Hierarchical Information Patterns and Distributed Strategy Synthesis

Dietmar Berwanger[1], Anup Basil Mathew[12], and Marie van den Bogaard[1]

[1] LSV, CNRS & Université Paris-Saclay, France
[2] IMSC, Chennai, India

Abstract Infinite games with imperfect information tend to be undecidable unless the information flow is severely restricted. One fundamental decidable case occurs when there is a total ordering among players, such that each player has access to all the information that the following ones receive.
In this paper we consider variations of this hierarchy principle for synchronous games with perfect recall, and identify new decidable classes for which the distributed synthesis problem is solvable with finite-state strategies. In particular, we show that decidability is maintained when the information hierarchy may change along the play, or when transient phases without hierarchical information are allowed.

1 Introduction

Realising systems that are correct by design is a persistent ambition of computing science. The stake is particularly high for systems that interact with an unpredictable environment over indeterminate time. Pioneering results in the area of synthesis, due to Büchi and Landweber [3], and Rabin [16] show that the task can be automatised for the case of a monolithic design, with correctness conditions specified by automata over infinite objects — words or trees representing computations. A most natural framework for representing the problem setting and its solution is in terms of infinite games with perfect information over finite graphs, as described by Pnueli and Rosner in [14], and by Thomas in [17].

For distributed systems, in which several components interact towards satisfying a global specification, the game-theoretical formulation of the synthesis problem leads to games with imperfect information and the question of whether there exists a winning strategy that can be distributed among several players. Unfortunately, such games are much less amenable to automated solutions: as pointed out by Peterson and Reif in [13], it is generally undecidable whether a solution, i.e., a distributed winning strategy, exists for a finitely presented game for two players against Nature (or the environment); furthermore, Janin [8] showed that, even if a solution exists, it may not be implementable by a finite-state device. As there is no hope for solving the distributed synthesis problem uniformly, it remains to look out for classes that allow for an algorithmic treatment. For

surveys on results in this direction, see e.g., the article [5] of Gastin, Sznajder, and Zeitoun, and the thesis of Puchala [15].

One fundamental case in which the distributed synthesis problem becomes decidable is that of hierarchical systems: these correspond to games where there is a total order among the players such that, informally speaking, each player has access to the information received by the players that come later in the order. Peterson and Reif [13] showed that for games in this setting, it is decidable whether distributed winning strategies exist – although, with nonelementary complexity –, and if yes, finite-state winning strategies can be effectively synthesised. The result was extended by Pnueli and Rosner [14] to the framework of distributed systems over fixed linear architectures where information can flow only in one direction. Later, Kupferman and Vardi developed a fundamental automata-theoretic approach [11] that allows to extend the decidability result from linear-time to branching-time specifications, and also removes some of the syntactic restrictions imposed by the fixed-architecture setting of Pnueli and Rosner. Finally, Finkbeiner and Schewe [4] give an effective characterisation of communication architectures on which distributed synthesis is decidable. The criterion requires absence of information forks, which implies a hierarchical order in which processes, or players, have access to the observations provided by the environment.

The setting of games is more liberal than that of architectures with fixed communication channels. For instance, Muscholl and Walukiewicz [12] present a decidable class of synthesis problems under different assumptions on the communication between processes that are not subsumed by information-fork free architectures. A rather general, though non-effective condition for games to admit finite-state distributed winning strategies is given in [2], based on epistemic models representing the knowledge acquired by players in a game with perfect recall. This condition suggests that, beyond the fork-free architecture classification there may be further natural classes of games for which the distributed synthesis problem is decidable.

In this paper, we study a relaxation of the hierarchic-information pattern underlying the basic decidability results on games with imperfect information. Firstly, we extend the assumption of hierarchical observation, that is *positional* information, by incorporating perfect recall. Rather than requiring that a player *observes* the signal received by a less-informed player, we will require that he can *deduce* it from his observation of the play history. It can be easily seen that this gives rise to a decidable class, and it is likely that previous authors had a perfect-recall interpretation in mind when describing hierarchical systems, even if the formal definitions in the relevant literature generally refer to observations.

Secondly, we investigate the case when the hierarchic information order is not fixed, but may change dynamically along the play. This allows to model situations where the schedule of the interaction allows a less-informed player to become more informed than others, or where the players may coordinate on designating one to receive certain signals, and thus become more informed than

others. We show that this condition of dynamic hierarchical observation also leads to a decidable class of the distributed synthesis problem.

As a third extension, we consider the case when the condition of hierarchic information (based on perfect recall) is intermittent. That is, along every play, it occurs infinitely often that the information sets of players are totally ordered; nevertheless, there may be histories, at which incomparable information sets arise, as it is otherwise typical of information forks. We show that, at least for the case of winning conditions over attributes observable by all players, this condition of recurring hierarchical observation is already sufficient for the decidability of the synthesis problem, and that finite-state winning strategies exist for all solvable instances.

For all the three conditions of hierarchic information, it is decidable with relatively low complexity whether they hold for a given game. However, the complexity of solving a game is nonelementary in all cases, as they are more general than the condition of hierarchic observation, known to admit no elementary lower bound [14].

2 Preliminaries

2.1 Games on graphs

We use the standard model of concurrent games with imperfect information, following the notation from [2]. There is a set $N = \{1, \dots, n\}$ of players and a distinguished agent called Nature. A list of elements $x = (x^i)_{i \in N}$, one for each player, is a *profile*. For each player i, we fix a set A^i of *actions* and a set B^i of *observations*; these are finite sets.

