Assume-Admissible Synthesis*

Romain Brenguier, Jean-François Raskin, and Ocan Sankur

Université Libre de Bruxelles, Brussels, Belgium

Abstract

In this paper, we introduce a novel rule for synthesis of reactive systems, applicable to systems made of n components which have each their own objectives. It is based on the notion of *admissible* strategies. We compare our novel rule with previous rules defined in the literature, and we show that contrary to the previous proposals, our rule define sets of solutions which are *rectangular*. This property leads to solutions which are robust and resilient. We provide algorithms with optimal complexity and also an abstraction framework.

1998 ACM Subject Classification D.2.4 Software/Program verification, F.3.1 Specifying and Verifying and Reasoning about Programs, I.2.2 Program synthesis

Keywords and phrases Multi-player games, controller synthesis, admissibility

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2015.100

1 Introduction

The automatic synthesis of reactive systems has recently attracted a considerable attention. The theoretical foundations of most of the contributions in this area rely on two-player zero sum games played on graphs: one player (player 1) models the system to synthesize, and the other player (player 2) models its environment. The game is zero-sum: the objective of player 1 is to enforce the specification of the system while the objective of player 2 is the *negation* of this specification. This is a *worst-case* assumption: because the cooperation of the environment cannot be assumed, we postulate that it is *antagonistic*.

A fully adversarial environment is usually a bold abstraction of reality. Nevertheless, it is popular because it is *simple* and *sound*: a *winning strategy* against an antagonistic player is winning against *any* environment which pursues its own objective. But this approach may fail to find a winning strategy even if there exist solutions when the objective of the environment is taken into account. Also, this model is for two players only: system *vs* environment. In practice, both the system and the environment may be composed of several parts to be constructed individually or whose objectives should be considered one at a time. It is thus crucial to take into account different players' objectives when synthesizing strategies; accordingly, alternative notions have been proposed in the literature.

A first classical alternative is to *weaken* the winning condition of player 1 using the objective of the environment, requiring the system to win only when the environment meets its objective. This approach together with its weaknesses have been discussed in [3], we will add to that later in the paper. A second alternative is to use concepts from n-players *non*-zero sum games. This is the approach taken both by *assume-guarantee synthesis* [6] (AG), and by *rational synthesis* [16] (RS). AG relies on *secure equilibria* [8] (SE), a refinement of Nash equilibria [23] (NE). In SE, objectives are lexicographic: players first try to maximize their own specifications, and then try to falsify the specifications of others. It is shown in [8]

* Supported by the ERC starting grant inVEST (FP7-279499).

that SE are those NE which represent enforceable contracts between the two players. In RS, the system is assumed to be monolithic and the environment is made of components that are *partially controllable*. In RS, we search for a profile of strategies where the system ensures its objective and the players that model the environment are given an "*acceptable*" strategy profiles, from which it is assumed that they will not deviate. "Acceptable" is formalized either by NE, dominating strategies (Dom), or subgame perfect equilibria (SPE).

Contributions. As a first and central contribution, we propose a novel notion of synthesis where we take into account different players' objectives using the concept of *admissible* strategies [1, 2, 4]. For a player with objective ϕ, a strategy σ is *dominated* by σ' if σ' does as well as σ w.r.t. ϕ against all strategies of the other players, and better for some of those strategies. A strategy σ is *admissible* if it is *not* dominated by another strategy. In [2], the admissibility notion was lifted to games played on graphs, and algorithmic questions left open were solved in [4], with the goal of *model checking* the set of runs that survive the iterative elimination of dominated strategies. Here, we use this notion to derive a meaningful notion to *synthesize* systems with several players, with the following idea. *Rational* players should only play admissible strategies since dominated strategies are clearly *suboptimal*. In *assume-admissible synthesis* (AA), we make the assumption that players play admissible strategies. Then for each player, we search for an admissible strategy that is *winning* against all admissible strategies of other players. AA is *sound*: any strategy profile that is winning against admissible strategies of other players, satisfies the objectives of all the players (Theorem 1).

As a second contribution, we compare the different synthesis rules. First we apply all the rules on a simple but representative example, and show the main advantages of AA w.r.t. the other rules. Then we compare systematically the different approaches. We show when a solution for one rule implies a solution for another rule and we prove that, contrary to other rules, AA yields rectangular sets of solutions (Theorem 4). We argue that the rectangularity property is essential for practical applications. As a third contribution, we provide algorithms to decide the existence of assume-admissible winning strategy profiles and prove the optimal complexity of our algorithm (Theorem 8): PSPACE-complete for Müller, and PTIME for Büchi objectives. As a last important contribution, we provide an abstraction framework which allows us to define sufficient conditions to compute sets of winning assume-admissible strategies for each player in the game compositionally (Theorem 12).

Additional pointers to related works. We have already mentioned assume-guarantee synthesis [6] and rational synthesis [16, 20]. Those are the closest related works to ours as they pursue the same scientific objective: to synthesis strategy profiles for non-zero sum multi-player games by taking into account the specification of each player. As those works are defined for similar formal setting, we are able to provide formal statements in the core of the paper that add elements of comparison with our work.

In [15], Faella studies several alternatives to the notion of winning strategy including the notion of admissible strategy. His work is for two-players only, and only the objective of one player is taken into account, the objective of the other player is left unspecified. Faella uses the notion of admissibility to define a notion of *best-effort* in synthesis while we use the notion of admissibility to take into account the objectives of the other players in an n player setting where each player has his own objective.

The notion of admissible strategy is definable in strategy logics [9, 22] and decision problems related to the AA rule can be reduced to satisfiability queries in such logics.

Nevertheless this would not lead to worst-case optimal algorithms. Based on our previous work [4], we develop in this paper worst-case optimal algorithms.

In [12], Damm and Finkbeiner use the notion of *dominant strategy* to provide a compositional semi-algorithm for the (undecidable) distributed synthesis problem. So while we use the notion of admissible strategy, they use a notion of dominant strategy. The notion of dominant strategy is *strictly stronger*: every dominant strategy is admissible but an admissible strategy is not necessary dominant. Also, in multiplayer games with omega-regular objectives with complete information (as considered here), admissible strategies are always guaranteed to exist [2] while it is not the case for dominant strategies. We will show in an example that the notion of dominant strategy is too strong for our purpose. Also, note that the objective of Damm and Finkbeiner is different from ours: they use dominance as a mean to formalize a notion of *best-effort* for components of a distributed system w.r.t. their common objective, while we use admissibility to take into account the objectives of the other components when looking for a winning strategy for one component to enforce its own objective. Additionally, our formal setting is different from their setting in several respects. First, they consider zero-sum games between a distributed team of players (processes) against a unique environment, each player in the team has the same specification (the specification of the distributed system to synthesize) while the environment is considered as adversarial and so its specification is the negation of the specification of the system. In our case, each player has his *own* objective and we do not distinguish between protagonist and antagonist players. Second, they consider distributed synthesis: each individual process has its own view of the system while we consider games with perfect information in which all players have a complete view of the system state. Finally, let us point out that Damm and Finkbeiner use the term *admissible* for specifications and *not* for strategies (as already said, they indeed consider dominant strategies and not admissible strategies). In our case, we use the notion of *admissible* strategy which is classical in game theory, see e.g. [17, 1]. This vocabulary mismatch is unfortunate but we decided to stick to the term of "admissible strategy" which is well accepted in the literature, and already used in several previous works on (multi-player) games played on graphs [2, 15, 4].

Structure of the paper. Sect. 2 contains definitions. In Sect. 3, we review synthesis rules introduced in the literature and define assume-admissible synthesis. In Sect. 4, we consider an example; this allows us to underline some weaknesses of the previous rules. Sect. 5 presents a formal comparison of the different rules. Sect. 6 contains algorithms for Büchi and Müller objectives, and Sect. 7 abstraction techniques applied to our rule.

