Open access • Book Chapter • DOI:10.1007/978-3-540-69507-3_26

On the (High) Undecidability of Distributed Synthesis Problems

David Janin

Institutions: University of Bordeaux

Published on: 20 Jan 2007 - Conference on Current Trends in Theory and Practice of Informatics

Topics: Undecidable problem, Decidability, Decision problem and Arithmetical hierarchy

On the (high) undecidability of distributed synthesis problems

David Janin

On the (high) undecidability of distributed synthesis problems

David Janin

LaBRI, Université de Bordeaux I,
351, cours de la libération,
F-33 405, Talence cedex, France
janin@labri.fr

Abstract. The distributed synthesis problem [11] is known to be undecidable. Our purpose here is to study further this undecidability.
For this, we consider distributed games [8], an infinite variant of Peterson and Reif multiplayer games with partial information [10], in which Pnueli and Rosner's distributed synthesis problem can be encoded and, when decidable [11, 6, 7], uniformly solved [8].
We first prove that even the simple problem of solving 2-process distributed game with reachability conditions is undecidable (Σ_1^0-complete). This decision problem, equivalent to two process distributed synthesis with fairly restricted FO-specification was left open [8]. We prove then that the safety case is Π_1^0-complete. More generally, we establish a correspondence between 2-process distributed game with Mostowski's weak parity conditions [9] and levels of the arithmetical hierarchy. finally, distributed games with general ω-regular infinitary conditions are shown to be highly undecidable (Σ_1^1-complete).

1 Introduction

In this paper, we study the undecidability of the distributed synthesis problem as introduced by Pnueli and Rosner [11]. This problem can be stated as follows:

> Given a *distributed architecture* (finitely many sites interconnected through a given network with some specified global input channels and global output channels) and a *global specification* of expected correct architecture's global behaviors (defining a set of mappings that map global input sequences to global output sequences say in First Order (FO) or even Monadic Second order (MSO) Logic), is there a *distributed program* (a set of mappings, one per site, that maps sequences of local inputs to sequences of local outputs) which *resulting global behavior satisfies the global specification* ?

With a specification language as simple FO, on the architecture defined by two independent sites with independent (global) input channels and (global) output channels (see Figure 1), this problem is known to be undecidable [11].

Analyzing Pnueli and Rosner's proof, one can observe that with reachability conditions (FO global specifications essentially stating that some properties eventually occur) the distributed synthesis problem is Σ_1^0-complete in the arithmetical hierarchy, i.e. inter-reducible to the halting problem of Turing Machine (TM).

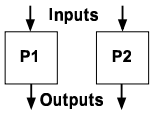

Fig. 1. An undecidable architecture

With safety conditions (FO global specification essentially stating that some bad properties never occur, allowing thus infinitary behaviors) one may conjecture (although the two kinds of problems are not dual) that the distributed synthesis problem is Π_1^0-complete in the arithmetical hierarchy.

More generally, one may ask what is the relationship between the expressiveness of the global specification language (say within FO logic or even within MSO logic with more advanced infinitary conditions) and the complexity of the resulting distributed synthesis problems. In this paper, we give an answer to this question.

Main results

We first prove the following refinement of Pnueli and Rosner's result:

Theorem 1. *The 2-process reachability distributed synthesis problem is Σ_1^0-complete even with fairly restricted global specification : universally quantified k-local FO-properties.*

Next, we prove that:

Theorem 2. *The 2-process safety distributed synthesis problem with FO specification is Π_1^0-complete.*

Since the set of finite state (or even computable) distributed programs is Σ_1^0-definable, this result also implies that :

Corollary 1. *There exist safety distributed synthesis problems with distributed solutions but no computable ones.*

We study then relationships between more general infinitary conditions for distributed synthesis problems and levels of the arithmetical (or the analytical) hierarchy. More precisely, we show that:

Theorem 3. *For every integer $n \geq 1$, solving distributed synthesis problem with two (or more) processes and weak parity conditions (in the sense of Mostowski [9]) of range $[0, n+1]$ (resp. $[1, n]$) is Π_n^0-complete (resp. Σ_n^0-complete).*

With more complex infinitary conditions:

Theorem 4. *The distributed synthesis problem with Büchi infinitary condition (or more general parity infinitary condition) is highly undecidable (Σ_1^1-complete).*

Related works

These results are achieved by proving analogous statements for 2-process distributed games [8]. These games are a variant (with infinite behaviors) of Peterson and Reif's multiplayer games with partial information [10] into which distributed synthesis problems can be encoded and, in most decidable cases [11, 6, 7], uniformly solved [8].