A *game graph* $G = (V, E, (\beta^i)_{i \in N})$ consists of a finite set V of nodes called *positions*, an edge relation $E \subseteq V \times A \times V$ representing simultaneous *moves* labelled by action profiles, and a profile of *observation* functions $\beta^i : V \to B^i$ that label every position with an observation, for each player. We assume that the graph has no dead ends, that is, for every position $v \in V$ and every action profile $a \in A$, there exists an outgoing move $(v, a, w) \in E$.

Plays start at a designated initial position $v_0 \in V$ and proceed in rounds. In a round at position v, each player i chooses simultaneously and independently an action $a^i \in A^i$, then Nature chooses a successor position v' reachable along a move $(v, a, v') \in E$. Now, each player i receives the observation $\beta^i(v')$, and the play continues from position v'. Thus, a *play* is an infinite sequence $\pi = v_0, v_1, v_2, \dots$ of positions, such that for all $\ell \geq 0$, there exists a move $(v_\ell, a, v_{\ell+1}) \in E$. A *history* is a nonempty prefix $\pi = v_0, v_1, \dots, v_\ell$ of a play; we refer to ℓ as the *length* of the history, and we denote by $\mathrm{Hist}(G)$ the set of all histories on the game graph G. The observation function extends from positions to histories[1] and plays as $\beta^i(\pi) = \beta^i(v_1), \beta^i(v_2), \dots$, and we write $\mathrm{Hist}^i(G) := \{\beta^i(\pi) \mid \pi \in \mathrm{Hist}(G)\}$ for the set of *observation histories* of

[1] Note that we discard the observation at the initial position; this will be technically convenient and does not restrict the model.

player i. We say that two histories π, π' are *indistinguishable* to player i, and write $\pi \sim^i \pi'$, if $\beta^i(\pi) = \beta^i(\pi')$. This is an equivalence relation, and its classes are called the *information sets*. The information set of player i at history π is $P^i(\pi) := \{\pi' \in \mathrm{Hist}(G) \mid \pi' \sim^i \pi\}$. Accordingly, our model is *synchronous* and we assume *perfect recall*.

A *strategy* for player i is a mapping $s^i : V^* \to A^i$ from histories to actions such that $s^i(\pi) = s^i(\pi')$, for any pair $\pi \sim^i \pi'$ of indistinguishable histories. We denote the set of all strategies of player i with S^i and the set of all strategy profiles by S. A history or play $\pi = v_0, v_1, \ldots$ *follows* the strategy $s^i \in S^i$, if, for every $\ell > 0$, we have $(v_\ell, a, v_{\ell+1}) \in E$ for some action profile a with $a^i = s^i(v_0, v_1, \ldots, v_\ell)$. The play π follows a strategy profile $s \in S$, if it follows all strategies s^i. The set of possible *outcomes* of a strategy profile s is the set of plays that follow s.

A *winning condition* over a game graph G is a set $W \subseteq V^\omega$ of plays. A *distributed game* $\mathcal{G} = (G, W)$ is described by a game graph and a winning condition. We say that a play π is winning in $\mathcal{G}$, if $\pi \in W$. A strategy profile s is winning in $\mathcal{G}$, if all its possible outcomes are so. In this case, we refer to s as a *distributed winning strategy*. *Solving* a game means to determine whether a distributed winning strategy exists, and if yes, to construct one.

Automata

Our focus is on finitely-represented games, where the game graphs are finite and the winning conditions described by finite-state automata. Specifically, winning conditions are given by a colouring function $\gamma : V \to C$ and an ω-regular set $W \subseteq C^\omega$ describing the set of plays $v_0, v_1, \ldots$ with $\gamma(v_0), \gamma(v_1), \cdots \in W$. In certain cases, we assume that the colouring is *observable* to each player i, that is, $\beta^i(v) \neq \beta^i(v')$ whenever $\gamma(v) \neq \gamma(v')$. For general background on automata for games, we refer to the survey [7].

Strategies shall also be represented as finite-state machines. A *Moore* machine over an input alphabet Σ and an output alphabet Γ is described by a tuple (M, m_0, μ, α) consisting of a finite set M of *memory states* with an initial state m_0, a memory *update* function $\mu : M \times \Sigma \to M$ and an *output* function $\nu : M \to \Gamma$ defined on memory states. Intuitively, the machine starts in the initial memory state m_0, and proceeds as follows: in state m, upon reading an input symbol $x \in \Sigma$, updates its memory state to $m' := \mu(m, x)$ and then outputs the letter $\nu(m)$. Formally, the update function μ is extended to input words in Σ^* by setting, $\mu(\varepsilon) := m_0$, for the empty word, and by setting, $\mu(x_0 \ldots x_{\ell-1} x_\ell) := \mu(\,\mu(x_0 \ldots x_{\ell-1}),\, x_\ell)$, for all nontrivial words $x_0 \ldots x_{\ell-1} x_\ell$. This gives rise to the function $M : \Sigma^* \to \Gamma^*$ *implemented* by M, defined by $(x_0, \ldots, x_\ell) := \nu(\,\mu(\,x_0 \ldots x_\ell)\,)$. A *strategy automaton* for player i on a game G, is a Moore machine M with input alphabet B^i and output alphabet A^i. The strategy implemented by M is defined as $s^i(v_0, \ldots, v_{\ell-1}) := M(\beta^i(v_0 \ldots v_{\ell-1}))$. A *finite-state strategy* is one implemented by a strategy automaton.

