24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Reasoning about Quality and Fuzziness of Strategic Behaviours

Patricia Bouyer[1], **Orna Kupferman**[2], **Nicolas Markey**[3], **Bastien Maubert**[4], **Aniello Murano**[4], **Giuseppe Perelli**[5]

1 Context and motivation

One of the significant developments in formal reasoning has been the use of *temporal logics* for the specification of on-going behaviours of reactive systems [4, 5]. Traditional temporal logics are interpreted over Kripke structures, modelling 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 behaviours in multi-agent systems [2, 3, 10]. These formalisms, most notably ATL, ATL*, and Strategy Logic (SL), include quantification of 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.

The duration of games in classical game theory is finite and the outcome of the game depends on its final position [11]. In contrast, agents in multi-agent systems maintain an *on-going interaction* with each other [7], and reasoning about their behaviour refers not to their final state (in fact, we consider non-terminating systems, with no final state) but rather to the *language* of computations that they generate. The logics ATL* and SL both extend the Linear Temporal Logic LTL [12], and thus can express rich on-going strategic behaviours. However, their semantics are Boolean: a system may satisfy a specification or it may not. 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 only after being convinced that the automatic procedure that replaces it generates 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 the strategies. This is what we do 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 multi-valued logic that augments LTL with quality operators [1]. The satisfaction value of an LTL[$\mathcal{F}$] formula is a real value in $[0, 1]$, where the higher the value, the higher the quality in which the computation satisfies the specification. The quality operators in $\mathcal{F}$ can prioritise different scenarios or reduce the satisfaction value of computations in which delays occur. 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. For example, consider a "carrier" drone c that tries to bring an artefact to a rescue point, while keeping it as far as possible from the "villain" adversarial drone v. They evolve in a three dimensional cube of side length 1 unit, 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 atomic proposition "safe" denote that the artefact has reached the rescue point. In this scenario, the quality of an execution, or path, can be formalised with the following LTL[$\mathcal{F}$] formula:

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

Indeed, the satisfaction value of ψ_{rescue} is 0 on every path in which the artefact 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.

2 Contribution

We introduce and study SL[$\mathcal{F}$], a logic that extends both LTL[$\mathcal{F}$] and SL: On the one hand, it lifts LTL[$\mathcal{F}$] to the strategic setting by introducing strategy quantifier $\langle\!\langle x \rangle\!\rangle$, which maximises the satisfaction value of the formula in computations that are outcomes of interactions that respect strategies in x, and binding operator (a, x), which assigns strategy x to agent a. On the other hand, it 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. Thus, a model-checking procedure for SL[$\mathcal{F}$], which is our main contribution, enables formal reasoning about both quality and fuzziness of strategic behaviours. In addition, our model-checking procedure can be used as a synthesis algorithm to produce witness strategies that maximise the value of the formula.

Specifying strategies' quality As a motivating example, consider security drones that may patrol different height levels. Using SL[$\mathcal{F}$], we can specify the quality of strategies for the drones, whose objectives are to fly above and below all uncontrollable drones and perform certain actions when uncontrollable drones exhibit some disallowed behaviour. Indeed, the multi-valued atomic propositions are used to specify the different heights, temporal operators are used for specifying on-going behaviours, and the functions in $\mathcal{F}$ may be used to refer

[1] LSV, CNRS & ENS Paris-Saclay, Université Paris-Saclay
[2] Hebrew University
[3] Irisa, CNRS & Inria & Université de Rennes
[4] Università degli Studi di Napoli "Federico II"
[5] University of Göteborg

to these behaviours in a quantitative manner, for example to compare heights and to specify the satisfaction level that the designer gives to different possible scenarios.

In the more concrete 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 minimum over all strategies y for the villain of the quality of the behaviour resulting from x against y. If the quality of a behaviour is specified by formula ψ_{rescue} as above, then the quality of a strategy x is 0 if the villain has a counterstrategy y to steal it (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 counterstrategies y of the villain. We are interested in maximising the quality of carrier's strategies, which is formalised with the following formula:

$$\varphi_{\text{rescue}} = \langle\!\langle x \rangle\!\rangle [\![y]\!] (c, x)(v, y)(\text{dist}(p_{\mathbin{\rotatebox{90}{$\curlyvee$}}}, q_{\mathbin{\rotatebox{90}{$\curlyvee$}}})\ \mathbf{U}\ \text{safe})$$

where $[\![y]\!]$ is the dual of $\langle\!\langle y \rangle\!\rangle$. Thus, for a given strategy x, formula $[\![y]\!](c, x)(v, y)(\text{dist}(p, q)\ \mathbf{U}\ \text{safe})$ minimises the quality over all possible strategies y of the villain. It is therefore the minimal quality that strategy x is guaranteed to enforce. The formula φ_{rescue} then maximises this value over all possible strategies x of the carrier.