2 Definitions

A *turn-based multiplayer arena* is a tuple $\mathsf{A} = \langle \mathcal{P}, (\mathsf{S}_i)_{i\in\mathcal{P}}, s_{\mathsf{init}}, (\mathsf{Act}_i)_{i\in\mathcal{P}}, \delta\rangle$ where $\mathcal{P}$ is a finite set of players; for $i \in \mathcal{P}$, S_i is a finite set of player-i states; we let $\mathsf{S} = \biguplus_{i\in\mathcal{P}} \mathsf{S}_i$; $s_{\mathsf{init}} \in \mathsf{S}$ is the initial state; for every $i \in \mathcal{P}$, Act_i is the set of player-i actions; we let $\mathsf{Act} = \bigcup_{i\in\mathcal{P}} \mathsf{Act}_i$; and $\delta\colon \mathsf{S} \times \mathsf{Act} \mapsto \mathsf{S}$ is the transition function. A *run* ρ is a sequence of alternating states and actions $\rho = s_1a_1s_2a_2\ldots \in (\mathsf{S}\cdot\mathsf{Act})^\omega$ such that for all $i \geq 1$, $\delta(s_i, a_i) = s_{i+1}$. We write $\rho_i = s_i$, and $\mathsf{act}_i(\rho) = a_i$. A *history* is a finite prefix of a run ending in a state. We denote by $\rho_{\leq k}$ the history $s_1a_1\ldots s_k$; and write $\mathsf{last}(\rho_{\leq k}) = s_k$, the last state of the history. The set of states *occurring infinitely often* in a run ρ is $\mathrm{Inf}(\rho) = \{s \in \mathsf{S} \mid \forall j \in \mathbb{N}.\ \exists i > j, \rho_i = s\}$.

A *strategy* of player i is a function $\sigma_i : (\mathsf{S}^*\cdot\mathsf{S}_i) \to \mathsf{Act}_i$. A *strategy profile* for the set of players $P \subseteq \mathcal{P}$ is a tuple of strategies, one for each player of P. We write $-i$ for the set

$\mathcal{P} \setminus \{i\}$. Let $\Sigma_i(\mathsf{A})$ be the set of the strategies of player i in A, written Σ_i if A is clear from context, and Σ_P the strategy profiles of $P \subseteq \mathcal{P}$.

A run ρ is *compatible* with strategy σ for player i if for all $j \geq 1$, $\rho_j \in \mathsf{S}_i$ and $\mathsf{act}_j(\rho) = \sigma(\rho_{\leq j})$. It is compatible with strategy profile $\sigma_{\mathcal{P}}$ if it is compatible with each σ_i for $i \in \mathcal{P}$. The *outcome* of a strategy profile $\sigma_{\mathcal{P}}$ is the unique run compatible with $\sigma_{\mathcal{P}}$ starting at s_{init}, denoted $\mathsf{Out}_{\mathsf{A}}(\sigma_{\mathcal{P}})$. We write $\mathsf{Out}_{\mathsf{A},s}(\sigma_{\mathcal{P}})$ for the outcome starting at state s. Given $\sigma_P \in \Sigma_P$ with $P \subseteq \mathcal{P}$, let $\mathsf{Out}_{\mathsf{A}}(\sigma_P)$ denote the set of runs compatible with σ_P, and extend it to $\mathsf{Out}_{\mathsf{A}}(\Sigma')$ where Σ' is a set of strategy profiles. For $E \subseteq \mathsf{S}_i \times \mathsf{Act}_i$, let $\mathsf{Strat}_i(E)$ denote the set of player-i strategies σ that only use actions in E in all outcomes compatible with σ.

An *objective* ϕ is a subset of runs. A strategy σ_i of player i is *winning* for objective ϕ_i if for all $\sigma_{-i} \in \Sigma_{-i}$, $\mathsf{Out}_{\mathsf{A}}(\sigma_i, \sigma_{-i}) \in \phi_i$. A *game* is an arena equipped with an objective for each player, written $\mathsf{G} = \langle \mathsf{A}, (\phi_i)_{i \in \mathcal{P}} \rangle$ where for each player i, ϕ_i is an objective. Given a strategy profile σ_P for the set of players P, we write $\mathsf{G}, \sigma_P \models \phi$ if $\mathsf{Out}_{\mathsf{A}}(\sigma_P) \subseteq \phi$. We write $\mathsf{Out}_{\mathsf{G}}(\sigma_P) = \mathsf{Out}_{\mathsf{A}}(\sigma_P)$, and $\mathsf{Out}_{\mathsf{G}} = \mathsf{Out}_{\mathsf{G}}(\Sigma)$. For any coalition $C \subseteq \mathcal{P}$, and objective ϕ, we denote by $\mathsf{Win}_C(\mathsf{A}, \phi)$ the set of states s such that there exists $\sigma_C \in \Sigma_C$ with $\mathsf{Out}_{\mathsf{G},s}(\sigma_C) \subseteq \phi$.

Although we prove some of our results for general objectives, we give algorithms for ω-regular objectives represented by Muller conditions. A Muller condition is given by a family $\mathcal{F}$ of sets of states: $\phi_i = \{\rho \mid \mathrm{Inf}(\rho) \in \mathcal{F}\}$. Following [19], we assume that $\mathcal{F}$ is given by a Boolean circuit whose inputs are S, which evaluates to true exactly on valuations encoding subsets $S \in \mathcal{F}$. We also use linear temporal logic (LTL) [24] to describe objectives. LTL formulas are defined by $\phi := \mathtt{G}\phi \mid \mathtt{F}\phi \mid \mathtt{X}\phi \mid \phi \mathtt{U} \phi \mid \phi \mathtt{W} \phi \mid S$ where $S \subseteq \mathsf{S}$ (We refer to [14] for the semantics.) We consider the special case of Büchi objectives, given by $\mathtt{GF}(B) = \{\rho \mid B \cap \mathrm{Inf}(\rho) \neq \varnothing\}$. Boolean combinations of formulas $\mathtt{GF}(S)$ define Muller conditions representable by polynomial-size circuits.

In any game G, a player i strategy σ_i is *dominated* by σ_i' if for all $\sigma_{-i} \in \Sigma_{-i}$, $\mathsf{G}, \sigma_i, \sigma_{-i} \models \phi_i$ implies $\mathsf{G}, \sigma_i', \sigma_{-i} \models \phi_i$ and there exists $\sigma_{-i} \in \Sigma_{-i}$, such that $\mathsf{G}, \sigma_i', \sigma_{-i} \models \phi_i$ and $\mathsf{G}, \sigma_i, \sigma_{-i} \not\models \phi_i$, (this is classically called *weak* dominance, but we call it dominance for simplicity). A strategy which is not dominated is *admissible*. Thus, admissible strategies are maximal, and incomparable, with respect to the dominance relation. We write $\mathsf{Adm}_i(\mathsf{G})$ for the set of *admissible* strategies in Σ_i, and $\mathsf{Adm}_P(\mathsf{G}) = \prod_{i \in P} \mathsf{Adm}_i(G)$ the product of the sets of admissible strategies for $P \subseteq \mathcal{P}$.

Strategy σ_i is *dominant* if for all σ_i', and σ_{-i}, $\mathsf{G}, \sigma_i', \sigma_{-i} \phi_i$ implies $\mathsf{G}, \sigma_i, \sigma_{-i} \models \phi_i$. The set of dominant strategies for player i is written $\mathsf{Dom}_i(\mathsf{G})$. A *Nash equilibrium* for G is a strategy profile $\sigma_{\mathcal{P}}$ such that for all $i \in \mathcal{P}$, and $\sigma_i' \in \Sigma_i$, $\mathsf{G}, \sigma_{-i}, \sigma_i' \models \phi_i$ implies $\mathsf{G}, \sigma_{\mathcal{P}} \models \phi_i$; thus no player can improve its outcome by deviating from the prescribed strategy. A Nash equilibrium for G from s, is a Nash equilibrium for G where the initial state is replaced by s. A *subgame-perfect equilibrium* for G is a strategy profile $\sigma_{\mathcal{P}}$ such that for all histories h, $(\sigma_i \circ h)_{i \in \mathcal{P}}$ is a Nash equilibrium in G from state $\mathsf{last}(h)$, where given a strategy σ, $\sigma \circ h$ denotes the strategy $\mathsf{last}(h) \cdot h' \mapsto \sigma(h \cdot h')$.

3 Synthesis Rules

In this section, we review synthesis rules proposed in the literature, and introduce a novel one: the *assume-admissible* synthesis rule (AA). Unless stated otherwise, we fix for this section a game G, with players $\mathcal{P} = \{1, \ldots, n\}$ and their objectives $\phi_1, \ldots, \phi_n$.

Rule Coop. The objectives are *achieved cooperatively* if there is a strategy profile $\sigma_{\mathcal{P}} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ such that $\mathsf{G}, \sigma_{\mathcal{P}} \models \bigwedge_{i \in \mathcal{P}} \phi_i$.

