Synthesis with Rational Environments

Orna Kupferman[1]*, Giuseppe Perelli[2]**, and Moshe Y. Vardi[3]***

The Hebrew University[1], University of Naples “Federico II”[2] Rice University[3]
orna@cs.huji.ac.il[1], giuseppe.perelli@unina.it[2], vardi@cs.rice.edu[3]

Abstract *Synthesis* is the automated construction of a system from its specification. The system has to satisfy its specification in all possible environments. The environment often consists of agents that have objectives of their own. Thus, it makes sense to soften the universal quantification on the behavior of the environment and take the objectives of its underlying agents into an account. Fisman et al. introduced *rational synthesis*: the problem of synthesis in the context of rational agents. The input to the problem consists of temporal-logic formulas specifying the objectives of the system and the agents that constitute the environment, and a solution concept (e.g., Nash equilibrium). The output is a profile of strategies, for the system and the agents, such that the objective of the system is satisfied in the computation that is the outcome of the strategies, and the profile is stable according to the solution concept; that is, the agents that constitute the environment have no incentive to deviate from the strategies suggested to them.
In this paper we continue to study rational synthesis. First, we suggest an alternative definition to rational synthesis, in which the agents are rational but not cooperative. In the non-cooperative setting, one cannot assume that the agents that constitute the environment take into account the strategies suggested to them. Accordingly, the output is a strategy for the system only, and the objective of the system has to be satisfied in all the compositions that are the outcome of a stable profile in which the system follows this strategy. We show that rational synthesis in this setting is 2EXPTIME-COMPLETE, thus it is not more complex than traditional synthesis or cooperative rational synthesis. Second, we study a richer specification formalism, where the objectives of the system and the agents are not Boolean but quantitative. In this setting, the goal of the system and the agents is to maximize their outcome. The quantitative setting significantly extends the scope of rational synthesis, making the game-theoretic approach much more relevant.

1 Introduction

Synthesis is the automated construction of a system from its specification. The basic idea is simple and appealing: instead of developing a system and verifying that it adheres to its

* The research leading to these results has received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 278410, by the Israel Science Foundation (grant 1229/10), and by the US-Israel Binational Science Foundation (grant 2010431).

** This research was partially done while visiting the Rice University.

*** NSF Expeditions in Computing project "ExCAPE: Expeditions in Computer Augmented Program Engineering

specification, we would like to have an automated procedure that, given a specification, constructs a system that is correct by construction. The first formulation of synthesis goes back to Church [8]; the modern approach to synthesis was initiated by Pnueli and Rosner, who introduced LTL (linear temporal logic) synthesis [20]. The LTL *synthesis problem* receives as input a specification in LTL and outputs a reactive system modeled by a finite-state transducer satisfying the given specification — if such exists. It is important to distinguish between input signals, assigned by the environment, and output signals, assigned by the system. A system should be able to cope with all values of the input signals, while setting the output signals to desired values [20]. Therefore, the quantification structure on input and output signals is different. Input signals are universally quantified while output signals are existentially quantified.

Modern systems often interact with other systems. For example, the clients interacting with a server are by themselves distinct entities (which we call agents) and are many times implemented by systems. In the traditional approach to synthesis, the way in which the environment is composed of its underlying agents is abstracted. In particular, the agents can be seen as if their only objective is to conspire to fail the system. Hence the term “hostile environment" that is traditionally used in the context of synthesis. In real life, however, many times agents have goals of their own, other than to fail the system. The approach taken in the field of algorithmic game theory [18] is to assume that agents interacting with a computational system are *rational*, i.e., agents act to achieve their own goals. Assuming agents rationality is a restriction on the agents behavior and is therefore equivalent to softening the universal quantification on the environment. [1] Thus, the following question arises: can system synthesizers capitalize on the rationality and goals of agents interacting with the system?

In [10], Fisman et al. positively answered this question by introducing and studying *rational synthesis*. The input to the rational-synthesis problem consists of LTL formulas specifying the objectives of the system and the agents that constitute the environment, and a solution concept, e.g., dominant strategies, Nash Equilibria, and the like. The atomic propositions over which the objectives are defined are partitioned among the system and the agents, so that each of them controls a subset of the propositions. The desired output is a strategy profile such that the objective of the system is satisfied in the computation that is the outcome of the profile, and the agents that constitute the environment have no incentive to deviate from the strategies suggested to them (formally, the profile is an equilibrium with respect to the solution concept). Fisman et al. showed that there are specifications that cannot be realized in a hostile environment but are realizable in a rational environment. Moreover, the rational-synthesis problem for LTL and common solution concepts used in game theory can be solved in 2EXPTIME, thus its complexity coincides with that of usual synthesis.

In this paper we continue the study of rational synthesis. We present the following three contributions. First, we suggest an alternative definition to rational synthesis, in which the agents are rational but not cooperative. Second, we study a richer specification formalism, where the objectives of the system and the agents are not Boolean but

[1] Early work on synthesis has realized that the universal quantification on the behaviors of the environment is often too restrictive. The way to address this point, however, has been by adding assumptions on the environment, which can be part of the specification (cf., [5]).

quantitative. Third, we show that all these variants of the rational synthesis problems can be reduced to model checking in fragments of *Strategy Logic* [6]. Before we describe our contributions in more detail, let us highlight a different way to consider rational synthesis and our contribution here. *Mechanism design* is a field in game theory and economics studying the design of games whose outcome (assuming agents rationality) achieves some goal [17, 18]. The outcome of traditional games depends on the final position of the game. In contrast, the systems we reason about maintain an *on-going interaction* with their environment, and we reason about their behavior by referring 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. Rational synthesis can be viewed as a variant of mechanism design in which the game is induced by the objective of the system, and the objectives of both the system and the agents refer to their on-going interaction and are specified by temporal-logic formulas. Our contributions here correspond to the classic setting assumed in mechanism design: the agents need not be cooperative, and the outcome is not Boolean.

We argue that the definition of rational synthesis in [10] is *cooperative*, in the sense that the agents that constitute the environment are assumed to follow the strategy profile suggested to them (as long as it is in an equilibrium). Here, we consider also a *non-cooperative* setting, in which the agents that constitute the environment may follow any strategy profile that is in an equilibrium, and not necessarily the one suggested to them by the synthesis algorithm. In many scenarios, the cooperative setting is indeed too optimistic, as the system cannot assume that the environment, even if it is rational, would follow a suggested strategy, rather than a strategy that is as good for it. Moreover, sometimes there is no way to communicate with the environment and suggest a strategy for it. From a technical point of view, we show that the non-cooperative setting requires reasoning about *all* possible equilibria, yet, despite this more sophisticated reasoning, it stays 2EXPTIME-COMPLETE. We achieve the upper bound by reducing rational synthesis to the model-checking problem for Strategy Logic (SL, for short). SL is a specification formalism that allows to explicitly quantify over strategies in games as first-order objects [6]. While the model-checking problem for strategy logic is in general non-elementary, we show that it is possible to express rational synthesis in the restricted *Nested-Goal* fragment of SL, introduced in [15], which leads to the desired complexity. It is important to observe the following difference between the cooperative and the non-cooperative settings. In the cooperative one, we synthesize strategies for all agents, with the assumption that the agent that corresponds to the system always follows his suggested strategy and the agents that constitute the environment decide in a rational manner whether to follow their strategies. On the other hand, in the non-cooperative setting, we synthesize a strategy only for the agent that corresponds to the system, and we assume that the agents that constitute the environment are rational, thus the suggested strategy has to win against all rational behaviors of the environment.

