Graded Strategy Logic: Reasoning about Uniqueness of Nash Equilibria *

Benjamin Aminof
Technische Universitat Wien, Austria
benj@forsyte.tuwien.ac.at

Vadim Malvone, Aniello Murano, Sasha Rubin
Università degli Studi di Napoli Federico II, Italy
{malvone,murano,rubin}@unina.it

Strategy Logic (SL) is a well established formalism for strategic reasoning in multi-agent systems. It is built over LTL and treats strategies as first-order objects that are associated with agents by means of a binding operator. In this work, we introduce Graded Strategy Logic (GRADED-SL), an extension of SL by graded quantifiers over tuples of strategy variables such as "there exist at least g different tuples $(x_1,...,x_n)$ of strategies". We study the model-checking problem of GRADED-SL and prove that it is no harder than for SL, i.e., it is non-elementary in the quantifier rank. We show that GRADED-SL allows one to count the number of different strategy profiles that are Nash equilibria (NE), or subgame-perfect equilibria (SPE). By analyzing the structure of the specific formulas involved, we conclude that the important problems of checking for the existence of a unique NE or SPE can both be solved in 2EXPTIME, which is not harder than merely checking for the existence of such equilibria.

1 Introduction

Strategy Logic (SL) is a powerful formalism for reasoning about strategies in multi-agent systems [24, 25]. Strategies tell an agent what to do — they are functions that prescribe an action based on the history. The key idea in SL is to treat strategies as first-order object. A strategy x can be quantified existentially $\langle\langle x\rangle\rangle$ (read: there exists a strategy x) and universally $[[x]]$ (read: for all strategies x). Strategies are not intrinsically glued to specific agents: the *binding* operator (α, x) allows one to bind an agent α to the strategy x. SL strictly subsumes several other logics for strategic reasoning including the well known ATL and ATL* [2]. Being a very powerful logic, SL can directly express many solution concepts [11, 17, 18, 24] among which that a strategy profile $\bar{x}$ is a Nash equilibrium, and thus also the existence of a Nash equilibrium (NE) as well as other important solution concepts such as subgame-perfect equilibrium (SPE) [32].

Nash equilibrium is one of the most important concepts in game theory, forming the basis of much of the recent fundamental work in multi-agent decision making. A challenging and important aspect is to establish whether a game admits a *unique* NE [1]. This problem impacts on the predictive power of NE since, in case there are multiple equilibria, the outcome of the game cannot be uniquely pinned down [12]. Unfortunately, uniqueness has mainly been established either for special cost functions [1], or for very restrictive game topologies [27]. More specifically, uniqueness of NE in game theory is proved with procedures that are separated from the ones that check for its existence [1]; these procedures require various transformations of the best response functions of individual contributors [14, 15], and there is no general theory that can be applied to different application areas [1].

In this paper we address and solve the problem of expressing the uniqueness of certain solution concepts (and NE in particular) in a principled and elegant way. We introduce an extension of SL called

*This is an expository paper reporting on the work [4].

To appear in EPTCS.

GRADED-SL and study its model-checking problem. We extend SL by replacing the quantification $\langle\langle x\rangle\rangle$ and $[[x]]$ over strategy variables with *graded quantification over tuples of strategy variables*: $\langle\langle x_1,\ldots,x_n\rangle\rangle^{\geq g}$ (read $\langle\langle x_1,\ldots,x_n\rangle\rangle^{\geq g}$ as "there exist at least g different tuples $(x_1,\ldots,x_n)$ of strategies") and its dual $[[x_1,\ldots,x_n]]^{<g}$ ($g\in\mathbb{N}$). Here, two strategies are different if they disagree on some history. That is, we count strategies syntactically as is usual in graded extensions of modal and description logics [8, 9, 16, 19]. The key for expressing uniqueness of NE is the combination of quantifying over tuples (instead of singleton variables), and adding counting (in the form of graded modalities).

We address the model-checking problem for GRADED-SL and prove that it has the same complexity as SL, i.e., it is non-elementary in the nesting depth of quantifiers. In particular, we show that model checking GRADED-SL formulas with a nesting depth $k>0$ of blocks of quantifiers (a block of quantifiers is a maximally-consecutive sequence of quantifiers of the same type, i.e., either all existential, or all universal) is in $(k+1)$EXPTIME, and that for the special case where the formula starts with a block of quantifiers, it is in kEXPTIME. Since many natural formulas have a small number of quantifiers, and even smaller nesting depth of blocks of quantifiers, the complexity of the model-checking problem is not as bad as it seems. Specifically, several solution concepts can be expressed as SL formulas with a small number of quantifiers [11, 17, 18, 24]. Since the existence of a NE, and the fact that there is at most one NE, can be expressed in GRADED-SL using simple formulas we are able to conclude that the problem of checking the uniqueness of a NE can be solved in 2EXPTIME. Previously, it was only known that existence of NE can be checked in 2EXPTIME [17, 24], and indeed it is 2EXPTIME-complete [3, 17, 28]. Thus, GRADED-SL is the first logic that can solve the existence and uniqueness of NE (as well as many other solution concepts) in a uniform way.

SL has a few natural and powerful syntactic fragments. In particular, the Nested-Goal fragment of SL requires that bindings and quantifications appear in exhaustive blocks (see Section 2 for details), and can express NE [24]. Similarly, we define Nested-Goal GRADED-SL, a syntactic fragment of GRADED-SL. We show that Nested-Goal GRADED-SL has the same model-checking complexity as Nested-Goal SL, i.e., non-elementary in the *alternation number* of the quantifiers appearing in the formula (the alternation number of a Nested-Goal formula is, roughly speaking, the maximum number of existential/universal quantifier switches in a consecutive sequence of quantifiers, see Appendix A). Since uniqueness of subgame-perfect equilibria (SPE) can be expressed in Nested-Goal GRADED-SL using alternation one, we get a proof that checking the SPE uniqueness is in 2EXPTIME (as well as an alternative proof for NE).

We focus on the *iterated prisoner's dilemma* (IPD) [29]. The prisoner's dilemma (PD) is a popular metaphor for the problem of stable cooperation and has been widely used in several application domains [5]. In the classic definition, it consists of two players, each of them has an option to defect or collaborate. More involved is the IPD in which the process is repeated and one can model reactive strategies in continuous play. The IPD has become a standard model for the evolution of cooperative behaviour within a community of egoistic agents, frequently cited for implications in both sociology and biology (see [5]). Our work shows that checking the uniqueness of a NE and of a SPE in an IPD can be solved in 2EXPTIME.

The work closest to ours is [22]: also motivated by counting NE, it introduces a graded extension of SL, called GSL. In contrast with our work, GSL has a very intricate way of counting strategies: it gives a semantic definition of equivalence of strategies, and the graded modalities count non-equivalent strategies. While this approach is sound, it heavily complicates the model-checking problem. Indeed, only a very weak fragment of GSL has been solved in [22] by exploiting an *ad hoc* solution that does not seem to be scalable to (all of) GSL.

