The Complexity of Admissibility in Omega-Regular Games *

Romain Brenguier
Département d'informatique, Université Libre de Bruxelles (U.L.B.), Belgium
romain.brenguier@ulb.ac.be

Jean-François Raskin
Département d'informatique, Université Libre de Bruxelles (U.L.B.), Belgium
jraskin@ulb.ac.be

Mathieu Sassolas
Université Paris-Est, LACL, Créteil, France
mathieu.sassolas@u-pec.fr

Abstract

Iterated admissibility is a well-known and important concept in classical game theory, e.g. to determine *rational* behaviors in multi-player matrix games. As recently shown by Berwanger, this concept can be soundly extended to infinite games played on graphs with ω-regular objectives. In this paper, we study the algorithmic properties of this concept for such games. We settle the exact complexity of natural decision problems on the set of strategies that survive iterated elimination of dominated strategies. As a byproduct of our construction, we obtain automata which recognize all the possible outcomes of such strategies.

Categories and Subject Descriptors F.1.2 [*Modes of computation*]: Interactive and reactive computation; D.2.4 [*Software/Program Verification*]: Model checking; I.2.8 [*Problem Solving, Control Methods, and Search*]: Control theory

Keywords Games; Admissible strategies; Synthesis; Safety; Muller games; LTL.

1. Introduction

Two-player games played on graphs are central in many applications of computer science. For example, in the synthesis problem for reactive systems, implementations are obtained from winning strategies in games with a ω-regular objectives [19]. To analyze systems composed of several components, two-player games are extended to multi-player games with non zero-sum objectives, i.e. each player has his own objective expressed as a ω-regular specification which is not necessarily adversarial w.r.t. the objectives of the other players.

To analyze multi-player games in normal form (a.k.a. *matrix games*), concepts like the celebrated Nash equilibrium [16] have been proposed. Another central concept is the notion of *dominated strategy* [17]. A strategy of a player *dominates* another one if the outcome of the first strategy is better than the outcome of the second no matter how the other players play. In two-player matrix game of Figure 1, strategies of player 1 (of player 2 respectively) are given as rows of the matrix (as columns respectively), and the payoffs to be maximized, are given as pairs of integers (the first for player 1 and the second for player 2). Strategy B of player 1 dominates strategy A: no matter how player 2 plays, B provides an outcome which is larger than or equal to the one of A, and if player 2 plays C then the outcome provided by B is strictly larger than the outcome of A. On the other hand, player 2 at first sight has no preference between C and D. But if player 2 knows that player 1 prefers strategy B to strategy A, then he will in turn prefer D to C, and it is then reasonable to predict that (B, D) will be played. This process is called the *iterated elimination of dominated strategies*, and it is valid under the hypothesis that rationality is *common knowledge* among the players [1]. Strategies that survive the iterated elimination of strategies are called *iteratively admissible strategies*.

	C	D
A	$(0, 2)$	$(1, 1)$
B	$(1, 1)$	$(1, 2)$

Figure 1: A two-player matrix game.

In [2], Berwanger initiated a fundamental study of the notion of *rational behaviour* in infinite duration games played on graph by generalising the notion of strategy dominance and iterated elimination of dominated strategies to that setting. This solution concept is a clear potential alternative to Nash equilibria for those games. As pointed out by Berwanger, one important advantage of admissible strategies is that they are compatible with the sequential nature of games played on graphs: *"in any position reachable with an admissible strategy, a strategy is admissible in the sub-game rooted in that position if, and only if, it is the restriction of an admissible strategy in the original game."* As a consequence, admissibility does not feature *non-credible threats* while it is well known that it is the case for Nash equilibria. Nonetheless, the extension of iterated strategy elimination to infinite duration games is challenging as the set of strategies is infinite and may lead to infinite dominance chains. Berwanger's main technical results are as follows: all iteration stages are dominated by admissible strategies, the iteration is non-stagnating, and, under regular objectives, admissible strategies form a *regular set*. In particular, for the last result, Berwanger suggests a procedure that uses tree automata to represent sets of strategies. The closure of tree automata to projection and Boolean operations naturally provides an algorithm to compute admissible strategies in parity games but this algorithm has *non-elementary complexity*.

In order to represent a viable alternative to Nash equilibria from a *computational point of view*, it is fundamental to better understand the complexity of iterated elimination of dominated strategies in ω-regular games, and see if the non-elementary complexity of the tree-automata based procedure can be avoided. We prove

* Work supported by ERC Starting Grant inVEST (279499).

CSL-LICS 2014, July 14–18, 2014, Vienna, Austria.

http://dx.doi.org/10.1145/2603088.2603143

here that this is indeed the case and that iterated elimination of dominated strategies has a computational complexity similar to the one of Nash equilibria. More precisely, we study games with weak Muller and (classical) Muller winning conditions given as circuits. Circuits offer a concise representation of Muller conditions and are closed (while remaining succinct) under Boolean operations. Weak Muller conditions define objectives based on the set of states that occur along a run, they generalize safety and reachability objectives. (Classical) Muller conditions define objectives based on the set of states that appear *infinitely often* along a run. They generalize Büchi and parity objectives and are canonical representations of ω-regular objectives as every regular language can be accepted by a deterministic Muller automaton. We study the *winning coalition problem*: given a game and two subsets W, L of players, to determine whether there exists an iteratively admissible profile of strategies that guarantees that (i) all players of W win the game, and (ii) all players of L lose the game (other players may either win or lose). For weak and classical Muller objectives, we provide a procedure in PSPACE, with matching lower-bounds for safety, reachability, and Muller objectives. For Büchi objectives, we provide an algorithm that calls a polynomial number of times an oracle solving parity games (hence this would lead to a polynomial time solution if a polynomial time algorithm is found for parity games – the current best known complexity is $\mathsf{UP} \cap \mathsf{coUP}$ [12], although a deterministic subexponential algorithm exists [13]).

As a byproduct of our constructions, we obtain an automaton on infinite words which recognizes *all the possible outcomes* of iteratively admissible strategies. Any regular query on this language can be solved using classical automata techniques. As a consequence, we can solve any variant of the winning coalition problem defined above, if this variant can be expressed as such a query. For example, we can solve the *model-checking under admissibility* problem: given φ, an LTL formula [18, 21], does the outcome of every iteratively admissible profile satisfy φ? We show that this problem is complete for the class PSPACE, so it retains the same complexity as the "classical" model-checking problem for this logic. Model-checking under admissibility is useful to reason about properties that naturally *emerge* in a system from the interaction of *rational* agents that purse *their own objectives*.

Related work Dominance can be expressed in strategy logics [6, 15] but not unbounded iterated dominance. Bounded iterated dominance is expressible but leads to classes of formulas with a non-elementary model-checking algorithm. Other paradigms of rationality have been studied for games on finite graphs, like Nash-equilibria [3, 14, 23] or *regret minimization* [10]. In [14], the authors build an automaton that recognizes outcomes of Nash equilibria. In turn-based game, finding a Nash equilibrium with a particular payoff is PSPACE-complete for Muller objectives [24], which is the same complexity we obtain for admissibility. In the case of Büchi objectives, a polynomial algorithm exists for Nash equilibria [23], while we have a $\mathsf{NP} \cap \mathsf{coNP}$ algorithm for admissibility.

In this paper, we concentrate on n-player turn-based perfect information finite game graph with ω-regular objective. This is the basic setting that needs to be studied before looking at richer models, like games with incomplete information [20], games with quantitative objectives like mean-payoff objectives [25], or concurrent games [8]. Our results and techniques are clearly prerequisites to study those richer settings.

Organization of the paper We first formalize the setting and notations in Section 2. Safety objectives are solved in Section 3. Muller objectives are solved in Section 4. We conclude by giving other applications of our techniques for Büchi or weak Muller objectives, and to the model-checking under admissibility problem, in Section 8.

A comprehensive example of iterated elimination of dominated strategies is provided in Appendix A.

Due to space constraints, most proofs have been omitted from this paper; they can be found in the full version [5].

2. Definitions

2.1 Multiplayer Games

Definition 1 (Multiplayer games)**.** *A* turn-based multiplayer (non zero-sum) game *is a tuple* $\mathcal{G} = \langle P, (V_i)_{i\in P}, E, (\textsc{Win}_i)_{i\in P}\rangle$ *where:*

- P *is the non-empty and finite set of players;*
- $V = \biguplus_{i\in P} V_i$ *and for every* i *in* P, V_i *is the finite set of player* i*'s states;*
- $E \subseteq V \times V$ *is the set of edges*[1]*; we write* $s \to t$ *for* $(s,t) \in E$ *when* E *is clear from context.*
- *For every* i *in* P, $\textsc{Win}_i \subseteq V^\omega$ *is a winning condition.*

A *path* ρ is a sequence of states $(\rho_j)_{0\le j<n}$ with $n \in \mathbb{N} \cup \{\infty\}$ s.t. for all $j < n-1$, $\rho_j \to \rho_{j+1}$. The *length* $|\rho|$ of the path ρ is n. A *history* is a finite path and a *run* is an infinite path. Given a run $\rho = (\rho_j)_{j\in\mathbb{N}}$ and an integer k, we write $\rho_{\le k}$ the history $(\rho_j)_{0\le j<k+1}$, that is, the prefix of length $k+1$ of ρ. For a history ρ and a (finite or infinite) path ρ', ρ is a *prefix* of ρ' is written $\rho \trianglelefteqslant \rho'$. The *last state* of a history ρ is $\mathrm{last}(\rho) = \rho_{|\rho|-1}$. The set of states *occuring* in a path ρ is $\mathrm{Occ}(\rho) = \{s \in V \mid \exists i \in \mathbb{N}.\ i < |\rho|, \rho_i = s\}$. The set of states *occuring infinitely often* in a run ρ is $\mathrm{Inf}(\rho) = \{s \in V \mid \forall j \in \mathbb{N}.\ \exists i > j, \rho_i = s\}$.