Our second contribution addresses a weakness of the classical synthesis problem, a weakness that is more apparent in the rational setting. In classical synthesis, the specification is Boolean and describes the expected behavior of the system. In many applications, systems can satisfy their specifications at different levels of quality. Thus, synthesis with respect to Boolean specifications does not address designers needs. This latter problem is a real obstacle, as 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. In the last years we see a lot of effort on developing formalisms that would enable the specification of such quality measures [4, 1].

Classical applications of game theory consider games with quantitative payoffs. In the Boolean setting, we assumed that the payoff of an agent is, say, 1, if its objective is satisfied, and is 0 otherwise. In particular, this means that agents whose objectives are not satisfied have no incentive to follow any strategy, even if the profile satisfies the solution concept. In real-life, rational objectives are rarely Boolean. Thus, even beyond our goal of synthesizing systems of high quality, the extension of the synthesis problem to the rational setting calls also for an extension to a quantitative setting. Unfortunately, the full quantitative setting is undecidable already in the context of model checking [11]. In [10], Fisman et al. extended cooperative rational synthesis to objectives in the multi-valued logic LLTL, where specifications take truth values from a finite lattice.

We introduce here a new quantitative specification formalism, termed *Objective* LTL, (OLTL, for short). We first define the logic, and then study its rational synthesis. Essentially, an OLTL specification is a pair $\theta = \langle \Psi, \mathsf{f} \rangle$, where $\Psi = \langle \psi_1, \psi_2, \ldots, \psi_m \rangle$ is a tuple of LTL formulas and $\mathsf{f} : \{0,1\}^m \to \mathbb{Z}$ is a *reward function*, mapping Boolean vectors of length m to an integer. A computation η then maps θ to a reward in the expected way, according to the subset of formulas that are satisfied in η. In the rational synthesis problem for OLTL, the input consists of OLTL specifications for the system and the other agents, and the goal of the system is to maximize its reward with respect to environments that are in an equilibrium. Again, we distinguish between a cooperative and a non-cooperative setting. Note that the notion of an equilibrium in the quantitative setting is much more interesting, as it means that all agents in the environment cannot expect to increase their payoffs. We show that the quantitative setting is not more complex than the non-quantitative one, thus quantitative rational synthesis is complete for 2EXPTIME in both the cooperative and non-cooperative settings.

2 Preliminaries

2.1 Games

A *concurrent game structure* (CGS, for short) [3] is a tuple $\mathcal{G} \triangleq \langle \Phi, \Omega, (A_i)_{i \in \Omega}, S, \lambda, \tau, s_0 \rangle$, where Φ and $\Omega = \{\alpha_0, \ldots, \alpha_k\}$ are finite sets of *atomic propositions* and *agents*, A_i are disjoint sets of *actions*, one for each agent α_i, S is a set of *states*, $s_0 \in S$ is a designated *initial state*, and $\lambda : S \to 2^{\Phi}$ is a *labeling function* that maps each state to the set of atomic propositions true in that state. By $A \triangleq \bigcup_{i \in \Omega} A_i$ we denote the union of all possible action for all the agents. Let $D \triangleq A_0 \times \ldots \times A_k$ be the set of *decisions*, i.e., $(k+1)$-tuples of actions representing the choices of an action for each agent. Then, $\tau : S \times D \to S$ is a deterministic *transition function* mapping a pair of a state and a decision to a state.

A *path* in a CGS $\mathcal{G}$ is an infinite sequence of states $\eta = \eta_0 \cdot \eta_1 \cdot \ldots \in S^{\omega}$ that agrees with the transition function, i.e., such that, for all $i \in \mathbb{N}$, there exists a decision $\mathsf{d} \in D$ such that $\eta_{i+1} = \tau(\eta_i, \mathsf{d})$. A *track* in a CGS $\mathcal{G}$ is a prefix ρ of a path η, also denoted by $\eta_{\leq n}$, for a suitable $n \in \mathbb{N}$. A track ρ is *non-trivial* if $|\rho| > 0$, i.e., $\rho \neq \varepsilon$.

We use $\mathrm{Pth} \subseteq S^{\omega}$ and $\mathrm{Trk} \subseteq S^{+}$ to denote the set of all paths and non-trivial tracks, respectively. Also, for a given $s \in S$, we use Pth^{s} and Trk^{s} to denote the subsets of paths and tracks starting from $s \in S$. Intuitively, the game starts in the state s_0 and, at each step, each agent selects an action in its set. The game then deterministically proceeds to the next state according to the corresponding decision. Thus, the outcome of a CGS is a path, regulated by individual actions of the agents.

A *strategy* for Agent α_i is a tool used to decide which decision to take at each phase of the game. Formally, it is a function $\pi_i : \mathrm{Trk} \to A_i$ that maps each non-trivial track to a possible action of Agent α_i. By Π_i we denote the set of all possible strategies for agent α_i. A *strategy profile* is a $(k+1)$-tuple $\mathsf{P} = \langle \pi_0, \dots, \pi_k \rangle \in \Pi_0 \times \ldots \times \Pi_k$ that assigns a strategy to each agent. We denote by $\mathcal{P} \triangleq \Pi_0 \times \ldots \times \Pi_k$ the set of all possible strategy profiles. For a strategy profile P and a state s, we use $\eta = \mathsf{play}(\mathsf{P}, s)$ to denote the path that is the outcome of a game that starts in s and agrees with P, i.e., for all $i \in \mathbb{N}$, it holds that $\eta_{i+1} = \tau(\eta_i, \mathsf{d}_i)$, where $\mathsf{d}_i = (\pi_0(\eta_{\leq i}), \dots, \pi_k(\eta_{\leq i}))$. By $\mathsf{play}(\mathsf{P}) = \mathsf{play}(\mathsf{P}, s_0)$ we denote the unique path starting from s_0 obtained from P.

We model reactive systems by deterministic transducers. A *transducer* is a tuple $\mathcal{T} = \langle \mathrm{I}, \mathrm{O}, \mathrm{S}, s_0, \delta, \mathsf{L} \rangle$, where I is a set of input signals assigned by the environment, O is a set of output signals, assigned by the system, S is a set of states, s_0 is an initial state, $\delta : \mathrm{S} \times 2^{\mathrm{I}} \to \mathrm{S}$ is a transition function, and $\mathsf{L} : \mathrm{S} \to 2^{\mathrm{O}}$ is a labeling function. When the system is in state $s \in \mathrm{S}$ and it reads an input assignment $\sigma \in 2^{\mathrm{I}}$, it changes its state to $s' = \delta(s, \sigma)$ where it outputs the assignment $\mathsf{L}(s')$. Given a sequence $\varrho = \sigma_1, \sigma_2, \sigma_3, \ldots \in (2^{\mathrm{I}})^{\omega}$ of inputs, the *execution* of $\mathcal{T}$ on ϱ is the sequence of states $s_0, s_1, s_2, \ldots$ such that for all $j \geq 0$, we have $s_{j+1} = \delta(s_j, \sigma_j)$. The *computation* $\eta \in (2^{\mathrm{I}} \times 2^{\mathrm{O}})^{\omega}$ of S on ϱ is then $\langle \mathsf{L}(s_0), \sigma_1 \rangle, \langle \mathsf{L}(s_1), \sigma_2 \rangle, \langle \mathsf{L}(s_2), \sigma_3 \rangle, \ldots$.

