quality Synthesis Against Stochastic Environments. In *Proceedings of the 25th EACSL Annual Conference on Computer Science Logic (CSL'16) (Leibniz International Proceedings in Informatics, Vol. 62)*, Jean-Marc Talbot and Laurent Regnier (Eds.). Leibniz-Zentrum für Informatik, 28:1–28:17. https://doi.org/10.4230/LIPIcs.CSL.2016.28

[5] Shaull Almagor and Orna Kupferman. 2017. Latticed-LTL synthesis in the presence of noisy inputs. *Discrete Event Dynamic Systems* 27, 3 (2017), 547–572. https://doi.org/10.1007/s10626-017-0242-0

[6] Shaull Almagor, Orna Kupferman, and Giuseppe Perelli. 2018. Synthesis of Controllable Nash Equilibria in Quantitative Objective Games. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, Jérôme Lang (Ed.). IJCAI organization, 35–41. https://doi.org/10.24963/ijcai.2018/5

[7] Shaull Almagor, Orna Kupferman, Jan Oliver Ringert, and Yaron Velner. 2017. High-Quality Synthesis Against Stochastic Environments. In *Proceedings of the 29th International Conference on Computer Aided Verification (CAV'17) (Lecture Notes in Computer Science, Vol. 10427)*, Rupak and Majumdar (Ed.). Springer-Verlag, 353–374. https://doi.org/10.1007/978-3-319-63390-9_19

[8] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 2002. Alternating-time Temporal Logic. *J. ACM* 49, 5 (Sept. 2002), 672–713. https://doi.org/10.1145/585265.585270

[9] Benjamin Aminof, Marta Kwiatkowska, Bastien Maubert, Aniello Murano, and Sasha Rubin. 2019. Probabilistic strategy logic. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*. IJCAI, IJCAI organization, 32–38. https://doi.org/10.24963/ijcai.2019/5

[10] Benjamin Aminof, Vadim Malvone, Aniello Murano, and Sasha Rubin. 2018. Graded modalities in Strategy Logic. *Information and Computation* 261, 4 (Aug. 2018), 634–649. https://doi.org/10.1016/j.ic.2018.02.021

[11] Eugene Asarin, Oded Maler, Amir Pnueli, and Joseph Sifakis. 1998. Controller Synthesis for Timed Automata. In *Proceedings of the 5th IFAC Cconference on System Structure and Control (SSC'98)*. Elsevier, 469–474.

[12] Christel Baier, Tomáš Brázdil, Marcus Größer, and Antonín Kučera. 2012. Stochastic Game Logic. *Acta Informatica* 49, 4 (June 2012), 203–224. https://doi.org/10.1007/s00236-012-0156-0

[13] Raphaël Berthon, Bastien Maubert, Aniello Murano, Sasha Rubin, and Moshe Y. Vardi. 2017. Strategy logic with imperfect information. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'17)*. IEEE Computer Society, 1–12. https://doi.org/10.1109/LICS.2017.8005136

[14] Dietmar Berwanger. 2007. Admissibility in Infinite Games. In *Proceedings of the 24th Annual Symposium on Theoretical Aspects of Computer Science (STACS'07) (Lecture Notes in Computer Science, Vol. 4393)*. Springer, 188–199. https://doi.org/10.1007/978-3-540-70918-3_17

[15] Udi Boker, Krishnendu Chatterjee, Thomas A. Henzinger, and Orna Kupferman. 2014. Temporal Specifications with Accumulative Values. *ACM Transactions on Computational Logic* 15, 4 (Aug. 2014), 27:1–27:25. https://doi.org/10.1145/2629686

[16] Benedikt Bollig, Normann Decker, and Martin Leucker. 2012. Frequency Linear-time Temporal Logic. In *Proceedings of the 6th International Symposium on Theoretical Aspects of Software Engineering (TASE'12)*, Tiziana Margaria, Zongyang Qiu, and Hongli Yang (Eds.). IEEE Comp. Soc. Press, 85–92. https://doi.org/10.1109/TASE.2012.43