2 Graded Strategy Logic

In this section we introduce Graded Strategy Logic, which we call GRADED-SL for short.

2.1 Models

Sentences of GRADED-SL are interpreted over *concurrent game structures*, just as for ATL and SL [2,24].

Definition 2.1 *A* concurrent game structure *(*CGS*) is a tuple* $\mathcal{G} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, s_I, \mathsf{ap}, \mathsf{tr} \rangle$*, where* AP*,* Ag*,* Ac*, and* St *are the sets of* atomic propositions*,* agents*,* actions *and* states*, respectively,* $s_I \in \mathrm{St}$ *is the* initial state*, and* $\mathsf{ap} : \mathrm{St} \to 2^{\mathrm{AP}}$ *is the* labeling function *mapping each state to the set of atomic propositions true in that state. Let* $\mathrm{Dc} \triangleq \mathrm{Ag} \to \mathrm{Ac}$ *be the set of* decisions*,* i.e.*, functions describing the choice of an action by every agent. Then,* $\mathsf{tr} : \mathrm{Dc} \to (\mathrm{St} \to \mathrm{St})$ *denotes the* transition function *mapping every decision* $\delta \in \mathrm{Dc}$ *to a function* $\mathsf{tr}(\delta) : \mathrm{St} \to \mathrm{St}$*.*

We will usually take the set Ag of agents to be $\{\alpha_1, \ldots, \alpha_n\}$. A *path (from s)* is a finite or infinite non-empty sequence of states $s_1 s_2 \ldots$ such that $s = s_1$ and for every i there exists a decision δ with $\mathsf{tr}(\delta)(s_i) = s_{i+1}$. The set of paths starting with s is denoted $\mathrm{Pth}(s)$. The set of finite paths from s, called the *histories (from s)*, is denoted $\mathrm{Hst}(s)$. A *strategy (from s)* is a function $\sigma \in \mathrm{Str}(s) \triangleq \mathrm{Hst}(s) \to \mathrm{Ac}$ that prescribes which action has to be performed given a certain history. We write $\mathrm{Pth}, \mathrm{Hst}, \mathrm{Str}$ for the set of all paths, histories, and strategies (no matter where they start). We use the standard notion of equality between strategies, [21], i.e., $\sigma_1 = \sigma_2$ iff for all $\rho \in \mathrm{Hst}$, $\sigma_1(\rho) = \sigma_2(\rho)$. This extends to equality between two n-tuples of strategies in the natural way, i.e., coordinate-wise.

2.2 Syntax

GRADED-SL extends SL by replacing the singleton strategy quantifiers $\langle\langle x \rangle\rangle$ and $[\![x]\!]$ with the graded (tupled) quantifiers $\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g}$ and $[\![x_1, \ldots, x_n]\!]^{<g}$, respectively, where each x_i belongs to a countable set of variables Vr and $g \in \mathbb{N}$ is called the *degree* of the quantifier. Intuitively, these are read as "there exist at least g tuples of strategies $(x_1, \ldots, x_n)$" and "all but less than g many tuples of strategies", respectively. The syntax (α, x) denotes a *binding* of the agent α to the strategy x.

Definition 2.2 GRADED-SL formulas *are built inductively by means of the following grammar, where* $p \in \mathrm{AP}$*,* $\alpha \in \mathrm{Ag}$*,* $x, x_1, \ldots x_n \in \mathrm{Vr}$ *such that* $x_i \neq x_j$ *for* $i \neq j$*, and* $g, n \in \mathbb{N}$*:*

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \,\mathsf{U}\, \varphi \mid \langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \varphi \mid (\alpha, x)\varphi.$$

Notation. Whenever we write $\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g}$ we mean that $x_i \neq x_j$ for $i \neq j$. Shorthands are derived as usual. Specifically, $\mathsf{true} \triangleq p \vee \neg p$, $\mathsf{false} \triangleq \neg\mathsf{true}$, $\mathsf{F}\varphi \triangleq \mathsf{true}\,\mathsf{U}\,\varphi$, and $\mathsf{G}\varphi \triangleq \neg\mathsf{F}\neg\varphi$. Also, we have that $[\![x_1, \ldots, x_n]\!]^{<g} \varphi \triangleq \neg\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \neg\varphi$.

In order to define the semantics, we first define the concept of *free placeholders* in a formula, which refer to agents and variables. Intuitively, an agent or variable is free in φ if it does not have a strategy associated with it (either by quantification or binding) but one is required in order to ascertain if φ is true or not. The set of *free agents* and *free variables* of a GRADED-SL formula φ is given by the function $\mathsf{free} : \text{GRADED-SL} \to 2^{\mathrm{Ag} \cup \mathrm{Vr}}$ defined as follows:

- $\mathsf{free}(p) \triangleq \emptyset$, where $p \in \mathrm{AP}$;
- $\mathsf{free}(\neg\varphi) \triangleq \mathsf{free}(\varphi)$;
- $\mathsf{free}(\varphi_1 \vee \varphi_2) \triangleq \mathsf{free}(\varphi_1) \cup \mathsf{free}(\varphi_2)$;

- $\mathsf{free}(\mathsf{X}\varphi) \triangleq \mathrm{Ag} \cup \mathsf{free}(\varphi)$;
- $\mathsf{free}(\varphi_1 \mathsf{U} \varphi_2) \triangleq \mathrm{Ag} \cup \mathsf{free}(\varphi_1) \cup \mathsf{free}(\varphi_2)$;
- $\mathsf{free}(\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \varphi) \triangleq \mathsf{free}(\varphi) \setminus \{x_1, \ldots, x_n\}$;
- $\mathsf{free}((\alpha, x)\varphi) \triangleq \mathsf{free}(\varphi)$, if $\alpha \notin \mathsf{free}(\varphi)$, where $\alpha \in \mathrm{Ag}$ and $x \in \mathrm{Vr}$;
- $\mathsf{free}((\alpha, x)\varphi) \triangleq (\mathsf{free}(\varphi) \setminus \{\alpha\}) \cup \{x\}$, if $\alpha \in \mathsf{free}(\varphi)$, where $\alpha \in \mathrm{Ag}$ and $x \in \mathrm{Vr}$.

A formula φ without free agents (resp., variables), i.e., with $\mathsf{free}(\varphi) \cap \mathrm{Ag} = \emptyset$ (resp., $\mathsf{free}(\varphi) \cap \mathrm{Vr} = \emptyset$), is called *agent-closed* (resp., *variable-closed*). If φ is both agent- and variable-closed, it is called a *sentence*.

A formula φ without free agents (resp., variables), i.e., with $\mathsf{free}(\varphi) \cap \mathrm{Ag} = \emptyset$ (resp., $\mathsf{free}(\varphi) \cap \mathrm{Vr} = \emptyset$), is called *agent-closed* (resp., *variable-closed*). Also, φ is a *sentence* if it is both agent- and variable-closed.