Definition 2 (Strategies)**.** *A* strategy *of player* i *is a function* $\sigma_i : (V^* \cdot V_i) \to V$*, such that if* $\sigma_i(\rho) = s$ *then* $(\mathrm{last}(\rho), s) \in E$*. A* strategy profile *for the set of players* $P' \subseteq P$ *is a tuple of strategies, one for each player of* P'*.*

Let $\mathcal{S}_i(\mathcal{G})$ be the set of all strategies of player i in $\mathcal{G}$; we write $\mathcal{S}_i$ when $\mathcal{G}$ is clear from the context. We write $\mathcal{S} = \prod_{i\in P} \mathcal{S}_i$ for the set of all strategy profiles for P, and $\mathcal{S}_{-i}$ for the set of strategy profiles for all players but i. If $\sigma_{-i} = (\sigma_j)_{j\in P\setminus\{i\}} \in \mathcal{S}_{-i}$, we write (σ_i, σ_{-i}) for $(\sigma_j)_{j\in P}$. Similarly, if S is a set of profiles, S_i denotes the i-th projection of S, i.e. a set of strategies for player i. A *rectangular set* of strategy profiles is a set that can be decomposed as a Cartesian product of sets of strategies, one for each player.

A strategy profile $\sigma_P \in \mathcal{S}$ defines a unique *outcome* from state s: $Out_s(\sigma_P)$ is the run $\rho = (\rho_j)_{j\in\mathbb{N}}$ s.t. $\rho_0 = s$ and for $j \ge 0$, if $\rho_j \in V_i$, then $\rho_{j+1} = \sigma_i(\rho_{\le j})$. If S_i is a set of strategies for player i, we write $Out_s(S_i)$ for $\{\rho \mid \exists \sigma_i \in S_i, \sigma_{-i} \in \mathcal{S}_{-i}.\ Out_s(\sigma_i, \sigma_{-i}) = \rho\}$. For a tuple of sets of strategies $S_{P'}$ with $P' \subseteq P$, we write $Out_s(S_{P'}) = \bigcap_{i\in P'} Out_s(S_i)$. A strategy σ_i of player i is said to be *winning from state* s *against a rectangular set* $S_{-i} \subseteq \mathcal{S}_{-i}$, if for all $\sigma_{-i} \in S_{-i}$, $Out_s(\sigma_i, \sigma_{-i}) \in \textsc{Win}_i$. It is simply said *winning from state* s if $S_{-i} = \mathcal{S}_{-i}$. For each player i, we write $\textsc{Win}_i^s(\sigma_P)$ if $Out_s(\sigma_P) \in \textsc{Win}_i$.

2.2 Winning conditions

Winning conditions for each player are given by accepting sets either on the set of states occurring along the run, or the set of states occurring infinitely often. Particular cases are safety, reachability, and Büchi winning conditions.

- A *safety condition* is defined by a set $Bad_i \subseteq V$: $\textsc{Win}_i = (V \setminus Bad_i)^\omega$.
- A *reachability condition* is defined by a set $Good_i \subseteq V$: $\textsc{Win}_i = V^* \cdot Good_i \cdot V^\omega$.

[1] It is assumed for convinience and w.l.o.g. that each state in V has at least one outgoing edge.

- A *Büchi condition* is defined by a set $F_i \subseteq V$: $\text{WIN}_i = (V^* \cdot F_i)^\omega$.
- A *Muller condition* is given by a family $\mathcal{F}$ of sets of states: $\text{WIN}_i = \{\rho \mid \text{Inf}(\rho) \in \mathcal{F}\}$. For a succinct representation, we assume $\mathcal{F}$ is given by a Boolean circuit whose inputs are the states of V, and which evaluates to true on valuation $v_S \colon s \mapsto 1$ if $s \in S$; 0 otherwise; if, and only if, $S \in \mathcal{F}$ [11].
- A *weak Muller condition* is given by a family $\mathcal{F}$ of sets of states: $\text{WIN}_i = \{\rho \mid \text{Occ}(\rho) \in \mathcal{F}\}$. We again assume that $\mathcal{F}$ is given by a Boolean circuit.

Muller conditions generalize Büchi and other classical conditions such as parity: these can be encoded by a circuit of polynomial size [11]. Note that Muller conditions are *prefix-independent*: for any finite path h and infinite path ρ', $h \cdot \rho' \in \text{WIN}_i \Leftrightarrow \rho' \in \text{WIN}_i$. In two-player zero-sum games with circuit conditions, deciding the winner is PSPACE-complete [11]. Weak Muller conditions generalize safety and reachability.

2.3 Admissibility

Definition 3 (Dominance for strategies)**.** *Let $S = \prod_{i \in P} S_i \subseteq \mathcal{S}$ be a rectangular set of strategy profiles. Let $\sigma, \sigma' \in S_i$. Strategy σ* very weakly dominates *strategy σ' with respect to S, written $\sigma \succcurlyeq_S \sigma'$, if from all states s:*

$$\forall \tau \in S_{-i}, \text{WIN}_i^s(\sigma', \tau) \Rightarrow \text{WIN}_i^s(\sigma, \tau).$$

Strategy σ weakly dominates *strategy σ' with respect to S, written $\sigma \succ_S \sigma'$, if $\sigma \succcurlyeq_S \sigma'$ and $\neg(\sigma' \succcurlyeq_S \sigma)$. A strategy $\sigma \in S_i$ is* dominated *in S if there exists $\sigma' \in S_i$ such that $\sigma' \succ_S \sigma$. A strategy that is not dominated in S is* admissible *in S. A profile σ_P such that for all $i \in P$, σ_i is admissible is called an* admissible profile.

The set $\mathcal{S}^*$ of *iteratively admissible* strategies is obtained by iteratively eliminating dominated strategies, starting from set $\mathcal{S}$. Formally, we consider the sequence:

- $\mathcal{S}^0 = \mathcal{S}$;
- $\mathcal{S}^{n+1} = \prod_{i \in P} \{\sigma_i \in \mathcal{S}_i^n \mid \sigma_i \text{ admissible in } \mathcal{S}^n\}$.

Then $\mathcal{S}^* = \bigcap_{n \in \mathbb{N}} \mathcal{S}^n$. When for all player $i \in P$, WIN_i is ω-regular winning conditions, $\mathcal{S}^*$ is reached after finitely many iterations and is not empty [2].

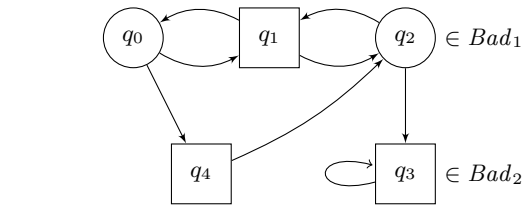
Note that our strategies are defined from all states while in [2] they are defined only for history starting from the initial state. A strategy here can be seen as a tuple of strategies (in the sense of [2]): one for each state. The set $\mathcal{S}^n$ we compute is then the cardinal product of admissible strategies from each state.

Example 1. *Figure 2(a) presents a safety game that starts in q_0. Strategies of player 1 that from q_0 go to q_4 are losing. Whereas for those that go to q_1, there is a strategy of player 2 which helps player 1 to win by playing back to q_0. Hence the former are dominated by the later, and so they are eliminated at the first elimination of dominated strategies and do not appear in $\mathcal{S}_1^1$. In the second step of iteration, if player 2 plays from q_1 to q_0, he is ensured to win if player 1 plays a strategy of $\mathcal{S}_1^1$. Therefore the strategies of player 2 that go to q_2 are dominated: there are strategies in $\mathcal{S}_1^1$ that make them lose. These latter strategies are therefore removed and do not appear in $\mathcal{S}_2^2$. The process then stabilizes: $\mathcal{S}^* = \mathcal{S}^2$.*

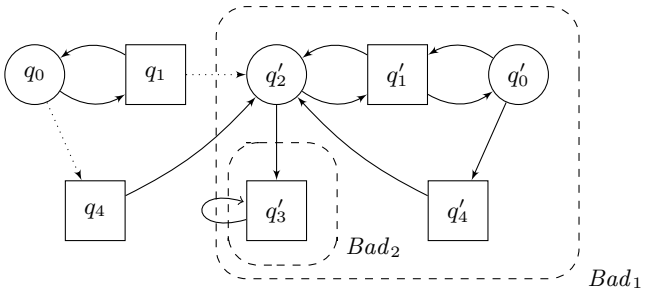
2.4 Decision problems

Winning coalition problem Given a game $\mathcal{G}$ and two subsets W, L of players, does there exist an iteratively admissible profile s.t. all players of W win the game, and all players of L lose the game (other players may either win or lose)?

Model-checking under admissibility problem Given a game and an LTL [18, 21] formula ψ, does the outcome of every iteratively admissible profile satisfy ψ?



(a) A safety game.



(b) Its unfolding. Transitions eliminated after two steps of elimination are dotted.

Figure 2: A safety game and its unfolding. States controlled by player 1 are represented with circles and states controlled by player 2 with squares. We keep this convention through the paper. The objective of player 1 is to avoid q_2 and that of player 2 is to avoid q_3.