This rule [21, 10] asks for a strategy profile that *jointly* satisfies the objectives of all the players. This rule makes *very strong assumptions*: players fully cooperate and strictly follow their respective strategies. This concept is *not robust* against deviations and postulates that the behavior of every component in the system is *controllable*. This weakness is well-known: see e.g. [6] where the rule is called *weak co-synthesis*.

Rule Win**.** The objectives are *achieved adversarially* if there is a strategy profile $\sigma_{\mathcal{P}} = (\sigma_1, \ldots, \sigma_n)$ such that for all $i \in \mathcal{P}$, $\mathsf{G}, \sigma_i \models \phi_i$.

This rule does *not* require any cooperation among players: the rule asks to synthesize for each player i a strategy which enforces his/her objective ϕ_i against all possible strategies of the other players. Strategy profiles obtained by Win are extremely *robust*: each player is able to ensure his/her objective no matter how the other players behave. Unfortunately, this rule is often not applicable in practice: often, none of the players has a winning strategy against *all* possible strategies of the other players. The next rules soften this requirement by taking into account the objectives of other players.

Rule Win-under-Hyp**.** Given a two-player game G with $\mathcal{P} = \{1, 2\}$ in which player 1 has objective ϕ_1, player 2 has objective ϕ_2, player 1 can *achieve adversarially* ϕ_1 under *hypothesis* ϕ_2, if there is a strategy σ_1 for player 1 such that $\mathsf{G}, \sigma_1 \models \phi_2 \rightarrow \phi_1$.

The rule *winning under hypothesis* applies for two-player games only. Here, we consider the synthesis of a strategy for player 1 against player 2 under the hypothesis that player 2 behaves according to his/her specification. This rule is a relaxation of the rule Win as player 1 is *only* expected to win when player 2 plays so that the outcome of the game satisfies ϕ_2. While this rule is often reasonable, it is *fundamentally* plagued by the following problem: instead of trying to satisfy ϕ_1, player 1 could try to falsify ϕ_2, see e.g. [3]. This problem disappears if player 2 has a winning strategy to enforce ϕ_2, and the rule is then safe. We come back to that later in the paper (see Lemma 1).

Chatterjee et al. in [6] proposed synthesis rules inspired by Win-under-Hyp but avoid the aforementioned problem. The rule was originally proposed in a model with two components and a scheduler. We study here two natural extensions for n players.

Rules $\mathsf{AG}^{\wedge}$ **and** $\mathsf{AG}^{\vee}$**.** The objectives are achieved by

($\mathsf{AG}^{\wedge}$**)** *assume-guarantee-*$\wedge$ if there exists a strategy profile $\sigma_{\mathcal{P}}$ such that

1. $\mathsf{G}, \sigma_{\mathcal{P}} \models \bigwedge_{i \in \mathcal{P}} \phi_i$,
2. for all players i, $\mathsf{G}, \sigma_i \models (\bigwedge_{j \in \mathcal{P} \setminus \{i\}} \phi_j) \Rightarrow \phi_i$.

($\mathsf{AG}^{\vee}$**)** *assume-guarantee-*$\vee$[1] if there exists a strategy profile $\sigma_{\mathcal{P}}$ such that

1. $\mathsf{G}, \sigma_{\mathcal{P}} \models \bigwedge_{i \in \mathcal{P}} \phi_i$,
2. for all players i, $\mathsf{G}, \sigma_i \models (\bigvee_{j \in \mathcal{P} \setminus \{i\}} \phi_j) \Rightarrow \phi_i$.

The two rules differ in the second requirement: $\mathsf{AG}^{\wedge}$ requires that player i wins whenever *all* the other players win, while $\mathsf{AG}^{\vee}$ requires player i to win whenever *one* of the other player wins. Clearly $\mathsf{AG}^{\vee}$ is stronger, and the two rules are equivalent for two-player games. As shown in [8], for two-player games, a profile of strategy for $\mathsf{AG}^{\wedge}$ (or $\mathsf{AG}^{\vee}$) is a Nash equilibrium in a derived game where players want, in lexicographic order, first to satisfy

[1] This rule was introduced in [5], under the name Doomsday equilibria, as a generalization of the AG rule of [6] to the case of n-players.

their own objectives, and then as a secondary objective, want to falsify the objectives of the other players. As NE, $\mathsf{AG}^\wedge$ and $\mathsf{AG}^\vee$ require players to *synchronize* on a particular strategy profiles. As we will see, this is not the case for the new rule that we propose.

[16] and [20] introduce two versions of *rational synthesis* (RS). In the two cases, one of the player, say player 1, models the system while the other players model the environment. The existential version ($\mathsf{RS}^\exists$) searches for a strategy for the system, and a profile of strategies for the environment, such that the objective of the system is satisfied, and the profile for the environment is *stable* according to a solution concept which is either NE, SPE, or Dom. The universal version ($\mathsf{RS}^\forall$) searches for a strategy for the system, such that for all environment strategy profiles that are *stable* according to the solution concept, the objective of the system holds. We write $\Sigma^{\mathsf{NE}}_{G,\sigma_1}$, resp. $\Sigma^{\mathsf{SPE}}_{\mathsf{G},\sigma_1}$, for the set of strategy profiles $\sigma_{-1} = (\sigma_2, \sigma_3, \dots, \sigma_n)$ that are NE (resp. SPE) equilibria in the game G when player 1 plays σ_1, and $\Sigma^{\mathsf{Dom}}_{G,\sigma_1}$ for the set of strategy profiles σ_{-1} where each strategy σ_j, $2 \leq j \leq n$, is dominant in the game G when player 1 plays σ_1.

Rules $\mathsf{RS}^{\exists,\forall}(\mathsf{NE},\mathsf{SPE},\mathsf{Dom})$ **.** Let $\gamma \in \{\mathsf{NE},\mathsf{SPE},\mathsf{Dom}\}$, the objective is achieved by:
($\mathsf{RS}^\exists(\gamma)$) existential rational synthesis under γ if there is a strategy σ_1 of player 1, and a profile $\sigma_{-1} \in \Sigma^\gamma_{\mathsf{G},\sigma_1}$, such that $\mathsf{G}, \sigma_1, \sigma_{-1} \models \phi_1$.
($\mathsf{RS}^\forall(\gamma)$) universal rational synthesis under γ if there is a strategy σ_1 of player 1, such that $\Sigma^\gamma_{\mathsf{G},\sigma_1} \neq \emptyset$, and for all $\sigma_{-1} \in \Sigma^\gamma_{\mathsf{G},\sigma_1}$, $\mathsf{G}, \sigma_1, \sigma_{-1} \models \phi_1$.

Clearly, ($\mathsf{RS}^\forall(\gamma)$) is stronger than ($\mathsf{RS}^\exists(\gamma)$) and more robust. As $\mathsf{RS}^{\exists,\forall}(\mathsf{NE},\mathsf{SPE})$ are derived from NE and SPE, they require players to synchronize on particular strategy profiles.

Novel rule. We now present our novel rule based on the notion of *admissible strategies*.

Rule AA**.** The objectives are achieved by *assume-admissible (*AA*) strategies* if there is a strategy profile $\sigma_\mathcal{P}$ such that:
1. for all $i \in \mathcal{P}$, $\sigma_i \in \mathsf{Adm}_i(\mathsf{G})$;
2. for all $i \in \mathcal{P}$, $\forall \sigma'_{-i} \in \mathsf{Adm}_{-i}(\mathsf{G})$. $\mathsf{G}, \sigma'_{-i}, \sigma_i \models \phi_i$.

A player-i strategy satisfying conditions 1 and 2 above is called *assume-admissible-winning* (AA-*winning*). A profile of AA-winning strategies is an AA-*winning strategy profile*. The rule AA requires that each player has a strategy *winning* against *admissible* strategies of other players. So we assume that players do not play strategies which are *dominated*, which is reasonable as dominated strategies are clearly *suboptimal options*.

Contrary to Coop, $\mathsf{AG}^\wedge$, and $\mathsf{AG}^\vee$, AA does not require that the strategy profile is winning for each player. As for Win, this is a consequence of the definition:

▶ **Theorem 1.** *For all* AA*-winning strategy profile* $\sigma_\mathcal{P}$, $\mathsf{G}, \sigma_\mathcal{P} \models \bigwedge_{i \in \mathcal{P}} \phi_i$.

The condition that AA strategies are admissible is necessary for Thm. 1; it does not suffice to have strategies that are winning against admissible strategies.