[17] Patricia Bouyer, Uli Fahrenberg, Kim Guldstrand Larsen, Nicolas Markey, and Jiří Srba. 2008. Infinite Runs in Weighted Timed Automata with Energy Constraints. In *Proceedings of the 6th International Conferences on Formal Modelling and Analysis of Timed Systems (FORMATS'08) (Lecture Notes in Computer Science, Vol. 5215)*, Franck Cassez and Claude Jard (Eds.). Springer-Verlag, 33–47. https://doi.org/10.1007/978-3-540-85778-5_4

[18] Patricia Bouyer, Patrick Gardy, and Nicolas Markey. 2015. Weighted Strategy Logic with Boolean Goals over One-counter Games. In *Proceedings of the 35th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'15) (Leibniz International Proceedings in Informatics, Vol. 45)*, Prahladh Harsha and G. Ramalingam (Eds.). Leibniz-Zentrum für Informatik, 69–83. https://doi.org/10.4230/LIPIcs.FSTTCS.2015.69

[19] Patricia Bouyer, Orna Kupferman, Nicolas Markey, Bastien Maubert, Nello Murano, and Giuseppe Perelli. 2019. Reasoning about Quality and Fuzziness of Strategic Behaviours. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*. IJCAI organization, 1588–1594. https://doi.org/10.24963/ijcai.2019/220

[20] Patricia Bouyer, Nicolas Markey, and Raj Mohan Matteplackel. 2014. Averaging in LTL. In *Proceedings of the 25th International Conference on Concurrency Theory (CONCUR'14) (Lecture Notes in Computer Science, Vol. 8704)*, Paolo Baldan and Daniele Gorla (Eds.). Springer-Verlag, 266–280. https://doi.org/10.1007/978-3-662-44584-6_19

[21] Romain Brenguier. 2016. Robust Equilibria in Mean-Payoff Games. In *Proceedings of the 19th International Conference on Foundations of Software Science and Computation Structures (FOSSACS'16) (Lecture Notes in Computer Science, Vol. 9634)*. Springer, 217–233. https://doi.org/10.1007/978-3-662-49630-5_13

[22] Romain Brenguier, Guillermo A. Pérez, Jean-François Raskin, and Mathieu Sassolas. 2016. Admissibility in Quantitative Graph Games. In *Proceedings of the 36th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'16) (Leibniz International Proceedings in Informatics)*, S. Akshay, Akash Lal, Saket Saurabh, and Sandeep Sen (Eds.). Leibniz-Zentrum für Informatik, 42:1–42:14. https://doi.org/10.4230/LIPIcs.FSTTCS.2016.42

[23] Romain Brenguier, Jean-François Raskin, and Ocan Sankur. 2017. Assume-admissible synthesis. *Acta Informatica* 54, 1 (2017), 41–83. https://doi.org/10.1007/s00236-016-0273-2

[24] Thomas Brihaye, Véronique Bruyère, and Julie De Pril. 2014. On Equilibria in Quantitative Games with Reachability/Safety Objectives. *Theory of Computing Systems* 54, 2 (Feb. 2014), 150–189. https://doi.org/10.1007/s00224-013-9495-7

[25] Thomas Brihaye, François Laroussinie, Nicolas Markey, and Ghassan Oreiby. 2007. Timed Concurrent Game Structures. In *Proceedings of the 18th International Conference on Concurrency Theory (CONCUR'07) (Lecture Notes in Computer Science, Vol. 4703)*, Luís Caires and Vasco T. Vasconcelos (Eds.). Springer-Verlag, 445–459. https://doi.org/10.1007/978-3-540-74407-8_30

[26] Véronique Bruyère, Noëmie Meunier, and Jean-François Raskin. 2014. Secure Equilibria in Weighted Games. In *Proceedings of the Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic and the 29th Annual ACM/IEEE Symposium on Logic In Computer Science (CSL/LICS'14)*. ACM Press. https://doi.org/10.1145/2603088.2603109