2.5 Values

Our algorithms are based on the notion of *value* of a history. It characterizes whether a player can win (alone) or cannot win (even with the help of other players), restricting the strategies to the ones that have not been eliminated so far. This notion is also a central tool in [2] to characterize admissible strategies. However, [2] gives no practical way to compute values. We will show in this paper, that these are indeed computable.

Definition 4 (Value)**.** *The* value *of history h for player i after the n-th step of elimination, written $\text{Val}_i^n(h)$, is given by:*

- *if there is no strategy profile σ_P in $\mathcal{S}^n$ whose outcome ρ from $\text{last}(h)$ is such that $h_{<|h|-1} \cdot \rho$ is winning for player i then $\text{Val}_i^n(h) = -1$;*
- *if there is a strategy of $\sigma_i \in \mathcal{S}_i^n$ such that for all strategy profiles σ_{-i} in $\mathcal{S}_{-i}^n$, the outcome ρ of (σ_i, σ_{-i}) from s is such that $h_{<|h|-1} \cdot \rho$ is winning for player i then $\text{Val}_i^n(h) = 1$;*
- *otherwise $\text{Val}_i^n(h) = 0$;*

By convention, $\text{Val}_i^{-1}(h) = 0$.

The following lemma illustrates a property of values and admissible strategies:

Lemma 1. *For all $n \in \mathbb{N}$, if $\text{last}(h) \in V_i$ and $\sigma_i \in \mathcal{S}_i^{n+1}$ then $\text{Val}_i^n(h) = \text{Val}_i^n(h \cdot \sigma_i(h))$.*

Hence a player that plays according to an admissible strategy cannot go to a state that changes the value of the current history. This condition is not always sufficient, but in the following sections

we characterize runs of admissible strategies relying on this notion of value.

3. Safety objectives

The main result of this section is a PSPACE algorithm for the winning coalition problem in safety games. This is based on a notion of dominance for transitions. We show that by iteratively removing dominated transitions of the game, we describe exactly the set of admissible strategies.

3.1 Making explicit the losing players

Let h be an history, the players *losing on* h are the players in

$$\lambda(h) = \{i \in P \mid \exists k < |h|,\ h_k \in Bad_i\}.$$

Proposition 1. *For safety winning conditions, the value of a history h only depends on $\lambda(h)$ and $\mathrm{last}(h)$.*

We can therefore write $\mathrm{Val}_i^n(\lambda(h), s)$ for $\mathrm{Val}_i^n(h)$, when $\mathrm{last}(h) = s$. We encode the set $\lambda(h)$ of losing players in the state of the game, at the price of an exponential blowup (in the number of players). The new game has states in $2^P \times V$ and set of transitions $(\lambda, s) \to (\lambda \cup \{i \mid s' \in Bad_i\}, s')$ for any $\lambda \subseteq P$, if $s \to s'$. In this partially unfolded game, the value depends only on the current state, hence is written $\mathrm{Val}_i^n(s)$. For example, the game of Figure 2(a) is unfolded as the game of Figure 2(b); states $q'_0, \ldots, q'_4$ are states where player 1 has already lost. Now, let us assume for the remainder of this section that the losing players in the game $\mathcal{G}$ are explicit.

3.2 Dominance of transitions

In the case of safety winning condition, the necessary condition of Lemma 1 becomes sufficient, as shown below. This yields a local notion of dominance, that can be expressed directly on transitions:

Definition 5. *We write T_i^n for the set of transitions $s \to s' \in E$, such that s is controlled by player i and $\mathrm{Val}_i^n(s) > \mathrm{Val}_i^n(s')$. Such transitions are said to be* dominated *after the n-th step of elimination. We write T^n for the union of all T_i^n.*

Definition 6 (Subgame)**.** *Let $\mathcal{G} = \langle P, V, E, \mathrm{WIN}_P\rangle$ be a game and $T \subseteq E$ a set of transitions. If each state $s \in V$ has at least one successor by $E \setminus T$, the game $\mathcal{G} \setminus T = \langle P, V, E \setminus T, \mathrm{WIN}_P\rangle$ is called a* subgame *of $\mathcal{G}$. We write $\mathcal{S}_i(\mathcal{G} \setminus T)$ the set of strategies $\sigma_i \in \mathcal{S}_i(\mathcal{G})$ such that for all history h of $\mathcal{G} \setminus T$, if $\mathrm{last}(h) \in V_i$ then $(\mathrm{last}(h), \sigma_i(h)) \notin T$.*

This notion yields a polynomial procedure in the size of the game where losing players are explicit, to compute the set of all iteratively admissible strategies, described in Algorithm 1. The loop is executed at most $|E|$ times, where $|E|$ is the number of transitions in the partially unfolded game.

However, this procedure assumes that the information of which players have already violated their safety condition is encoded in the state. So in the general case, the procedure has a complexity which is exponential in the number of players and polynomial in the number of states of the game. In the case of the *winning coalition problem*, we can however reduce this complexity to PSPACE.

We now show the correctness of the procedure. We first prove a link between the notions of dominance for strategies and for transitions. Note that since all states s have at least one successor with a value greater or equal to that of s, removing transitions of T_i^n yield what we call a subgame.

Proposition 2. *All admissible strategies w.r.t. $\mathcal{S}^n$ of player i are strategies of $\mathcal{S}_i(\mathcal{G} \setminus T_i^n)$.*

Proof. We show that if player i plays a strategy σ_i admissible w.r.t. $\mathcal{S}^n$, i.e. $\sigma_i \in \mathcal{S}_i^{n+1}$, then the value cannot decrease on a transition controled by player i. Let $\rho \in Out(\sigma_i, \sigma_{-i})$ with $\sigma_{-i} \in \mathcal{S}_{-i}^n$ and $\sigma_i \in \mathcal{S}_i^{n+1}$, and $\rho_k \in V_i$. Let $s' = \sigma_i(\rho_{\leq k})$:

- If $\mathrm{Val}_i^n(\rho_k) = 1$, then σ_i has to be winning against all strategies of $\mathcal{S}_{-i}^n$, otherwise it would be weakly dominated by such a strategy. Since there is no such strategy from a state with value $\mathrm{Val}_i^n \leq 0$, $\mathrm{Val}_i^n(s') = 1$.
- If $\mathrm{Val}_i^n(s) = 0$, then there is a profile $\sigma_{-i} \in \mathcal{S}_{-i}^n$ such that $\rho = Out(\sigma_i, \sigma_{-i}) \in \mathrm{WIN}_i$. Note that $h \cdot s \cdot s' \trianglelefteq \rho$. If $\mathrm{Val}_i^n(s') = -1$, there can be no such profile, thus $\mathrm{Val}_i^n(s') \geq 0$.
- If $\mathrm{Val}_i^n(s) = -1$, the value cannot decrease. □

Algorithm 1: Computing the set of iteratively admissible strategies

```
n := 0 ; T_i^{-1} := ∅ ;
repeat
    forall the s ∈ V do
        if there is a winning strategy for player i from s in
        G \ T^{n-1} then Val_i^n(s) := 1;
        else if there is no winning run for player i from s in
        G \ T^{n-1} then Val_i^n(s) := −1;
        else Val_i^n(s) := 0;
    forall the i ∈ P do
        T_i^n := T_i^{n-1} ∪ {(s, s') ∈ E | s ∈ V_i ∧ Val_i^n(s) >
        Val_i^n(s')};
    n := n + 1 ;
until ∀i ∈ P. T_i^n = T_i^{n-1};
```

Example 2. *In Figure 2(a), initially, q_4 has value -1 for player 1, but q_0 has value 0 since it is possible to loop in q_1 and q_0 (if player 2 helps). So, the transition to state q_4 is dominated and removed at the first iteration. Then, player 2 has a winning strategy from q_1, by always going back to q_0, whereas the state q'_2 has value 0 for him. Hence $q_1 \to q'_2$ is removed after this iteration. The fix-point is obtained at that step, it is represented in Figure 2(b).*

We have seen that removing dominated transitions only removes strictly dominated strategies. The converse is also true, all strategies that remain are not dominated:

Proposition 3. *All strategies of $\mathcal{S}_i^n \cap \mathcal{S}_i(\mathcal{G} \setminus T_i^n)$ are admissible with respect to $\mathcal{S}^n$.*

Proof. Let $\sigma_i, \sigma'_i \in \mathcal{S}_i^n \cap \mathcal{S}_i(\mathcal{G} \setminus T_i^n)$ and assume $\sigma'_i \succ_{\mathcal{S}^n} \sigma_i$. Then there is a state s and strategy profile $\sigma_{-i} \in \mathcal{S}_{-i}^n$ such that $\mathrm{WIN}_i^s(\sigma'_i, \sigma_{-i}) \wedge \neg\mathrm{WIN}_i^s(\sigma_i, \sigma_{-i})$. Let $\rho = Out_s(\sigma_i, \sigma_{-i})$ and $\rho' = Out_s(\sigma'_i, \sigma_{-i})$. Consider the first position where these runs differ: write $\rho = w \cdot s' \cdot s_2 \cdot w'$ and $\rho' = w \cdot s' \cdot s_1 \cdot w''$. Note that s' belongs to player i.

First remark that since $\mathrm{WIN}_i(\sigma'_i, \sigma_{-i})$, it is clear that $\mathrm{Val}_i^n(s_1) \geq 0$. Moreover, since $s' \to s_1$ and $s \to s_2$ do not belong to T_i^n, states s', s_1 and s_2 must have the same value.