A 2-process distributed game is equivalent to a 2-site distributed synthesis problem with fairly restricted specification. In fact, the global specification in a distributed game is implicitly encoded into (1) the description of players possible moves and (2) an infinitary condition specifying the allowed infinite behaviors.

In logical terms, moves are specified by universally quantified k-local formulas. With reachability condition, i.e. when no infinite behavior is allowed, a 2-process distributed game is thus a fairly restricted 2-site distributed synthesis problem. General 2-site distributed synthesis problems with arbitrary LTL specification are not known to be reducible to 2-process distributed games [8]. The best known reduction is to add a third process that plays the role of (an automaton translation of) the external (LTL or even MSO) specification [2].

This makes our undecidability result in the reachability case stronger than Pnueli and Rosner's result [11].

In Mohalik and Walukiewicz's work [8], the decidability of 2-process distributed games was also left open. Our results show they are not decidable even with the simplest infinitary conditions.

2 Background

A word on an alphabet A is a function $w : \omega \to A$ with prefix-closed domain. When $dom(w)$ is finite, we say w is a finite word, otherwise, w is an infinite word. The set of finite words (resp. finite non empty words) on alphabet A is written A^* (resp. A^+), the set of infinite words is written A^ω, the set of finite or infinite words is written A^∞. The empty word is written ϵ. The catenation of every word $u \in A^*$ and every word $v \in A^\infty$ is written $u.v$.

Given two sets A and B, we write π_A or π_1 (resp. π_B or π_2) the projection from $A \times B$ onto A (resp. from $A \times B$ onto B). These notations are extended to any subset of $A \times B$ and words on $A \times B$. Given $P \subseteq A \times B$ (resp. $w \in (A \times B)^\infty$), we also write $P[1] = \pi_A(P)$ and $P[2] = \pi_B(P)$ (resp. $w[1] = \pi_A(w)$ and $w[2] = \pi_B(w)$). Given $w \in (A + B)^\infty$, we also write $\pi_A(w)$ (resp. $\pi_B(w)$) the word obtained from w by deleting all letters not in A (resp. not in B).

In the sequel, we will use languages of infinite words as infinitary acceptance conditions. We first review here the definition we will use.

Definition 1 (Parity and weak parity condition [9]). *Let $L \subseteq A^\omega$ be a language of infinite words. Language L is called a* parity condition *when there are some integers m and $n \in \mathbb{N}$ with $m \leq n$ and some priority mapping $\Omega : A \to [m, n]$ such that $L = \{w \in A^\omega : \liminf \Omega(w) \equiv 0(2)\}$, i.e. L is the set of infinite*

sequences where the least priority that occurs infinitely often is even. Language L *is a* weak parity condition *when there is a priority mapping* $\Omega : A \to [m, n]$ *as above such that* L *is moreover restricted to sequence* $w \in A^\omega$ *such that* $\Omega(w)$ *is an increasing sequence of priorities.*

In both cases, interval $[m, n]$ *is called the* range *of the parity condition.*

A safety condition *is a parity (or weak parity) condition with range* $[0]$ *(hence with* $L = A^\omega$*) and a* reachability condition *is a parity (or weak parity) condition with range* $[1]$ *(hence with* $L = \emptyset$*).*

Distributed games [8] are special kind of multiplayer games with partial information [10] extended to infinite plays with a cooperating Process team playing against a unique Environment player.