[27] Nils Bulling and Valentin Goranko. 2013. How to Be Both Rich and Happy: Combining Quantitative and Qualitative Strategic Reasoning about Multi-Player Games (Extended Abstract). In *Proceedings of the 1st International Workshop on Strategic Reasoning (SR'13) (Electronic Proceedings in Theoretical Computer Science, Vol. 112)*. 33–41. https://doi.org/10.4204/EPTCS.112.8

[28] Arindam Chakrabarti, Luca de Alfaro, Thomas A. Henzinger, and Mariëlle Stoelinga. 2003. Resource Interfaces. In *Proceedings of the 3rd International Conference on Embedded Software (EMSOFT'03) (Lecture Notes in Computer Science, Vol. 2855)*, Rajeev Alur and Insup Lee (Eds.). Springer-Verlag, 117–133. https://doi.org/10.1007/978-3-540-45212-6_9

[29] Krishnendu Chatterjee, Laurent Doyen, Emmanuel Filiot, and Jean-François Raskin. 2014. Doomsday Equilibria for Omega-Regular Games. In *Proceedings of the 15th International Workshop on Verification, Model Checking, and Abstract Interpretation (VMCAI'14) (Lecture Notes in Computer Science, Vol. 8318)*, Kenneth L. McMillan and Xavier Rival (Eds.). Springer-Verlag, 78–97. https://doi.org/10.1007/978-3-642-54013-4_5

[30] Krishnendu Chatterjee, Thomas A Henzinger, and Marcin Jurdziński. 2006. Games with secure equilibria. *Theoretical Computer Science* 365, 1-2 (2006), 67–82. https://doi.org/10.1016/j.tcs.2006.07.032

[31] Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. 2010. Strategy Logic. *Information and Computation* 208, 6 (June 2010), 677–693. https://doi.org/10.1016/j.ic.2009.07.004

[32] Taolue Chen and Jian Lu. 2007. Probabilistic Alternating-time Temporal Logic and Model Checking Algorithm. In *Proceedings of the 4th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD'07)*, J. Lei (Ed.). IEEE Comp. Soc. Press, 35–39. https://doi.org/10.1109/FSKD.2007.458

[33] Hana Chockler, Orna Kupferman, and Moshe Y. Vardi. 2006. Coverage Metrics for Formal Verification. *Software Tools for Technology Transfer* 8, 4-5 (2006), 373–386. https://doi.org/10.1007/s10703-006-0001-6

[34] Edmund M. Clarke and E. Allen Emerson. 1982. Design and Synthesis of Synchronization Skeletons using Branching-Time Temporal Logic. In *Proceedings of the 3rd Workshop on Logics of Programs (LOP'81) (Lecture Notes in Computer Science, Vol. 131)*, Dexter C. Kozen (Ed.). Springer-Verlag, 52–71. https://doi.org/10.1007/BFb0025774

[35] Richard Cole, Yevgeniy Dodis, and Tim Roughgarden. 2006. How much can taxes help selfish routing? *J. Comput. System Sci.* 72, 3 (2006), 444–467. https://doi.org/10.1016/j.jcss.2005.09.010

[36] Luca de Alfaro, Marco Faella, Thomas A. Henzinger, Rupak Majumdar, and Mariëlle Stoelinga. 2005. Model checking discounted temporal properties. *Theoretical Computer Science* 345, 1 (Nov. 2005), 139–170. https://doi.org/10.1016/j.tcs.2005.07.033

[37] Filipe Costa de Souza and Leandro Chaves Rego. 2016. Mixed equilibrium in 2×2 normal form games: when burning money is rational. *Pesquisa Operacional* 36 (04 2016), 81–99.

[38] Dario Della Monica and Aniello Murano. 2018. Parity-energy ATL for Qualitative and Quantitative Reasoning in MAS. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'18)*, Elisabeth André, Sven Koenig, Mehdi Dastani, and Gita Sukthankar (Eds.). International Foundation for Autonomous Agents and Multiagent Systems, 1441–1449.