4 Synthesis Rules at the Light of an Example

We illustrate the synthesis rules on an example of a real-time scheduler with two tasks. The system is composed of Sched (player 1) and Env (player 2). Env chooses the truth value for r_1, r_2 (r_i is a request for task i), and Sched controls q_1, q_2 (q_i means that task i has been scheduled). Our model is a turn-based game: first, Env chooses a value for r_1, r_2, then in

the next round Sched chooses a value for q_1, q_2, and we repeat forever. The requirements for Sched and Env are as follows:

1. Sched is not allowed to schedule the two tasks at the same time. When r_1 is true, then task 1 must be scheduled (q_1) *within* three rounds. When r_2 is true, task 2 must be scheduled (q_2) in *exactly* three rounds.
2. Whenever Env issues r_i then it does not issue this request again before the occurrence of the grant q_i. Env issues infinitely many requests r_1 and r_2.

We say that a request r_i is *pending* whenever the corresponding grant has not yet been issued. Those requirements can be expressed in LTL as follows:

- $\phi_{\mathsf{Sched}} = \mathtt{G}(r_1 \rightarrow \mathtt{X}q_1 \vee \mathtt{XXX}q_1) \wedge \mathtt{G}(r_2 \rightarrow \mathtt{XXX}q_2) \wedge \mathtt{G}\neg(q_1 \wedge q_2)$.
- $\phi_{\mathsf{Env}} = \mathtt{G}(r_1 \rightarrow \mathtt{X}(\neg r_1 \mathtt{W} q_1)) \wedge \mathtt{G}(r_2 \rightarrow \mathtt{X}(\neg r_2 \mathtt{W} q_2)) \wedge (\mathtt{GF}r_1) \wedge (\mathtt{GF}r_2)$.

A solution compatible with the previous rules in the literature. First, we note that there is no winning strategy neither for Sched, nor for Env. In fact, first let $\hat{\sigma_1}$ be the strategy of Sched that never schedules any of the two tasks, i.e. leaves q_1 and q_2 constantly false. This is clearly forcing $\neg\phi_{\mathsf{Env}}$ against all strategies of Env. Second, let $\hat{\sigma_2}$ be s.t. Env always requests the scheduling of both task 1 and task 2, i.e. r_1 and r_2 are constantly true. It is easy to see that this enforces $\neg\phi_{\mathsf{Sched}}$ against any strategy of Sched. So, there is no solution with rule Win[2]. But clearly those strategies are also not compatible with the objectives of the respective players, so this leaves the possibility to apply successfully the other rules. We now consider a strategy profile which is a solution for all the rules except for AA.

Let (σ_1, σ_2) be strategies for player 1 and 2 respectively, such that the outcome of (σ_1, σ_2) is "Env emits r_1, then Sched emits q_1, Env emits r_2, then Sched waits one round and emits q_2, and repeat." If a deviation from this *exact* execution is observed, then the two players switch to strategies $\hat{\sigma_1}$ and $\hat{\sigma_2}$ respectively, i.e. to the strategies that falsify the specification of the other players. The reader can now convince himself/herself that (σ_1, σ_2) is a solution for Coop, AG and $\mathsf{RS}^{\exists}(\mathsf{NE}, \mathsf{SPE}, \mathsf{Dom})$. Furthermore, we claim that σ_1 is a solution for Win-under-Hyp and $\mathsf{RS}^{\forall}(\mathsf{NE}, \mathsf{SPE}, \mathsf{Dom})$. But, assume now that Env is a device driver which requests the scheduling of tasks by the scheduler of the kernel of an OS when the device that it supervised requires it. Clearly (σ_1, σ_2), which is compatible with all the previous rules (but Win), makes little sense in this context. On the other hand, ϕ_{Sched} and ϕ_{Env} are natural specifications for such a system. So, there is clearly room for other synthesis rules!

Solutions provided by AA, our novel rule. For Env, we claim that the set of *admissible strategies*, noted $\mathsf{Adm}(\phi_{\mathsf{Env}})$, are exactly those that (*i*) do not emit a new request before the previous one has been acknowledged, and (*ii*) do always eventually emit a (new) request when the previous one has been granted. Indeed as we have seen above, Env and Sched can cooperate to satisfy $\phi_{\mathsf{Sched}} \wedge \phi_{\mathsf{Env}}$, so any strategy of Env which would imply the falsification of ϕ_{Env} is dominated and so it is not admissible. Also, we have seen that Env does not have a winning strategy for ϕ_{Env}, so Env cannot do better.

Now, let us consider the following strategy for Sched. (*i*) if pending requests r_1 and r_2 were made one round ago, then grant q_1; if pending requests r_1 and r_2 were made three rounds ago, then behave arbitrarily (it is no more possible to satisfy the specification); (*ii*) if pending request r_2 was made three rounds ago, but not r_1, then grant q_2; (*iii*) if pending r_1 was

[2] Also, it is easy to see that Env does not have a dominant strategy for his specification. So, considering dominant strategies as *best-effort strategies* would not lead to a solution for this example. To find a solution, we need to take into account the objectives of the other players.

made three rounds ago, but not r_2, then grant q_1. We claim that this strategy is *admissible* and while it is *not* winning against *all* possible strategies of Env, it is *winning* against *all admissible* strategies of Env. So, this strategy enforces ϕ_{Sched} against all reasonable strategies of Env w.r.t. to his/her own objective ϕ_{Env}. In fact, there is a whole set of such strategies for Sched, noted $\mathsf{WinAdm}_{\mathsf{Sched}}$. Similarly, there is a whole set of strategies for Env which are both *admissible* and winning against the *admissible* strategies of Sched, noted $\mathsf{WinAdm}_{\mathsf{Env}}$. We prove in the next section that the solutions to AA are *rectangular sets*: they are exactly the solutions in $\mathsf{WinAdm}_{\mathsf{Sched}} \times \mathsf{WinAdm}_{\mathsf{Env}}$. This ensures that AA leads to *resilient* solutions: players do not need to synchronize with the other players on a particular strategy profile but they can arbitrarily choose inside their sets of strategies that are admissible and winning against the admissible strategies of the other players.

5 Comparison of Synthesis Rules

In this section, we compare the synthesis rules to understand which ones yield solutions more often, and to assess their robustness. Some relations are easy to establish; for instance, rules $\mathsf{Win}, \mathsf{AG}^{\vee}, \mathsf{AG}^{\wedge}, \mathsf{AA}$ imply Coop by definition (and Thm. 1). We summarize the implication relations between the rules in Fig. 1. We present the rules $\mathsf{AG}^{\vee}, \mathsf{AG}^{\wedge}$, and the variants of $\mathsf{RS}^{\cdot}(\cdot)$ in one group, respectively. A dashed arrow from A to B means that rule A implies *some* rule in B; while a plain arrow means that A implies *all* rules in B (*e.g.* AA implies $\mathsf{AG}^{\wedge}$ but not $\mathsf{AG}^{\vee}$; while Win implies both rules.) An absence of path means that A does not imply any variant of B. Thus the figure explains which approaches yield solutions more often, by abstracting away the precise variants. The following theorem states the correctness of our diagram.

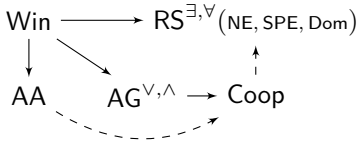

Figure 1 Comparison of synthesis rules.

▶ **Theorem 2.** *The implication relations of Fig. 1 hold.*

In the controller synthesis framework using two-player games between a controller and its environment, some works advocate the use of environment objectives which the environment can guarantee against any controller [7]. Under this assumption, Win-under-Hyp implies AA:

▶ **Lemma 3.** *Let* $\mathsf{G} = \langle \mathsf{A}, \phi_1, \phi_2 \rangle$ *be a two-player game. If player* 2 *has a winning strategy for* ϕ_2 *and Win-under-Hyp has a solution, then* AA *has a solution.*

We now consider the *robustness* of the profiles synthesized using the above rules. An AA-winning strategy profile $\sigma_{\mathcal{P}}$ is robust in the following sense: The set of AA-winning profiles is *rectangular*, *i.e. any* combination of AA-winning strategies independently chosen for each player, is an AA-winning profile. Second, if one replaces *any* subset of strategies in AA-winning profile $\sigma_{\mathcal{P}}$ by arbitrary admissible strategies, the objectives of all the other players still hold. Formally, a *rectangular set* of strategy profiles is a set that is a Cartesian product of sets of strategies, given for each player. A synthesis rule is *rectangular* if the set of strategy profiles satisfying the rule is rectangular. The RS rules require a specific definition since player 1 has a particular role: we say that $\mathsf{RS}^{\forall,\exists}(\gamma)$ is rectangular if for any strategy σ_1 witnessing the rule, the set of strategy profiles $(\sigma_2, \dots, \sigma_n) \in \Sigma^{\gamma}_{\mathsf{G},\sigma_1}$ s.t. $\mathsf{G}, \sigma_1, \dots, \sigma_n \models \phi_1$ is rectangular. We show that apart from AA, only Win and $\mathsf{RS}^{\forall}(\mathsf{Dom})$ are rectangular.