Definition 2 (Distributed Game Arenas). *A one-Process (two Players) game arena is a quadruple* $\mathcal{G} = \langle P, E, T, e\rangle$ *with set of Process positions* P*, set of Environment positions* E*, set of possible transition moves* $T \subseteq P \times E \cup E \times P$ *and initial position* $e \in E$*.*

Given n *one-Process game arenas* $\mathcal{G}_i = \langle P_i, E_i, T_i, e_i\rangle$ *for* $i \in [1, n]$*, a* synchronous distributed game arena $\mathcal{G}$ *built from the local game arenas* $\mathcal{G}_1, \ldots, \mathcal{G}_n$*, is a game arena* $\mathcal{G} = \langle P, E, T, e\rangle$ *with* $P = \prod_i P_i$*,* $E = \prod_i E_i$*,* $e = (e_1, \cdots, e_n)$ *and such that the set of moves* T *satisfies the following conditions: for every* $u \in P$ *and* $v \in E$

- *P-moves :* $(u, v) \in T$ *if and only if for every* $i \in [1, n]$*,* $(u[i], v[i]) \in T_i$*,*
- *E-moves : if* $(v, u) \in T$ *then for every* $i \in [1, n]$*,* $(u[i], v[i]) \in T_i$*.*

Observe that there is a unique distributed game arena built from the local arenas $\mathcal{G}_1, \ldots, \mathcal{G}_n$ with maximal set of Environment moves. This arena, written $\mathcal{G}_1 \otimes \cdots \otimes \mathcal{G}_n$, is called the free synchronous product of the games $\mathcal{G}_1, \ldots, \mathcal{G}_n$. Observe that any other distributed arena $\mathcal{G}$ built from the same local games can just be seen as a subgame of the free product obtained by possibly disallowing some Environment moves. We simply write $\mathcal{G} \subseteq \mathcal{G}_1 \otimes \cdots \otimes \mathcal{G}_n$ to denote that.

In [8], a more general notion of distributed with *asynchronous* moves is defined. The additional expressiveness gained with asynchronism is not used in this paper. Since we are essentially establishing lower bounds result, this fact makes statements even stronger.

Definition 3 (Plays and strategies). *Given a two player game arena* $\mathcal{G} = \langle P, E, T, e\rangle$*, a* strategy *for the Process player (resp. a* strategy *for the Environment player) is a mapping* $\sigma : P^+ \to E$ *(resp. a mapping* $\tau : E^+ \to P$*).*

The play induced by strategies σ *and* τ*, written* $\sigma * \tau$*, is defined to be the maximal word* $w \in (P + E)^\infty$ *such that* $w(0) = e$ *and, for every* $i \in dom(w)$ *with* $i > 0$*,* $(w(i-1), w(i)) \in T$ *and, given* $w' = w(0) \cdots w(i-1)$*, if* $w(i-1) \in P$ *then* $w(i) = \sigma \circ \pi_P(w')$ *and if* $w(i-1) \in E$ *then* $w(i) = \tau \circ \pi_E(w')$*.*

A process strategy σ *is* non blocking *when, for every counter strategy* τ*,* $\sigma * \tau \in (P + E)^*.E \cup (P + E)^\omega$*.*

Given an n-process game arena $\mathcal{G} \subseteq \mathcal{G}_1 \otimes \cdots \otimes \mathcal{G}_n$, a Process strategy $\sigma : P^+ \to E$ is a distributed strategy *where there is a set of local process strategies $\{\sigma_i : P_i^+ \to E\}_{i \in [1,n]}$ such that, for every word $w \in P^+$, $\sigma(w) = (\sigma_1 \circ \pi_{P_1}(w), \cdots, \sigma_n \circ \pi_{P_n}(w)$. In other words, a Process strategy is distributed when every local Process player only plays following its own local view of a global play.*

Definition 4 (Games and winning strategies). *A (weak or parity) game is an tuple $\mathcal{G} = \langle P, E, T, e, Acc \rangle$ where $\langle P, E, T, e \rangle$ is a game arena and $Acc \subseteq (P + E)^\omega$ is an additional (weak or parity) infinitary condition*

A game $\mathcal{G}$ (resp. a distributed game) is winning for player P (resp. for the Process team) when there is a Process strategy (resp. a distributed Process strategy) $\sigma : P^+ \to E$ such that,for every counter strategy $\tau : E^+ \to P$, $\sigma * \tau \in (P + E)^*.E \cup Acc$, i.e. every maximal play allowed by σ is either finite and ends in a position of player E, or is infinite and belongs to Acc.