Sometimes it is convenient to refer to Mealy machines rather than Moore machines. These are finite-state machines of similar format, with the only differ-

ence that the output function $\nu : M \times \Sigma \to \Gamma$ refers to transitions rather than their target state.

In the following we will speak of several classes $\mathcal{C}$ of finite games, always assuming that winning conditions are given as ω-regular languages. We say that the *synthesis problem for $\mathcal{C}$ is finite-state solvable*, if

(i) it is decidable whether a given game $G \in \mathcal{C}$ admits a distributed winning strategy, and
(ii) if yes, we can effectively construct a profile of finite-state machines that implements a distributed winning strategy for G.

3 Static information hierarchies

3.1 Hierarchical observation

We set out from the basic pattern of hierarchical information underlying the decidability results cited in the introduction [13, 14, 11]. These results rely on a positional interpretation of information, i.e., on observations.

Definition 1. A game graph yields *hierarchical observation*, if there exists a total order $\preceq$ among the players such that whenever $i \preceq j$, then for all pairs v, v' of positions, $\beta^i(v) = \beta^i(v')$ implies $\beta^j(v) = \beta^j(v')$

In other words, if $i \preceq j$, then the observation of player i determines the observation of player j.

Kupferman and Vardi present an automata-theoretic construction [11] for solving the distributed synthesis problem for such games. The key operation is that of *widening* – a finite-state interpretation of strategies for a less-informed player j within the strategies of a more-informed player $i \preceq j$. This allows to first solve a game as if all the moves were performed by the most-informed player, which comes first in the order $\preceq$, and successively discard solutions that cannot be implemented by the less-informed players, i.e., those which involve strategies that are not in the image of the widening interpretation.

Theorem 2 ([11]). *For games with hierarchical observation, the synthesis problem is finite-state solvable.*

3.2 Incorporating perfect recall

In a first step, we extend the notion of hierarchical information to incorporate the power of perfect recall that players have. While maintaining the requirement of a fixed order, we now ask that the *information set* of a player determines the information sets of those who follow in the order.

Definition 3. A game graph yields *(static) hierarchical information*, if there exists a total order $\preceq$ among the players such that, for all histories π, if $i \preceq j$ then $P^i(\pi) \subseteq P^j(\pi)$.

The following lemma provides an operational characterisation of the condition. We detail the proof, as its elements will be used later.

Lemma 4. *A game graph G yields static hierarchical information, if and only if, for every pair $i \preceq j$ of players, there exists a Moore machine that outputs $\beta^j(\pi)$ on input $\beta^i(\pi)$, for every history π in G.*

Proof. For an arbitrary game graph G, let us denote the relation between the observations of two players i and j along the histories of G by

$$T^{ij} := \{(\beta^i(\pi), \beta^j(\pi)) \in (B^i \times B^j)^* \mid \pi \in \mathrm{Hist}(G)\}.$$

This is a regular relation, recognised by the game graph G when viewed as a finite-word automaton A^{ij}_G over the alphabet of observation pairs $B^i \times B^j$. Concretely, $A^{ij}_G := (V, B^i \times B^j, v_0, \Delta, V)$ is a nondeterministic automaton on states corresponding to positions of G, with transitions $(v, (b^i, b^j), v') \in \Delta$ if there exists a move $(v, a, v') \in E$ such that $\beta^i(v') = b^i$ and $\beta^j(v') = b^j$; all states are accepting.

($\Leftarrow$) If there exists a Moore machine that recognises T^{ij}, then T^{ij} is actually a function. Thus, $\pi \sim^i \pi'$ implies $\beta^j(\pi) = T^{ij}(\beta^i(\pi)) = T^{ij}(\beta^i(\pi')) = \beta^j(\pi')$, and therefore $\pi \sim^j \pi'$.

($\Rightarrow$) Assuming that G yields static hierarchical information, consider the automaton M^{ij} obtained by determinising A^{ij}_G and trimming the result, that is, removing all states that do not lead to an accepting state. As G yields hierarchical information, the relation T^{ij} recognised by M^{ij} is functional, and hence M^{ij} is deterministic in the input component i: for any state v there exists precisely one outgoing transition along each observation $b^i \in B^i$. In other words, M^{ij} is a Mealy machine, which we can transform into an equivalent Moore machine, as desired. □

Theorem 5. *For games with static hierarchical information, the synthesis problem is finite-state solvable.*

Proof. Intuitively, we transform an arbitrary game graph $G = (V, E, \beta)$ with static hierarchical *information* into one with hierarchical *observation*, by taking the synchronised product of G with automata that signal to each player i the observations of all players $j \succeq i$. We shall see that this preserves the solutions to the distributed synthesis problem, for any winning condition on G.

To make the construction precise, let us fix a pair $i \preceq j$ of players, and consider the Moore machine $M^{ij} = (M, m_0, \mu, \nu)$ from the proof of Lemma 4, which translates the observations $\beta^i(\pi)$ into $\beta^j(\pi)$, for every history π in G. We define the product $G \times M^{ij}$ as a new game graph with the same sets of actions as G, and the same observation alphabets $(B^k)_{k \neq i}$, except for player i, for which we expand the alphabet to $B^i \times B^j$ to also include observations of player j. The new game is over positions in $V \times M$ with moves $((v, m), a, (v', m'))$, if $(v, a, v') \in E$ and $\mu(m, \beta^i(v)) = m'$. The observations for player i are given by $\beta^i(v, m) = (\beta^i(v), \nu(m))$, whereas they remain unchanged for all other players $\beta^k(v, m) = \beta^k(v)$, for all $k \neq i$.