▶ **Theorem 4.** *We have:*

1. *Rule* AA *is rectangular; and for all games* G*, for all* AA*-winning strategy profile* σ_P*, coalition* $C \subseteq \mathcal{P}$*, if* $\sigma'_C \in \mathsf{Adm}_C(\mathsf{G})$*, then* $\mathsf{G}, \sigma_{-C}, \sigma'_C \models \bigwedge_{i \in -C} \phi_i$*.*
2. *The rules* Win *and* $\mathsf{RS}^{\forall}(\mathsf{Dom})$ *are rectangular; the rules* Coop*,* $\mathsf{AG}^{\vee}$*,* $\mathsf{AG}^{\wedge}$*,* $\mathsf{RS}^{\exists}(\mathsf{NE}, \mathsf{SPE}, \mathsf{Dom})$*, and* $\mathsf{RS}^{\forall}(\mathsf{NE}, \mathsf{SPE})$ *are not rectangular.*

6 Algorithm for Assume-Admissible Synthesis

We now recall the characterization of the outcomes of admissible strategy profiles given in [4], and derive algorithms for the AA rule. We use the game of Fig. 2 as a running example for this section. Clearly, none of the players of this game has a winning strategy for his own objective when not taking into account the objective of the other player, but, as we will see, both players have an admissible and winning strategy against the *admissible* strategies of the other player, and so the AA rule applies.

The notion of *value* associated to the states of a game plays an important role in the characterization of admissible strategies and their outcomes [2, 4]. Fix a game G. A state s has value 1 for player i, written $\mathsf{Val}_i(s) = 1$, if player i has a winning strategy from s; $\mathsf{Val}_i(s) = -1$ if for all strategy profiles $\sigma_{\mathcal{P}} \in \Sigma_{\mathcal{P}}$, $\mathsf{Out}_{\mathsf{G},s}(\sigma_{\mathcal{P}})$ does not satisfy ϕ_i; and otherwise $\mathsf{Val}_i(s) = 0$. A *player* j *decreases its own value in history* h if there is a position k such that $\mathsf{Val}_j(h_k) > \mathsf{Val}_j(h_{k+1})$ and $h_k \in \mathsf{S}_j$. We proved in [4], that admissible strategies do not decrease their own values. Let us call such strategies *value-preserving*. In fact, if the current state has value 1, there is a winning strategy which stays within the winning region; if the value is 0, then although other players may force the play into states of value -1, a good strategy for player i will not do this by itself.

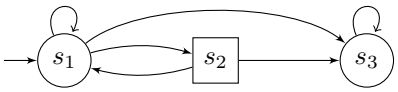

Figure 2 Game G with two players $\mathcal{P} = \{1, 2\}$. Player 1 controls the round states, and has objective $\mathtt{GF}s_2$, and player 2 controls the square state and has objective $\mathtt{GF}s_1$.

▶ **Lemma 5** ([4, Lem. 1])**.** *For all games* G*, players* i*, and histories* ρ*, if* $\mathsf{last}(\rho) \in \mathsf{S}_i$ *and* $\sigma_i \in \mathsf{Adm}_i$ *then* $\mathsf{Val}_i(\delta(\mathsf{last}(\rho), \sigma_i(\rho))) = \mathsf{Val}_i(\mathsf{last}(\rho))$*.*

For player i, let us define the sets $V_{i,x} = \{s \mid \mathsf{Val}_i(s) = x\}$ for $x \in \{-1, 0, 1\}$, which partition S. We define the set of *value-preserving edges* for player i as $E_i = \{(s, a) \in \mathsf{S} \times \mathsf{Act} \mid s \in \mathsf{S}_i \Rightarrow \mathsf{Val}_i(\delta(s, a)) = \mathsf{Val}_i(s)\}$. Observe that value-preserving strategies for player i are exactly those respecting E_i.

In our running example of Fig. 2, it should be clear that any strategy that chooses a transition that goes to s_3 is *not* admissible nor for Player 1 neither for Player 2, as by making this choice both players are condemned to lose their own objective while their other choices leave a chance to win; so the choice of going to s_3 would decrease their own value. So, we can already conclude that Player 2 always chooses $s_2 \mapsto s_1$, his only admissible strategy.

Not all value-preserving strategies are admissible: for Müller objectives, staying inside the winning region does not imply the objective. Moreover, in states of value 0, admissible strategies must visit states where other players can "help" satisfy the objective. Formally, *help states* for player i are other players' states with value 0 and at least two different successors of value 0 or 1. Let $H_i = \{s \in \mathsf{S} \setminus \mathsf{S}_i \mid \mathsf{Val}_i(s) = 0 \wedge \exists s' \neq s''.\ s' \in \delta(s, \mathsf{Act}) \wedge s'' \in \delta(s, \mathsf{Act}) \wedge \mathsf{Val}_i(s') \geq 0 \wedge \mathsf{Val}_i(s'') \geq 0\}$. Given this, the following lemma, adapted from [4], characterizes the outcomes of admissible strategies. We denote by $\mathtt{G}(E_i)$ the set of runs that respect E_i, *i.e.* $\mathtt{G}(\bigvee_{(s,a) \in E_i} s \wedge \mathtt{X}(\delta(s, a)))$.

▶ **Lemma 6.** *For all games* G*, and players* i*,* $\mathsf{Out}_{\mathsf{G}} \cap \Phi_i = \mathsf{Out}_{\mathsf{G}}(\mathsf{Adm}_i, \Sigma_{-i})$*, where* $\Phi_i = \mathtt{G}(E_i) \wedge (\mathtt{GF}(V_{i,1}) \Rightarrow \phi_i) \wedge (\mathtt{GF}(V_{i,0}) \Rightarrow \phi_i \vee \mathtt{GF}(H_i))$.

In our running example of Fig. 2, a strategy of Player 1 which, after some point, always chooses $s_1 \mapsto s_1$ is dominated by strategies that chose infinitely often $s_1 \mapsto s_2$. This is a corollary of the lemma above. Indeed, while all those strategies only visit states with value 0 (and so do not decrease the value for Player 1), the strategy that always chooses $s_1 \mapsto s_1$ has an outcome which is loosing for Player 1 while the other strategies are compatible with outcomes that are winning for Player 1. So, outcome of admissible strategies for Player 1 that always visit states with values 0, also visits s_2 infinitely often. Using the fact that strategies are value-preserving and the last observation, we can now conclude that both players have (admissible) winning strategies against the admissible strategies of the other players. For instance when Player 1 always chooses to play $s_1 \mapsto s_2$, he wins against the admissible strategies of Player 2.

Note that Φ_i can be decomposed into a safety condition $S_i = \mathtt{G}(E_i)$ and a *prefix independent* condition $M_i = (\mathtt{GF}(V_{i,1}) \Rightarrow \phi_i) \wedge (\mathtt{GF}(V_{i,0}) \Rightarrow (\phi_i \vee \mathtt{GF}(H_i))$ which can be expressed by a Müller condition described by a circuit of polynomial size.

For player i, we let $\Omega_i = \mathsf{Out}_{\mathsf{G}}(\mathsf{Adm}_i) \wedge (\mathsf{Out}_{\mathsf{G}}(\mathsf{Adm}_{-i}) \Rightarrow \phi_i)$, which describes the outcomes of admissible strategies of player i, which satisfy objective ϕ_i under the hypothesis that they are compatible with other players' admissible strategies. In fact, it follows from [4] that Ω_i captures the *outcomes* of AA-winning strategies for player i.