3 Tilings and quasi-tilings

In order to prove lower bounds results in next section, we review in this section the notions of finite and infinite tilings [1, 5].

Definition 5 (Tilings). *Let $\{n, s, w, e\}$ be the four cardinal directions of the plan. Given a finite set of colors C with a distinguished color $\#$ called the border color, a* tile *is a mapping $t : \{n, s, w, e\} \to C$ that assigns to each cardinal direction a color of C with the additional requirement that $t(s) \neq \#$ and $t(w) \neq \#$, i.e. color $\#$ will only be used to define $East$ or $North$ borders.*

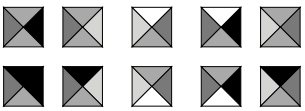

Fig. 2. A finite set of tiles

Given a finite set S of tiles (see Figure 2), a tiling *is a partial function $m : \omega \times \omega \to S$ such that $dom(m) = [0, M-1] \times [0, N-1]$ for some $(M, N) \in \omega \times \omega$ when m is a finite tiling or $dom(m) = \omega \times \omega$ when m is an infinite tiling, such that: for all $(i, j) \in dom(m_\sigma)$,* N/S-compatibility: *if $(i, j + 1) \in dom(m)$ then $m(i, j)(n) = m(i, j + 1)(s)$,* E/W-compatibility: *if $(i + 1, j) \in dom(m)$ then $m(i, j)(w) = m(i+1, j)(e)$,* E-border condition: *$(i+1, j) \notin dom(m)$ if and only if $m(i, j)(e) = \#$, and* N-border condition: *$(i, j + 1) \notin dom(m)$ if and only if $m(i, j)(n) = \#$ (see Figure 3 with color black standing for the border color $\#$).*

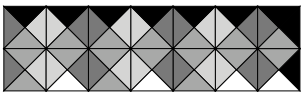

Fig. 3. A tiling

Theorem 5 (Berger [1], Harel [5]). *Given a set of colors C and a set S of tiles with a distinguished tile $t_0 \in S$: (1) the problem of finding M and N and a finite $M \times N$-tiling m such that $m(0,0) = t_0$ is Σ_1^0-complete, (2) the problem of finding an infinite tiling m such that $m(0,0) = t_0$ is Π_1^0-complete, and (3) the problem of finding an infinite tiling m such that $m(0,0) = t_0$ and one given color , say* red*, occurs infinitely often is Σ_1^1-complete.*

4 Towards the proofs : quasi-tilings

The notion of quasi-tiling defined below and encoded into one process game is essential for our encoding (in the remaining sections) of tilings into 2-process distributed games.

Definition 6 (Quasi-tilings). *A function $m : \omega \times \omega \to S$ is a* quasi-tiling *(see Figure 4) when it satisfies N/S-compatibility and N-border condition on every column, E-border condition on every line, and E/W-compatibility on the first line.*

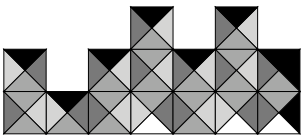

Fig. 4. A quasi-tiling

It occurs that, for every finite set of tiles S and initial tile $t_0 \in S$, there exists a one process (two player) game $\mathcal{G}_{S,t_0}$ that encodes all quasi-tiling $m : \omega \times \omega \to S$ as non blocking strategies for player E.

Definition 7 (Quasi-tiling games). *Given a finite set of color C, a finite set of C-colored tiles S and an initial tile t_0, let $\mathcal{G}_{S,t_0} = \langle P, E, T, i \rangle$ be the two player game arena defined by:*

- $P = (\{e,n\} \times S \times \{Proc\}) \cup \{\bot\}$, $E = (\{e,n\} \times S \times \{Env\}) \cup \{*\}$ *and* $i = *$,
- T *is the set of all pairs of the form* $((d,t,Proc),(d,t',Env)) \in P \times Env$ *such that, if* $d = e$ *then* $t'(w) = t(e)$ *and if* $d = n$ *then* $t'(s) = t(n)$ *and* $t'(e) = \#$ *if and only if* $t(e) = \#$ *(Process moves) plus the set of all pairs of the form* $(*,(x,t_0,Proc))$ *or* $((x,t,Env),\bot)$ *plus all pairs of the form* $((d,t,Env),(d',t,Proc)) \in E \times P$ *such that, if* $d = e$ *then* $d' \in \{e,n\}$ *and if* $d = n$ *or* $t(e) = \#$ *and then* $d' = n$ *(Environment moves).*