2.2 Strategy Logic

Strategy Logic [6, 16, 14] (SL, for short) is a logic that allows to quantify over strategies in games as explicit first-order objects. Intuitively, such quantification, together with a syntactic operator called *binding*, allows us to restrict attention to restricted classes of strategy profiles, determining a subset of paths, in which a temporal specification is desired to be satisfied. Since nesting of quantifications and bindings is possible, such temporal specifications can be recursively formulated by an SL subsentence. From a syntactic point of view, SL is an extension of LTL with strategy variables $\mathrm{Var}_0, \dots, \mathrm{Var}_k$ for the agents, existential ($\langle\langle x_i \rangle\rangle$) and universal ($[[x_i]]$) strategy quantifiers, and a binding operator of the form (α_i, x_i) that couples an agent α_i with one of its variables $x_i \in \mathrm{Var}_i$.

We first introduce some technical notation. For a tuple $t = (t_0, \dots, t_k)$, by $t[i \leftarrow d]$ we denote the tuple obtained from t by replacing the i-th component with d. We use $\vec{x}$ as an abbreviation for the tuple $(x_0, \dots, x_k) \in \mathrm{Var}_0 \times \ldots \times \mathrm{Var}_k$. By $\langle\langle \vec{x} \rangle\rangle = \langle\langle x_0 \rangle\rangle \ldots \langle\langle x_k \rangle\rangle$, $[[\vec{x}]] = [[x_0]] \ldots [[x_k]]$, and $\flat(\vec{x}) = (\alpha_0, x_0) \ldots (\alpha_k, x_k)$ we denote the existential and universal quantification, and the binding of all the agents to the strategy profile variable $\vec{x}$, respectively. Finally, by $\flat(\vec{x}_{-i}, y_i) = (\alpha_0, x_0) \ldots (\alpha_i, y_i) \ldots (\alpha_k, x_k)$ we denote the changing of binding for Agent α_i from the strategy variable x_i to the strategy variable y_i in the global binding $\flat(\vec{x})$.

Here we define and use a slight variant of the *Nested-Goal* fragment of SL, namely SL[NG], introduced in [15]. Formulas in SL[NG] are defined with respect to a set Φ of atomic proposition, a set Ω of agents, and sets Var_i of strategy variables for Agent $\alpha_i \in \Omega$. The set of SL[NG] formulas is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \varphi\,\mathsf{R}\,\varphi \mid \langle\!\langle x_i \rangle\!\rangle\varphi \mid [\![x_i]\!]\varphi \mid \flat(\vec{x})\varphi,$$

where, $p \in \Phi$ is an atomic proposition, $x_i \in \mathrm{Var}_i$ is a variable, and $\vec{x} \in \mathrm{Var}_0 \times \ldots \times \mathrm{Var}_k$ is a tuple of variables, one for each agent.

The LTL part has the classical meaning. The formula $\langle\!\langle x_i \rangle\!\rangle\varphi$ states that there exists a strategy for Agent α_i such that the formula φ holds. The formula $[\![x_i]\!]\varphi$ states that, for all possible strategies for Agent α_i, the formula φ holds. Finally, the formula $\flat(\vec{x})\varphi$ states that the formula φ holds under the assumption that the agents in Ω adhere to the strategy evaluation of the variable x_i coupled in $\flat(\vec{x})$.

As an example, $\langle\!\langle x_0 \rangle\!\rangle[\![x_1]\!]\flat(\vec{x})(p\mathsf{U}\,q) \vee [\![y_0]\!]\langle\!\langle y_1 \rangle\!\rangle\flat(\vec{y})(\mathsf{G}\,\mathsf{F}\,p \wedge \mathsf{G}\,\neg q)$ is an SL[NG] formula stating that either the system α_0 has a strategy x_0 to enforce $p\mathsf{U}\,q$ or, for all possible behaviors y_0, the environment has a strategy y_1 to enforce both $\mathsf{G}\,\mathsf{F}\,p$ and $\mathsf{G}\,\neg q$.

We denote by $\mathsf{free}(\varphi)$ the set of strategy variables occurring in φ but not in a scope of a quantifier. A formula φ is *closed* if $\mathsf{free}(\varphi) = \emptyset$.

Similarly to the case of first order logic, an important concept that characterizes the syntax of SL is the one of *alternation depth* of quantifiers, i.e., the maximum number of quantifier switches $\langle\!\langle x_i \rangle\!\rangle[\![x_j]\!]$, $[\![x_i]\!]\langle\!\langle x_j \rangle\!\rangle$, in the formula. A precise formalization of the concepts of alternation depth can be found in [16, 14].

Now, in order to define the semantics of SL, we use the auxiliary concept of assignment. Let $\mathrm{Var} = \bigcup_{i=0}^{k} \mathrm{Var}_i$ be a set of variables for the agents in Ω, an *assignment* is a function $\chi : \mathrm{Var} \cup \Omega \to \Pi$ mapping variables and agents to a relevant strategy, i.e., for all $\alpha_i \in \Omega$ and $x_i \in \mathrm{Var}_i$, we have that $\chi(\alpha_i), \chi(x_i) \in \Pi_i$. Let $\mathrm{Asg} \triangleq \Pi^{\mathrm{Var} \cup \Omega}$ denote the set of all assignments. For an assignment χ and elements $l \in \mathrm{Var} \cup \Omega$, we use $\chi[l \mapsto \pi] \in \mathrm{Asg}$ to denote the new assignment that returns π on l and the value of χ on the other ones, i.e., $\chi[l \mapsto \pi](l) \triangleq \pi$ and $\chi[l \mapsto \pi](l') \triangleq \chi(l')$, for all $l' \in (\mathrm{Var} \cup \Omega) \setminus \{l\}$. By $\mathsf{play}(\chi, s)$ we denote the path $\mathsf{play}(\mathsf{P}, s)$, for the strategy profile P that is compatible with χ.

We now describe when a given game $\mathcal{G}$ and a given assignment χ satisfy an SL formula φ, where $\mathsf{dom}(\chi)$ [2] $= \mathsf{free}(\varphi) \cup \Omega$. We use $\mathcal{G}, \chi, s \models \varphi$ to indicate that the path $\mathsf{play}(\chi, s)$ satisfies φ over the CGS $\mathcal{G}$. For φ in LTL, the semantics is as usual [13]. For the other operators, the semantics is as follows.