SL has a few natural syntactic fragments, the most powerful of which is called Nested-Goal SL. Similarly, we define *Nested-Goal* GRADED-SL (abbreviated GRADED-SL[NG]), as a syntactic fragment of GRADED-SL. As in SL[NG], in GRADED-SL[NG] we require that bindings and quantifications appear in exhaustive blocks. I.e., whenever there is a quantification over a variable in a formula ψ it is part of a consecutive sequence of quantifiers that covers all of the free variables that appear in ψ, and whenever an agent is bound to a strategy then it is part of a consecutive sequence of bindings of all agents to strategies. Finally, formulas with free agents are not allowed. To formalize GRADED-SL[NG] we first introduce some notions. A *quantification prefix* over a finite set $\mathrm{V} \subseteq \mathrm{Vr}$ of variables is a sequence $\wp \in \{\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g}, [\![x_1, \ldots, x_n]\!]^{<g} : x_1, \ldots, x_n \in \mathrm{V} \wedge g \in \mathbb{N}\}^*$ such that each $x \in \mathrm{V}$ occurs exactly once in $\wp$. A *binding prefix* is a sequence $\flat \in \{(a, x) : \alpha \in \mathrm{Ag} \wedge x \in \mathrm{Vr}\}^*$ such that each $\alpha \in \mathrm{Ag}$ occurs exactly once in $\flat$. We denote the set of binding prefixes by Bn, and the set of quantification prefixes over V by $\mathrm{Qn}(\mathrm{V})$.

Definition 2.3 GRADED-SL[NG] *formulas are built inductively using the following grammar, with* $p \in \mathrm{AP}$, $\wp \in \mathrm{Qn}(\mathrm{V})$ *(*$\mathrm{V} \subseteq \mathrm{Vr}$*), and* $\flat \in \mathrm{Bn}$*:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \wp\varphi \mid \flat\varphi,$$

where in the rule $\wp\varphi$ *we require that* φ *is agent-closed and* $\wp \in \mathrm{Qn}(\mathsf{free}(\varphi))$.

Formulas of GRADED-SL[NG] can be classified according to their *alternation number*, i.e., the maximum number of quantifier switches in a quantification prefix. See Appendix A for the formal definition.

The *quantifier rank* of φ is the maximum nesting of quantifiers in φ, e.g., $\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g}(\alpha_1, x_1) \ldots (\alpha_n, x_n) \bigwedge_{i=1}^{n} (\langle\langle \mathrm{y} \rangle\rangle(\alpha_i, \mathrm{y})\psi_i) \rightarrow \psi_i$ has quantifier rank 2 if each ψ_i is quantifier free. Moreover, a *quantifier-block of* φ is a maximally-consecutive sequence of quantifiers in φ of the same type (*i.e.*, either all existential, or all universal). The *quantifier-block rank of* φ is exactly like the quantifier rank except that a quantifier block of j quantifiers contributes 1 instead of j to the count.

We conclude this subsection by introducing *One-Goal* GRADED-SL, written GRADED-SL[1G]. The importance of this fragment in SL stems from the fact that it strictly includes ATL* while maintaining the same complexity for both the model checking and the satisfiability problems, *i.e.* 2EXPTIME-COMPLETE [23, 24]. However, it is commonly believed that Nash Equilibrium cannot be expressed in this fragment. The definition of GRADED-SL[1G] follows.

Definition 2.4 GRADED-SL[1G] *formulas are built inductively using the following grammar, with* $p \in \mathrm{AP}$, $\wp \in \mathrm{Qn}(\mathrm{V})$ *(*$\mathrm{V} \subseteq \mathrm{Vr}$*), and* $\flat \in \mathrm{Bn}$*:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \wp\flat\varphi,$$

where $\wp$ *is a quantification prefix over* $\mathsf{free}(\flat\varphi)$.

2.3 Semantics

As for SL, the interpretation of a GRADED-SL formula requires a valuation of its free placeholders. This is done via *assignments (from s)*, *i.e.*, functions $\chi \in \mathrm{Asg}(s) \triangleq (\mathrm{Vr} \cup \mathrm{Ag}) \to \mathrm{Str}(s)$ mapping variables/agents to strategies. We denote by $\chi[e \mapsto \sigma]$, with $e \in \mathrm{Vr} \cup \mathrm{Ag}$ and $\sigma \in \mathrm{Str}(s)$, the assignment that differs from χ only in the fact that e maps to σ. Extend this definition to tuples: for $\overline{e} = (e_1, \ldots, e_n)$ with $e_i \neq e_j$ for $i \neq j$, define $\chi[\overline{e} \mapsto \overline{\sigma}]$ to be the assignment that differs from χ only in the fact that e_i maps to σ_i (for each i). For an assignment $\chi \in \mathrm{Asg}(s)$ the (χ, s)-*play* denotes the path $\pi \in \mathrm{Pth}(s)$ such that for all $i \in \mathbb{N}$, it holds that $\pi_{i+1} = \mathsf{tr}(\mathsf{dc})(\pi_i)$, where $\mathsf{dc}(\alpha) \triangleq \chi(\alpha)(\pi_{\leq i})$ for $\alpha \in \mathrm{Ag}$. The function $\mathsf{play} : \mathrm{Asg} \times \mathrm{St} \to \mathrm{Pth}$, with $\mathsf{dom}(\mathsf{play}) \triangleq \{(\chi, s) : \chi \in \mathrm{Asg}(s)\}$, maps (χ, s) to the (χ, s)-play $\mathsf{play}(\chi, s) \in \mathrm{Pth}(s)$. The notation $\pi_{\leq i}$ (resp. $\pi_{<i}$) denotes the prefix of the sequence π of length i (resp. $i-1$). Similarly, the notation π_i denotes the ith symbol of π. Thus, $\mathsf{play}(\chi, s)_i$ is the ith state on the play determined by χ from s. For $\chi \in \mathrm{Asg}(s)$ and $i \in \mathbb{N}$, writing $\rho \triangleq \mathsf{play}(\chi, s)_{\leq i}$ (the prefix of the play up to i) and $t \triangleq \mathsf{play}(\chi, s)_i$ (the last state of ρ) define $\chi_i \in \mathrm{Asg}(t)$ to be the assignment from t that maps $e \in \mathrm{Vr} \cup \mathrm{Ag}$ to the strategy that maps $h \in \mathrm{Hst}(t)$ to the action $\chi(e)(\rho_{<i} \cdot h)$.

The semantics of GRADED-SL mimics the one for SL as given in [24]. Given a CGS $\mathscr{G}$, for all states $s \in \mathrm{St}$ and assignments $\chi \in \mathrm{Asg}(s)$, the relation $\mathscr{G}, \chi, s \models \varphi$ is defined inductively on the structure of φ. In addition to the SL case, we have the existential quantifier $\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \varphi$ interpreted as follows:

$$\mathscr{G}, \chi, s \models \langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \varphi \text{ iff there exist } g \text{ many tuples } \overline{\sigma_1}, \ldots, \overline{\sigma_g} \text{ of strategies such that:}$$
$$\overline{\sigma_i} \neq \overline{\sigma_j} \text{ for } i \neq j, \text{ and}$$
$$\mathscr{G}, \chi[\overline{x} \mapsto \overline{\sigma_i}], s \models \varphi \text{ for } 1 \leq i \leq g.$$

Intuitively, $\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \varphi$ counts the number of distinct tuples of strategies that satisfy φ.