The intuition behind this definition is that player E chooses along a word of the form $e^i.n^\omega$ and, for every prefix $e^i.n^j$ of this word, player P answers by choosing a tile for position (i,j). Since player E chooses where to turn the full set $\omega \times \omega$ is potentially covered. It turns out, as precisely stated in next Lemma, that player P non blocking strategies just define all quasi-tilings.

Lemma 1 (Quasi-tilings and strategies). *For every non blocking strategy $\sigma : P^+ \to E$, in game $\mathcal{G}_{S,t_0}$, there is a unique quasi-tiling $m_\sigma : \omega \times \omega \to S$ such that for all $(i,j) \in \omega \times \omega$, $(i,j) \in dom(m_\sigma)$ if and only if there is counter strategy $\tau : E^+ \to P$ such that $\pi_1 \circ \pi_P(\sigma * \tau) = e^i.n^j$ and $\pi_2(\pi_P(\sigma * \tau)(i+j)) = m_\sigma(i,j)$ (with in particular, $m_\sigma(0,0) = t_0$).*

Conversely, for every quasi-tiling m such that $m(0,0) = t_0$ there is a non blocking strategy σ_m in game $\mathcal{G}_{m,t_0}$ such that $m_{\sigma_m} = m$.

Proof. By construction, in every play, player E's task is to chose, at every step, a direction e or n and, when direction n has been chosen, or when the East border is reached, to choose repeatedly direction n. It follows that every (blocking) strategy for player E that avoids position $\bot$ can be described by (1) choosing some $(i,j) \in \omega \times \omega$ and (2) playing the successive directions described by the word $e^i.n^j$ - provided player P does not create the East border.

Against player E, player P's strategy just amounts to choose, for every $(i',j') \leq_x (i,j)$ a tile $t_{i',j'}$. It should be clear that this choice is independent from (i,j) (chosen but by player E but unknown to player P) so we can define $m_\sigma(i',j') = t_{i',j'}$.

The fact that m_σ is a quasi-tiling immediately follows from game G_{S,t_0} definition. The converse property is also immediate. □

Observe that, in game $\mathcal{G}_{S,t_0}$, player P chooses to define a tiling bounded in the East direction by choosing the first tile t such that $t(e) = \#$.

5 Undecidability results: ground cases

Theorem 6 (Safety case). *The problem of finding a winning distributed strategy in a 2-process distributed game with safety condition is Π_1^0-complete.*

Proof. Clearly, solving a safety distributed game is Π_1^0. It remains to prove that it is also Π_1^0-hard. In order to do so, we encode the infinite tiling problem into a safety distributed game.

Let S be a finite set of tiles and let $t_0 \in S$ be a given initial tile. The idea is to build a distributed game $\mathcal{G}$ from the free product $\mathcal{G}_{S,t_0} \otimes \mathcal{G}_{S,t_0}$ with safety condition in such a way that player E checks that (1) if a distributed strategy $\sigma_1 \otimes \sigma_2$ is non blocking then $\sigma_1 = \sigma_2$, and (2) a distributed strategy of the form $\sigma \otimes \sigma$ is winning if and only if the quasi-tiling $m_\sigma : \omega \times \omega \to S$ is also a tiling of $\omega \times \omega$, i.e. it satisfies the E/W-compatibility condition on all lines.

This is done as follows. We first assume, without lost of generality, that every position in game game $\mathcal{G}_{S,t_0}$ is (1) extended with a new bit value that is positioned by Environment player's first move as explained below and (2) also extended in such a way the last two tiles t (current) and t' (previous) chosen by player Process are readable in Environment positions.