Assume $\mathrm{Val}_i^n(s') = 0$. We show that there is a profile[2] $\sigma_{-i}^2 \in \mathcal{S}_{-i}^n$ such that $\mathrm{WIN}_i(\sigma_i, \sigma_{-i}^2)$ from s_2. Let h be a history such that $\mathrm{last}(h) \notin V_i$, if for all $\sigma_{-i}^2 \in \mathcal{S}_{-i}^n$, $\mathrm{Val}_i^n(\sigma_{-i}^2(h)) = -1$ then $\mathrm{Val}_i^n(\mathrm{last}(h)) = -1$. Therefore it is possible to define a strategy profile $\sigma_{-i}^2 \in \mathcal{S}_{-i}^n$ that never decreases the value from 0 or 1 to -1. The strategy σ_i itself does not decrease the value of

[2] Although the definition of the value yields the existence of a profile winning for i, it remains to be shown that there is such profile where i plays strategy σ_i.

player i because it does not take transitions of T_i^n. So the outcome of $(\sigma_i, \sigma^2_{-i})$ never reaches a state of value -1. Hence it never reaches a state in Bad_i and therefore it is winning for player i. Now, $\mathrm{Val}_i^n(s_1) = 0$ so there is no winning strategy for player i from s_1 against all strategies of $\mathcal{S}^n_{-i}$. Then there exists a strategy profile $\sigma^1_{-i} \in \mathcal{S}^n_{-i}$ such that σ'_i loses from s_1. Now consider strategy profile σ'_{-i} that plays like σ_{-i} if the play does not start with w, then σ^1_{-i} after s_1 and σ^2_{-i} after s_2. Given a history h:

$$\sigma'_{-i}(h) = \begin{cases} \sigma^1_{-i}(h') & \text{if } w \cdot s_1 \trianglelefteq h \text{ and } w \cdot s_1 \cdot h' = h \\ \sigma^2_{-i}(h') & \text{if } w \cdot s_2 \trianglelefteq h \text{ and } w \cdot s_2 \cdot h' = h \\ \sigma_{-i}(h) & \text{otherwise} \end{cases}$$

Clearly we have $\mathrm{WIN}_i^s(\sigma_i, \sigma'_{-i}) \wedge \neg \mathrm{WIN}_i^s(\sigma'_i, \sigma'_{-i})$, which contradicts $\sigma'_i \succcurlyeq_{\mathcal{S}^n} \sigma_i$.

Now assume $\mathrm{Val}_i^n(s_2) = 1$. Since $\neg \mathrm{WIN}_i(\sigma_i, \sigma_{-i})$, the produced outcome ρ reaches a state of Bad_i, hence the value of states along ρ is -1 after some point. Consider the first state ρ_k which has value smaller or equal to 0: $k = \min_{k'}\{\rho_{k'} \mid \mathrm{Val}_i^n(\rho_{k'}) \leq 0\}$. The state ρ_{k-1} has value 1, it is necessarily controlled by a player j different from player i, since transitions of T_i^n cannot be taken by σ_i. Since there exists a winning strategy $\sigma_i \in \mathcal{S}_i^n$ from ρ_{k-1} against strategies of $\mathcal{S}^n_{-i}$, then this strategy is still winning at ρ_k. Therefore $\mathrm{Val}_i^n(\rho_k) = 1$, which is a contradiction. □

3.3 The winning coalition problem for safety objectives

Theorem 1. *The winning coalition problem with safety winning conditions is* PSPACE*-complete. However, if the number of players is fixed, the problem becomes* P*-complete.*

Proof sketch. To decide the winning coalition problem, only the existence of a particular profile is required and the explicit construction of the unfolded graph is not necessary. By guessing a lasso path produced by such a profile, and checking recursively that it does not contain dominated transition, we get PSPACE membership.

The hardness proof is done by encoding instances of QSAT. Instead of detailing the whole construction, we illustrate it on an example in Figure 3 for the following formula $\mu = \exists x_1 \forall x_2 \exists x_3 (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$. There are two players x and $\neg x$ for each variable x, plus two players `Eve` and `Adam`. The moves of `Eve` and `Adam` in the left part of the game determine a valuation: x_i is said to be true if Bad_i was reached. If a player x_i has not yet lost, in the right part of the game, it is better for this player to visit the losing state of `Eve` than its own. Hence, at the first step of elimination, the edges removed in the unfolded game correspond to the ones going to a state Bad_{x_i} if x_i is false (and $\neg x_i$ if x_i is true). At the second step of elimination, `Eve` should avoid whenever possible, states corresponding to a literal whose valuation is not true, since those states will necessarily lead to $Bad_\exists$. If the valuation satisfies each clause, then she has the possibility to do so, and one admissible profile is winning for her: so μ is true if, and only if, there is a admissible profile where `Eve` is winning. □

4. Muller objectives

Our main result for this section is stated in the following theorem:

Theorem 2. *The winning coalition problem with a Muller condition for each player is* PSPACE*-complete. The problem is* PSPACE*-hard even when restricted to two players.*

PSPACE-hardness follows from PSPACE-hardness of two-player games with Muller conditions [7].

The idea of the algorithm is to construct a graph representation of the outcomes of admissible profiles. While the construction also relies on the notion of value, it is more involved than for safety conditions.

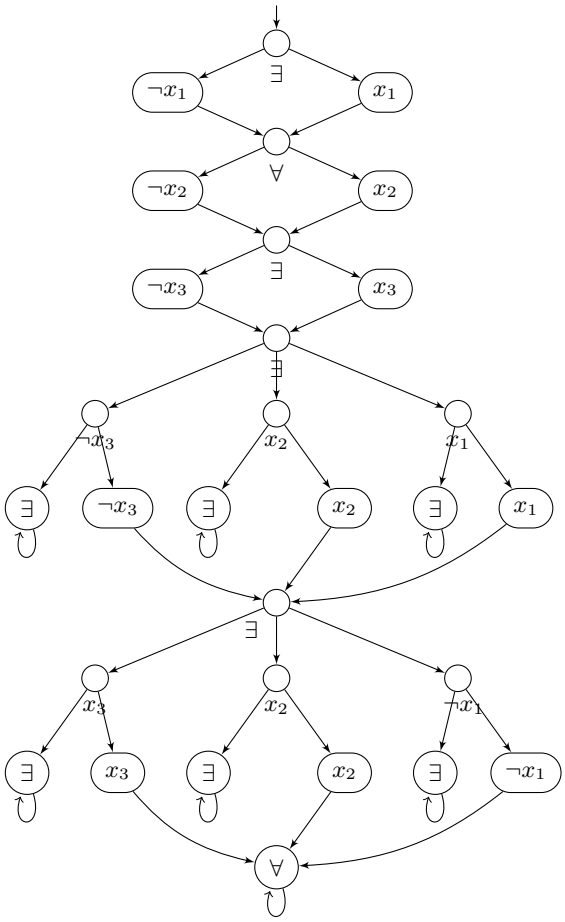


Figure 3: Game $\mathcal{G}_\mu$ with $\mu = \exists x_1 \forall x_2 \exists x_3 (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$. A label y *inside* a state s denotes that $s \in Bad_y$; a label y *below* a state s denotes that $s \in V_y$. Note that `Eve` is abbreviated to $\exists$ and `Adam` is abbreviated to $\forall$.

4.1 Characterizing outcomes of admissible strategies using the sequence of their value

Proposition 4. *For prefix-independent objectives, the value depends only on the last state of the history.*

Hence we write $\mathrm{Val}_j^n(s)$ instead of $\mathrm{Val}_j^n(h)$ when $\mathrm{last}(h) = s$. Since Val_j^n is here a function from V to $\{-1, 0, 1\}$, it can be extended to runs: $\mathrm{ValR}_j^n(\rho)$ is the word $w \in \{-1, 0, 1\}^\omega$ such that $w_k = \mathrm{Val}_j^n(\rho_k)$ for all k.

The following lemma shows that in terms of the sequence of value for a player, we can distinguish three types of plays that are outcome of admissible strategies.

Lemma 2. *Let $s \in V$, $\rho \in Out_s(\mathcal{S}^n)$, and $i \in P$.*

If $\rho \in Out_s(\mathcal{S}_i^{n+1})$ then $\mathrm{ValR}_i^n(\rho) \in 0^*1^\omega + 0^\omega + 0^*(-1)^\omega$.

Now, we characterize outcomes of admissible strategies according to whether they end with value 1, -1, or 0. We do so for each player individually.