▶ **Lemma 7.** *A player* i *strategy is* AA*-winning iff it is winning for objective* Ω_i*.*

Objective Ω_i is not directly expressible as a Müller condition, since Φ_i and $\bigwedge_j \Phi_j$ contain safety parts. Nevertheless, the information whether $\mathtt{G}(E_i)$, or $\mathtt{G}(\cup_{j \neq i} E_j)$ has been violated can be encoded in the state space. Formally, for each player i, we define game G'_i by taking the product of G with $\{\top, 0, \bot\}$; that is, the states are $\mathsf{S} \times \{\top, 0, \bot\}$, and the initial state $(s_{\mathsf{init}}, 0)$. The transitions are defined as for G for the first component; while from state $(s, 0)$, any action a outside E_i leads to $(\delta(s, a), \bot)$, and any action a outside E_j, $j \neq i$, leads to $(\delta(s, a), \top)$. The second component is absorbing at $\bot, \top$. We now rewrite the condition Ω_i for G'_i as $\Omega'_i = \big(\mathtt{GF}(\mathsf{S} \times \{0\}) \wedge M'_i \wedge (\wedge_{j \neq i} M'_j \Rightarrow \phi'_i)\big) \vee (\mathtt{GF}(\mathsf{S} \times \{\top\}) \wedge M'_i)$, where M'_i is the set of runs of G'_i whose projections to G are in M_i, and similarly for ϕ'_i.

Now, checking AA-synthesis is reduced to solving games with Müller conditions. Moreover, we also obtain a polynomial-time algorithm when all objectives are Büchi conditions, by showing that Ω'_i is expressible by a parity condition with four colors.

▶ **Theorem 8.** AA*-synthesis in multiplayer games is* PSPACE*-complete, and* P*-complete for Büchi objectives. Player* i *wins for objective* Ω_i *in* G *iff he wins for objective* Ω'_i *in* G'_i*.*

7 Abstraction

We present abstraction techniques to compute assume-admissible strategy profiles following the *abstract interpretation* framework [11]; see [18] for games. Abstraction is a crucial feature for scalability in practice, and we show here that the AA rule is amenable to abstraction techniques. The problem is not directly reducible to computing AA-winning strategies in abstract games obtained as *e.g.* in [13]; in fact, it can be easily seen that the set of admissible strategies of an abstract game is incomparable with those of the concrete game in general.

Overview. Informally, to compute an AA-winning strategy for player i, we construct an abstract game $\mathcal{A}'_i$ with objective $\underline{\Omega'_i}$ s.t. winning strategies of player i in $\mathcal{A}'_i$ map to AA-winning strategies in G. To define $\mathcal{A}'$, we re-visit the steps of the algorithm of Section 6 by defining approximations computed on the abstract state space. More precisely, we show how to compute under- and over-approximations of the sets $V_{x,k}$, namely $\underline{V}_{x,k}$ and $\overline{V}_{x,k}$, using fixpoint computations on the abstract state space only. We then use these sets to define approximations of the value preserving edges ($\underline{E}_k$ and $\overline{E}_k$) and those of the help states ($\underline{H}_k$ and $\overline{H}_k$). These are then combined to define objective $\underline{\Omega'}_k$ s.t. if player k wins the abstract game for $\underline{\Omega'}_k$, then he wins the original game for Ω'_k, and thus has an AA-winning strategy.

Abstract Games. Consider $\mathsf{G} = \langle \mathsf{A}, (\phi_i)_{i\in\mathcal{P}}\rangle$ with $\mathsf{A} = \langle \mathcal{P}, (\mathsf{S}_i)_{i\in\mathcal{P}}, s_{\mathsf{init}}, (\mathsf{Act}_i)_{i\in\mathcal{P}}, \delta\rangle$ where each ϕ_i is a Müller objective given by a family of sets of states $(\mathcal{F}_i)_{i\in\mathcal{P}}$. Let $\mathsf{S}^{\mathsf{a}} = \biguplus_{i\in\mathcal{P}} \mathsf{S}^{\mathsf{a}}_i$ denote a finite set, namely the *abstract state space*. A *concretization function* $\gamma\colon \mathsf{S}^{\mathsf{a}} \mapsto 2^{\mathsf{S}}$ is a function such that:

1. the abstract states partitions the state space: $\biguplus_{s^{\mathsf{a}}\in\mathsf{S}^{\mathsf{a}}} \gamma(s^{\mathsf{a}}) = \mathsf{S}$,
2. it is compatible with players' states: for all players i and $s^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}_i$, $\gamma(s^{\mathsf{a}}) \subseteq \mathsf{S}_i$.

We define the corresponding *abstraction* function $\alpha : \mathsf{S} \to \mathsf{S}^{\mathsf{a}}$ where $\alpha(s)$ is the unique state s^{a} s.t. $s \in \gamma(s^{\mathsf{a}})$. We also extend α, γ naturally to sets of states; and to histories, by replacing each element of the sequence by its image.

We further assume that γ is *compatible* with all objectives $\mathcal{F}_i$ in the sense that the abstraction of a set S is sufficient to determine whether $S \in \mathcal{F}_i$: for all $i \in \mathcal{P}$, for all $S, S' \subseteq \mathsf{S}$ with $\alpha(S) = \alpha(S')$, we have $S \in \mathcal{F}_i \Leftrightarrow S' \in \mathcal{F}_i$. If the objective ϕ_i is given by a circuit, then the circuit for the corresponding abstract objective ϕ^{a}_i is obtained by replacing each input on state s by $\alpha(s)$. We thus have $\rho \in \phi_i$ if, and only if, $\alpha(\rho) \in \phi^{\mathsf{a}}_i$.

The *abstract transition relation* Δ^{a} induced by γ is defined by: $(s^{\mathsf{a}}, a, t^{\mathsf{a}}) \in \Delta^{\mathsf{a}} \Leftrightarrow \exists s \in \gamma(s^{\mathsf{a}}), \exists t \in \gamma(t^{\mathsf{a}}), t = \delta(s, a)$. We write $\mathsf{post}_\Delta(s^{\mathsf{a}}, a) = \{t^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}} \mid \Delta(s^{\mathsf{a}}, a, t^{\mathsf{a}})\}$, and $\mathsf{post}_\Delta(s^{\mathsf{a}}, \mathsf{Act}) = \cup_{a\in\mathsf{Act}} \mathsf{post}_\Delta(s^{\mathsf{a}}, a)$. For each coalition $C \subseteq \mathcal{P}$, we define a game in which players C play together against coalition $-C$; and the former resolves non-determinism in Δ^{a}. Intuitively, the winning region for C in this abstract game will be an over-approximation of the original winning region. Given C, the *abstract arena* $\mathcal{A}^C$ is $\langle \{C, -C\}, (\mathsf{S}_C, \mathsf{S}_{-C}), \alpha(s_{\mathsf{init}}), (\mathsf{Act}_C, \mathsf{Act}_{-C}), \delta^{\mathsf{a},C}\rangle$, where $\mathsf{S}_C = \big(\bigcup_{i\in C} \mathsf{S}^{\mathsf{a}}_i\big) \cup \big(\bigcup_{i\in\mathcal{P}} \mathsf{S}^{\mathsf{a}}_i \times \mathsf{Act}_i\big)$, $\mathsf{S}_{-C} = \bigcup_{i\notin C} \mathsf{S}^{\mathsf{a}}_i$; and $\mathsf{Act}_C = \big(\bigcup_{i\in C} \mathsf{Act}_i\big) \cup \mathsf{S}^{\mathsf{a}}$ and $\mathsf{Act}_{-C} = \bigcup_{i\in -C} \mathsf{Act}_i$. The relation $\delta^{\mathsf{a},C}$ is given by: if $s^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}$, then $\delta^{\mathsf{a},C}(s^{\mathsf{a}}, a) = (s^{\mathsf{a}}, a)$. If $(s^{\mathsf{a}}, a) \in \mathsf{S}^{\mathsf{a}} \times \mathsf{Act}$ and $t^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}$ satisfies $(s^{\mathsf{a}}, a, t^{\mathsf{a}}) \in \Delta^{\mathsf{a}}$ then $\delta^{\mathsf{a},C}((s^{\mathsf{a}}, a), t^{\mathsf{a}}) = t^{\mathsf{a}}$; while for $(s^{\mathsf{a}}, a, t^{\mathsf{a}}) \notin \Delta^{\mathsf{a}}$, the play leads to an arbitrarily chosen state u^{a} with $\Delta(s^{\mathsf{a}}, a, u^{\mathsf{a}})$. Thus, from states (s^{a}, a), coalition C chooses a successor t^{a}.