Environment moves in the product $\mathcal{G}_{S,t_0} \otimes \mathcal{G}_{S,t_0}$ are then defined as follows. From the initial position $(*,*)$ player E moves to a position with an arbitrary pair of bit values (one on every side), and, according to these values (that remained unchanged later on):

1. with bit values $(0,0)$, $(0,1)$ or $(1,0)$: player E plays the same directions in both local games and checks process strategy equality,
2. with bit values $(1,1)$: player E delays by one step the North turn in the second local game and, after this turn, repeatedly check that $t'_1(e) = t_2(w)$ where t'_1 is the "previous" choice of tiles made by player P_1 and t_2 is the "current" choice of tiles made by player P_2.

Player E moves to $(\bot, \bot)$ if any of these checks fails or if any of the Process players chooses a tile that contains the border color $\#$.

The winning condition for the Process team is to avoid position $(\bot, \bot)$. This is a safety condition.

Let then $\sigma_1 \otimes \sigma_2$ be a distributed winning strategy on such a game. By checking equality with bit values $(0,0)$, $(0,1)$ or $(1,0)$ Environment makes sure that Process player does play the same strategy $\sigma = \sigma_1 = \sigma_2$ regardless of the initial bit value that he has chosen. Given then the induced quasi-tiling m_σ (see Lemma 1), one can check that when bit values are $(1,1)$ Environment does indeed check E/W-compatibility. It follows that m_σ is a tiling, infinite by the safety condition.

Conversely, for any infinite tiling m such that $m(0,0) = t_0$ one can check that $\sigma_m \otimes \sigma_m$ is a winning distributed strategy.

We conclude applying Theorem 5. □

Theorem 7 (Reachability case). *The problem of finding a winning distributed strategy in a 2-process distributed game with reachability condition is Σ^0_1-complete.*

Proof. Again, clearly, this problem is Σ^0_1. It remains to prove that it is Σ^0_1-hard. In order to do so, we encode into reachability distributed games the finite tiling problem.

The encoding is similar to the encoding in the proof of Theorem 6 except that (1) player E now allows players P_1 and P_2 to play tiles that contains the border color $\#$ and (2) the winning condition for Process team is to reach, at the end of every local play, a tile t with $t(n) = \#$.

Observing that player P will force the East-border by playing a tile t with $t(e) = \#$ makes it clear that there is a winning distributed strategy in the new (reachability) distributed game $\mathcal{G}$ if and only if there is a finite tiling m such that $m(0,0) = t_0$ □

6 Within and above the arithmetical hierarchy

The relationship with the arithmetical hierarchy is achieved through the observation that, by Post's Theorem, every level of the arithmetical hierarchy has a computational interpretation by means of Alternating Turing Machines [3] extended with infinite runs and weak parity acceptance conditions.

Theorem 8. *For every integer $n > 0$, a language $L \subseteq \Sigma^*$ is Π^0_n-definable (resp. Σ^0_n-definable) if and only if it is definable by an Alternating Turing Machine with infinitary weak parity condition with range $[0, n-1]$ (resp. $[1, n]$).*

Proof. By ATM we mean here ATM with universal and existential states only (no negation states). ATM are extended with infinite runs (with infinitary conditions) in the following sense: a computation tree of an ATM is accepting if every finite branch ends in an accepting control state, and, for every infinite branches, the corresponding infinite sequence of control states satisfies the infinitary condition.

Applying [3], we know that standard ATM (with reachability conditions) capture the level Σ_1^0 of the arithmetical hierarchy. By duality, ATM with safety conditions (hence infinite runs) capture the level Π_1^0.

For higher levels, the proof is based on the observation that alternation allows a machine to (1) guess the answer of an oracle and, at the same time, to (2) start a computation of the oracle (or its complement) that checks the guessing is correct. By construction, since no acknowledgment is expected, the resulting infinitary conditions are weak in the sense of Mostowski [9]. Post's Theorem ensures such a construction captures, level by level, the arithmetical hierarchy. □

Theorem 9 (The weak case). *For every integer $n > 0$, the problem of solving 2-process distributed weak game with Mostowski range $[0, n-1]$ (resp. $[1, n]$) is Π_n^0-complete (resp. Σ_n^0-complete).*

Proof. (sketch) Upper bounds should be clear. It remains to prove the lower bound. The main idea is to encode into (winning) distributed strategy the (accepting) runs of ATM. This can be achieved as follows.