Note that in this example, the formula φ_{rescue} does not just specify the ability of the carrier to behave in some desired manner. Rather, it associates a satisfaction value in $[0, 1]$ with each strategy x. This suggests that SL$[\mathcal{F}]$ can be used not only to specify strategic behaviours with quantitative objectives, but also for quantizing notions from game theory that are traditionally Boolean. For example, beyond specifying that the agents are in a Nash Equilibrium, we can specify how far they are from an equilibrium, namely how much an agent may gain by a deviation that witnesses the instability. As a result we can express concepts such as ϵ-Nash Equilibria [11].

Synthesising 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, and which consists in reducing the problem to the model checking of (some appropriate extension of) Quantified CTL* [9]. This is, however, the first time this approach is used in a quantitative setting.

To do so we first need to define a fitting quantitative extension of Quantified CTL* (the extension of CTL* with second-order quantifiers on atomic propositions [9]). We do so by extending CTL$^\star[\mathcal{F}]$ [1] with quantifiers over atomic propositions where, similarly to strategy quantifiers in SL$[\mathcal{F}]$, existential/universal quantifications on atomic propositions are seen as maximisations/minimisations over possible valuations of the quantified propositions on the models, which are weighted infinite trees. However, while atomic propositions may have values in $[0, 1]$ in the models, we restrict propositional quantifiers to quantify only over Boolean valuations. The resulting logic is called Booleanly-Quantified CTL$^\star[\mathcal{F}]$, or BQCTL$^\star[\mathcal{F}]$. We show that the restriction to Booleanly quantified valuations is sufficient to simulate quantification on strategies. Moreover, without it, this restriction, the model-checking problem of QCTL$^\star[\mathcal{F}]$ is undecidable.

A general technique to model check CTL* 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 [6]. For CTL$^\star[\mathcal{F}]$, it was possible to extend this technique, using weighted fresh atomic propositions [1]. However this approach does not work for BQCTL$^\star[\mathcal{F}]$: indeed, the externally quantified atomic propositions may appear in different subformulas, and we cannot evaluate them one by one without fixing the same assignment for the quantified atomic propositions. Instead, we build upon both the automata theoretic approach to CTL* model checking [8] and the word automata construction developed for LTL$[\mathcal{F}]$ [1], 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 weighted trees t such that the satisfaction value of φ on t is in P. The translation, and hence the complexity of the model-checking problem, is non-elementary: we show that it is $(k + 1)$-EXPTIME-complete for formulas involving at most k nested quantifications on atomic propositions, and we show a similar complexity result for SL$[\mathcal{F}]$, in terms of nesting of strategy quantifiers. Similarly to LTL$[\mathcal{F}]$ [1], 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 [13], whenever the answer to the model-checking problem is positive, we can synthesise witness strategies. More precisely, if a formula $\varphi = \langle\!\langle x_1 \rangle\!\rangle \dots \langle\!\langle x_n \rangle\!\rangle \varphi'$ 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 synthesise strategies $x_1, \dots, x_n$ that maximise the quality as specified by φ'.

REFERENCES

[1] Shaull Almagor, Udi Boker, and Orna Kupferman, 'Formally reasoning about quality', *JACM*, **63**(3), 24:1–24:56, (2016).

[2] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman, 'Alternating-time temporal logic', *JACM*, **49**(5), 672–713, (September 2002).

[3] Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman, 'Strategy logic', *I& C*, **208**(6), 677–693, (June 2010).

[4] Edmund M. Clarke and E. Allen Emerson, 'Design and synthesis of synchronization skeletons using branching-time temporal logic', in *LOP'81*, ed., Dexter C. Kozen, volume 131 of *Lecture Notes in Computer Science*, pp. 52–71. Springer-Verlag, (1982).

[5] E. Allen Emerson and Joseph Y. Halpern, '"Sometimes" and "not never" revisited: On branching versus linear time temporal logic', *JACM*, **33**(1), 151–178, (January 1986).

[6] E. Allen Emerson and Chin-Laung Lei, 'Modalities for model checking: Branching time logic strikes back', *Science of Computer Programming*, **8**, 275–306, (1987).

[7] David Harel and Amir Pnueli, 'On the development of reactive systems', in *LMCS*, volume F-13, 477–498, Springer, (1985).

[8] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper, 'An automata-theoretic approach to branching-time model checking', *Journal of the ACM*, **47**(2), 312–360, (2000).

[9] François Laroussinie and Nicolas Markey, 'Quantified CTL: Expressiveness and complexity', *LMCS*, **10**(4), (2014).

[10] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi, 'Reasoning about strategies: On the model-checking problem', *ToCL*, **15**(4), 34:1–34:47, (August 2014).

[11] Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, *Algorithmic Game Theory*, Cambridge University Press, 2007.

[12] Amir Pnueli, 'The temporal semantics of concurrent programs', *Theoretical Computer Science*, **13**, 45–60, (1981).

[13] M.Y. Vardi and P. Wolper, 'Automata-theoretic techniques for modal logics of programs', *Journal of Computer and Systems Science*, **32**(2), 182–221, (1986).