[39] Manfred Droste, Werner Kuich, and George Rahonis. 2008. Multi-Valued MSO Logics over Words and Trees. *Fundamenta Informaticae* 84, 13-4 (2008), 305–327.

[40] Manfred Droste, Temur Kutsia, George Rahonis, and Wolfgang Schreiner. 2017. MK-fuzzy Automata and MSO Logics. In *Proceedings of the 8th International Symposium on Games, Automata, Logics and Formal Verification (GandALF'17) (Electronic Proceedings in Theoretical Computer Science, Vol. 256)*, Patricia Bouyer, Andrea Orlandini, and Pierluigi San Pietro (Eds.). 106–120. https://doi.org/10.4204/EPTCS.256.8

[41] Andrzej Ehrenfeucht and Jan Mycielski. 1979. Positional strategies for mean payoff games. *International Journal of Game Theory* 8, 2 (June 1979), 109–113. https://doi.org/10.1007/BF01768705

[42] E. Allen Emerson and Joseph Y. Halpern. 1986. "Sometimes" and "Not Never" Revisited: On Branching versus Linear Time Temporal Logic. *J. ACM* 33, 1 (Jan. 1986), 151–178.

[43] E. Allen Emerson and Charanjit S. Jutla. 1991. Tree Automata, Mu-Calculus and Determinacy. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science (FOCS'91)*. IEEE Comp. Soc. Press, 368–377. https://doi.org/10.1109/SFCS.1991.185392

[44] E. Allen Emerson and Chin-Laung Lei. 1987. Modalities for Model Checking: Branching Time Logic Strikes Back. *Science of Computer Programming* 8 (1987), 275–306. https://doi.org/10.1016/0167-6423(87)90036-0

[45] Nathanaël Fijalkow, Bastien Maubert, Aniello Murano, and Sasha Rubin. 2018. Quantifying Bounds in Strategy Logic. In *Proceedings of the 27th EACSL Annual Conference on Computer Science Logic (CSL'18) (Leibniz International Proceedings in Informatics, Vol. 119)*. Leibniz-Zentrum für Informatik, 23:1–23:23. https://doi.org/10.4230/LIPIcs.CSL.2018.23

[46] Jerzy Filar and Koos Vrieze. 1997. *Competitive Markov decision processes*. Springer-Verlag.

[47] Dana Fisman, Orna Kupferman, and Yoad Lustig. 2010. Rational synthesis. In *Proceedings of the 16th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS'10) (Lecture Notes in Computer Science, Vol. 6015)*, Javier Esparza and Rupak Majumdar (Eds.). Springer-Verlag, 190–204. https://doi.org/10.1007/978-3-642-12002-2_16

[48] Tim French. 2001. Decidability of quantifed propositional branching time logics. In *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence (AJCAI'01)*. 165–176. https://doi.org/10.1007/3-540-45656-2_15

[49] Achille Frigeri, Liliana Pasquale, and Paola Spoletini. 2014. Fuzzy time in linear temporal logic. *ACM Transactions on Computational Logic* 15, 4 (2014), 30. https://doi.org/10.1145/2629606

[50] David Harel and Amir Pnueli. 1985. On the development of reactive systems. In *Logics and Models of Concurrent Systems*. NATO Advanced Summer Institutes, Vol. F-13. Springer, 477–498.

[51] Jason D. Hartline and Tim Roughgarden. 2008. Optimal mechanism design and money burning. In *Proceedings of the 40th ACM Symposium on Theory of Computing (STOC'08)*. 75–84. https://doi.org/10.1145/1374376.1374390

[52] Thomas A. Henzinger and Vinayak S. Prabhu. 2006. Timed Alternating-Time Temporal Logic. In *Proceedings of the 4th International Conferences on Formal Modelling and Analysis of Timed Systems (FORMATS'06) (Lecture Notes in Computer Science, Vol. 4202)*, Eugene Asarin and Patricia Bouyer (Eds.). Springer-Verlag, 1–17. https://doi.org/10.1007/11867340_1

[53] Wojciech Jamroga, Beata Konikowska, and Wojciech Penczek. 2016. Multi-Valued Verification of Strategic Ability. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'16)*, Catholijn M. Jonker, Stacy Marsella, John Thangarajah, and Karl Tuyls (Eds.). International Foundation for Autonomous Agents and Multiagent Systems, 1180–1189.

