On Computational Tractability for Rational Verification

Julian Gutierrez
Monash University
Melbourne, Australia
Julian.Gutierrez@monash.edu

Muhammad Najib
TU Kaiserslautern
Kaiserslautern, Germany
najib@informatik.uni-kl.de

Giuseppe Perelli
University of Rome, La Sapienza
Rome, Italy
perelli@diag.uniroma1.it

Michael Wooldridge
University of Oxford
Oxford, UK
mjw@cs.ox.ac.uk

ACM Reference Format:
Julian Gutierrez, Muhammad Najib, Giuseppe Perelli, and Michael Wooldridge. 2020. On Computational Tractability for Rational Verification. In *Proceedings of the 10th Workshop on Logical Aspects of Multi-Agent Systems (LAMAS 2020), Bastien Maubert, Aniello Murano, Sasha Rubin (eds.), Auckland, New Zealand, May 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

The formal verification of systems using temporal logics such as LTL and CTL [6] is a major research area, which has led to the development of an impressive number of industrial-strength verification tools and techniques. Arguably the most successful technique within formal verification is model checking, which can be done in polynomial space for LTL specifications and even in polynomial time for CTL specifications [5]. In the context of multiagent systems, *rational verification* forms a natural counterpart of model checking [9, 10, 18]. This is the problem of checking whether a given property ϕ, expressed as a temporal logic formula, is satisfied in a computation of a system that might be generated if agents within the system choose strategies for selecting actions that form a game-theoretic (*e.g.*, Nash) equilibrium – a decision problem that in case of Nash equilibria is denoted as ENASH [10]. Unlike model checking, rational verification is still in its infancy: the main ideas, formal models, and reasoning techniques underlying rational verification are under development, while current tool support is limited and cannot yet handle systems of industrial size [13, 16].

One key difficulty is that rational verification is computationally much harder than model checking, because checking equilibrium properties requires quantifying over the strategies available to players in the system. Rational verification is also different from model checking in the kinds of properties that each technique tries to check: while model checking is interested in correctness with respect to *any* possible behaviour of a system, rational verification is interested only in behaviours that can be *sustained by a Nash equilibrium*, when a multiagent system is modelled as a multi-player game. This, in particular, adds a new ingredient to the verification problem, as it is now necessary to take into account the *preferences* of players with respect to the possible runs of the system. Typically, in rational verification, such preferences are given by associating an

Proceedings of the 10th Workshop on Logical Aspects of Multi-Agent Systems (LAMAS 2020), Bastien Maubert, Aniello Murano, Sasha Rubin (eds.), May 2020, Auckland, New Zealand.

Goals	Spec.	ENASH
LTL	LTL	2EXPTIME-complete [10]
GR(1)	LTL	PSPACE-complete
GR(1)	GR(1)	FPT
mp	LTL	PSPACE-complete
mp	GR(1)	NP-complete

Table 1: Summary of main complexity results.

LTL goal γ_i with each player i in the game. In this case, rational verification with respect to a specification ϕ is 2EXPTIME-complete, regardless of whether the representation of the system is given succinctly [9, 10] or explicitly as a finite-state labelled transition graph [8]. In fact, the problem is 2EXPTIME-hard even if $\phi = \top$ since in such a simpler case any LTL synthesis problem can be encoded using only the LTL goals of the players in the game.

Here, we address this issue and provide complexity results that greatly improve on the 2EXPTIME upper bound of the general case. In particular, we consider games where the goals of players are represented as either *GR(1) formulae* (an important fragment of LTL that can express most response properties of a concurrent and reactive system [2]), or *mean-payoff utility functions* (one of the most studied reward and quality measures used in games for automated formal verification). In each case, we study the rational verification problem for system specifications ϕ given as GR(1) formulae and as LTL formulae, with respect to system models that are formally represented as concurrent game structures [1]. Our main results, summarised in Table 1, show that in the cases above mentioned, the 2EXPTIME upper bound can be dramatically improved to settings where rational verification can be solved in polynomial space, NP, or even in polynomial time if the number of players is fixed.

Related Work:

Rational verification has been studied for a number of settings, including iterated Boolean games, reactive modules games, and concurrent game structures [8–11]. In all cases, the problem is 2EXPTIME-complete, and even undecidable if imperfect information is allowed [15]. This work also relates to mean-payoff (mp) games at large, which are NP-complete for multi-player games [17] and in NP ∩ coNP for two-player games [19] – and in fact solvable in quasipolynomial time since they can be polynomially reduced to two-player turn-based perfect-information parity games [4].