As usual, if χ and χ' agree on $\mathsf{free}(\varphi)$, then $\mathscr{G}, \chi, s \models \varphi$ if and only if $\mathscr{G}, \chi', s \models \varphi$, i.e., the truth of φ does not depend on the values the assignment takes on placeholders that are not free. Thus, for a sentence φ, we write $\mathscr{G} \models \varphi$ to mean that $\mathscr{G}, \chi, s_I \models \varphi$ for some (equivalently, for all) assignments χ, and where s_I is the initial state of $\mathscr{G}$.

3 Illustrating GRADED-SL: uniqueness of some solution concepts

In game theory, players have objectives that are summarised in a payoff function that maps plays to real numbers. In order to specify such payoffs we follow a formalisation from [18] called *objective* LTL. This will allow us to model the Prisoner's Dilemma (PD), probably the most famous game in game-theory, as well as an iterated version (IPD). We then discuss appropriate solution concepts, and show how to express these in GRADED-SL.

Let $\mathscr{G}$ be a CGS with n agents. Let $m \in \mathbb{N}$ and fix, for each agent $\alpha_i \in \mathrm{Ag}$, an *objective* tuple $S_i \triangleq \langle \mathsf{f}_i, \varphi_i^1, \ldots, \varphi_i^m \rangle$, where $\mathsf{f}_i : \{0,1\}^m \to \mathbb{N}$, and each φ_i^j is an LTL formula over AP. If π is a play, then agent α_i receives payoff $\mathsf{f}_i(\overline{h}) \in \mathbb{N}$ where the j'th bit $\overline{h}_j$ of $\overline{h}$ is 1 if and only if $\pi \models \varphi_i^j$. For instance, f may count the number of formulas that are true.

Prisoner's Dilemma – One shot and Iterated

For convenience, we describe the Prisoner's Dilemma (one-shot and iterated) using the same CGS in Figure 1. The difference is in how we define the objectives. For the one-shot version, the objective of agent α_i is $S_i \triangleq \langle \mathsf{f}_i, \varphi_i^1, \varphi_i^2, \varphi_i^3, \varphi_i^4 \rangle$ where $\varphi_i^1 \triangleq \mathsf{X}\mathbf{S}_i$, $\varphi_i^2 \triangleq \mathsf{X}\mathbf{P}_i$, $\varphi_i^3 \triangleq \mathsf{X}\mathbf{R}_i$, and $\varphi_i^4 \triangleq \mathsf{X}\mathbf{T}_i$ and f_i returns the value of its input vector interpreted as a binary number, *e.g.*, $\mathsf{f}_i(0100) = 4$.

In words, we have two agents α_1 and α_2. Each agent has two actions: cooperate (C) and defect (D), corresponding, respectively, to the options of remaining silent or confessing. For each possible pair

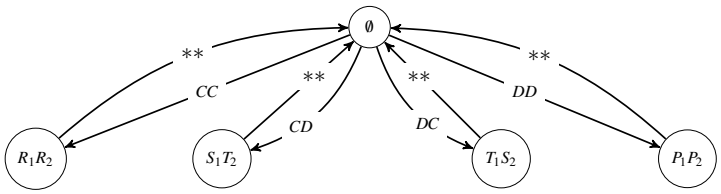

Figure 1: Iterated Prisoner's dilemma.

of moves, the game goes in a state whose atomic propositions represent the payoffs: $\mathbf{R}_i$ represents the *reward* payoff that α_i receives if both cooperate; $\mathbf{P}_i$ is the *punishment* that α_i receives if both defect; $\mathbf{T}_i$ is the *temptation* that α_i receives as a sole defector, and $\mathbf{S}_i$ is the *sucker* payoff that α_i receives as a sole cooperator. The payoffs satisfy the following chain of inequalities: $\mathbf{T}_i > \mathbf{R}_i > \mathbf{P}_i > \mathbf{S}_i$.

The Iterated Prisoner's Dilemma (IPD) is used to model a series of interactions. This is like PD except that after the first choice, both agents have another choice and so on. The main difference between the one-shot and iterated PD is that in the latter the agents' actions (may) depend on the past behaviour. Convenient payoff functions for the IPD are the mean-average and discounted-payoff [7]. Such quantitative payoffs can be replaced, in a first approximation, by LTL payoffs such as "maximise the largest payoff I receive infinitely often". This is formalised, for agent α_i, by the objective $S_i \triangleq \langle \mathsf{f}_i, \varphi_i^1, \varphi_i^2, \varphi_i^3, \varphi_i^4 \rangle$ where $\varphi_i^1 \triangleq \mathsf{GF}\mathbf{S}_i$, $\varphi_i^2 \triangleq \mathsf{GF}\mathbf{P}_i$, $\varphi_i^3 \triangleq \mathsf{GF}\mathbf{R}_i$, and $\varphi_i^4 \triangleq \mathsf{GF}\mathbf{T}_i$, and f_i is as in the one-shot PD.

Solution Concepts. Solution concepts are criteria by which one captures what rational agents would do. This is especially relevant in case each agent has its own objective. The central solution concept in game theory is the Nash Equilibrium. A tuple of strategies, one for each player, is called a *strategy profile*. A strategy profile is a *Nash equilibrium (NE)* if no agent can increase his payoff by unilaterally choosing a different strategy. A game may have zero, one, or many NE.

Consider first a CGS $\mathscr{G}$ with n agents, where the objective of the agent $\alpha_i \in \mathrm{Ag}$ contains a single LTL formula φ_i (with a larger payoff if it holds than if it doesn't). It is not hard to see that the following formula expresses that $\overline{x} \triangleq (x_1 \dots x_n)$ is a Nash Equilibrium:

$$\phi_{NE}^1(\overline{x}) \triangleq [\![\mathsf{y}_1]\!] \dots [\![\mathsf{y}_n]\!] \bigwedge_{i=1}^{n} (\flat_i \varphi_i) \rightarrow \flat \varphi_i$$

where $\flat = (\alpha_1, x_1) \dots (\alpha_n, x_n)$, and $\flat_i = (\alpha_1, x_1) \dots (\alpha_{i-1}, \mathsf{x}_{i-1})(\alpha_i, \mathsf{y}_i)(\alpha_{i+1}, \mathsf{x}_{i+1}) \dots (\alpha_n, x_n)$.