We extend γ to histories of $\mathcal{A}^C$ by first removing states of $(\mathsf{S}^{\mathsf{a}}_i \times \mathsf{Act}_i)$; and extend α by inserting these intermediate states. Given a strategy σ of player k in $\mathcal{A}^C$, we define its *concretization* as the strategy $\gamma(\sigma)$ of G that, at any history h of G, plays $\gamma(\sigma)(h) = \sigma(\alpha(h))$. We write $\mathsf{Win}_D(\mathcal{A}^C, \phi^{\mathsf{a}}_k)$ for the states of S^{a} from which the coalition D has a winning strategy in $\mathcal{A}^C$ for objective ϕ^{a}_k, with $D \in \{C, -C\}$. Informally, it is easier for coalition C to achieve an objective in $\mathcal{A}^C$ than in G, that is, $\mathsf{Win}_C(\mathcal{A}^C, \phi^{\mathsf{a}}_k)$ over-approximates $\mathsf{Win}_C(\mathsf{A}, \phi_k)$:

▶ **Lemma 9.** *If the coalition C has a winning strategy for objective ϕ_k in G from s then it has a winning strategy for ϕ^{a}_k in $\mathcal{A}^C$ from $\alpha(s)$.*

Value-Preserving Strategies. We now provide under- and over-approximations for value-preserving strategies for a given player. We start by computing approximations $\underline{V}_{k,x}$ and $\overline{V}_{k,x}$ of the sets $V_{k,x}$, and then use these to obtain approximations of the value-preserving edges E_k.

Fix a game G, and a player k. Let us define the *controllable predecessors* for player k as $\mathsf{CPRE}_{\mathcal{A}^{\mathcal{P}\setminus\{k\}},k}(X) = \{s^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}_k \mid \exists a \in \mathsf{Act}_k, \mathsf{post}_\Delta(s^{\mathsf{a}}, a) \subseteq X\} \cup \{s^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}_{\mathcal{P}\setminus\{k\}} \mid \forall a \in \mathsf{Act}_{-k}, \mathsf{post}_\Delta(s^{\mathsf{a}}, a) \subseteq X\}$. We let

$$\begin{aligned}
&\overline{V}_{k,1} = \mathsf{Win}_{\{k\}}(\mathcal{A}^{\{k\}}, \phi^{\mathsf{a}}_k), && \overline{V}_{k,-1} = \mathsf{Win}_{\emptyset}(\mathcal{A}^{\emptyset}, \neg\phi^{\mathsf{a}}_k),\\
&\overline{V}_{k,0} = \mathsf{Win}_{\mathcal{P}\setminus\{k\}}(\mathcal{A}^{\mathcal{P}\setminus\{k\}}, \neg\phi^{\mathsf{a},k}) \cap \mathsf{Win}_{\mathcal{P}}(\mathcal{A}^{\mathcal{P}}, \phi^{\mathsf{a}}_k),\\
&\underline{V}_{k,1} = \mathsf{Win}_{\{k\}}(\mathcal{A}^{\mathcal{P}\setminus\{k\}}, \phi^{\mathsf{a}}_k), && \underline{V}_{k,-1} = \mathsf{Win}_{\emptyset}(\mathcal{A}^{\mathcal{P}}, \neg\phi^{\mathsf{a}}_k)\\
&\underline{V}_{k,0} = \nu X.\big(\mathsf{CPRE}_{\mathcal{A}^{\mathcal{P}\setminus\{k\}},k}(X \cup \underline{V}_{k,1} \cup \underline{V}_{k,-1}) \cap F\big),\\
&\qquad \text{where } F = \mathsf{Win}_{\mathcal{P}\setminus\{k\}}(\mathcal{A}^{\{k\}}, \neg\phi^{\mathsf{a}}_k) \cap \mathsf{Win}_{\mathcal{P}}(\mathcal{A}^{\emptyset}, \phi^{\mathsf{a}}_k).
\end{aligned}$$

The last definition uses the $\nu X.f(X)$ operator which is the greatest fixpoint of f. These sets define approximations of the sets $V_{k,x}$. Informally, this follows from the fact that to define *e.g.* $\overline{V}_{k,1}$, we use the game $\mathcal{A}^{\{k\}}$, where player k resolves itself the non-determinism, and thus has more power than in G. In contrast, for $\underline{V}_{k,1}$, we solve $\mathcal{A}^{\mathcal{P}\setminus\{k\}}$ where the adversary resolves non-determinism. We state these properties formally:

▶ **Lemma 10.** *For all players k and $x \in \{-1, 0, 1\}$, $\gamma(\underline{V}_{k,x}) \subseteq V_{k,x} \subseteq \gamma(\overline{V}_{k,x})$.*

We thus have $\cup_x \gamma(\overline{V}_{k,x}) = \mathsf{S}$ (as $\cup_x V_{k,x} = \mathsf{S}$) but this is not the case for $\underline{V}_{k,x}$; so let us define $\underline{V} = \cup_{j\in\{-1,0,1\}} \underline{V}_{k,j}$. We now define approximations of E_k based on the above sets.

$$\begin{aligned}
\overline{E}_k &= \{(s^{\mathsf{a}}, a) \in \mathsf{S}^{\mathsf{a}} \times \mathsf{Act} \mid s^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}_k \Rightarrow \exists x, s^{\mathsf{a}} \in \overline{V}_{k,x}, \mathsf{post}_\Delta(s^{\mathsf{a}}, a) \cap \cup_{l\geq x} \overline{V}_{k,l} \neq \emptyset\},\\
\underline{E}_k &= \{(s^{\mathsf{a}}, a) \in \mathsf{S}^{\mathsf{a}} \times \mathsf{Act} \mid s^{\mathsf{a}} \in \mathsf{S}^{\mathsf{a}}_k \Rightarrow \exists x, s^{\mathsf{a}} \in \underline{V}_{k,x}, \mathsf{post}_\Delta(s^{\mathsf{a}}, a) \subseteq \cup_{l\geq x} \underline{V}_{k,l}\}\\
&\qquad\qquad \cup \{(s^{\mathsf{a}}, a) \mid s^{\mathsf{a}} \notin \underline{V}\}.
\end{aligned}$$

Intuitively, $\overline{E}_k$ is an over-approximation of E_k, and $\underline{E}_k$ under-approximates E_k when restricted to states in $\underline{V}$ (notice that $\underline{E}_k$ contains all actions from states outside $\underline{V}$). In fact, our under-approximation will be valid only inside $\underline{V}$; but we will require the initial state to be in this set, and make sure the play stays within $\underline{V}$. We show that sets $\underline{E}_k$ and $\overline{E}_k$ provide approximations of value-preserving strategies.

▶ **Lemma 11.** *For all games G, and players k, $\mathsf{Strat}_k(E_k) \subseteq \gamma(\mathsf{Strat}_k(\overline{E}_k))$, and if $s_{\mathsf{init}} \in \gamma(\underline{V})$, then $\emptyset \neq \gamma(\mathsf{Strat}_k(\underline{E}_k)) \subseteq \mathsf{Strat}_k(E_k)$.*

Abstract Synthesis of AA-winning strategies. We now describe the computation of AA-winning strategies in abstract games. Consider game G and assume sets $\underline{E}_i, \overline{E}_i$ are computed for all players i. Roughly, to compute a strategy for player k, we will constrain him to play only edges from $\underline{E}_k$, while other players j will play in $\overline{E}_j$. By Lemma 11, any strategy of player k maps to value-preserving strategies in the original game, and all value-preserving strategies for other players are still present. We now formalize this idea, incorporating the help states in the abstraction.

We fix a player k. We construct an abstract game in which winning for player k implies that player k has an effective AA-winning strategy in G. We also define $\mathcal{A}'_k = \langle\{\{k\}, -k\}, (\mathsf{S'}^{\mathsf{a}}_k, \mathsf{S'}^{\mathsf{a}}_{-k} \cup \mathsf{S'}^{\mathsf{a}} \times \mathsf{Act}), \alpha(s_{\mathsf{init}}), (\mathsf{Act}_k, \mathsf{Act}_{-k}), \delta_{\mathcal{A}^k}\rangle$, where $\mathsf{S'}^{\mathsf{a}} = \mathsf{S}^{\mathsf{a}} \times \{\bot, 0, \top\}$; thus we modify $\mathcal{A}^{\mathcal{P}\setminus\{k\}}$ by taking the product of the state space with $\{\top, 0, \bot\}$. Intuitively, as in Section 6, initially the second component is 0, meaning that no player has violated the value-preserving edges. The component becomes $\bot$ whenever player k plays an action outside of $\underline{E}_k$; and $\top$ if another player j plays outside $\overline{E}_j$. We extend γ to $\mathcal{A}'_k$ by $\gamma((s^{\mathsf{a}}, x)) = \gamma(s^{\mathsf{a}}) \times \{x\}$, and extend it to histories of $\mathcal{A}'_k$ by first removing the intermediate states $\mathsf{S'}^{\mathsf{a}} \times \mathsf{Act}$. We thus see $\mathcal{A}'_k$ as an abstraction of $\mathsf{A'}$ of Section 6.