1. $\mathcal{G}, \chi, s \models \langle\!\langle x_i \rangle\!\rangle\varphi$ if there exists a strategy π_i for α_i such that $\mathcal{G}, \chi[x_i \mapsto \pi_i], s \models \varphi$;
2. $\mathcal{G}, \chi, s \models [\![x_i]\!]\varphi$ if, for all strategies π_i for α_i, it holds that $\mathcal{G}, \chi[x_i \mapsto \pi_i], s \models \varphi$;
3. $\mathcal{G}, \chi, s \models \flat(\vec{x})\varphi$ if it holds that $\mathcal{G}, \chi[\alpha_0 \mapsto x_0] \ldots [\alpha_k \mapsto x_k], s \models \varphi$.

Finally, we say that $\mathcal{G}$ satisfies φ, and write $\mathcal{G} \models \varphi$, if there exists an assignment χ such that $\mathcal{G}, \chi, s_0 \models \varphi$.

Intuitively, at Items 1 and 2, we evaluate the existential and universal quantifiers over a variable x_i by associating with it a suitable strategy. At Item 3 we commit the agents to use the strategy contained in the tuple variable $\vec{x}$.

Theorem 1. [15] *The model-checking problem for* SL[NG] *is* $(d+1)$EXPTIME *with* d *being the alternation depth of the specification.*

[2] By $\mathsf{dom}(\mathsf{f})$ we denote the domain of the function f.

2.3 Rational Synthesis

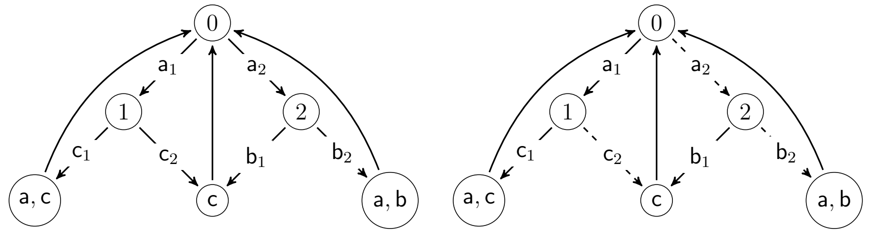
In this subsection we define two variants of rational synthesis. The first, *cooperative rational synthesis*, was introduced in [10]. The second, *non-cooperative rational synthesis*, is new.

We work with the following model: the world consists of a *system* and an environment composed of k agents: $\alpha_1, \ldots, \alpha_k$. For uniformity, we refer to the system as Agent α_0. We assume that Agent α_i controls a set X_i of propositions, and the different sets are pairwise disjoint. At each point in time, each agent sets his propositions to certain values. Let $\mathrm{X} = \bigcup_{0 \leq i \leq k} \mathrm{X}_i$, and $\mathrm{X}_{-i} = \mathrm{X} \setminus \mathrm{X}_i$. Each agent α_i (including the system) has an objective φ_i, specified as an LTL formula over X.

This setting induces the CGS $\mathcal{G} = \langle \Phi, \Omega, (A_i)_{i \in \Omega}, S, \lambda, \tau, s_0 \rangle$ defined as follows. The set of agents $\Omega = \{\alpha_0, \alpha_1, \ldots, \alpha_k\}$ consists of the system and the agents that constitute the environment. The actions of Agent α_i are the possible assignments to its variables. Thus, $A_i = 2^{\mathrm{X}_i}$. We use A and A_{-i} to denote the sets 2^{X} and $2^{\mathrm{X}_{-i}}$, respectively. The nodes of the game record the current assignment to the variables. Hence, $S = A$, and for all $s \in S$ and $\langle \sigma_0, \ldots, \sigma_k \rangle \in A_0 \times A_1 \times \cdots \times A_k$, we have $\delta(s, \sigma_0, \ldots, \sigma_k) = \langle \sigma_0, \cdots, \sigma_k \rangle$.

A strategy for the system is a function $\pi_0 : \mathrm{Trk} \to A_0$. In the standard synthesis problem, we say that π_0 realizes φ_0 if, no matter which strategies the agents composing the environment follow, all the paths in which the system follows π_0 satisfy φ_0. In rational synthesis, on the other hand, we assume that the agents that constitute the environment are rational, which soften the universal quantification on the behavior of the environment.

Recall that the rational-synthesis problem gets a solution concept as a parameter. As discussed in Section 1, the fact that a strategy profile is a solution with respect to the concept guarantees that it is not worthwhile for the agents constituting the environment to deviate from the strategies assigned to them. Several solution concepts are studied and motivated in game theory. Here, we focus on the concepts of *dominant strategy* and *Nash equilibrium*, defined below.



(a) The game (b) Nash equilibrium

Figure 1: A game.

The common setting in game theory is that the objective for each agent is to maximize his *payoff* – a real number that is a function of the outcome of the game. We use $\mathsf{payoff}_i : \mathrm{Pth} \to \mathbb{R}$ to denote the payoff function of Agent α_i. That is, payoff_i assigns to each possible path η a real number $\mathsf{payoff}_i(\eta)$ expressing the payoff of α_i on η. For a strategy profile P, we use $\mathsf{payoff}_i(\mathsf{P})$ to abbreviate $\mathsf{payoff}_i(\mathsf{play}(\mathsf{P}, s_0))$. In the case of an LTL goal ψ_i, we define $\mathsf{payoff}_i(\eta) = 1$ if $\eta \models \psi_i$ and $\mathsf{payoff}_i(\eta) = 0$, otherwise.

The simplest and most appealing solution concept is dominant-strategies solution [19]. A *dominant strategy* is a strategy that an agent can never lose by adhering to, regardless of the strategies of the other agents. Therefore, if there is a profile of strategies $\mathsf{P} = \langle \pi_0, \ldots, \pi_k \rangle$ in which all strategies π_i are dominant, then no agent has an incentive to deviate from the strategy assigned to him in P. Formally, P is a *dominant strategy profile* if for every $1 \leq i \leq k$ and for every (other) profile P', we have that $\mathsf{payoff}_i(\mathsf{P}') \leq \mathsf{payoff}_i(\mathsf{P}'[i \leftarrow \pi_i])$.

As an example, consider the game in Figure 1(a), played by three agents, Alice, Bob, and Charlie, whose actions are $\{\mathsf{a}_1, \mathsf{a}_2\}$, $\{\mathsf{b}_1, \mathsf{b}_2\}$, and $\{\mathsf{c}_1, \mathsf{c}_2\}$, respectively. The arrows are labeled with the possible action of the agents. Each agent wants to visit a state marked with his initial letter, infinitely often. In this game, the strategy for Bob of always choosing b_2 on his node 2 is dominant, while all the possible strategies for Charlie are dominant. On the other hand, Alice has no dominant strategies, since her goal essentially depends on the strategies adopted by the other agents. In several games, it can happen that agents have not any dominant strategy. For this reason, one would consider also other kind of solution concepts.

Another well known solution concept is Nash equilibrium [19]. A strategy profile is a *Nash equilibrium* if no agent has an incentive to deviate from his strategy in P provided that the other agents adhere to the strategies assigned to them in P. Formally, P is a *Nash equilibrium profile* if for every $1 \leq i \leq k$ and for every (other) strategy π'_i for agent α_i, we have that $\mathsf{payoff}_i(\mathsf{P}[i \leftarrow \pi'_i]) \leq \mathsf{payoff}_i(\mathsf{P})$. An important advantage of Nash equilibrium is that it is more likely to exist than an equilibrium of dominant strategies [19] [3]. A weakness of Nash equilibrium is that it is not nearly as stable as a dominant-strategy solution: if one of the other agents deviates from his assigned strategy, nothing is guaranteed.