Consider now the general case, where each agent α_i has an objective tuple $S_i \triangleq \langle \mathsf{f}_i, \varphi_i^1, \dots, \varphi_i^m \rangle$. Given a vector $\overline{h} \in \{0,1\}^m$, let $gd_i(\overline{h}) \triangleq \{\overline{t} \in \{0,1\}^m \mid \mathsf{f}_i(\overline{t}) \geq \mathsf{f}_i(\overline{h})\}$ be the set of vectors $\overline{t}$ for which the payoff for agent α_i is not worse than for $\overline{h}$. Also, let $\eta_i^{\overline{h}}$ be the formula obtained by taking a conjunction of the formulas $\varphi_i^1, \dots, \varphi_i^m$ or their negations according to $\overline{h}$, i.e., by taking φ_a^j if the j'th bit in $\overline{h}$ is 1, and otherwise taking $\neg\varphi_a^j$. Formally, $\eta_i^{\overline{h}} \triangleq \wedge_{j \in \{1 \leq j \leq m \mid h_j = 1\}} \varphi_i^j \wedge \wedge_{j \in \{1 \leq j \leq m \mid h_j = 0\}} \neg \varphi_i^j$. Observe that the following formula says that $\overline{x} \triangleq (x_1 \dots x_n)$ is a Nash Equilibrium:

$$\phi_{NE}(\overline{x}) \triangleq [\![\mathsf{y}_1]\!] \dots [\![\mathsf{y}_n]\!] \bigwedge_{i=1}^{n} \bigwedge_{\overline{h} \in \{0,1\}^m} (\flat_i \eta_i^{\overline{h}}) \rightarrow \bigvee_{\overline{t} \in gd_i(\overline{h})} \flat \eta_i^{\overline{t}}$$

Going back to the PD example, due to the simplicity of the payoff functions, we have that:

$$\phi_{PD}(\overline{x}) \triangleq [\![\mathsf{y}_1]\!] \dots [\![\mathsf{y}_n]\!] \bigwedge_{i=1}^{2} \bigwedge_{j=1}^{4} (\flat_i \varphi_i^j) \Rightarrow (\bigvee_{r \geq j} \flat \varphi_i^r)$$

As it turns out (again due to the simplicity of the payoff functions), the formula above is also correct for the IPD. It has been argued (in [18, 32]) that NE may be implausible when used for sequential games (of which iterated one shot games are central examples), and that a more robust notion is subgame-perfect equilibrium [30]. Given a game $\mathscr{G}$, a strategy profile is a *subgame-perfect equilibrium (SPE)* if for every possible history of the game, the strategies are an NE. The following formula expresses that $\bar{x} \triangleq (\alpha_1, x_1) \dots (\alpha_n, x_n)$ is an SPE:

$$\phi_{SPE}(\bar{x}) \triangleq [[z_1, \dots, z_n]](\alpha_1, z_1) \dots (\alpha_n, z_n)\, \mathsf{G}\, \phi_{NE}(\bar{x})$$

Using graded modalities, we can thus express the uniqueness of a NE using the following GRADED-SL[NG] formula:

$$\langle\langle x_1, \dots, x_n \rangle\rangle^{\geq 1} \phi_{NE}(\bar{x}) \wedge \neg \langle\langle x_1, \dots, x_n \rangle\rangle^{\geq 2} \phi_{NE}(\bar{x})$$

By replacing ϕ_{NE} with ϕ_{SPE} in the formula above, we can express the uniqueness of a SPE.

4 The Model-checking procedure

In this section we study the model-checking problem for GRADED-SL and show that it is decidable with a time-complexity that is non-elementary (i.e., not bounded by any fixed tower of exponentials). However, it is elementary if the number of blocks of quantifiers is fixed. For the algorithmic procedures, we follow an *automata-theoretic approach* [20], reducing the decision problem for the logic to the emptiness problem of an automaton. The procedure we propose here extends that used for SL in [24]. The only case that is different is the new graded quantifier over tuples of strategies.

We start with the central notions of automata theory, and then show how to convert a GRADED-SL sentence φ into an automaton that accepts exactly the (tree encodings) of the concurrent game structures that satisfy φ. This is used to prove the main result about GRADED-SL model checking.

Automata Theory. A Σ-*labeled* Υ-*tree* T is a pair $\langle T, V \rangle$ where $T \subseteq \Upsilon^+$ is prefix-closed (i.e., if $t \in T$ and $s \in \Upsilon^+$ is a prefix of t then also $s \in T$), and $V : T \to \Sigma$ is a labeling function. Note that every word $w \in \Upsilon^+ \cup \Upsilon^\omega$ with the property that every prefix of w is in T, can be thought of as a path in T. Infinite paths are called *branches*. *Nondeterministic tree automata* (NTA) are a generalization to infinite trees of the classical automata on words [31]. *Alternating tree automata* (ATA) are a further generalization of nondeterministic tree automata [13]. Intuitively, on visiting a node of the input tree, while an NTA sends exactly one copy of itself to each of the successors of the node, an ATA can send several copies to the same successor. We use the parity acceptance condition [20]. *Alternating parity tree automata* (APT) are defined in Appendix B. The *language* $\mathrm{L}(\mathscr{A})$ of the APT $\mathscr{A}$ is the set of trees T accepted by $\mathscr{A}$. Two automata are *equivalent* if they have the same language. The *emptiness problem* for alternating parity tree-automata is to decide, given $\mathscr{A}$, whether $\mathrm{L}(\mathscr{A}) = \emptyset$. The *universality problem* is to decide whether $\mathscr{A}$ accepts all trees.

4.1 From Logic to Automata

Following an automata-theoretic approach, we reduce the model-checking problem of GRADED-SL to the emptiness problem for alternating parity tree automata [24]. The main step is to translate every GRADED-SL formula φ (i.e., φ may have free placeholders), concurrent-game structure $\mathscr{G}$, and state s, into an APT that accepts a tree if and only if the tree encodes an assignment χ such that $\mathscr{G}, \chi, s \models \varphi$.

We first describe the encoding, following [24]. Informally, the CGS $\mathscr{G}$ is encoded by its "tree-unwinding starting from s" whose nodes represent histories, i.e., the St-labeled St-tree $\mathsf{T} \triangleq \langle \mathrm{Hst}(s), \mathrm{u} \rangle$

such that $\mathsf{u}(h)$ is the last symbol of h. Then, every strategy $\chi(e)$ with $e \in \mathsf{free}(\varphi)$ is encoded as an Ac-labelled tree over the unwinding. The unwinding and these strategies $\chi(e)$ are viewed as a single $(\textsc{Val} \times \textsc{St})$-labeled tree where $\textsc{Val} \triangleq \mathsf{free}(\varphi) \to \textsc{Ac}$.

Definition 4.1 *The* encoding of χ (w.r.t. $\varphi, \mathcal{G}, s$) *is the* $(\textsc{Val} \times \textsc{St})$*-labeled* St*-tree* $\mathsf{T} \triangleq \langle \mathsf{T}, \mathsf{u} \rangle$ *such that* T *is the set of histories* h *of* $\mathcal{G}$ *starting with* s *and* $\mathsf{u}(h) \triangleq (\mathsf{f}, q)$ *where* q *is the last symbol in* h *and* $\mathsf{f} : \mathsf{free}(\varphi) \to \textsc{Ac}$ *is defined by* $\mathsf{f}(e) \triangleq \chi(e)(h)$ *for all* $e \in \mathsf{free}(\varphi)$.[1]

The following lemma is proved by induction on the structure of the formula φ, as in [24]. The idea for handling the new case, i.e., the graded quantifier $\langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \psi$, is to build an APT that is a projection of an APT that itself checks that each of the g tuples of strategies satisfies ψ and that each pair of g tuples is distinct.