The obtained product graph is equivalent to the original game graph G, in the sense that they have the same tree unravelling, and the additional components in the observations of player i (representing observations of player j, given by the Moore machine M^{ij}) are already determined by his own observation history, so player i cannot distinguish any pair of histories in the new game that he could not distinguish in the original game. Accordingly, the strategies on the expanded game graph $G \times M^{ij}$ correspond to strategies on G, such that the outcomes of any distributed strategy are preserved. In particular, for any winning condition over G, a distributed strategy is winning in the original game if, and only if, it is winning in the expanded game $G \times M^{ij}$. On the other hand, the (positional) observations of Player i in the expanded game determine the observations of Player j.

By applying the transformation for each pair $i \preceq j$ of players successively, we obtain a game graph that is equivalent to G under every winning condition, and which additionally yields hierarchic observation. Due to Theorem 2, we can thus conclude that, under ω-regular winning condition, the synthesis problem is finite-state solvable for games with static hierarchical information. □

To decide whether a given game graph yields static hierarchical information, the collection of Moore machines according to Lemma 4, for all players i, j, may be used as a witness. However, this yields an inefficient procedure, as the determinisation of a functional transducer involves an exponential blowup; precise bounds for such translations are given by Weber and Klemm in [18]. More directly, one could verify that each of the transductions A_G^{ij} relating observation histories of Players i, j, as defined in the proof of Lemma 4, is functional. This can be done in polynomial time using, e.g., the procedure described by Béal et al. in [1].

We can give a precise bound in terms of nondeterministic complexity.

Lemma 6. *The problem of deciding whether a game yields static hierarchical information is* Co-NLogSpace*-complete.*

Proof. The complement problem, of deciding whether for a given game there exists a pair of players i, j that cannot be ordered in either way, is solved by the following nondeterministic procedure: guess a pair of players, then check that $i \not\preceq j$, by following nondeterministically a pair of histories $\pi \sim^i \pi'$, such that $\pi \not\sim^j \pi'$; symmetrically, check that $j \not\preceq i$ — this needs only logarithmic space to maintain pointers to two players and four positions for tracking the histories. Hence, the complement problem is in NLogSpace, which means that our decision problem of whether a game yields static hierarchical information belongs to Co-NLogSpace.

For hardness, we reduce the emptiness problem for nondeterministic finite automata to the problem of deciding whether the following game for two players playing on the graph of the automaton yields hierarchical information: Nature chooses a run, the players can only observe the input letters, until an accepting state is reached; if this happens, Nature sends to each player privately one bit,

which violates the condition of hierarchical information. Thus, the game has hierarchic information if, and only if, no input word is accepted. □

3.3 Signals and game transformations

Functions that return information about the current history, such as constructed in the proof of Lemma 4 will be a useful tool in our exposition, especially when the information that can be made observable to some players.

Given a game graph G, a *signal* is a function defined on the set of histories in G, or on the set of observation histories of some player i. We say that a signal $f : \mathrm{Hist}(G) \to C$ is *information-consistent* for player i, if any two histories that are indistinguishable to player i have the same image under f. A finite-state signal is one implemented by a Moore machine. Any finite-state signal $f : \mathrm{Hist}(G) \to C$ can also be implemented by a Moore machine M^i over the observation alphabet B^i, such that that $M(\pi) = M^i(\beta^i(\pi))$, for every history π. The *synchronisation* of G with a finite-state signal f is the expanded game graph (G, f) obtained by taking the synchronised product $G \times M$, as described in the proof of Lemma 4. In case f is information-consistent for player i, it can be made *positionally observable* to this player, without changing the game essentially. The result is a game graph (G, f^i) that expands (G, f) with an additional observation component $f^i(v)$ for player i at every position v, such that $f(\pi) = f^i(v)$ for each history π that ends at v. Under any winning strategy, the game (G, f^i) is finite-state equivalent to G, in the sense that winning strategies of the original game can be transformed into winning strategies of the synchronised game via standard finite-state operations. In particular, the transformation preserves solutions to the finite-state synthesis problem.

4 Dynamic hierarchies

In this section, we maintain the requirement on the information sets of players to be totally ordered at every history. However, in contrast to the case of static hierarchical information, we allow the order to depend on the history, and to change dynamically along a play.

Definition 7. We say that a history π yields *hierarchical information*, if the information sets $\{P^i(\pi) \mid i \in N\}$ are totally ordered by inclusion. A game graph G yields *dynamic* hierarchical information, if every history yields hierarchical information.

In other words, a game has dynamic hierarchical information, if there is no history at which the information sets of two players are incomparable. To decide whether this is the case, we can use a nondeterministic procedure similar to the one in Lemma 6, to guess two players i, j and three histories $\pi \sim^i \pi'$ and $\pi'' \sim^j \pi$, such that $\pi' \not\sim^i \pi''$ and $\pi' \not\sim^j \pi''$. Since, for a history π, witnesses π', π'' can be guessed and verified by a nondeterministic automaton, it also follows that, for every finite game, the set of histories that yield hierarchical information is regular.