Value 1 To be admissible in $\mathcal{S}^n$ from a state of value 1, a strategy has to be winning against all strategies of $\mathcal{S}^n$:

Lemma 3. *Let $s \in V$, $i \in P$ and $\rho \in Out_s(\mathcal{S}^n)$ be such that $\mathrm{ValR}_i^n(\rho) \in 0^*1^\omega$.*

$$\rho \in Out_s(\mathcal{S}_i^{n+1}) \text{ if, and only if, } \rho \in \mathrm{WIN}_i.$$

Value -1 If the run reaches a state of value -1, then, from there, there is no possibility of winning, so any strategy is admissible but the state of value -1 must not have been reached by player i's fault:

Lemma 4. *Let $s \in V$, $i \in P$ and $\rho \in Out_s(\mathcal{S}^n)$ be such that $\mathrm{ValR}_i^n(\rho) \in 0^*(-1)^\omega$. Let k be the index such that $\mathrm{Val}_i^n(\rho_k) = 0 \wedge \mathrm{Val}_i^n(\rho_{k+1}) = -1$.*

$$\rho \in Out_s(\mathcal{S}_i^{n+1}) \text{ if and only if, } \rho_k \notin V_i.$$

Example 3. *As an illustration of Lemmata 3 and 4, consider the left game in Figure 4. Both runs $s_0 \cdot s_1 \cdot s_2 \cdot Good_1^\omega$ and $s_0 \cdot s_1 \cdot s_2 \cdot s_3^\omega$ are outcomes of non dominated strategies of player 1. Indeed, the play that goes to $Good_1$ is winning and the value of the path that goes to s_3 belongs to $0^*(-1)^\omega$ and the value decreases on a transition by player 2.*

Value 0 This case is more involved. From a state of value 0, an admissible strategy of player i should allow a winning run for player i with the help of other players. We write H_i^n for set of states s controlled by a player $j \neq i$ that have at least two successors that *(i)* have value 0 or 1 for player i and *(ii)* have the same value for player j than s after iteration $n-1$. Formally, for $n \geq 0$, the "Help!"-states of player i are defined as:

$$H_i^n = \bigcup_{j \in P \setminus \{i\}} \left\{ s \in V_j \;\middle|\; \begin{array}{l} \exists s', s'',\ s' \neq s'' \\ \wedge\ s \to s' \wedge s \to s'' \\ \wedge\ \mathrm{Val}_i^n(s') \geq 0 \\ \wedge\ \mathrm{Val}_i^n(s'') \geq 0 \\ \wedge\ \mathrm{Val}_j^{n-1}(s) = \mathrm{Val}_j^{n-1}(s') \\ \wedge\ \mathrm{Val}_j^{n-1}(s) = \mathrm{Val}_j^{n-1}(s'') \end{array} \right\}$$

These states have the following property.

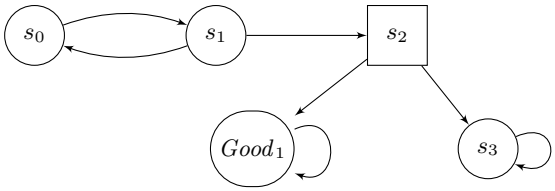
Lemma 5. *Let $s \in V$, $i \in P$ and $\rho \in Out_s(\mathcal{S}^n)$ be such that $\mathrm{ValR}_i^n(\rho) = 0^\omega$.*

$$\rho \in Out_s(\mathcal{S}_i^{n+1}) \text{ if, and only if, } \rho \in \mathrm{WIN}_i \text{ or } \mathrm{Inf}(\rho) \cap H_i^n \neq \emptyset$$

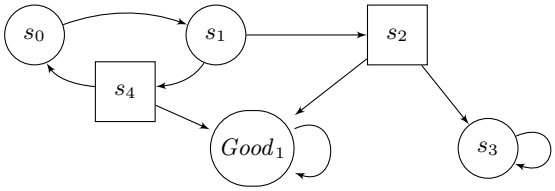
Proof sketch. Let $\sigma_P \in \mathcal{S}^n$ such that $\rho = Out_s(\sigma_P)$. Assume first that there is only a finite number of visits to H_i^n and that ρ is not winning for player i. Let k be the greatest index such that $\rho_k \in H_i^n$. After the prefix $\rho_{\leq k}$ no profile of $\mathcal{S}^n$ can make σ_i win, since no "Help!"-state is visited. While since $\mathrm{Val}_i^n(\rho_k) = 0$ there is a strategy profile σ_P'' of $\mathcal{S}^n$ whose outcome is winning for i from ρ_k. Hence the strategy that follows σ_i until the prefix $\rho_{\leq k}$ and then switches to σ_i'' weakly dominates σ_i, and $\sigma_i \notin \mathcal{S}_i^{n+1}$.

Assume now that either $\rho \in \mathrm{WIN}_i$ or there is an infinite number of indexes k such that $\rho_k \in H_i^n$. We define a strategy σ_i that follows ρ when possible and otherwise plays a non-dominated strategy. Assume that there is a strategy profile $\sigma_P' \in \mathcal{S}^n$ such that $\rho' = Out(\sigma_{-i}', \sigma_i') \in \mathrm{WIN}_i$ and $\rho'' = Out(\sigma_{-i}', \sigma_i) \notin \mathrm{WIN}_i$. We show that there is a profile that makes σ_i win and σ_i' lose, so that σ_i' does not weakly dominate σ_i.

For instance in the case where ρ'' continues to follow ρ after it has diverged from ρ', we know it will encounter a "Help!"-state ρ_k. From there there is a strategy profile σ_{-i}^d that is in $\mathcal{S}_{-i}^n$ thanks to the condition $\mathrm{Val}_j^{n-1}(\rho_k) = \mathrm{Val}_j^{n-1}(s'')$ with $s'' \neq \rho_{k+1}$, and which is winning for σ_i thanks to the condition $\mathrm{Val}_i^n(s'') \geq 0$, and the fact that σ_i reverts to a non-dominated strategy outside of ρ. This strategy profile can moreover be made losing for σ_i' from the point where it deviates from ρ, since this deviation is on a state controled by player i and its value for player i is 0. □



(a) A first reachability game



(b) A second reachability game

Figure 4: Two games. The goal for player 1 is to reach $Good_1$.

Example 4. *As an illustration of Lemma 5, consider the two games in Figure 4. In the game of Figure 4(a), a strategy of player 1 that stays in the loop $(s_0 \cdot s_1)$ forever is weakly dominated. The value of this run is 0^ω and visits no "Help!"-state, since in that game $H_1^0 = \emptyset$. Intuitively, the strategy is dominated because it has no chance of winning, while getting out after k steps can be winning if player 2 helps.*

However, in the game of Figure 4(b), the strategy that always chooses s_4 from s_1 is admissible. The run $(s_0 \cdot s_1 \cdot s_4)^\omega$ also has value 0^ω, but this time $H_1^0 = \{s_4\}$, which is visited infinitely often by the loop.

4.2 Automata for $Out(\mathcal{S}^n)$

Our goal is to obtain an automaton which recognizes $Out(\mathcal{S}^n)$. We decompose this construction for each player by noting that:

$$Out(\mathcal{S}^{n-1}) \cap \bigcap_{i \in P} Out(\mathcal{S}_{-i}^{n-1}, \mathcal{S}_i^n) = Out(\mathcal{S}^n)$$

From the characterization of admissible strategies w.r.t. values of state, we define an automaton $\mathcal{A}_i^n$ that selects $Out(\mathcal{S}_i^n)$ within $Out(\mathcal{S}^{n-1})$, i.e. such that $\mathcal{L}(\mathcal{A}_i^n) \cap Out(\mathcal{S}^{n-1})$ is equal to $Out(\mathcal{S}_i^n, \mathcal{S}_{-i}^{n-1})$. This construction relies on values at previous iterations. For $n > 0$, $\mathcal{A}_i^n$ is the automaton where:

- The set of states is V, i.e. the same states as in $\mathcal{G}$;
- From the transitions in $\mathcal{G}$ we remove those controlled by player i that decrease his value. Formally:

$$T = E \setminus \{(s, s') \mid s \in V_i \wedge \mathrm{Val}_i^{n-1}(s) > \mathrm{Val}_i^{n-1}(s')\}.$$

- A run ρ is accepted by $\mathcal{A}_i^n$ if one of the following conditions is satisfied:
  1. $\mathrm{ValR}_i^{n-1}(\rho) \in 0^*(-1)^\omega$;
  2. $\mathrm{ValR}_i^{n-1}(\rho) \in 0^*1^\omega$ and $\rho \in \mathrm{WIN}_i$;
  3. $\mathrm{ValR}_i^{n-1}(\rho) \in 0^\omega$ and $\rho \in \mathrm{WIN}_i$ or $\rho \in (V^* H_i^{n-1})^\omega$.

Note that the structure of the automaton is a subgame of $\mathcal{G}$. Since the transitions of $\mathcal{G}$ bear no label, the "language" $\mathcal{L}(\mathcal{A})$ of an automaton $\mathcal{A}$ is here considered to be the set of accepting runs. The following lemma shows the relation between automaton $\mathcal{A}_i^n$ and outcomes of admissible strategies at step n.

Lemma 6. $Out(\mathcal{S}^{n-1}) \cap \mathcal{L}(\mathcal{A}_i^n) = Out(\mathcal{S}_i^n, \mathcal{S}_{-i}^{n-1})$

Sketch of proof. For all run ρ, thanks to Lemma 2, there are three cases to distinguish, based on $\mathrm{ValR}_i^{n-1}(\rho)$. The case 0^*1^ω is solved by Lemma 3, the case $0^*(-1)^\omega$ by Lemma 4, and the case 0^ω by Lemma 5. □

The following property is a direct consequence of Lemma 6.

Lemma 7. $Out(\mathcal{S}^{n-1}) \cap \bigcap_{i\in P} \mathcal{L}(\mathcal{A}_i^n) = Out(\mathcal{S}^n)$

As $Out(\mathcal{S}^0)$ is the set of all runs over $\mathcal{G}$, the previous lemma allows the construction of $Out(\mathcal{S}^n)$ by induction, for any n. Note that the size of the automaton accepting $Out(\mathcal{S}^n)$ does not grow since all automata in the intersection share the same structure as $\mathcal{G}$ (although some edges have been removed). The accepting condition, however, becomes more and more complex. Nonetheless, if for all $j \in P$, WIN_j is a Muller condition given by a Boolean circuit, the condition of the automaton accepting $Out(\mathcal{S}^n)$ is a Boolean combination of such conditions, thus it is expressible by a Boolean circuit, of polynomial size. For example, condition $\mathrm{ValR}_i^{n-1}(\rho) \in 0^*1^\omega$ can be expressed by a circuit that is the disjunction of all states of value 1 at step $n-1$.