For the case of repeated-turn games like infinite games, a suitable refinement of Nash Equilibria is the *Subgame perfect Nash-equilibrium* [21] (SPE, for short). A strategy profile $\mathsf{P} = \langle \pi_0, \ldots, \pi_k \rangle$ is an SPE if for every possible history of the game, no agent α_i has an incentive to deviate from her strategy π_i, assuming that the other agents follow their strategies in P. Intuitively, an SPE requires the existence of a Nash Equilibrium for each subgame starting from a randomly generated finite path of the original one. In [10], the authors have studied cooperative rational synthesis also for the solution concept of SPE. To do this, the synthesis algorithm in [10] was extended to consider all possible histories of the game. In SL such a path property can be expressed combining strategy quantifiers with temporal operators. Indeed, the formula $\varphi = [\![\vec{x}]\!]\flat(\vec{x})\mathsf{G}\ \psi(\vec{y})$, with $\mathsf{free}(\varphi) = \vec{y}$, states that, for all possible profile strategies the agents can follow, the game always reaches a position in which the formula $\psi(\vec{y})$ holds. Thus, for all possible paths that can be generated by agents, the property holds. By replacing $\psi(\vec{y})$ with the above formula, we then obtain a formula that represents SPEs. Hence, the cooperative and non-cooperative synthesis problems can be asserted in SL also for SPE, and our results hold also for this solution concept.

[3] In particular, all k-agent turn-based games with ω-regular objectives have Nash equilibrium [7].

In rational synthesis, we control the strategy of the system and assume that the agents that constitute the environment are rational. Consider a strategy profile $\mathsf{P} = \langle \pi_0, \ldots, \pi_k \rangle$ and a solution concept γ (that is, γ is either "dominant strategies" or "Nash equilibrium"). We say that P is *correct* if $\mathsf{play}(\mathsf{P})$ satisfies φ_0. We say that P is in a π_0*-fixed* γ*-equilibrium* if the agents composing the environment have no incentive to deviate from their strategies according to the solution concept γ, assuming that the system continues to follow π_0. Thus, P is a π_0-fixed dominant-strategy equilibrium if for every $1 \leq i \leq k$ and for every (other) profile P' in which Agent 0 follows π_0, we have that $\mathsf{payoff}_i(\mathsf{P}') \leq \mathsf{payoff}_i(\mathsf{P}'[i \leftarrow \pi_i])$. Note that for the case of Nash equilibrium, adding the requirement that P is π_0-fixed does not change the definition of an equilibrium.

In the context of objectives in LTL, we assume the following simple payoffs. If the objective φ_i of Agent α_i holds, then his payoff is 1, and if φ_i does not hold, then the payoff of Agent i is 0. Accordingly, $\mathsf{P} = \langle \pi_0, \ldots, \pi_k \rangle$ is in a dominant-strategy equilibrium if for every $1 \leq i \leq k$ and profile $\mathsf{P}' = \langle {\pi_0}', \ldots, {\pi_k}' \rangle$ with $\pi_0' = \pi_0$, if $\mathsf{play}(\mathsf{P}') \models \varphi_i$, then $\mathsf{play}(\mathsf{P}'[i \leftarrow \pi_i]) \models \varphi_i$. Also, P is in a Nash-equilibrium if for every $1 \leq i \leq k$ and strategy π_i', if $\mathsf{play}(\mathsf{P}[i \leftarrow \pi_i']) \models \varphi_i$, then $\mathsf{play}(\mathsf{P}) \models \varphi_i$.

Definition 1 (Rational synthesis).
The input to the rational-strategy problem is a set X *of atomic propositions, partitioned into* $\mathrm{X}_0, \ldots, \mathrm{X}_k$*,* LTL *formulas* $\varphi_0, \ldots, \varphi_k$*, describing the objectives of the system and the agents composing the environment, and a solution concept* γ*. We distinguish between two variants of the problem:*

1. *In* Cooperative rational synthesis [10]*, the desired output is a strategy profile* P *such that* $\mathsf{play}(\mathsf{P})$ *satisfies* φ_0 *and* P *is a* π_0*-fixed* γ*-equilibrium.*
2. *In* Non-cooperative rational synthesis*, the desired output is a strategy* π_0 *for the system such that for every strategy profile* P *that includes* π_0 *and is a* π_0*-fixed* γ*-equilibrium, we have that* $\mathsf{play}(\mathsf{P})$ *satisfies* φ_0.

Thus, in the cooperative variant of [10], we assume that once we suggest to the agents in the environment strategies that are in a γ-equilibrium, they will adhere to the suggested strategies. In the non-cooperative variant we introduce here, the agents may follow any strategy profile that is in a γ-equilibrium, and thus we require the outcome of all these profiles to satisfy φ_0. It is shown in [10] that the cooperative rational synthesis problem is 2EXPTIME-COMPLETE.

Note that the input to the rational synthesis problem may not have a solution, so when we solve the rational-synthesis problem, we first solve the *rational-realizability* problem, which asks if a solution exists. As with classical synthesis, the fact that SL model-checking algorithms can be easily modified to return a regular witness for the involved strategies in case an existentially quantified strategy exists, makes the realizability and synthesis problems strongly related.

Example 1. Consider a file-sharing network with the system and an environment consisting of two agents. The system controls the signal d_1 and d_2 (Agent α_1 and α_2 can download, respectively) and it makes sure that an agent can download only when the other agent uploads. The system's objective is that both agents will upload infinitely often. Agent α_1 controls the signal u_1 (Agent α_1 uploads), and similarly for Agent α_2 and u_2. The goal of both agents is to download infinitely often.

Formally, the set of atomic propositions is $\mathrm{X} = \{\mathsf{d}_1, \mathsf{d}_2, \mathsf{u}_1, \mathsf{u}_2\}$, partitioned into $\mathrm{X}_0 = \{\mathsf{d}_1, \mathsf{d}_2\}$, $\mathrm{X}_1 = \{\mathsf{u}_1\}$, and $\mathrm{X}_2 = \{\mathsf{u}_2\}$. The objectives of the system and the environment are as follows.

– $\varphi_0 = \mathsf{G}\,(\neg\mathsf{u}_1 \rightarrow \neg\mathsf{d}_2) \wedge \mathsf{G}\,(\neg\mathsf{u}_2 \rightarrow \neg\mathsf{d}_1) \wedge \mathsf{G}\,\mathsf{F}\,\mathsf{u}_1 \wedge \mathsf{G}\,\mathsf{F}\,\mathsf{u}_2$,
– $\varphi_1 = \mathsf{G}\,\mathsf{F}\,\mathsf{d}_1$,
– $\varphi_2 = \mathsf{G}\,\mathsf{F}\,\mathsf{d}_2$.