2 GAMES WITH TEMPORAL LOGIC GOALS

In our setting, games are played on (deterministic) concurrent games structures, certain transition-state structures (*i.e.*, labelled graphs) where at every state of the structure, players – concurrently and independently – make choices which determine a unique successor state. A game is played by them repeating this process ad infinitum and, in such a way, defining a unique path in the graph. Because the states of such graphs are labelled with atomic propositions, the infinite path built in this way (a play, an execution run of the system) can be used as a model of a temporal logic formula. An outcome of a game is such a run, which is then used to check whether the temporal logic goal of every player is satisfied or not. Then, for each outcome of the game, the set of players is divided between "winners" and "losers", that is, those who get their goal achieved and those who do not, respectively. As expected, players will prefer execution runs that satisfy their goal over runs that do not, giving them an incentive to deviate (*i.e.*, do something different, use a different strategy) whenever their goal is not achieved if such a situation can be modified, that is, if they could become winners by unilaterally changing their current strategy. With this setting in mind, the concept of Nash equilibrium can be formally defined.

ENASH is the decision problem that asks whether some temporal logic formula ϕ, which represents a property that an external principal wants to see satisfied, is satisfied on an execution run of the system that can be sustained by some Nash equilibrium (a Nash equilibrium run) of the game. Then, while players only care about the satisfaction of their temporal logic goals, the external principal only cares about the satisfaction of ϕ, in particular, in any Nash equilibrium run of the concurrent game structure. Based on this informal description of the game, we now let players' goals γ be given by GR(1) formulae and consider two cases: games where ϕ is an LTL formula and games where ϕ is a GR(1) formula.

2.1 Solving ENASH

A general algorithm to solve ENASH, regardless of whether ϕ is a GR(1) formula or an LTL formula, is the following procedure:

(1) Guess a set W of "winners" in the game;
(2) For each "loser" j in the game, compute the set of states in the concurrent game structure from which j can be prevented from having a beneficial deviation – its punishment region;
(3) Remove states and transitions in the concurrent game structure according to the punishment regions of all losers;
(4) Check whether there exists an execution run π of the resulting (pruned) structure such that $\pi \models \phi \wedge \bigwedge_{i \in W} \gamma_i$ holds.

The bottlenecks of the above procedure are steps 1 and 4. If ϕ is an LTL formula, step 1 is not an issue since we can do the rest of the algorithm using only polynomial space (note that step 4 is an existential LTL model checking problem) for every W, leading to an overall PSPACE complexity. On the other hand, if ϕ is a GR(1) formula, we can do something better. In step 4, we can, instead, transform the problem into an emptiness check of a deterministic Streett word automaton, which can be solved in time that is polynomial in the automaton's index, given by the goals in the game, and singly exponential in the number of players. Assuming that the set of players will be fixed, we can then conclude that the problem is FPT with respect to the number of players in the game.

3 GAMES WITH MEAN-PAYOFF GOALS

We now consider concurrent game structures where, in addition to the propositional variables labelling the states of the structure, one also has, for each player, integers labelling such states. Then, every execution run will also be associated with an infinite sequence of n-tuples of integer numbers $(w_1, \ldots, w_n)(w'_1, \ldots, w'_n) \ldots$, which will define, for each player, a mean-payoff value for such a player. Then, in this case, instead of having goals given by GR(1) formulae, each player i will simply want to maximise a mean-payoff value associated with the outcomes of the game, that is, each player i will want to maximise the mean-payoff value of the infinite sequence $w_i w'_i w''_i \ldots$. Naturally, a player will prefer higher mean-payoff values than lower ones. Such a preference relation will define a notion of Nash equilibrium and consequently will determine a collection of Nash equilibrium execution runs in the system.

3.1 Solving ENASH

As before, we will consider two cases: games where ϕ is a GR(1) formula, and games where ϕ is an LTL formula. In the latter case, we can use a decision procedure that is slightly similar to the one described in the previous section. Instead of guessing "winners" in the game, we guess punishment values for every player in the game. Using such punishment values we can prune the concurrent game structure so that the game will only contain outcomes without states from which a player can have a beneficial deviation, that is, enforce an execution run with a higher mean-payoff value. Once this is done, we can use LTL^{Lim}, an extension of LTL where statements about mean-payoff values over a given (weighted/quantitative) graph as the one we consider here can be made [3]. Model checking in our case can be done in PSPACE, which provides the desired upper bound. And, in case ϕ is a GR(1) specification, again, we can do something better. Following [12], we define a linear program that characterises the existence of Nash equilibrium runs in this setting, and that can be used to provide an NP upper bound for the problem.