4.3 Inductive computation of values

Now, we show how to compute the values of the states for all players. Initially, at "iteration -1", all values are assumed to be 0, and $\mathcal{S}^{-1} = \mathcal{S}$. Computing the values at the next iteration relies on solving two-player zero-sum games with objectives based on outcomes of admissible strategies. For example, in order to decide whether a state s has value 1 for player i at iteration 0, i.e. whether $\mathrm{Val}_i^0(s) = 1$, one must decide whether player i has a winning strategy from s when playing against all other players. This corresponds to the game with winning condition WIN_i where vertices of all players but i belong to a single opponent. To decide whether $\mathrm{Val}_i^0(s) > -1$, all players try together to make i win. Therefore this is a one-player game (or emptiness check) on $\mathcal{G}$ with condition WIN_i. All the other states have value -1.

This idea is extended to subsequent iterations, but the objectives of the games become more complex in order to take into account the previous iterations: the objectives need to enforce that only outcomes of admissible strategies (at previous iterations) are played. Hence the construction relies on the automata $\mathcal{A}_i^n$ built above. Assuming that winning conditions for all players are Muller conditions, this yields a polynomial space algorithm to compute the values.

4.3.1 Characterizing states of value -1

A state s has a value > -1 for player i at step n if there is a strategy profile $\sigma_P \in \mathcal{S}^n$ s.t. $Out_s(\sigma_P) \in \mathrm{WIN}_i$. This is expressed by: $\exists \sigma_P \in \mathcal{S}.\ \sigma_P \in \mathcal{S}^n \wedge Out_s(\sigma_P) \in \mathrm{WIN}_i$. This prompts the definition of the following objective:

$$\Psi_i^n(s) = Out_s(\mathcal{S}^n) \cap \mathrm{WIN}_i.$$

Lemma 8. *If $\Psi_i^n(s) \neq \emptyset$, then $\exists \sigma_P \in \mathcal{S}^n : Out_s(\sigma_P) \in \mathrm{WIN}_i$.*

Therefore, since the set $Out_s(\mathcal{S}^n)$ and WIN_i can both be expressed as the language of an automaton, testing whether a state has value -1 boils down to check emptiness.

4.3.2 Characterizing states of value 1

A state s has value 1 for player i at step n if he has a strategy σ_i in $\mathcal{S}_i^n$ such that for all strategies σ_{-i} in $\mathcal{S}_{-i}^n$, $Out_s(\sigma_i, \sigma_{-i}) \in \mathrm{WIN}_i$. This is expressed by the formula:

$$\exists \sigma_i \in \mathcal{S}_i,\ \forall \sigma_{-i} \in$$

$$\mathcal{S}_{-i},\ (\sigma_i \in \mathcal{S}_i^n \wedge (\sigma_{-i} \in \mathcal{S}_{-i}^n \Rightarrow \mathrm{WIN}_i^s(\sigma_i, \sigma_{-i}))).$$

This prompts the definition of the following objective:

$$\Omega_i^n(s) = Out_s(\mathcal{S}_i^n) \cap (Out_s(\mathcal{S}^n) \Rightarrow \mathrm{WIN}_i).$$

We show that there is indeed a correspondence between this objective Ω_i^n and states s such that $\mathrm{Val}_i^n(s) = 1$.

Proposition 5. *A strategy of player i is a strategy of $\mathcal{S}_i^n$ which is winning from state s against all strategies of $\mathcal{S}_{-i}^n$ if, and only if, it is winning for objective $\Omega_i^n(s)$.*

Note that we cannot directly obtain a description of $\Omega_i^n(s)$ in term of automata using Lemma 6, since we need to recognize $Out_s(\mathcal{S}_i^n)$. To characterize runs of $Out_s(\mathcal{S}_i^n)$ we will both use a condition on transition similar to the safety case, and a condition on the long run. Recall that $T^n = \cup_{i\in P} T_i^n$, where:

$$T_i^n = \{(s, s') \in E \mid s \in V_i \wedge \mathrm{Val}_i^{n-1}(\rho_{k+1}) < \mathrm{Val}_i^{n-1}(\rho_k)\}.$$

If a player $j \neq i$ takes a transition that decreases its value, then we immediately know that player i wins for objective Ω_i^n by playing an admissible strategy. We thus define:

$$C_i^n = \{\rho \in V^\omega \mid \exists k,\ \exists j \neq i,\ (\rho_k, \rho_{k+1}) \in T_j^n\};$$

We also define $C_i^{\leq n} = \bigcup_{m=1}^n C_i^m$ and $Out(\mathcal{G} \setminus T^{\leq n}) = \bigcap_{m=1}^n Out(\mathcal{G} \setminus T^n)$. Objective Ω_i^n can be further decomposed with respect to C_i^n and T^n.

Proposition 6. *Player i, has a winning strategy for $\Omega_i^n(s)$ in $\mathcal{G}$ if, and only, if he has one for:*

$$\Theta_i^n(s) = C_i^{\leq n} \cup \left(Out(\mathcal{G} \setminus T^{\leq n}) \cap \Omega_i^n(s)\right).$$

4.3.3 Using the automata

We can now use the acceptance conditions of automata $(\mathcal{A}_j^m)_{m\leq n, j\in P}$ to rewrite Θ_i^n. If we expand the definition of Ω_i^n in Θ_i^n, we obtain that $\Theta_i^n(s)$ equals:

$$C_i^{\leq n} \cup \left(Out(\mathcal{G} \setminus T^{\leq n}) \cap Out_s(\mathcal{S}_i^n) \cap (Out_s(\mathcal{S}^n) \Rightarrow \mathrm{WIN}_i)\right).$$

The set $C_i^{\leq n}$ and WIN_i are easily definable by an automaton. An automata for $Out(\mathcal{S}^n)$ is constructed through an intersection of automata $(\mathcal{A}_j^m)_{m\leq n, j\in P}$, as shown in Lemma 7. Hence, we now have to construct automata recognizing and $Out(\mathcal{S}_i^n) \cap Out(\mathcal{G} \setminus T^{\leq n})$.

Computing $Out(\mathcal{S}_i^n) \cap Out(\mathcal{G} \setminus T^{\leq n})$ This construction is also based on the automaton $\mathcal{A}_i^n$. For this, we show that it can be defined through a combination of $Out(\mathcal{S}_i^n, \mathcal{S}_{-i}^{n-1})$ and of outcomes of admissible strategies at the previous iteration. Namely:

Lemma 9. *$Out(\mathcal{S}_i^n) \cap Out(\mathcal{G} \setminus T^{\leq n})$ is equal to:*
$Out(\mathcal{S}_i^{n-1}) \cap Out(\mathcal{G} \setminus T^{\leq n-1})_i \cap \left(Out(\mathcal{S}^{n-1}) \Rightarrow \mathcal{L}(\mathcal{A}_i^n)\right) \cap Out(\mathcal{G} \setminus T^{\leq n})$.

The previous lemma provides a recurrence relation to compute the intersection $Out(\mathcal{S}_i^n) \cap Out(\mathcal{G} \setminus T^n)$, with base case being all the runs of $\mathcal{G}$.

4.4 Bounding the number of iteration phases

We can show that $\mathrm{Val}_i^n(s) = 1 \Rightarrow \mathrm{Val}_i^{n+1}(s) = 1$ as winning strategies are admissible and $\mathcal{S}_{-i}^{n+1} \subseteq \mathcal{S}_{-i}^n$ implies that winning strategies remain winning. Similarly, we can show $\mathrm{Val}_i^n(s) = -1 \Rightarrow \mathrm{Val}_i^{n+1}(s) = -1$. So the number of times that the value function changes is bounded by $|P| \cdot |V|$. This allows us to bound the number of iterations necessary to reach the fix-point:

Proposition 7. $\mathcal{S}^* = \mathcal{S}^{|P|\cdot|V|}$.

4.5 Algorithm for Muller conditions

The above construction yields procedures to compute the values, and in turn to solve the winning coalition problem. The algorithm works as follow, starting from $n = 0$ and incrementing it at each loop:

1. compute an automaton representing $Out(\mathcal{S}^n)$ with the help of $\mathcal{L}(\mathcal{A}^n)$, as described in Section 4.2;
2. compute for each player i the objective Θ_i^n as explained in Sections 4.3.3 and 4.3.3;
3. compute for each player i the objective Ψ_i^n (see Section 4.3.1);
4. compute for each player i and state s, $\mathrm{Val}_i^n(s)$: we will show how to do this in the remainder of this section;
5. compute for each player i the set H_i^n.

Proposition 8. *Checking whether* $\mathrm{Val}_i^n(s) = 1$ *is in* PSPACE.

Proof. We define the following two-player game $\mathcal{G}_i^n$ as follows. $\mathcal{G}_i^n$ has the structure of $\mathcal{G}$, except that the edges corresponding to a player decreasing his own value at any previous iteration have been removed (thus avoiding $T^{\leq n}$ syntactically). The players are `Eve` and `Adam`, the states of `Eve` are V_i and the states of `Adam` are $\bigcup_{j\neq i} V_j$. The winning condition for `Eve` is Θ_i^n, which is a Muller condition.

If `Eve` has a winning strategy starting from state s, then $\mathrm{Val}_i^n(s) = 1$, as shown by Propositions 5 and 6. Recall that for Muller condition, deciding whether there exists a winning strategy for `Eve` is in PSPACE [11]. □

Proposition 9. *Checking whether* $\mathrm{Val}_i^n(s) = -1$ *is in* coNP.