First, note that in standard synthesis, φ_0 is not realizable, as a hostile environment needs not upload. In the cooperative setting, the system can suggest to both agents the following TIT FOR TAT strategy: upload at the first time step, and from that point onward upload iff the other agent uploads. The system itself follows a strategy π_0 according to which it enables downloads whenever possible (that is, d_2 is valid whenever Agent α_1 uploads, and d_1 is valid whenever Agent α_2 uploads). It is not hard to see that the above three strategies form a triple of dominant strategies. Indeed, all the three objectives are satisfied. Thus, the triple of strategies is a solution for the cooperative setting, for both solution concepts.

What about the non-cooperative setting? Consider the above strategy π_0 of the system, and consider strategies for the agents that never upload. The tuple of the three strategies is in a π_0-fixed Nash equilibrium.

This ensures strategies for the environment to be dominant. Indeed, if Agent α_2 changes her strategy, φ_1 is still satisfied and vice-versa. Indeed, as long as Agent α_2 sticks to his strategy, Agent α_1 has no incentive to change his strategy, and similarly for Agent α_2. Thus, π_0 is not a solution to the non-cooperative rational synthesis problem for the solution concept of Nash equilibrium. On the other hand, we claim that π_0 is a solution to the non-cooperative rational synthesis problem for the solution concept of dominant strategies. For showing this, we argue that if π_1 and π_2 are dominant strategies for α_1 and α_2, then φ_0 is satisfied in the path that is the outcome of the profile $\mathsf{P} = \langle \pi_0, \pi_1, \pi_2 \rangle$. To see this, consider such a path $\eta = \mathsf{play}(\pi_0, \pi_1, \pi_2)$. We necessarily have that $\eta \models \varphi_1 \wedge \varphi_2$. Indeed, otherwise π_1 and π_2 would not be dominant, as we know that with the strategies described above, α_1 and α_2 can satisfy their objectives. Now, since $\eta \models \varphi_1 \wedge \varphi_2$, we know that u_2 and u_1 hold infinitely often in η. Also, it is not hard to see that the formulas $\mathsf{G}\,(\neg\mathsf{u}_1 \rightarrow \neg\mathsf{d}_2)$ and $\mathsf{G}\,(\neg\mathsf{u}_2 \rightarrow \neg\mathsf{d}_1)$ are always satisfied in the context of π_0, no matter how the other agents behave. It follows that $\eta \models \varphi_0$, thus π_0 is a solution of the non-cooperative rational synthesis problem for dominant strategies.

3 Qualitative Rational Synthesis

In this section we study cooperative and non-cooperative rational synthesis and show that they can be reduced to the model-checking problem for SL[NG]. The cooperative and non-cooperative rational synthesis problems for several solution concepts can be stated in SL[NG].

We first show how to state that a given strategy profile $\vec{y} = (y_0, \ldots, y_k)$ is in a y_0-fixed γ-equilibrium. For $\alpha_i \in \Omega$, let φ_i be the objective of Agent α_i. For a solution concept γ and a strategy profile $\vec{y} = (y_0, \ldots, y_k)$, the formula $\varphi^\gamma(\vec{y})$, expressing that the profile $\vec{y}$ is a y_0-fixed γ-equilibrium, is defined as follows.

- For the solution concept of dominant strategies, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{z}]\!] \bigwedge_{i=1}^{k} (\flat(\vec{z})\varphi_i \rightarrow \flat(\vec{z}_{-i}, y_i)\varphi_i)$.
- For the solution concept of Nash equilibrium, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{z}]\!] \bigwedge_{i=1}^{k} (\flat(\vec{y}_{-i}, z_i)\varphi_i \rightarrow \flat(\vec{y})\varphi_i)$.
- For the solution concept of Subgame Perfect Equilibrium, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{x}]\!]\flat(\vec{x}_{-0}, y_0)\mathsf{F} \bigwedge_{i=1}^{k} [\![z_i]\!]\flat(\vec{y}_{-i}, z_i)\varphi_i \rightarrow \flat(\vec{y})\varphi_i$.

We can now state the existence of a solution to the cooperative and non-cooperative rational-synthesis problem, respectively, with input $\varphi_0, \ldots, \varphi_k$ by the closed formulas:

1. $\varphi^\gamma_{cRS} := \langle\!\langle y_0 \rangle\!\rangle \langle\!\langle y_1 \rangle\!\rangle \ldots \langle\!\langle y_k \rangle\!\rangle (\varphi^\gamma(\vec{y}) \wedge \varphi_0)$;
2. $\varphi^\gamma_{noncRS} := \langle\!\langle y_0 \rangle\!\rangle [\![y_1]\!] \ldots [\![y_k]\!] (\varphi^\gamma(\vec{y}) \rightarrow \varphi_0)$.

Indeed, the formula 1 specifies the existence of a strategy profile $\mathsf{P} = \langle \pi_0, \ldots, \pi_k \rangle$ that is π_0-fixed γ-equilibrium and such that the outcome satisfies φ_0. On the other hand, the formula 2 specifies the existence of a strategy π_0 for the system such that the outcome of all profiles that are in a π_0-fixed γ-equilibrium satisfy φ_0.

As shown above, all the solution concepts we are taking into account can be specified in SL[NG] with formulas whose length is polynomial in the number of the agents and in which the alternation depth of the quantification is 1. Hence we can apply the known complexity results for SL[NG]:

Theorem 2 (Cooperative and non-cooperative rational-synthesis complexity). *The cooperative and non-cooperative rational-synthesis problems in the qualitative setting are* 2EXPTIME*-complete.*

Proof. Consider an input $\varphi_0, \ldots, \varphi_k$, X, and γ to the cooperative or non-cooperative rational-synthesis problem. As explained in Section 2.3, the input induces a game $\mathcal{G}$ with nodes in 2^X. As detailed above, there is a solution to the cooperative (resp., non-cooperative) problem iff the SL[NG] formula φ^γ_{cRS} (resp., φ^γ_{noncRS}) is satisfied in $\mathcal{G}$. The upper bound then follows from the fact that the model checking problem for SL[NG] formulas of alternation depth 1 is in 2EXPTIME in the size of the formula (Cf., Theorem 1). Moreover, the model-checking algorithm can return finite-state transducers that model strategies that are existentially quantified.

For the lower bound, it is easy to see that the classical LTL synthesis problem is a special case of the cooperative and non-cooperative rational synthesis problem. Indeed, $\varphi(I, O)$ is realizable against a hostile environment iff the solution to the non-cooperative rational synthesis problem for a system that has an objective φ and controls I and an environment that consists of a single agent that controls O and has an objective *True*, is positive.

4 Quantitative Rational Synthesis

As discussed in Section 1, a weakness of classical synthesis algorithms is the fact that specifications are Boolean and miss a reference to the quality of the satisfaction. Applications of game theory consider games with quantitative payoffs. Thus, even more than the classical setting, the rational setting calls for an extension of the synthesis problem to a quantitative setting. In this section we introduce *Objective* LTL, a quantitative extension of LTL, and study an extension of the rational-synthesis problem for specifications in Objective LTL. As opposed to other multi-valued specification formalisms used in the context of synthesis of high-quality systems [4, 1], Objective LTL uses the syntax and semantics of LTL and only augments the specification with a reward function that enables a succinct and convenient prioritization of sub-specifications.