Lemma 8. (i) *For every finite game, we can construct a deterministic finite automaton that recognises the set of histories that yield hierarchical information.*
(ii) *The problem of deciding whether a game yields dynamic hierarchical information is* Co-NLogSpace*-complete.*

In the remainder of the section, we show that, under this more liberal condition, distributed games are still decidable.

Theorem 9. *For games with dynamic hierarchical information, the synthesis problem is finite-state solvable.*

For the proof, we transform an arbitrary game $\mathcal{G}$ with dynamic hierarchical information into one with static hierarchical information, among a different set of n *shadow* players $1', \ldots, n'$, where each shadow player i' plays the role of the i-most informed player in the original game, in a sense that we will make precise soon. The information sets of the shadow players follow their nominal order, that is if $i < j$ then $P^{i'}(\pi) \subseteq P^{j'}(\pi)$. The resulting shadow game inherits the graph structure of the orginal game, and we will ensure that, for every history π,

(i) each shadow player i' has the same information (set) as the i-most informed actual player, and
(ii) each shadow player i' has the same choice of actions as the i-most informed actual player.

This shall guarantee that the shadow game preserves the winning status of the original game.

The construction proceeds in two phases. Firstly, we expand the game graph G so that the correspondence between actual and shadow players does not depend on the history, but only on the current position. This is done by synchronising G with a finite-state machine that signals to each player his rank in the information hierarchy at the current history. Secondly, we modify the game graph, where the shadow-player correspondence is recorded as a positional attribute, such that the observation of each player is received by his shadow player, at every position; similarly, the actions of each player are transferred to his shadow player. Finally, we show how finite-state winning strategies for the shadow game can be re-distributed to the actual players to yield a winning profile of finite-state winning strategies for the original game.

4.1 Information rank signals

For the following, let us fix a game $\mathcal{G}$ with dynamic hierarchical information, with the usual notation. For a history π, we write $\preceq_\pi$ for the total order among players induced by the inclusions between their information sets at π. To formalise the notion of an i-most informed player, we use the shortcut $i \approx_\pi j$ for $i \preceq_\pi j$ and $j \preceq_\pi i$; likewise, we write $i \prec_\pi j$ for $i \preceq_\pi j$ and not $j \preceq_\pi i$.

Then, the *information rank* of player i over the game graph G is a signal $\mathrm{rank}^i : \mathrm{Hist}(G) \to N$ defined by

$$\mathrm{rank}^i(\pi) := |\{j \in N \mid j \prec_\pi i \text{ or } (j < i \text{ and } j \approx_\pi i)\}|.$$

Likewise, we define the *order* of player i *relative* to player j as a signal $\preceq^i_j$: $\mathrm{Hist}(G) \to \{0,1\}$ with $\preceq^i_j(\pi) = 1$ if, and only if $i \preceq_\pi j$.

Lemma 10. *The information rank of each player i, and his order relative to any player j are finite-state signals that are information-consistent to player i.*

Proof. We detail the argument for the rank, the case of relative order is similar and simpler.

Given a game G as in the statement, let us verify that the signal rank^i is information-consistent, for each player i. Towards this, consider two histories $\pi \sim^i \pi'$ in G, and suppose that some player j does not count for the rank of i at π, in the sense that either $i \prec_\pi j$ or ($i \approx_\pi j$ and $i < j$) — in both cases, it follows that $\pi \sim^j \pi'$, hence $P^j(\pi) = P^j(\pi')$, which implies that j does not count for the rank of i at π' either. Hence, the set of players that count for the rank of player i is the same at π and at π', which means that $\mathrm{rank}^i(\pi) = \mathrm{rank}^i(\pi')$.

To see that the signal rank^i can be implemented by a finite-state machine, we first build, for every pair i, j of players, a nondeterministic automaton A^j_i that accepts the histories π where $j \prec_\pi i$, by guessing a history $\pi' \sim^i \pi$ and verifying that $\pi' \not\sim^j \pi$. To accept the histories that satisfy $i \approx_\pi j$, we take the product of the automata A^j_i and A^i_j for $i \preceq_\pi j$ and $j \preceq_\pi i$ and accept if both accept. Combining the two constructions allows us to describe, for every player j, an automaton A_j to recognise the set of histories at which j counts for $\mathrm{rank}^i(\pi)$.

Next, we determinise each of the automata A_j and take appropriate Boolean combinations to obtain a Moore machine M^i with input alphabet V and output alphabet $\mathcal{P}(N)$, which upon reading a history π in G, outputs the set of players that count for $\mathrm{rank}^i(\pi)$. Finally we replace each set in the output of M^i by its size to obtain a Moore machine that returns on input $\pi \in V^*$, the rank of player i at the actual history π in G.

As we showed that rank^i is an information-consistent signal, we can conclude that there exists a Moore machine that inputs observation histories $\beta^i(\pi)$ of player i and outputs $\mathrm{rank}^i(\pi)$. □

One consequence of this construction is that we can view the signals rank^i and $\preceq^i_j$ as attributes of positions rather than properties of histories. Accordingly, we can assume without loss of generality that the observations of each player i have an extra rank component taking values in N and that the symbol j is observed at history π in this component if, and only if, $\mathrm{rank}^i(\pi) = j$. When referring to the positional attribute $\preceq^i_j$ at v, it is more convenient to write $i \preceq_v j$ rather than $\preceq^i_j$.