At first sight, the tiling encoding defined in the previous section fails to apply here since a tiling only encode the run of a *non alternating* TM (say with TM configurations encoded by means of east colors of tiles in a line).

However, in this encoding, somehow as in a solitaire domino game, the process team defines (playing identically in copies of local game $\mathcal{G}_{S,t_0}$) one tiling (equivalently one accepting TM run) while player E's role is bound to check that all required space is covered and all tiling rules are satisfied (equivalently it checks that the process team defines indeed a TM run). The idea to encode the run of an ATM is thus to let player E chooses some of the tiles, say one over two in a line, in a modified local game $\tilde{\mathcal{G}}'_{S,t_0}$. In this shift from a solitaire to a two player domino like game, all branches of an ATM run are encoded by the many tilings that are produced following player E's moves.

An analogous synchronization (restriction of player E's global moves) in a distributed game $\mathcal{G} \subseteq \tilde{\mathcal{G}}_{S,t_0} \otimes \tilde{\mathcal{G}}_{S,t_0}$ can force both Environment and Process players to play only real tilings (and not quasi-tilings). As the infinitary condition of the ATM immediately transfers to an infinitary condition of the distributed game, this concludes the proof. □

Theorem 10 (The Büchi case). *The problem of solving a 2-process (or more) distributed game with Büchi condition (or higher parity condition) is Σ_1^1-complete.*

Proof. It should be clear that solving an n-process distributed game with an arbitrary ω-regular infinitary condition is Σ_1^1. Conversely, we encode the con-

struction of an infinite tiling with infinitely many occurrences of color *red* (see Theorem 5).

From the encoding of the infinite tiling problem in the proof of Theorem 6, the idea is to add in local game $\mathcal{G}_{S,t_0}$ a non deterministic tree automaton [12, 4] that checks that, given a local strategy σ followed by player P, the induced quasi-tiling m_σ (seen as a a sub tree of the binary tree $t : (e+t)^* \to S$) uses infinitely many tiles with color *red*.

Such an automaton can be defined with Büchi acceptance criterion that, in turn, defines the winning condition for the Process team. □

7 Conclusion

We have established a correspondence between infinitary conditions in distributed games and levels of the arithmetical (or analytical) hierarchy. These results already hold for the 2-process case (implying undecidability in this very restricted setting).

Strictly speaking, they have no application. However, a clear understanding of the source of undecidability may help, in future work, to extend the known decidable classes of distributed synthesis problem (or distributed games).

Acknowledgment

Thanks to Anne Dicky for her help revising a former version of this paper.

References

1. R. Berger. The undecidability of the dominoe problem. *Memoirs of the American Mathematical Society*, 66:1–72, 1966.
2. J. Bernet and D. Janin. Tree automata and discrete distributed games. In *Foundation of Computing Theory*, volume 3623 of *LNCS*, pages 540–551. Springer-Verlag, 2005.
3. A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, january 1981.
4. E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics and Infinite Games*, volume 2500 of *LNCS Tutorial*. Springer, 2002.
5. D. Harel. Effective transformations on infinite trees, with applications to high undecidability. *J. ACM*, 33(1):224–248, 1986.
6. O. Kupferman and M. Y. Vardi. Synthesizing distributed systems. In *IEEE Symp. on Logic in Computer Science (LICS)*, pages 389–398, 2001.
7. P. Madhusudan. *Control and Synthesis of Open Reactive Systems*. PhD thesis, University of Madras, 2001.
8. S. Mohalik and I. Walukiewicz. Distributed games. In *Found. of Soft. tech and Theor. Comp. Science*, volume 2914 of *LNCS*, pages 338–351. Springer-Verlag, 2003.
9. A. W. Mostowski. Hierarchies of weak automata on weak monadic formulas. *Theoretical Comp. Science*, 83:323–335, 1991.

10. G.L. Peterson and J.H. Reif. Multiple-person alternation. In *20th Annual IEEE Symposium on Foundations of Computer Sciences*, pages 348–363, october 1979.
11. A. Pnueli and R. Rosner. Distributed reactive systems are hard to synthesize. In *IEEE Symposium on Foundations of Computer Science*, pages 746–757, 1990.
12. M. O. Rabin. Decidability of second order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, 1969.