4.1 The quantitative setting

Objective LTL (OLTL, for short) is an extension of LTL in which specifications consist of sets of LTL formulas weighted by functions. Formally, an OLTL *specification* over a set X of atomic propositions is a pair $\theta = \langle \Psi, \mathsf{f} \rangle$, where $\Psi = \langle \psi_1, \psi_2, \dots, \psi_m \rangle$ is a tuple of LTL formulas over X and $\mathsf{f} : \{0,1\}^m \to \mathbb{Z}$ is a *reward function*, mapping Boolean vectors of length m to integers. We assume that f is given by a polynomial. We use $|\Psi|$ to denote $\sum_{i=1}^{m} |\psi_i|$. For a computation $\eta \in (2^{\mathrm{X}})^\omega$, the *signature of* Ψ *in* η, denoted $\mathsf{sig}(\Psi, \eta)$, is a vector in $\{0,1\}^m$ indicating which formulas in Ψ are satisfied in η. Thus, $\mathsf{sig}(\Psi, \eta) = \langle v_1, v_2, \dots, v_m \rangle$ is such that for all $1 \leq i \leq m$, we have that $v_i = 1$ if $\eta \models \psi_i$ and $v_i = 0$ if $\eta \not\models \psi_i$. The *value of* θ *in* η, denoted $\mathsf{val}(\theta, \eta)$ is then $\mathsf{f}(\mathsf{sig}(\Psi, \eta))$. Thus, the interpretation of an OLTL specification is quantitative. Intuitively, $\mathsf{val}(\theta, \eta)$ indicates the level of satisfaction of the LTL formulas in Ψ in η, as determined by the priorities induced by f. We note that the input of weighted LTL formulas studied in [9] is a special case of Objective LTL.

Example 2. Consider a system with m buffers, of capacities $\mathsf{c}_1, \dots, \mathsf{c}_m$. Let full_i, for $1 \leq i \leq m$, indicate that buffer i is full. The OLTL specification $\theta = \langle \Psi, \mathsf{f} \rangle$, with $\Psi = \langle \mathsf{F}\,\mathsf{full}_1, \mathsf{F}\,\mathsf{full}_2, \dots, \mathsf{F}\,\mathsf{full}_m \rangle$ and $\mathsf{f}(\mathsf{v}) = \mathsf{c}_1 \cdot \mathsf{v}_1 + \cdots + \mathsf{c}_m \cdot \mathsf{v}_m$ enables us to give different satisfaction values to the objective of filling a buffer. Note that $\mathsf{val}(\langle \Psi, \mathsf{f} \rangle, \eta)$ in a computation η is the sum of capacities of all filled buffers.

In the quantitative setting, the objective of Agent α_i is given by means of an OLTL specification $\theta_i = \langle \Psi_i, \mathsf{f}_i \rangle$, specifications describe the payoffs to the agents, and the objective of each agent (including the system) is to maximize his payoff. For a strategy profile P, the payoff for Agent α_i in P is simply $\mathsf{val}(\theta_i, \mathsf{play}(\mathsf{P}))$.

In the quantitative setting, rational synthesis is an optimization problem. Here, in order to easily solve it, we provide a decision version by making use of a threshold. It is clear that the optimization version can be solved by searching for the best threshold in the decision one.

Definition 2 (Quantitative rational synthesis). *The input to the quantitative rational-strategy problem is a set* X *of atomic propositions, partitioned into* $\mathrm{X}_0, \dots, \mathrm{X}_k$*,* OLTL *specifications* $\theta_0, \theta_1, \dots, \theta_k$*, with* $\theta_i = \langle \Psi_i, \mathsf{f}_i \rangle$*, and a solution concept* γ*. We distinguish between two variants of the problem:*

1. *In* cooperative quantitative rational synthesis*, the desired output for a given threshold* $t \in \mathbb{N}$ *is a strategy profile* P *such that* $\mathsf{payoff}_0(\mathsf{P}) \geq t$ *and* P *is in a* π_0*-fixed* γ*-equilibrium.*
2. *In* non-cooperative quantitative rational synthesis*, the desired output for a given threshold* $t \in \mathbb{N}$ *is a strategy* π_0 *for the system such that, for each strategy profile* P *including* π_0 *and being in a* π_0*-fixed* γ*-equilibrium, we have that* $\mathsf{payoff}_0(\mathsf{P}) \geq t$*.*

Now, we introduce some auxiliary formula that helps us to formulate also the quantitative rational-synthesis problem in SL[NG].

For a tuple $\Psi = \langle \psi_1, \ldots, \psi_m \rangle$ of LTL formulas and a signature $v = \{v_1, \ldots, v_m\} \in \{0,1\}^m$, let $\mathsf{mask}(\Psi, v)$ be an LTL formula that characterizes computations η for which $\mathsf{sig}(\Psi, \eta) = v$. Thus, $\mathsf{mask}(\Psi, v) = (\bigwedge_{i:v_i=0} \neg\psi_i) \wedge (\bigwedge_{i:v_i=1} \psi_i)$.

We adjust the SL formulas $\Phi^\gamma(\vec{y})$ described in Section 3 to the quantitative setting. Recall that $\Phi^\gamma(\vec{y})$ holds iff the strategy profile assigned to $\vec{y}$ is in a π_0-fixed γ-equilibrium. There, the formula is a conjunction over all agents in $\{\alpha_1, \ldots, \alpha_k\}$, stating that Agent α_i does not have an incentive to change his strategy. In our quantitative setting, this means that the payoff of Agent α_i in an alternative profile is not bigger than his payoff in $\vec{y}$. For two strategy profiles, assigned to $\vec{y}$ and $\vec{y}'$, an SL formula that states that Agent α_i has no incentive that the profile would change from $\vec{y}$ to $\vec{y}'$ can state that the signature of Ψ in $\mathsf{play}(\vec{y}')$ results in a payoff to Agent α_i that is smaller than his current payoff. Formally, we have that:

$$\Phi_i^{eq}(\vec{y}, \vec{y}') = \bigvee_{v \in \{1,\ldots,m_i\}: \mathsf{f}_i(v) \leq \mathsf{payoff}_i(\vec{y})} \flat(\vec{y}')\mathsf{mask}(\Psi_i, v).$$

We can now adjust $\Phi^\gamma(\vec{y})$ for all the cases of solution concepts we are taking into account.