[54] Orna Kupferman. 1999. Augmenting branching temporal logics with existential quantification over atomic propositions. *Journal of Logic and Computation* 9, 2, 135–147. https://doi.org/10.1093/logcom/9.2.135

[55] Orna Kupferman, Parthasarathy Madhusudan, Pazhamaneri Subramaniam Thiagarajan, and Moshe Y. Vardi. 2000. Open Systems in Reactive Environments: Control and Synthesis. In *Proceedings of the 11th International Conference on Concurrency Theory (CONCUR'00)*. 92–107. https://doi.org/10.1007/3-540-44618-4_9

[56] Orna Kupferman, Giuseppe Perelli, and Moshe Y. Vardi. 2016. Synthesis with rational environments. *Annals of Mathematics and Artificial Intelligence* 78, 1 (2016), 3–20. https://doi.org/10.1007/s10472-016-9508-8

[57] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. 2000. An Automata-Theoretic Approach to Branching-Time Model Checking. *Journal of the ACM* 47, 2 (2000), 312–360. https://doi.org/10.1145/333979.333987

[58] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. 2000. An automata-theoretic approach to branching-time model checking. *Journal of the ACM* 47, 2 (2000), 312–360. https://doi.org/10.1145/333979.333987

[59] Khaled Ben Lamine and Froduald Kabanza. 2000. Using fuzzy temporal logic for monitoring behavior-based mobile robots. In *Proc. of IASTED Int. Conf. on Robotics and Applications*. 116–121.

[60] François Laroussinie and Nicolas Markey. 2014. Quantified CTL: Expressiveness and Complexity. *Logical Methods in Computer Science* 10, 4 (2014). https://doi.org/10.2168/LMCS-10(4:17)2014

[61] François Laroussinie and Nicolas Markey. 2015. Augmenting ATL with strategy contexts. *Information and Computation* 245 (2015), 98–123. https://doi.org/10.1016/j.ic.2014.12.020

[62] François Laroussinie, Nicolas Markey, and Ghassan Oreiby. 2006. Model-Checking Timed ATL for Durational Concurrent Game Structures. In *Proceedings of the 4th International Conferences on Formal Modelling and Analysis of Timed Systems (FORMATS'06) (Lecture Notes in Computer Science, Vol. 4202)*, Eugene Asarin and Patricia Bouyer (Eds.). Springer-Verlag, 245–259. https://doi.org/10.1007/11867340_18

[63] Jan Łukasiewicz. 1920. O Logice Trójwartościowej. *Studia Filozoficzne* (1920).

[64] Rupak Majumdar, Elaine Render, and Paulo Tabuada. 2011. Robust discrete synthesis against unspecified disturbances. In *Proceedings of the 14th ACM International Conference on Hybrid Systems: Computation and Control (HSCC'11)*. ACM, 211–220. https://doi.org/10.1145/1967701.1967732

[65] Vadim Malvone, Aniello Murano, and Loredana Sorrentino. 2016. Concurrent Multi-Player Parity Games. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'16)*. ACM, 689–697.