Lemma 4.1 *For every* GRADED-SL *formula* φ*,* CGS $\mathcal{G}$*, and state* $s \in \textsc{St}$*, there exists an* APT $\mathcal{A}_\varphi$ *such that for all assignments* χ*, if* T *is the encoding of* χ *(w.r.t.* $\varphi, \mathcal{G}, s$*), then* $\mathcal{G}, \chi, s \models \varphi$ *iff* $\mathsf{T} \in \mathrm{L}(\mathcal{A}_\varphi)$.

We make some remarks about this lemma. First, all the cases in the induction incur a linear blowup except for the quantification case (recall that the translation from an APT to an NPT results in an exponentially larger automaton [20]). Thus, the size of the APT for φ is non-elementary in the quantifier-rank of φ. However, we can say a little more. Note that a block of k identical quantifiers only costs, in the worst case, a single exponential (and not k many exponentials) because we can extend the proof above to deal with a block of quantifiers at once. Thus, we get that the size of the APT for φ is non-elementary in the quantifier-block rank of φ.

Theorem 4.1 *The model-checking problem for* GRADED-SL *is* PTIME-COMPLETE *w.r.t. the size of the model and* $(k+1)$EXPTIME *if* $k \geq 1$ *is the quantifier-block rank of* φ*. Moreover, if* φ *is the form* $\wp\psi$*, where* $\wp$ *is a quantifier-block, and* ψ *is of quantifier-block rank* $k-1$*, then the complexity is* kEXPTIME.

Proof. The lower-bound w.r.t the size of the model already holds for SL [24]. For the upper bound, use Lemma 4.1 to transform the CGS and φ into an APT and test its emptiness. The complexity of checking emptiness (or indeed, universality) of an APT is in EXPTIME [20]. As discussed after Lemma 4.1, the size of the APT is a tower of exponentials whose height is the quantifier-block rank of φ. This gives the $(k+1)$EXPTIME upper bound. □

Theorem 4.2 *The model-checking problem for* GRADED-SL[NG] *is* PTIME-COMPLETE *w.r.t. the size of the model and* $(k+1)$-EXPTIME *when restricted to formulas of maximum alternation number* k.

Proof. The lower bound already holds for SL[NG] [24], and the upper bound is obtained by following the same reasoning for SL[NG] of the singleton existential quantifier [24] but using the automaton construction as in Theorem 4.1. □

Directly from the statements reported above, we get the following results:

Theorem 4.3 *Checking the uniqueness of NE, and checking the uniqueness of SPE, can be done in* 2EXPTIME.

We conclude this section with the complexity of the model checking problem for GRADED-SL[1G]. Also in this case one can derive the lower bound from the one holding for the corresponding sub-logic in SL (SL[1G]) and the upper bound by using the same algorithm for SL[1G] but plugging a (yet no more

[1] In case $\mathsf{free}(\varphi) = \emptyset$, then f is the (unique) empty function. In this case, the encoding of every χ may be viewed as the tree-unwinding from s.

complex) different automata construction for the new existential quantifier modality. Indeed the model checking problem for GRADED-SL[1G] is 2EXPTIME-COMPLETE. It is worth recalling that SL[1G] strictly subsumes ATL* [24]. It is quite immediate to see that this also holds in the graded setting (note that ATL* already allows quantifying over tuples of agents' (bound) strategies). As the model checking for ATL* is already 2EXPTIME-HARD, we get that also for the graded extension for this logic, which we name GATL*, the model checking problem is 2EXPTIME-COMPLETE. The model checking results for both GATL* and GRADED-SL[1G] are reported in the following theorem.

Theorem 4.4 *The model-checking problem for* GATL* *and* GRADED-SL[1G] *is* PTIME-COMPLETE w.r.t. *the size of the model and* 2-EXPTIME-COMPLETE *in the size of formula.*

5 Conclusion

In this paper we introduced GRADED-SL to address and solve the unique NE problem. We have demonstrated that GRADED-SL is elegant, simple, and very powerful, and can solve the unique NE problem for LTL objectives in 2EXPTIME, and thus at the same complexity that is required to merely decide if a NE exists. We also instantiate our formalism by considering the well-known prisoner's dilemma and its iterated version. We have also shown that using the same approach one can express (and solve) the uniqueness of other standard solution concepts, e.g., subgame-perfect equilibria, again in 2EXPTIME. Finally, our work gives the first algorithmic solution to the model-checking problem of a graded variant of ATL*, and proves it to be 2EXPTIME-COMPLETE.

The positive results presented in this paper open several directions for future work. First, one may add other types of graded quantifiers, i.e., "there exists infinitely many strategies" and "there exists countably many strategies", and still retain decidability using the results of [6]. Second, one may study the expressive power of SL with the atomic expression $x = y$ (saying that strategies x and y are the same), just as one does for first-order logic with and without equality. Third, one might extend fragments of GRADED-SL to quantitative objectives, such as mean-payoff or discounted-sum. Finally, one may implement GRADED-SL and its model-checking procedure in a formal verification tool such as SLK-MCMAS [10].

Acknowledgments

We thank Michael Wooldridge for suggesting uniqueness of Nash Equilibria as an application of graded strategy logic. Benjamin Aminof is supported by the Austrian National Research Network S11403-N23 (RiSE) of the Austrian Science Fund (FWF) and by the Vienna Science and Technology Fund (WWTF) through grant ICT12-059. Sasha Rubin is a Marie Curie fellow of the Istituto Nazionale di Alta Matematica. Aniello Murano is partially supported by the GNCS 2016 project: Logica, Automi e Giochi per Sistemi Auto-adattivi.

References

[1] E. Altman, H. Kameda & Y. Hosokawa (2002): *Nash Equilibria in Load Balancing in Distributed Computer Systems*. *IGTR* 4(2), pp. 91–100.

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

[3] R. Alur, S. La Torre & P. Madhusudan (2003): *Playing Games with Boxes and Diamonds*. In: *CONCUR 2003*, *LNCS 2761* 2761, Springer, pp. 127–141.

[4] B. Aminof, V. Malvone, A. Murano & S. Rubin (2016, (to appear)): *Graded Strategy Logic: Reasoning about Uniqueness of Nash Equilibria*. In: *AAMAS'16*, IFAAMAS.

[5] R. Axelrod (1987): *The evolution of strategies in the iterated prisoners dilemma*. The dynamics of norms, pp. 1–16.

[6] Vince Bárány, Lukasz Kaiser & Alexander Moshe Rabinovich (2010): *Expressing Cardinality Quantifiers in Monadic Second-Order Logic over Trees*. Fundam. Inform. 100(1-4), pp. 1–17.