- For the solution concept of dominant strategies, we define:
  $\Phi^\gamma(\vec{y}) := \bigwedge_{i \in \{1,\ldots,n\}} [\![\vec{z}]\!] \Phi_i^{eq}(\vec{y}, (\vec{z}[\alpha_0 \leftarrow y_0]);$
- For the solution concept of Nash equilibrium, we define:
  $\Phi^\gamma(\vec{y}) := \bigwedge_{i \in \{1,\ldots,n\}} [\![\vec{z}]\!] \Phi_i^{eq}(\vec{y}, (\vec{y}[\alpha_i \leftarrow z_i]);$
- For the solution concept of Subgame Perfect Equilibrium, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{x}]\!] \flat(\vec{x}[\alpha_0 \leftarrow y_0])\mathsf{F} \bigwedge_{i \in \{1,\ldots,n\}} [\![\vec{z}]\!] \Phi_i^{eq}(\vec{y}, (\vec{z}[\alpha_0 \leftarrow y_0])).$

Once we adjust $\Phi^\gamma(\vec{y})$ to the quantitative setting, we can use the same SL formula used in the non-quantitative setting to state the existence of a solution to the rational synthesis problem. We have the following:

- $\Phi_{RS}^\gamma := \langle\!\langle y_0 \rangle\!\rangle \langle\!\langle y_1 \rangle\!\rangle \ldots \langle\!\langle y_k \rangle\!\rangle (\Phi^\gamma(\vec{y}) \wedge \varphi_0);$
- $\Phi_{nonRS}^\gamma := \langle\!\langle y_0 \rangle\!\rangle [\![y_1]\!] \ldots [\![y_k]\!] (\Phi^\gamma(\vec{y}) \rightarrow \varphi_0).$

Theorem 3. *The cooperative and non-cooperative quantitative rational-synthesis problems are 2*EXPTIME-COMPLETE.

Proof. We can reduce the problems to the model-checking problem of the SL formulas Φ_{RS}^γ and Φ_{nonRS}^γ, respectively. We should, however, take care when analyzing the complexity of the procedure, as the formulas $\Phi_i^{eq}(\vec{y}, \vec{y}')$, which participate in Φ_{RS}^γ

and Φ^{γ}_{nonRS} involve a disjunction over vectors in $\{0,1\}^{m_i}$, resulting in Φ^{γ}_{nonRS} of an exponential length.

While the above prevents us from using the doubly exponential known bound on SL model checking for formulas of alternation depth 1 as is, it is not difficult to observe that the run time of the model-checking algorithm in [15], when applied to Φ^{γ}_{nonRS}, is only doubly exponential. The reason is the fact that the inner exponent paid in the algorithm for SL model checking is due to the blow-up in the translation of the formula to a nondeterministic Büchi automaton over words (NBW, for short). In this translation, the exponentially many disjuncts are dominated by the exponential translation of the innermost LTL to NBW. Thus, the running time of the algorithm is doubly exponential, and it can return the witnessing strategies.

Hardness in 2EXPTIME follows easily from hardness in the non-quantitative setting.

5 Discussion

The understanding that synthesis corresponds to a game in which the objective of each player is to satisfy his specification calls for a mutual adoption of ideas between formal methods and game theory. In *rational synthesis*, introduced in [10], synthesis is defined in a way that takes into an account the rationality of the agents that constitute the environment and involves and assumption that an agent cooperates with a strategy in which his objective is satisfied. Here, we extend the idea and consider also *non-cooperative* rational synthesis, in which agents need not cooperate with suggested strategies and may prefer different strategies that are at least as beneficial for them.

Many variants of the classical synthesis problem has been studied. It is interesting to examine the combination of the rational setting with the different variants. To start, the cooperative and non-cooperative settings can be combined into a framework in which one team of agents is competing with another team of agents, where each team is internally cooperative, but the two teams are non-cooperative. Furthermore, we plan to study rational synthesis with incomplete information. In particular, we plan to study rational synthesis with *incomplete information* [12], where agents can view only a subset of the signals that other agents output, and rational *stochastic* synthesis [7], which models the unpredictability of nature and involves stochastic agents that assign values to their output signals by means of a distribution function. Beyond a formulation of the richer settings, one needs a corresponding extension of strategy logic and its decision problems.

As discussed in Section 1, classical applications of game theory consider games with quantitative payoffs. We added a quantitative layer to LTL by introducing Objective-LTL and studying its rational synthesis. In recent years, researchers have developed more refined quantitative temporal logics, which enable a formal reasoning about the quality of systems. In particular, we plan to study rational synthesis for the multi-valued logics LTL[F] [1], which enables a prioritization of different satisfaction possibilities, and LTL[D] [2], in which discounting is used in order to reduce the satisfaction value of specifications whose eventualities are delayed. The rational synthesis problem for these logics induce a game with much richer, possibly infinitely many, profiles, making the search for a stable solution much more challenging.

References

[1] S. Almagor, U. Boker, and O. Kupferman. Formalizing and Reasoning about Quality. In *ICALP'13*, volume 7966 of *LNCS*, pages 15–27, 2013.

[2] S. Almagor, U. Boker, and O. Kupferman. Discounting in LTL. In *TACAS'14*, volume 8413 of *LNCS*, pages 424–439. Springer, 2014.

[3] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *JACM*, 49(5):672–713, 2002.

[4] R. Bloem, K. Chatterjee, T. Henzinger, and B. Jobstmann. Better Quality in Synthesis through Quantitative Objectives. In *CAV'09*, volume 5643 of *LNCS*, pages 140–156. Springer, 2009.

[5] K. Chatterjee, T. Henzinger, and B. Jobstmann. Environment Assumptions for Synthesis. In *CONCUR'08*, volume 5201 of *LNCS*, pages 147–161. Springer, 2008.

[6] K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. In *CONCUR'07*, LNCS 4703, pages 59–73. Springer, 2007.

[7] K. Chatterjee, R. Majumdar, and M. Jurdzinski. On Nash Equilibria in Stochastic Games. In *CSL'04*, volume 3210 of *LNCS*, pages 26–40. Springer, 2004.

[8] A. Church. Logic, Arithmetics, and Automata. In *Proc. Int. Congress of Mathematicians*, pages 23–35. Institut Mittag-Leffler, 1963.

[9] C. Courcoubetis and M. Yannakakis. Markov Decision Processes and Regular Events. *IEEE Transaction on automatic control*, 43(10):1399 – 1418, 1998.

[10] D. Fisman, O. Kupferman, and Y. Lustig. Rational Synthesis. In *TACAS'10*, LNCS 6015, pages 190–204. Springer, 2010.

[11] T.A. Henzinger. From Boolean to Quantitative Notions of Correctness. In *POPL'10*, pages 157–158. ACM, 2010.

[12] O. Kupferman and M. Y. Vardi. Church's problem revisited. *Bulletin of Symbolic Logic*, 5(2):245–263, 1999.

[13] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems - Specification*. Springer, 1992.

[14] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi. What Makes ATL* Decidable? A Decidable Fragment of Strategy Logic. In *CONCUR'12*, volume 7454 of *LNCS*, pages 193–208, 2012.

[15] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. volume 15, 2014. to appear.

[16] F. Mogavero, A. Murano, and M.Y. Vardi. Reasoning About Strategies. In *FSTTCS'10*, LIPIcs 8, pages 133–144, 2010.

[17] N. Nisan and A. Ronen. Algorithmic Mechanism Design. *Games and Economic Behavior*, 35(1-2):166–196, 2001.

[18] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[19] M.J. Osborne and A. Rubinstein. *A Course in Game Theory.* MIT Press, 1994.

[20] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *POPL'89*, pages 179–190, 1989.

[21] R. Selten. Reexamination of the Perfectness Concept for Equilibrium Points in Extensive Games. *International Journal of Game Theory*, 4(1):25–55, 1975.