In order to define the objective of $\mathcal{A}'_k$, let us first define approximations of the help states H_k, where we write $\Delta(s^{\mathsf{a}}, \mathsf{Act}, t^{\mathsf{a}})$ to mean $\exists a \in \mathsf{Act}, \Delta(s^{\mathsf{a}}, a, t^{\mathsf{a}})$.

$$\begin{aligned}\overline{H}_k &= \{s^{\mathsf{a}} \in \overline{V}_{k,0} \setminus \mathsf{S}^{\mathsf{a}}_k \mid \exists t^{\mathsf{a}}, u^{\mathsf{a}} \in \overline{V}_{k,0} \cup \overline{V}_{k,1}.\ \Delta(s^{\mathsf{a}}, \mathsf{Act}, t^{\mathsf{a}}) \wedge \Delta(s^{\mathsf{a}}, \mathsf{Act}, u^{\mathsf{a}})\}\\ \underline{H}_k &= \{s^{\mathsf{a}} \in \underline{V}_{k,0} \setminus \mathsf{S}^{\mathsf{a}}_k \mid \exists a \neq b \in \mathsf{Act}, \mathsf{post}_\Delta(s^{\mathsf{a}}, a) \cap \mathsf{post}_\Delta(s^{\mathsf{a}}, b) = \emptyset,\\ &\qquad\qquad \mathsf{post}_\Delta(s^{\mathsf{a}}, a) \cup \mathsf{post}_\Delta(s^{\mathsf{a}}, b) \subseteq \underline{V}_{k,0} \cup \underline{V}_{k,1}\}.\end{aligned}$$

We define the following approximations of the objectives M'_k and Ω'_k in $\mathcal{A}'_k$.

$$\begin{aligned}\underline{M'}_k &= (\mathtt{GF}(\overline{V}_{k,1}) \Rightarrow \phi^{\mathsf{a}}_k) \wedge \left(\mathtt{GF}(\overline{V}_{k,0}) \Rightarrow (\phi^{\mathsf{a}}_k \vee \mathtt{GF}(\underline{H}_k))\right),\\ \overline{M'}_k &= (\mathtt{GF}(\underline{V}_{k,1}) \Rightarrow \phi^{\mathsf{a}}_k) \wedge \left(\mathtt{GF}(\underline{V}_{k,0}) \Rightarrow (\phi^{\mathsf{a}}_k \vee \mathtt{GF}(\overline{H}_k))\right),\\ \underline{\Omega'}_k &= \left(\mathtt{GF}(\mathsf{S}^{\mathsf{a}} \times \{0\}) \wedge \underline{M'}_k \wedge \left(\textstyle\bigwedge_{j \neq k} \overline{M'}_j \Rightarrow \phi^{\mathsf{a}}_k\right)\right) \vee (\mathtt{GF}(\mathsf{S}^{\mathsf{a}} \times \{\top\}) \wedge \underline{M'}_k).\end{aligned}$$

▶ **Theorem 12.** *For all games* G*, and players* k*, if* $s_{\mathsf{init}} \in \underline{V}$*, and player* k *has a winning strategy in* $\mathcal{A}'_k$ *for objective* $\underline{\Omega}'_k$*, then he has a winning strategy in* G'_k *for* Ω_k*; and thus a* AA*-winning strategy in* G*.*

Now, if Theorem 12 succeeds to find an AA-winning strategy for each player k, then the resulting strategy profile is AA-winning.

8 Conclusion

In this paper, we have introduced a novel synthesis rule, called the *assume admissible synthesis*, for the synthesis of strategies in non-zero sum n players games played on graphs with omega-regular objectives. We use the notion of admissible strategy, a classical concept from game theory, to take into account the objectives of the other players when looking for winning strategy of one player. We have compared our approach with other approaches such as assume guarantee synthesis and rational synthesis that target the similar scientific objectives. We have developed worst-case optimal algorithms to handle our synthesis rule as well as dedicated abstraction techniques. As future works, we plan to develop a tool prototype to support our assume admissible synthesis rule.

References

1 Brandenburger Adam, Friedenberg Amanda, H Jerome, et al. Admissibility in games. *Econometrica*, 2008.

2 Dietmar Berwanger. Admissibility in infinite games. In *Proc. of STACS'07*, volume 4393 of *LNCS*, pages 188–199. Springer, February 2007.

3 Roderick Bloem, Rüdiger Ehlers, Swen Jacobs, and Robert Könighofer. How to handle assumptions in synthesis. In *SYNT'14*, volume 157 of *EPTCS*, pages 34–50, 2014.

4 Romain Brenguier, Jean-François Raskin, and Mathieu Sassolas. The complexity of admissibility in omega-regular games. In *CSL-LICS'14, 2014*. ACM, 2014.

5 Krishnendu Chatterjee, Laurent Doyen, Emmanuel Filiot, and Jean-François Raskin. Doomsday equilibria for omega-regular games. In *VMCAI'14*, volume 8318, pages 78–97. Springer, 2014.

6 Krishnendu Chatterjee and Thomas A Henzinger. Assume-guarantee synthesis. In *TACAS'07*, volume 4424 of *LNCS*. Springer, 2007.

7 Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Environment assumptions for synthesis. In *CONCUR 2008*, volume 5201 of *LNCS*, pages 147–161. Springer, 2008.

8 Krishnendu Chatterjee, Thomas A Henzinger, and Marcin Jurdziński. Games with secure equilibria. *Theoretical Computer Science*, 365(1):67–82, 2006.

9 Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. *Inf. Comput.*, 208(6):677–693, 2010.

10 Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logics of Programs*, volume 131 of *LNCS*, pages 52–71. Springer, 1981.

11 Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL'77*. ACM, 1977.

12 Werner Damm and Bernd Finkbeiner. Automatic compositional synthesis of distributed systems. In *FM 2014*, volume 8442 of *LNCS*, pages 179–193. Springer, 2014.

13 Luca de Alfaro, Patrice Godefroid, and Radha Jagadeesan. Three-valued abstractions of games: Uncertainty, but with precision. In *LICS'04*. IEEE, 2004.

14 E Allen Emerson. Temporal and modal logic. *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, 995:1072, 1990.

15 Marco Faella. Admissible strategies in infinite games over graphs. In *MFCS 2009*, volume 5734 of *Lecture Notes in Computer Science*, pages 307–318. Springer, 2009.

16 Dana Fisman, Orna Kupferman, and Yoad Lustig. Rational synthesis. In *TACAS'10*, volume 6015 of *LNCS*, pages 190–204. Springer, 2010.

17 Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, Cambridge, MA, 1991. Translated into Chinesse by Renin University Press, Bejing: China.

18 Thomas A. Henzinger, Rupak Majumdar, Freddy Y. C. Mang, and Jean-François Raskin. Abstract interpretation of game properties. In *SAS*, pages 220–239, 2000.

19 Paul Hunter. *Complexity and Infinite Games on Finite Graphs*. PhD thesis, Computer Laboratory, University of Cambridge, 2007.

20 O. Kupferman, G. Perelli, and M.Y. Vardi. Synthesis with rational environments. In *Proc. 12th European Conference on Multi-Agent Systems*, LNCS. Springer, 2014.

21 Zohar Manna and Pierre Wolper. Synthesis of communicating processes from temporal logic specifications. In *Logics of Programs*, volume 131 of *LNCS*, pages 253–281. Springer, 1981.

22 Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi. Reasoning about strategies. In *FSTTCS 2010*, volume 8 of *LIPIcs*. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2010.

23 John Nash. Equilibrium points in n-person games. *Proc. NAS*, 1950.

24 Amir Pnueli. The temporal logic of programs. In *Foundations of Computer Science, 1977., 18th Annual Symposium on*, pages 46–57. IEEE, 1977.