[66] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. 2012. What Makes ATL* Decidable? A Decidable Fragment of Strategy Logic. In *Proceedings of the 23rd International Conference on Concurrency Theory (CONCUR'12) (Lecture Notes in Computer Science, Vol. 7454)*, Maciej Koutny and Irek Ulidowski (Eds.). Springer-Verlag, 193–208. https://doi.org/10.1007/978-3-642-32940-1_15

[67] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. 2014. Reasoning About Strategies: On the Model-Checking Problem. *ACM Transactions on Computational Logic* 15, 4 (Aug. 2014), 34:1–34:47. https://doi.org/10.1145/2631917

[68] David E. Muller and Paul E. Schupp. 1995. Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of the Theorems of Rabin, McNaughton and Safra. *Theoretical Computer Science* 141, 1&2 (1995), 69–107. https://doi.org/10.1016/0304-3975(94)00214-4

[69] Noam Nisan and Amir Ronen. 1999. Algorithmic Mechanism Design. In *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC'99)*. ACM, 129–140. https://doi.org/10.1145/301250.301287

[70] Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani. 2007. *Algorithmic Game Theory*. Cambridge University Press. https://doi.org/10.1017/CBO9780511800481

[71] Nir Piterman. 2007. From Nondeterministic Büchi and Streett Automata to Deterministic Parity Automata. *Logical Methods in Computer Science* 3, 3 (2007), 1–21. https://doi.org/10.2168/LMCS-3(3:5)2007

[72] Amir Pnueli. 1981. The Temporal Semantics of Concurrent Programs. *Theoretical Computer Science* 13 (1981), 45–60. https://doi.org/10.1016/0304-3975(81)90110-9

[73] Amir Pnueli and Roni Rosner. 1989. On the Synthesis of a Reactive Module. In *Conference Record of the 16th ACM Symposium on Principles of Programming Languages (POPL'89)*. ACM Press, 179–190. https://doi.org/10.1145/75277.75293

[74] Michael O Rabin. 1969. Decidability of second-order theories and automata on infinite trees. *TAMS* 141 (1969), 1–35. https://doi.org/10.1090/S0002-9947-1969-0246760-1

[75] George Rahonis. 2009. Fuzzy Languages. In *Handbook of Weighted Automata*, Manfred Droste, Werner Kuich, and Walter Vogler (Eds.). Springer-Verlag, Chapter 12, 481–517. https://doi.org/10.1007/978-3-642-01492-512

[76] Lloyd S. Shapley. 1953. Stochastic games. *Proceedings of the National Academy of Sciences* 39, 10 (Oct. 1953), 1095–1100. https://doi.org/10.1073/pnas.39.10.1095

[77] A Prasad Sistla. 1983. *Theoretical Issues in the Design and Certification of Distributed Systems*. Ph.D. Dissertation. Harvard University, Cambridge, MA, USA.

[78] Ufuk Topcu, Necmiye Ozay, Jun Liu, and Richard M. Murray. 2012. On synthesizing robust discrete controllers under modeling uncertainty. In *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control (HSCC'12)*. ACM, 85–94. https://doi.org/10.1145/2185632.2185648

[79] Michael Ummels. 2010. *Stochastic Multiplayer Games: Theory and Algorithms*. Master's thesis. RWTH Aachen, Germany.

[80] M.Y. Vardi and P. Wolper. 1986. Automata-Theoretic Techniques for Modal Logics of Programs. *Journal of Computer and Systems Science* 32, 2 (1986), 182–221. https://doi.org/10.1016/0022-0000(86)90026-7

[81] Steen Vester. 2015. On the Complexity of Model-checking Branching and Alternating-time Temporal Logics in One-counter systems. In *Proceedings of the 13th International Symposium on Automated Technology for Verification and Analysis (ATVA'15) (Lecture Notes in Computer Science, Vol. 9364)*, Bernd Finkbeiner, Geguang Pu, and Lijun Zhang (Eds.). Springer-Verlag, 361–377. https://doi.org/10.1007/978-3-319-24953-7_27

[82] Hao Wang. 1990. *Computation, Logic, Philosophy*. Mathematics ans its applications, Vol. 2. Springer-Verlag. https://doi.org/10.1007/978-94-009-2356-0

[83] Uri Zwick and Mike Paterson. 1996. The complexity of mean payoff games on graphs. *Theoretical Computer Science* 158, 1-2 (1996), 343–359.