Proof. This is equivalent to the emptiness of Ψ_i^n when starting from state s. Again, Ψ_i^n is a circuit condition on $\mathcal{G}$ where edges decreasing the value of their owner are removed. Testing non-emptiness amounts to guessing a lasso path starting from s, and checking that the states visited infinitely often correspond to a set of $\mathcal{F}$. This NP algorithm answers true when $\mathrm{Val}_i^n(s) \neq -1$. We therefore have a coNP algorithm. □

Proof of Theorem 2. Let W, L be the set of players that must win and lose, respectively, in the instance of the winning coalition problem. The winning coalition problem is thus equivalent to the non-emptiness of

$$\Phi = Out(\mathcal{S}^*) \cap \bigcap_{i\in W} \mathrm{WIN}_i \cap \bigcap_{i\in L} \neg\mathrm{WIN}_i$$

over $\mathcal{G}$, which is a Muller condition. This check is done in NP.

This however requires the computation of the condition corresponding to $Out(\mathcal{S}^*)$, hence of the values for all intermediate iterations before the fix-point. Each iteration means solving a Muller game, which is done in polynomial space. Moreover, by Proposition 7, there are at most $|P| \cdot |V|$ iterations. Thus the overall complexity is in PSPACE. □

5. Büchi objectives

In this section, We assume that each winning condition WIN_i is given by a Büchi set F_i. A careful analysis of condition Θ_i^n shows that it can be reformulated as a parity condition. Hence computing the value of a state boils down to solving two-player parity games. Parity games are known to be in UP ∩ coUP, but the question whether a polynomial algorithm exists for parity games has been open for several years [9, 12]. Our algorithm thus works in polynomial time with call to oracles in NP ∩ coNP, hence is also in NP ∩ coNP [4]. This idea is the basis for the proof of Theorem 3, which is detailed in the remainder of the section.

Theorem 3. *The winning coalition problem with Büchi objectives is in* NP ∩ coNP. *Moreover, if there exists a polynomial algorithm for solving two-player parity games, then the winning coalition problem with Büchi objectives is in* P.

Definition 7. *A* parity condition *is given by a coloring function* $\chi : V \to \{0, \ldots, M\}$ *with* $M \in \mathbb{N}$. *Accepted runs with respect to the coloring functions are the ones where the maximal color visited infinitely often is even:*

$$\mathrm{WIN}_\chi = \{\rho \mid \max(\mathrm{Inf}(\chi(\rho))) \text{ is even}\}.$$

5.0.1 Expressing Θ_i as a parity condition

First note that the acceptance condition of $\mathcal{L}(\mathcal{A}_i^n)$ can be expressed by the following Büchi set, assuming that we remove all the edges that decrease the value of player i:

$$\begin{aligned} K_i^n =& \{s \mid \mathrm{Val}_i^{n-1}(s) = -1\} \cup \{s \mid \mathrm{Val}_i^{n-1}(s) = 1 \wedge s \in F_i\} \\ &\cup \{s \mid \mathrm{Val}_i^{n-1}(s) = 0 \wedge s \in (F_i \cup H_i^{n-1})\} \end{aligned}$$

Note that $F_i \subseteq K_i^n$ for every n.

$Out(\mathcal{S}^0)$ is accepted by the automaton with the same structure as $\mathcal{G}$ and where the Büchi set is V. Since $Out(\mathcal{S}^{n-1}) \cap \bigcap_{i\in P} \mathcal{L}(\mathcal{A}_i^n) = Out(\mathcal{S}^n)$ by Lemma 7, $Out(\mathcal{S}^n)$ is recognized by an automaton whose acceptance condition is a conjunction of $n \times |P|$ Büchi conditions. By taking a product of the game with an automaton of size $n \times |P|$, a condition $Out(\mathcal{S}^n)$ can be expressed as a single Büchi set that we write D^n.

Recall that condition Θ_i^n is

$$C_i^{\leq n} \cup \left(Out(\mathcal{G} \setminus T^{n-1}) \cap Out(\mathcal{S}_i^n) \cap (Out(\mathcal{S}^n) \Rightarrow \mathrm{WIN}_i)\right).$$

We isolate the prefix-independent part of Θ_i^n by defining

$$\Gamma_i^n = Out(\mathcal{S}_i^n) \cap (Out(\mathcal{S}^n) \Rightarrow \mathrm{WIN}_i)$$

which, by unfolding the recurrence relation of Lemma 9, can be rewritten:

$$\Gamma_i^n = \bigcap_{m=0}^{n-1} \left(Out(\mathcal{S}^m) \Rightarrow \left(\mathcal{L}(\mathcal{A}_i^{m+1}) \cap Out(\mathcal{S}^m)\right)\right) \cap (Out(\mathcal{S}^n) \Rightarrow \mathrm{WIN}_i)$$

Also remark that the "prefix-dependent" part of Θ_i^n can be expressed by either removing edges (to avoid sets T_i^n) or adding an additional winning state (for sets C_i).

Each set $Out(\mathcal{S}^m)$ is expressed by the Büchi condition D^m. Similarly, condition $\left(\mathcal{L}(\mathcal{A}_i^{m+1}) \cap Out(\mathcal{S}^m)\right)$ is a conjunction of $m \times |P| + 1$ Büchi conditions. Therefore, in order to consider all these different conditions on the same game, it can be assumed that $\mathcal{G}$ is synchronized with a product of size[3] $O(n^2 \cdot |P|)$. Then each condition $\left(\mathcal{L}(\mathcal{A}_i^{m+1}) \cap Out(\mathcal{S}^m)\right)$ can be expressed as a single Büchi set that we write E_i^{m+1}.

Notice that we have a "chain" of inclusion of the various objectives. For each index m and player i:

$$Out(\mathcal{S}^m) \cap \mathcal{L}(\mathcal{A}_i^{m+1}) \subseteq Out(\mathcal{S}^m) \subseteq Out(\mathcal{S}^{m-1}) \cap \mathcal{L}(\mathcal{A}_i^m). \tag{1}$$

Hence, if the left operand of an implication $Out(\mathcal{S}^m) \Rightarrow \left(Out(\mathcal{S}^m) \cap \mathcal{L}(\mathcal{A}_i^{m+1})\right)$ is satisfied, then all such implications are satisfied for indexes $k < m$. And if the right operand is satisfied, then it is the case for indexes $k \leq m$. The condition Γ_i^n can then be defined by the parity condition given by the following coloring function χ_i^n:

[3] Precisely $|P| \cdot (n-1) \cdot (n-2) + n$.

- if $s \in F_i$ then $\chi_i^n(s) = 2n + 2$
- otherwise $\chi_i^n(s)$ is the maximum value between 0, $2 \cdot \max\{m \mid s \in E_i^m\}$, and $2 \cdot \max\{m \mid s \in D^m\} + 1$, with the convention that $\max \emptyset = -\infty$.

Lemma 10. *For all index n and player i, $\rho \in \Gamma_i^n$ if, and only if, ρ satisfies the parity condition χ_i^n.*

5.0.2 Solving the winning coalition problem for Büchi objectives

We showed that all objectives encountered in the computation of values boil down to parity objectives. This yields the following proof:

Proof of Theorem 3. By reformulating Θ_i^n as a parity condition, checking whether the value of a state is 1 amounts to solving a two player game with parity condition. This is known to be in $\mathsf{UP} \cap \mathsf{coUP}$, although suspected to be in P. Similarly, condition Ψ_i^n is a conjunction of Büchi objectives, hence solving its emptiness — which means deciding whether the value of a state is -1 — is in P.

Let n_0 be the index where the sets of admissible strategies stabilize (we have $n_0 \leq |P| \cdot |V|$). Let W, L be the set of players that should win and lose, respectively. Solving the winning coalition problem amounts to solving emptiness of

$$\Phi = Out(\mathcal{S}^{n_0}) \cap \bigcap_{i \in W} \textsc{Win}_i \cap \bigcap_{i \in L} \neg \textsc{Win}_i.$$

As the conjunction of $n_0 + |W|$ Büchi conditions and a coBüchi condition, it can be expressed as a parity condition with 3 colors. □

6. Weak Muller conditions

Assuming that the winning conditions are weak Muller conditions, we can show the following theorem, which generalizes Theorem 1:

Theorem 4. *The winning coalition problem for weak Muller conditions is* PSPACE*-complete. However, if the number of players is fixed, the problem becomes* P*-complete.*

Proof sketch. The core of the argument consists in proving that the value of a history only depends on the set $\mathrm{Occ}(h)$ of states that have been visited and on the last state of the history. Then, as in the case of safety conditions, we unfold the game, yielding prefix-independent winning conditions. Therefore we can reuse the techniques of previous section to compute the values. A recursive algorithm taking advantage of the structure of the unfolding limits the complexity and allows a computation in PSPACE. □

7. Model-checking under admissibility

Given a LTL formula ϕ, the model-checking under admissibility problem is equivalent to the emptiness of $\Phi = \mathcal{L}(\neg\phi) \cap Out(\mathcal{S}^*)$. It was proven in [22] that the language $\mathcal{L}(\neg\phi)$ can be represented by a Büchi automaton $\mathcal{A}_{\neg\phi}$ which can be accessed (*i.e.* the existence of a transition, whether a state is initial, accepting…) in polynomial space.

Nondeterministically guessing a path (the length of which can be bounded) in the product of $\mathcal{A}_{\neg\phi}$ and $(\mathcal{G}, \psi)$, the automaton with Muller condition for $Out(\mathcal{S}^*(\mathcal{G}_0))$, yields a polynomial space algorithm to solve the model-checking under admissibility problem. Conversely LTL model-checking is PSPACE-hard [21], hence providing the lower bound:

Theorem 5. *The model-checking under admissibility problem is* PSPACE*-complete.*

8. Conclusion

This paper provides new fundamental complexity and algorithmic results on the iterated elimination of dominated strategies in the context of turn-based n-player perfect information ω-regular games. We have shown that the non-elementary complexity of the plain tree-automata-based procedure suggested in [2] can be avoided, and our precise complexity results show that iterated elimination of dominated strategies is a *computationally viable alternative* to Nash equilibria.