4.2 Smooth overtaking

As we suggested in the proof outline, each player i and his shadow player, identified by the observable signal rank^i, should be equally informed. To achieve

this, we will let the observation of player i be received by his shadow, in every round of a play. However, since the rank of players, and hence the identity of the shadow, changes with the history, an information loss can occur when the information order between two players, say $1 \prec 2$ along a move is swapped to become $2 \prec 1$ in the next round. Intuitively, the observation received by player 2 after this move contains one piece of information that allows him to catch up with player 1, and another piece of information to overtake player 1. Due to their rank change along the move, the players would now also change shadows. Consequently, the shadow of 1 at the target position, who was previously as (little) informed as player 2, just receives the new observation of player 1, but he may miss the piece of information that allowed player 2 to catch up (and which player 1 had).

We describe a transformation to smoothen the switches in the information order, such that this artefact does no longer occur. Formally, for a play π in a game, we say that Player i and j cross at stage ℓ, if $P^i(\pi_\ell) \subsetneq P^j(\pi_\ell)$ and $P^j(\pi_{\ell+1}) \subsetneq P^i(\pi_{\ell+1})$. We say that a game with dynamic hierarchical information is *cross-free*, if there are no crossing players in any play.

Lemma 11. *Every game with dynamic hierarchical information is finite-state equivalent to a game that is cross-free.*

Proof (sketch). We define a signal for each pair of players i, j that represents the knowledge that player j has about the current observation of player i. If this signal is made observable to Player i only at histories π at which $i \preceq_\pi j$, the game remains essentially unchanged, as players only receive information from less-informed players, which they could hence deduce from their observation. Concretely, we define the signal $\lambda^i_j : V^* \to \mathcal{P}(B^i)$ by

$$\lambda^i_j(\pi) := \{\beta^i(v') \ : \ v' \text{ is the last state of some history } \pi' \in P^j(\pi)\}.$$

Clearly, this is a finite-state signal.

Now we look at the synchronised product of G with the signals $(\lambda^i_j)_{i,j \in N}$ and the relative-order signal $\preceq^i_j$ constructed in the proof of Lemma 10. In the resulting game graph, we add to every move (v, a, w) an intermediary position u, at which we assign, for every player i the observation $\{\lambda^i_j(w) \ : \ i \preceq_w j\}$. Intuitively, this can be viewed as a half-step lookahed signal that player i receives from player j who may have been more informed at the source position v – thus the signal is not necessarily information-consistent for player i. Still, adding the signal leaves the game essentially unchanged, as the players cannot react to the received observation before reaching the target w, at which point the information is no longer relevant. On the other hand, along moves at which the information order between players switches, the intermediary position ensures that the players attain equal information.

To adjust the ω-regular winning conditions for $\mathcal{G}$ to the new game, and to turn any finite-state distributed winning strategy for the new game corresponds into one for the original game, we may just ignore the added intermediary positions. In summary, the construction yields a game graph with no crossings that is finite-state equivalent to the original game graph. □

4.3 Shadow players

We are now ready to describe the construction of the shadow game associated to a game $\mathcal{G} = (V, E, \beta, W)$ with dynamic hierarchical information. Without loss of generality, we can assume that every position in G is marked with the attributes $\mathrm{rank}^i(v)$ and $\sim^i_j$, for all players i, j according to Lemma 10 and that the game graph is cross-free, according to Lemma 11.

The shadow game $\mathcal{G}' = (V \cup \{\ominus\}, E', \beta', W)$ is also played by n players and has the same winning condition as $\mathcal{G}$. The action and the observation alphabet of each shadow player consists of the union of the action and observation alphabets of all actual players. The game graph G' has the same positions as G, plus one sink $\ominus$ that absorbs all moves along unused action profiles. The moves of G' are obtained from G by assigning the actions of each player i to his shadow player $j = \mathrm{rank}^i(v)$ as follows: for every move $(v, a, v') \in E$, there is a move $(v, x, v') \in E'$ labelled with the action profile x obtained by a permutation of a corresponding to the rank order, that is, $a^i = x^j$ for $j = \mathrm{rank}^i(v)$, for all players i. Finally, at every position $v \in V$, the observation of any player i in the original game G is assigned to his shadow player, that is $\beta'^j(v) := \beta^i(v)$, for $j = \mathrm{rank}^i(v)$.

By construction, the shadow game yields static hierarchical information, according to the nominal order of the players. We can verify, by induction on the length of histories, that for every history π, the information set of player i at π in G is the same as the one of his shadow player $\mathrm{rank}^i(\pi)$ in G'.

Finally, we show that the distributed synthesis problem for G reduces to the one on G', and vice versa. To see that $\mathcal{G}'$ admits a winning strategy if $\mathcal{G}$ does, let us fix a distributed strategy s for the actual players in $\mathcal{G}$. We define a signal $\sigma^j : \mathrm{Hist}(G') \to A$ for each player in $\mathcal{G}'$, by setting $\sigma^j(\pi) := s^i(\pi)$ if $j = \mathrm{rank}^i(\pi)$, for each history π. This signal is information-consistent for player j, since, at any history π, his information set is the same as for the actual player i with $\mathrm{rank}^i(\pi) = j$, and because the strategy of the actual player i is information-consistent for himself. Hence, σ^j is a strategy for player j in G'. Furthermore, at every history, the action taken by the shadow player $j = \mathrm{rank}^i(\pi)$ has the same outcome as if it was taken by the actual player i in G. Hence, the set of play outcomes of the profiles s and σ are the same and we can conclude that, if there exists a distributed winning strategy for G, then there also exists one for G'. Notice that this implication holds under any winning condition, without assuming ω-regularity.