[7] K. G. Binmore (1992): *Fun and Games: A Text on Game Theory*. D.C. Heath.

[8] P.A. Bonatti, C. Lutz, A. Murano & M.Y. Vardi (2008): *The Complexity of Enriched muCalculi. LMCS* 4(3), pp. 1–27.

[9] D. Calvanese, G. De Giacomo & M. Lenzerini (1999): *Reasoning in Expressive Description Logics with Fixpoints based on Automata on Infinite Trees*. In: *IJCAI 99*, pp. 84–89.

[10] P. Čermák, A. Lomuscio, F. Mogavero & A. Murano (2014): *MCMAS-SLK: A Model Checker for the Verification of Strategy Logic Specifications.* In: *CAV'14*, LNCS 8559, Springer, pp. 524–531.

[11] K. Chatterjee, T.A. Henzinger & N. Piterman (2010): *Strategy Logic. IC* 208(6), pp. 677–693.

[12] J. Bramel D. Simchi-Levi, X. Chen (2013): *The Logic of Logistics: Theory, Algorithms, and Applications for Logistics Management*. Science and Business Media, Springer.

[13] E. A. Emerson & C. S. Jutla (1991): *Tree Automata, Mu-Calculus and Determinacy (Extended Abstract)*. In: *ASFCS*, pp. 368–377.

[14] Clive D. Fraser (1992): *The uniqueness of Nash equilibrium in the private provision of public goods: an alternative proof.* Journal of Public Economics 49(3), pp. 389–390.

[15] A. Glazer & K. A. Konrad (1993): *Private Provision of Public Goods, Limited Tax Deducibility, and Crowding Out*. FinanzArchiv / Public Finance Analysis 50(2), pp. 203–216.

[16] Erich Grädel (1999): *On The Restraining Power of Guards*. J. Symb. Log. 64(4), pp. 1719–1742.

[17] J. Gutierrez, P. Harrenstein & M. Wooldridge (2014): *Reasoning about Equilibria in Game-Like Concurrent Systems*. In: *KR 14*.

[18] O. Kupferman, G. Perelli & M. Y. Vardi (2014): *Synthesis with Rational Environments*. In: *EUMAS 2014*, pp. 219–235.

[19] O. Kupferman, U. Sattler & M.Y. Vardi (2002): *The Complexity of the Graded muCalculus.* In: *CADE'02*, LNCS 2392, Springer, pp. 423–437.

[20] O. Kupferman, M.Y. Vardi & P. Wolper (2000): *An Automata Theoretic Approach to Branching-Time Model Checking. JACM* 47(2), pp. 312–360.

[21] K. Leyton-Brown & Y. Shoham (2008): *Essentials of Game Theory: A Concise, Multidisciplinary Introduction (Synthesis Lectures on Artificial Intelligence and Machine Learning)*. M&C.

[22] V. Malvone, F. Mogavero, A. Murano & L. Sorrentino (2015): *On the Counting of Strategies*. In: *TIME 15 (To appear)*.

[23] F. Mogavero, A. Murano, G. Perelli & M.Y. Vardi (2012): *What Makes* ATL* *Decidable? A Decidable Fragment of Strategy Logic.* In: *CONCUR'12*, LNCS 7454, Springer, pp. 193–208.

[24] F. Mogavero, A. Murano, G. Perelli & M.Y. Vardi (2014): *Reasoning About Strategies: On the Model-Checking Problem. TOCL* 15(4), pp. 34:1–42.

[25] F. Mogavero, A. Murano & M.Y. Vardi (2010): *Reasoning About Strategies.* In: *FSTTCS'10*, LIPIcs 8, Leibniz-Zentrum fuer Informatik, pp. 133–144.

[26] D. E. Muller & P. E. Schupp (1995): *Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of the Theorems of Rabin, McNaughton and Safra*. Theor. Comput. Sci. 141(1&2), pp. 69–107.

[27] A. Orda, R. Rom & N. Shimkin (1993): *Competitive routing in multiuser communication networks*. *IEEE/ACM Trans. Netw.* 1(5), pp. 510–521.

[28] A. Pnueli & R. Rosner (1990): *Distributed reactive systems are hard to synthesize*. In: *FOCS'90*, IEEE, pp. 746–757.

[29] Ariel Rubinstein (1986): *Finite automata play the repeated prisoner's dilemma*. *Journal of Economic Theory* 39(1), pp. 83–96.

[30] R. Selten (1965): *Spieltheoretische Behandlung eines Oligopolmodells mit Nachfragetragheit. Zeitschrift fur die gesamte Staatswissenschaft* 121, pp. 301–324.

[31] Wolfgang Thomas (1990): *Infinite Trees and Automaton Definable Relations over Omega-Words*, pp. 263–277.

[32] M. Ummels (2006): *Rational Behaviour and Strategy Construction in Infinite Multiplayer Games*. In: *FSTTCS*, pp. 212–223.

A Definition of Alternation Number

Definition A.1 *The* alternation number *of a* GRADED-SL[NG] *formula is given by:*

- $\mathsf{alt}(p) \triangleq 0$*, where* $p \in \mathrm{AP}$*;*
- $\mathsf{alt}(\mathsf{OP}\varphi) \triangleq \mathsf{alt}(\varphi)$*, where* $\mathsf{OP} \in \{\neg, \mathsf{X}, \flat\}$*;*
- $\mathsf{alt}(\varphi_1 \mathsf{OP} \varphi_2) \triangleq \mathsf{max}(\mathsf{alt}(\varphi_1), \mathsf{alt}(\varphi_2))$ *where* $\mathsf{OP} \in \{\vee, \mathsf{U}\}$*;*
- $\mathsf{alt}(\wp\varphi) \triangleq \mathsf{max}(\mathsf{alt}(\varphi), \mathsf{alt}(\wp))$ *where* $\wp$ *is a quantification prefix;*
- $\mathsf{alt}(\wp) \triangleq \sum_{i=1}^{|\wp|-1} \mathsf{switch}(\wp_i, \wp_{i+1})$*, where* $\mathsf{switch}(Q, Q') = 1$ *if* Q *and* Q' *are either both universal or both existential quantifiers, and* 0 *otherwise.*

B Alternating Parity Tree Automata (APT)

For a set X, let $\mathtt{B}^+(X)$ be the set of positive Boolean formulas over X, including the constants **true** and **false**. A set $Y \subseteq X$ satisfies a formula $\theta \in \mathtt{B}^+(X)$, written $Y \models \theta$, if assigning **true** to elements in Y and **false** to elements in $X \setminus Y$ makes θ true.