Future directions include investigating extensions with imperfect information, with quantitative objectives, or richer interaction models like concurrent games.

References

[1] R. Aumann. Agreeing to disagree. *Annals of Statistics*, 4(6):1236–1239, 1976.

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

[3] P. Bouyer, R. Brenguier, N. Markey, and M. Ummels. Concurrent games with ordered objectives. In *Proc. of FoSSaCS'12*, volume 7213 of *LNCS*, pages 301–315. Springer, Mar. 2012. URL `http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/BBMU-fossacs12.pdf`.

[4] G. Brassard. A note on the complexity of cryptography. *IEEE Transactions on Information Theory*, 25(2):232–233, Mar. 1979.

[5] R. Brenguier, J.-F. Raskin, and M. Sassolas. The complexity of admissibility in omega-regular games. *CoRR*, abs/1304.1682, Apr. 2013. URL `http://arxiv.org/abs/1304.1682`.

[6] K. Chatterjee, T. A. Henzinger, and N. Piterman. Strategy logic. *Information and Computation*, 208(6):677–693, 2010.

[7] A. Dawar, F. Horn, and P. Hunter. Complexity Bounds for Muller Games. *Theoretical Computer Science*, 2013. Submitted.

[8] L. de Alfaro, T. A. Henzinger, and O. Kupferman. Concurrent reachability games. *Theoretical Computer Science*, 386(3):188–217, 2007.

[9] E. Emerson and C. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. of FCS'91*, pages 368–377. IEEE, 1991.

[10] E. Filiot, T. Le Gall, and J.-F. Raskin. Iterated regret minimization in game graphs. In *Proc. of MFCS'10*, pages 342–354, 2010.

[11] P. Hunter. *Complexity and Infinite Games on Finite Graphs*. PhD thesis, Computer Laboratory, University of Cambridge, 2007. URL `http://web.comlab.ox.ac.uk/oucl/work/paul.hunter/papers/thesis.pdf`

[12] M. Jurdziński. Deciding the winner in parity games is in UP and in co-UP. *Information Processing Letters*, 68(3):119–124, 1998.

[13] M. Jurdziński, M. Paterson, and U. Zwick. A deterministic subexponential algorithm for solving parity games. In *Proc. of SODA'06*, pages 117–123. ACM Press, Jan. 2006.

[14] M. Klimoš, K. G. Larsen, F. Stefanak, and J. Thaarup. Nash equilibria in concurrent priced games. In *Proc. of LATA'12*, volume 7183 of *LNCS*. Springer, 2012.

[15] F. Mogavero, A. Murano, and M. Y. Vardi. Reasoning about strategies. In *Proc. of FSTTCS'10*, volume 8 of *LIPIcs*, pages 133–144. Schloss Dagstuhl - LZI, Dec. 2010.

[16] J. Nash. Equilibrium points in n-person games. *Proc. of the National Academy of Sciences*, 36(1):48–49, 1950.

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

[18] A. Pnueli. The temporal logic of programs. In *Proc. of the 18th Annual Symposium on Foundations of Computer Science (FoCS'77)*, pages 46–57. IEEE Computer Society, Oct. 1977.

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

[20] J. H. Reif. The complexity of two-player games of incomplete information. *Journal of Computer and System Sciences*, 29(2):274–301, 1984.

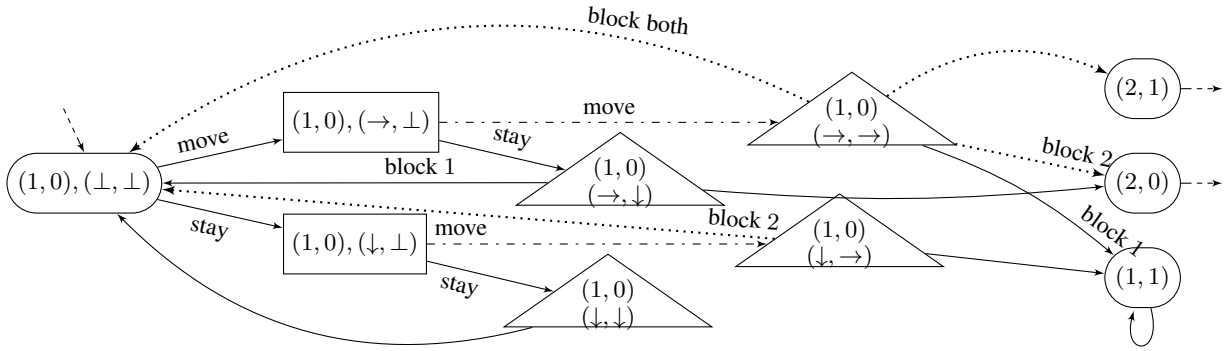


Figure 5: Part of the game played by the trains and the environment when trying to move from the original position. Round states belong to train t_1, square ones to train t_2, and triangular ones to the environment. Dotted edges are eliminated in the first iteration. Dashed ones in the second iteration.

[21] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, July 1985. ISSN 0004-5411. .

[22] A. P. Sistla, M. Y. Vardi, and P. Wolper. The complementation problem for Büchi automata with appplications to temporal logic. *Theorethical Computer Science*, 49:217–237, 1987.

[23] M. Ummels. The complexity of Nash equilibria in infinite multiplayer games. In *Proc. of FoSSaCS'12*, volume 4962 of *LNCS*, pages 20–34. Springer, 2008. URL `http://www.logic.rwth-aachen.de/ ummels/fossacs08.pdf`.

[24] M. Ummels. *Stochastic Multiplayer Games: Theory and Algorithms*. PhD thesis, RWTH Aachen University, 2010. URL `http://www.logic.rwth-aachen.de/ ummels/diss.pdf`.

[25] U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1&2):343–359, 1996.

A. Example: a metro system

A.1 The model of the system

We illustrate the use of model-checking under admissibility over the following metro system. The metro track is composed of a directed ring divided in n slots (one can assume for example that even-numbered slots represent a station, while odd-numbered slots represent a section of track between two stations). There are p trains numbered 1 to p on the track, initially at positions $p-1,\dots,0$ (hence train number 1 starts in front of the others (see Figure 6). In the sequel we study the case of a track of length 6 with two trains t_1 and t_2.

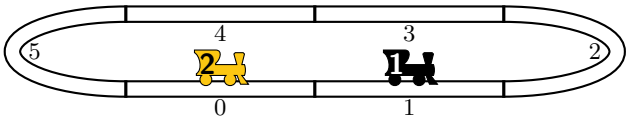


Figure 6: Metro track with $n=6$ and $p=2$, at their initial position $(1,0)$.

At each step, all the trains successively declare whether they want to advance or not. Once everyone has chosen, all trains try to move synchronously. However, there is an evil environment *env* that can prevent trains from moving (for example by activating an alarm on the section the train is trying to move in). That means some trains that wanted to move may in fact remain in their position. Nevertheless, all other trains must comply with their original choice, modeling the fact that trains cannot communicatein real-time. Additionally, if at any point two trains are on the same track section, they collide and the game stops. (Part of) the game graph for this protocol is depicted in Figure 5. In this game, the configuration of the track is given by the pair of positions for t_1 and t_2, and their respective wish to move or stay in place by the symbols "$\rightarrow$" (advance) or "$\downarrow$" (stay in place).

The objective of both trains is to loop infinitely often, which can be expressed by a generalized Büchi winning condition[4] requiring visiting infinitely often both sections 0 and 1. The objective of the environment is to give rise to a collision: since collision are sink states it is equivalent to a Büchi winning condition over the set of collision states. Globally, we are interested in knowing whether the LTL formula $\psi_{\neg\text{coll}}$ that expresses that no collision ever happens is satisfied under admissibility. Remark that here:

- no player has a winning strategy alone;
- the formula $\psi_{\neg\text{coll}}$ is not verified on all paths of the model;
- that the players are not *a priori* trying to satisfy $\psi_{\neg\text{coll}}$: indeed, the environment's objective is to negate $\psi_{\neg\text{coll}}$.

A.2 The set of iteratively admissible strategies

After iteratively eliminating dominated strategies (the computation is detailed in [5]) we obtain the set of strategies with the following properties.

- If given the opportunity (e.g. two trains following each other both willing to advance), the environment will create a collision.
- In this set, one can see that whenever t_2 is right behind t_1:
  - t_2 never tries to move;
  - t_1 infinitely tries to move;

  This prevents giving a collision opportunity to the environment.

As a result, one can see there can be no collision, hence the objective $\psi_{\neg\text{coll}}$ is satisfied. Hence, the model-checking under admissibility problem answers positively for $\psi_{\neg\text{coll}}$, while the ("classical") model-checking problem for the same formula answers negatively.

This also shows for example that the winning coalition problem with $env \in W$ answers negatively: there is no admissible profile that makes player *env* win.

On the other hand, it cannot be assured that t_1 and t_2 will win for their objective: the environment can choose to always block their movement, preventing them from looping infinitely often on the track.

[4] This condition can therefore be expressed by a Büchi condition on a graph with memory of visits to 0 or 1. It can also be expressed by a circuit condition encoding formula $\bigvee(0,\_) \wedge \bigvee(1,\_)$.