4 OTHER VERIFICATION PROBLEMS

ENASH is, arguably, the most fundamental problem in the rational verification framework, but it is not the only one. The two other key problems are ANASH and NONEMPTINESS. The former is the dual problem of ENASH, which asks, given a game G and a specification ϕ, whether ϕ is satisfied on *all* Nash equilibria of G. The latter simply asks whether the game G has at least one Nash equilibrium. We conclude from our results, that while ANASH for GR(1) games is also PSPACE and FPT, respectively, in case of LTL and GR(1) specifications, for mp games the problem is, respectively, PSPACE and coNP, in each case. In addition, we also conclude that whereas NONEMPTINESS for GR(1) games is FPT, for mp games is NP-complete. These results contrast with those when players' goals are general LTL formulae, where all problems are 2EXPTIME-complete. These results also contrast with those presented in [7], where it is shown that with LTL goals all problems in the rational verification framework can be reduced to NONEMPTINESS, which cannot be the case here, unless the polynomial hierarchy collapses.

Acknowledgement. This extended abstract informally describes and summarises work already presented at IJCAI [14] (Aug., 2019).

REFERENCES

[1] Rajeev Alur, Thomas Henzinger, and Orna Kupferman. 2002. Alternating-Time Temporal Logic. *Journal of the ACM* 49, 5 (2002), 672–713.

[2] Roderick Bloem, Barbara Jobstmann, Nir Piterman, Amir Pnueli, and Yaniv Sa'ar. 2012. Synthesis of Reactive(1) designs. *J. Comput. System Sci.* 78, 3 (2012), 911–938.

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

[4] Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. 2017. Deciding Parity Games in Quasipolynomial Time. In *STOC*. ACM, 252–263.

[5] Edmund M. Clarke, Orna Grumberg, Daniel Kroening, Doron Peled, and Helmut Veith. 2018. *Model Checking (2nd edition)*. MIT Press.

[6] E. Allen Emerson. 1990. Temporal and Modal Logic. In *Handbook of Theoretical Computer Science Volume B: Formal Models and Semantics*. Elsevier, 996–1072.

[7] Tong Gao, Julian Gutierrez, and Michael Wooldridge. 2017. Iterated Boolean Games for Rational Verification. In *AAMAS*. ACM, 705–713.

[8] Julian Gutierrez, Paul Harrenstein, and Michael Wooldridge. 2015. Expresiveness and Complexity Results for Strategic Reasoning. In *CONCUR (LIPIcs)*, Vol. 42. Schloss Dagstuhl, 268–282.

[9] Julian Gutierrez, Paul Harrenstein, and Michael Wooldridge. 2015. Iterated Boolean Games. *Information and Computation* 242 (2015), 53–79.

[10] Julian Gutierrez, Paul Harrenstein, and Michael Wooldridge. 2017. From Model Checking to Equilibrium Checking: Reactive Modules for Rational Verification. *Artificial Intelligence* 248 (2017), 123–157.

[11] Julian Gutierrez, Paul Harrenstein, and Michael Wooldridge. 2017. Reasoning about Equilibria in Game-like Concurrent Systems. *Annals of Pure and Applied Logic* 168, 2 (2017), 373–403.

[12] Julian Gutierrez, Aniello Murano, Giuseppe Perelli, Sasha Rubin, and Michael Wooldridge. 2017. Nash Equilibria in Concurrent Games with Lexicographic Preferences. In *IJCAI*. 1067–1073. https://doi.org/10.24963/ijcai.2017/148

[13] Julian Gutierrez, Muhammad Najib, Giuseppe Perelli, and Michael Wooldridge. 2018. EVE: A Tool for Temporal Equilibrium Analysis. In *ATVA (LNCS)*, Vol. 11138. Springer, 551–557.

[14] Julian Gutierrez, Muhammad Najib, Giuseppe Perelli, and Michael J. Wooldridge. 2019. On Computational Tractability for Rational Verification. In *IJCAI*, Sarit Kraus (Ed.). ijcai.org, 329–335.

[15] Julian Gutierrez, Giuseppe Perelli, and Michael Wooldridge. 2018. Imperfect Information in Reactive Modules games. *Information and Computation* 261, Part (2018), 650–675.

[16] Alexis Toumi, Julian Gutierrez, and Michael Wooldridge. 2015. A Tool for the Automated Verification of Nash Equilibria in Concurrent Games. In *ICTAC (LNCS)*, Vol. 9399. Springer, 583–594.

[17] Michael Ummels and Dominik Wojtczak. 2011. The Complexity of Nash Equilibria in Limit-Average Games. In *CONCUR*. 482–496. https://doi.org/10.1007/978-3-642-23217-6_32

[18] Michael Wooldridge, Julian Gutierrez, Paul Harrenstein, Enrico Marchioni, Giuseppe Perelli, and Alexis Toumi. 2016. Rational Verification: From Model Checking to Equilibrium Checking. In *AAAI*. AAAI Press, 4184–4191.

[19] Uri Zwick and Mike Paterson. 1996. The Complexity of Mean Payoff Games on Graphs. *Theoretical Computer Science* 158, 1 (1996), 343 – 359. https://doi.org/10.1016/0304-3975(95)00188-3