For the converse implication, let us suppose that the shadow game $\mathcal{G}'$ admits a winning profile σ of finite-state strategies. We consider, for each actual player i of G, the signal $s^i : \mathrm{Hist}(G) \to A^i$ that maps every history π to the action $s^i(\pi) := \sigma^j(\pi)$ of the shadow player $j = \mathrm{rank}^i(\pi)$. This is a finite-state signal, as we can implement it by synchronising G with rank^i, the observations on the shadow players, and the winning strategies σ^j, for all shadow players j. Moreover, s^i is information-consistent to the actual player i, because all histories $\pi \in P^i(\pi)$, have the same value $\mathrm{rank}^i(\pi) =: j$, and, since s^j is information-consistent for player j, the actions prescribed by $s^j(\pi)$ must be the same, for all

$\pi \in P^j(\pi) = P^i(\pi)$. In conclusion, the signal s^i represents a finite-state strategy for player i. The profile s has the same set of play outcomes outcome as σ, so s is indeed a distributed finite-state strategy, as desired.

In summary, we have shown that any game G with dynamic hierarchical information admits a winning strategy if, and only if, the associated shadow game with static hierarchical observation admits a finite-state winning strategy. The latter question is decidable according to Theorem 5. We showed that for every positive instance G', we can construct a finite-state distributed strategy for G. This concludes the proof of Theorem 9.

5 Transient perturbations

As a third pattern of hierarchical information, we consider the case where incomparable information sets may occur at some histories along a play, but it is guaranteed that a total order will be re-established in a finite number of rounds.

Definition 12. A play yields *recurring hierarchical information* if there are infinitely many histories that yield hierarchical information. A game yields *recurring hierarchical information* if all its plays do so.

Given a play π, we call a *gap* any interval $[t, t+\ell]$ of rounds such that the players do not have hierarchic information at any round in $[t, t+\ell]$; the length of the gap is $\ell + 1$. A game has *gap size* m, if the length of all gaps in its plays are uniformly bounded by m. One important insight is that, in finite games with recurring hierarchical information, only gaps of uniformly bounded size can arise.

Lemma 13. *If a game yields recurring hierarchical information, then its gap size is finite.*

Proof. Let G be an arbitrary finite game graph. As we pointed out in Lemma 8, a history π does not yield hierarchical information, if there exist two players i, j with incomparable information sets; the set of histories at which this occurs can be recognised by a deterministic word automaton. Let $\mathcal{A}$ be a deterministic automaton for the complement language intersected with $\mathrm{Hist}(G)$, that is, $\mathcal{A}$ accepts all histories with hierarchical information. If we now view $\mathcal{A}$ as a Büchi automaton, which accepts all words with infinitely many prefixes accepted by $\mathcal{A}$, we obtain a deterministic ω-word automaton that recognises the set of plays with recurring hierarchical information in G. Applying a standard pumping argument, we can conclude that, if the graph G at the outset yields recurring hierarchical information, its gap size is bounded by the number of states in the automaton $\mathcal{A}$. □

To determine whether a game yields recurring hierarchical information, it is sufficient to check whether the automaton constructed in the proof of Lemma 13 accepts every play.

Corollary 14. *It is decidable whether a game yields recurring hierarchical information.*

We can show that the synthesis problem for this class of games is finite state-solvable, by using the information tracking construction from [2].

Theorem 15. *For games with recurring hierarchical information and observable ω-regular winning conditions, the synthesis problem is finite-state solvable.*

Proof. The tracking construction of [2] reduces the problem of solving a distributed game with imperfect information for n players to that of solving a zero-sum game with perfect information for two players. This is done by unravelling the given game graph G and labelling every history with the *epistemic model* that represents the current knowledge of players, that is, a structure over $\mathrm{Hist}(G)$ equipped with the indistinguishability relations $\sim^i$ and an attribute designating the last state the history; for the epistemic model at a history π, only histories accessible from π via a sequence of $\sim^i$-relations matter. The unravelling generates a game on an infinite tree with perfect information, from which winning strategies can be translated back and forth to the original game.

The main result of [2] shows that, whenever two nodes of the unravelling tree carry homomorphically equivalent labels, they can be identified without changing the solution of the game, at least for observable ω-regular winning conditions. Consequently, the strategy synthesis problem is decidable for a class of games, whenever the unravelling process of any game in the class is guaranteed to generate only finitely many epistemic models, up to homomorphic equivalence.

Games graphs with recurring hierarchical information satisfy this condition. Firstly, for a fixed game, there exist only finitely many epistemic models, up to homomorphic equivalence, where the $\sim^i$-relations are totally ordered by inclusion [2, Section 5]. In other words, epistemic models of bounded size are sufficient to describe all histories with hierarchical information. Secondly, by Lemma 13, from any history with hierarchical information, the (finitely branching) tree of continuation histories with incomparable information is of bounded depth, hence only finitely many epistemic models can occur in the unravelling. Overall, this implies that every game with recurring hierarchical information and observable winning condition has a finite quotient under homomorphic equivalence. According to [2], we can conclude that the distributed strategy problem for the class is finite-state solvable. □

6 Discussion

The bottom-line message of our investigation is that the principle of ordered information flow can afford some flexibility. Still, this may not open floodgates for natural applications to automated synthesis under imperfect information. Rather than expecting information in a real-world system to respect a total order, we see applications in high-level synthesis towards systems on which hierarchical information patterns are enforced to allow for further refinement.