Definition B.1 *An Alternating Parity Tree-Automaton (*APT*) is a tuple* $\mathscr{A} \triangleq \langle \Sigma, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$*, where* Σ *is the* input alphabet*,* Δ *is a set of* directions*,* Q *is a finite set of* states*,* $q_0 \in \mathrm{Q}$ *is an* initial state*,* $\delta : \mathrm{Q} \times \Sigma \to \mathtt{B}^+(\Delta \times \mathrm{Q})$ *is an* alternating transition function*, and* $\aleph$*, an* acceptance condition*, is of the form* $(\mathrm{F}_1, \ldots, \mathrm{F}_k) \in (2^{\mathrm{Q}})^+$ *where* $\mathrm{F}_1 \subseteq \mathrm{F}_2 \ldots \subseteq \mathrm{F}_k = \mathrm{Q}$*.*

The set $\Delta \times \mathrm{Q}$ is called the set of *moves*. An NTA is an ATA in which each conjunction in the transition function δ has exactly one move (d, q) associated with each direction d.

An *input tree* for an APT is a Σ-labeled Δ-tree $\mathsf{T} = \langle \mathrm{T}, \mathsf{v} \rangle$. A *run* of an APT on an input tree $\mathsf{T} = \langle \mathrm{T}, \mathsf{v} \rangle$ is a $(\Delta \times \mathrm{Q})$-tree R such that, for all nodes $x \in \mathrm{R}$, where $x = (d_1, q_1) \ldots (d_n, q_n)$ (for some $n \in \mathbb{N}$), it holds that *(i)* $y \triangleq (d_1, \ldots, d_n) \in \mathrm{T}$ and *(ii)* there is a set of moves $\mathrm{S} \subseteq \Delta \times \mathrm{Q}$ with $\mathrm{S} \models \delta(q_n, \mathsf{v}(y))$ such that $x \cdot (d, q) \in \mathrm{R}$ for all $(d, q) \in \mathrm{S}$.

The acceptance condition allows us to say when a run is successful. Let R be a run of an APT $\mathscr{A}$ on an input tree T and $u \in (\Delta \times \mathrm{Q})^\omega$ one of its branches. Let $\mathsf{inf}(u) \subseteq \mathrm{Q}$ denote the set of states that occur in infinitely many moves of u. Say that u *satisfies the parity acceptance condition* $\aleph = (\mathrm{F}_1, \ldots, \mathrm{F}_k)$ if the least index $i \in [1, k]$ for which $\mathsf{inf}(u) \cap \mathrm{F}_i \neq \emptyset$ is even. A run is *successful* if all its branches satisfy the parity acceptance condition $\aleph$. An APT *accepts* an input tree T iff there exists a successful run R of $\mathscr{A}$ on T.

C Proof of Lemma 4.1.

As in [24] we induct on the structure of the formula φ to construct the corresponding automaton $\mathscr{A}_\varphi$. The Boolean operations are easily dealt with using the fact that disjunction corresponds to non-determinism, and negation corresponds to dualising the automaton. Note (†) that thus also conjunction is dealt with due to De Morgan's laws. The temporal operators are dealt with by following the unique play (determined by the given assignment) and verifying the required subformulas, e.g., for $\mathsf{X}\psi$ the automaton, after taking one step along the play, launches a copy of the automaton for ψ. All of these operations incur a linear blowup in the size of the automaton. The only case that differs from [24] is the quantification, i.e., we need to handle the case that $\varphi = \langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \psi$. Recall that $\mathscr{G}, \chi, s \models \langle\langle x_1, \ldots, x_n \rangle\rangle^{\geq g} \psi$ iff there exists g many tuples $\overline{\sigma_1}, \ldots, \overline{\sigma_g}$ of strategies such that: $\overline{\sigma_a} \neq \overline{\sigma_b}$ for $a \neq b$, and $\mathscr{G}, \chi[\bar{x} \mapsto \overline{\sigma_i}], s \models \psi$ for $1 \leq i \leq g$.

We show how to build an NPT for φ that mimics this definition: it will be a projection of an APT $\mathscr{D}_\psi$, which itself is the intersection of two automata, one checking that each of the g tuples of strategies satisfies ψ, and the other checking that each pair of the g tuples of strategies is distinct.

In more detail, introduce a set of fresh variables $X \triangleq \{x_i^j \in \mathrm{Vr} : i \leq n, j \leq g\}$, and consider the formulas ψ^j (for $j \leq g$) formed from ψ by renaming x_i (for $i \leq n$) to x_i^j. Define $\psi' \triangleq \wedge_{j \leq g} \psi^j$. Note that, by induction, each ψ^j has a corresponding APT, and thus, using the conjunction-case (†) above, there is an APT $\mathscr{B}$ for ψ'. Note that the input alphabet for $\mathscr{B}$ is $(\mathsf{free}(\psi') \to \mathrm{Ac}) \times \mathrm{St}$ and that $X \subseteq \mathsf{free}(\psi')$.

On the other hand, let $\mathscr{C}$ be an APT with input alphabet $(\mathsf{free}(\psi') \to \mathrm{Ac}) \times \mathrm{St}$ that accepts a tree $\mathsf{T} = \langle \mathrm{T}, \mathsf{v} \rangle$ if and only if for every $a \neq b \leq g$ there exists $i \leq n$ and $h \in \mathrm{T}$ such that $\mathsf{v}(h) = (f, q)$ and $f(x_i^a) \neq f(x_i^b)$.

Form the APT $\mathscr{D}_\psi$ for the intersection of $\mathscr{B}$ and $\mathscr{C}$.

Now, using the classic transformation [26], we remove alternation from the APT $\mathscr{D}_\psi$ to get an equivalent NPT $\mathscr{N}$ (note that this step costs an exponential). Finally, use the fact that NPTs are closed under projection (with no blowup) to get an NPT for the language $proj_X(L(\mathscr{N}))$ of trees that encode assignments χ satisfying φ.

For completeness we recall this last step. If L is a language of Σ-labeled trees with $\Sigma \triangleq \mathrm{A} \to \mathrm{B}$, and $X \subset \mathrm{A}$, then the X-*projection of* L, written $proj_{\mathrm{X}}(L)$, is the language of Σ'-labeled trees with $\Sigma' \triangleq \mathrm{A} \setminus \mathrm{X} \to \mathrm{B}$ such that $\mathsf{T} \triangleq \langle \mathrm{T}, \mathsf{v} \rangle \in proj_{\mathrm{X}}(L)$ if and only if there exists an X-labeled tree $\langle \mathrm{T}, \mathsf{w} \rangle$ such that the language L contains the tree $\langle \mathrm{T}, \mathsf{u} \rangle$ where $\mathsf{u} : \mathrm{T} \to (\mathrm{A} \to \mathrm{B})$ maps $t \in \mathrm{T}$ to $\mathsf{v}(t) \cup \mathsf{w}(t)$. Now, if $\mathscr{N}$ is an NPT with input alphabet $\Sigma \triangleq \mathrm{A} \to \mathrm{B}$, and if $\mathrm{X} \subset \mathrm{A}$, then there is an NPT with input alphabet $\Sigma' \triangleq \mathrm{A} \setminus \mathrm{X} \to \mathrm{B}$ with language $proj_{\mathrm{X}}(L(\mathscr{N}))$.

The proof that the construction is correct is immediate. □