One possible scenario is inspired from multi-level synthesis as proposed in [9] for program repair. Here, the objective is to synthesise a system in several steps: firstly, construct a high-level strategy for a system prototype, in which only a

subset of actions is controllable or/and not all observations are reliable, and subsequently refine this strategy to fulfil further specifications, by controlling more actions or relying on more observations.

For our concrete setting, the first-level synthesis problem can be formulated as follows: given an arbitrary distributed game, determine whether it admits a distributed finite-state winning strategy such that the synchronised product with the original game yields a residual game graph with hierarchical information; if possible, construct one. For the next level, the residual game graph can then be equipped with another winning condition, and the actions or observations may be refined. In either case, the condition of hierarchical information enforced by the the first-level procedure is in place and guarantees decidability of the synthesis problem, for each subsequent level.

It can be easily seen that for any arbitrary graph game, the set of strategies that maintain dynamic hierarchical information is regular. In this case, the multi-level synthesis approach can hence be combined with existing automata-theoretic methods. Unfortunately, this would not work out when the objective is to synthesise a graph with recurring hierarchical information; already the problem of eventually attaining dynamic hierarchical information is undecidable.

Finally, a promising approach towards handling coordination problems under imperfect information is proposed in recent work of Genest, Katz, Peled and Schewe [10, 6], in which strategies are viewed by separating the control and communication layers. The shadow game in our reduction of dynamic to static hierarchical information can be understood as an instance of this idea, with the scheduling of shadow players corresponding to a communication layer, and the actual execution of their strategy (as in the static hierarchical game), to the control layer.

References

[1] M.-P. Béal, O. Carton, C. Prieur, and J. Sakarovitch, *Squaring transducers: An efficient procedure for deciding functionality and sequentiality of transducers*, in LATIN 2000: Theoretical Informatics, G. Gonnet and A. Viola, eds., vol. 1776 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2000, pp. 397–406.

[2] D. Berwanger, Ł. Kaiser, and B. Puchala, *Perfect-information construction for coordination in games*, in Proc. of Foundations of Software Technology and Theoretical Computer Science (FSTTCS'11), vol. 13 of LIPiCS, Leibniz-Zentrum für Informatik, 2011, pp. 387–398.

[3] J. R. Büchi and L. H. Landweber, *Solving sequential conditions by finite-state strategies*, Transactions of the American Mathematical Society, 138 (1969), pp. pp. 295–311.

[4] B. Finkbeiner and S. Schewe, *Uniform distributed synthesis*, in Proc. of LICS '05, IEEE, 2005, pp. 321–330.

[5] P. Gastin, N. Sznajder, and M. Zeitoun, *Distributed synthesis for well-connected architectures*, Formal Methods in System Design, 34 (2009), pp. 215–237.

[6] B. Genest, D. Peled, and S. Schewe, *Knowledge = observation + memory + computation*, in Foundations of Software Science and Computation Structures, A. Pitts, ed., vol. 9034 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2015, pp. 215–229.

[7] E. Grädel, W. Thomas, and T. Wilke, eds., *Automata, Logics, and Infinite Games*, no. 2500 in Lecture Notes in Computer Science, Springer-Verlag, 2002.

[8] D. Janin, *On the (high) undecidability of distributed synthesis problems*, in Proc. of Theory and Practice of Computer Science (SOFSEM 2007), vol. 4362 of Lecture Notes in Computer Science, Springer, 2007, pp. 320–329.

[9] B. Jobstmann, A. Griesmayer, and R. Bloem, *Program repair as a game*, in Computer Aided Verification, CAV 2005, Proc., vol. 3576 of Lecture Notes in Computer Science, Springer, 2005, pp. 226–238.

[10] G. Katz, D. Peled, and S. Schewe, *Synthesis of distributed control through knowledge accumulation*, in Computer Aided Verification, G. Gopalakrishnan and S. Qadeer, eds., vol. 6806 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2011, pp. 510–525.

[11] O. Kupferman and M. Y. Vardi, *Synthesizing distributed systems*, in Proc. of LICS '01, IEEE Computer Society Press, June 2001, pp. 389–398.

[12] A. Muscholl and I. Walukiewicz, *Distributed synthesis for acyclic architectures*, in Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014, Proc., vol. 29 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014, pp. 639–651.

[13] G. L. Peterson and J. H. Reif, *Multiple-Person Alternation*, in Proc 20th Annual Symposium on Foundations of Computer Science, (FOCS 1979), IEEE, 1979, pp. 348–363.

[14] A. Pnueli and R. Rosner, *Distributed reactive systems are hard to synthesize*, in Proceedings of the 31st Annual Symposium on Foundations of Computer Science, FoCS '90, IEEE Computer Society Press, 1990, pp. 746–757.

[15] B. Puchala, *Synthesis of Winning Strategies for Interaction under Partial Information*, PhD thesis, RWTH Aachen University, 2013.

[16] M. O. Rabin, *Automata on infinite objects and Church's thesis*, no. 13 in Regional Conference Series in Mathematics, American Mathematical Society, 1972.

[17] W. Thomas, *On the synthesis of strategies in infinite games*, in STACS, 1995, pp. 1–13.

[18] A. Weber and R. Klemm, *Economy of description for single-valued transducers*, Inf. Comput., 118 (1995), pp. 327–340.