Foundations of Software Technology and Theoretical Computer Science (Bangalore) 2008.
Editors: R. Hariharan, M. Mukund, V. Vinay; pp -

On the Power of Imperfect Information*

Dietmar Berwanger[1] and Laurent Doyen[2]

[1] RWTH Aachen, Germany

[2]EPFL Lausanne, Switzerland

ABSTRACT. We present a polynomial-time reduction from parity games with imperfect information to safety games with imperfect information. Similar reductions for games with perfect information typically increase the game size exponentially. Our construction avoids such a blow-up by using imperfect information to realise succinct counters which cover a range exponentially larger than their size. In particular, the reduction shows that the problem of solving imperfect-information games with safety conditions is EXPTIME-complete.

1 Introduction

Nondeterminism is a notorious source of complexity in automata. The process of determinisation, which consists in monitoring the uncertainty about the flow of control in a nondeterministic device, typically involves a power-set construction and an exponential blow-up of the state space. Reversing the argument, a nondeterministic automaton may be considerably more succinct than any equivalent deterministic automaton.

When we shift from automata to games, a similar jump in complexity arises as an effect of imperfect information of players about the history of a play. Already in the basic setting of two-player zero-sum games, the construction of a perfect-information game monitoring the uncertainty of a player about the flow of information in an imperfect-information game requires a powerset construction [13, 5].

The shape of winning conditions constitutes a further source of complexity in games. In particular in parity games, the range of the priority function is perceived as a key factor. For games with perfect information, the current situation is as follows. While games with two priorities can be solved in quadratic time, the complexity of the best known deterministic algorithms is exponential in the number of priorities. Several procedures for reducing the priorities in a parity game to a fixed small number have been proposed, all leading to an exponential blow-up in the size of the game [3, 9, 15]. A polynomial-time reduction of this kind would prove that parity games can be solved in polynomial time, which is a major open problem.

The question of parity-range reduction has also been investigated in the context of the modal μ-calculus, an expressive logic that subsumes many important specification formalisms. The model-checking problem for this logic corresponds to the problem of solving a parity game with as many priorities as there are alternations of fixed-point quantifiers in

*Research supported in part by the ESF programmes AutoMathA and GAMES, by the Swiss National Science Foundation, and by the European COMBEST project.

the formula [7]. A uniform method for reducing the number of quantifier alternations in formulae would thus lead to tractable model-checking games. For a particular fragment of distributive formulae, a reduction to formulae with only one alternation is presented in [12]. However, the fact that the μ-calculus alternation hierarchy is strict [11, 4, 1], implies that a uniform reduction, which depends only on the formula, cannot exist for the general case. In [15], Seidl proposes a reduction that removes fixed-point alternations syntactically in a non-uniform way, depending on the model, yielding one of the best algorithms for μ-calculus model checking, or equivalently, for solving parity games with perfect information.

In this paper, we consider parity games with imperfect information. We present a polynomial-time reduction of parity games into safety games that preserves the existence of winning strategies. This shows that, in the setting of imperfect information, parity games with only two priorities are able to simulate parity games with arbitrarily many priorities in a succinct way. In other words, the complexity arising from imperfect information preempts the complexity inherent to the winning condition.

The reduction implements a variant of the progress-measure algorithm for solving parity games proposed by Jurdzinski in [8]. We use the power of imperfect information in two ways: firstly, to design counters that cover a range exponentially larger than their size and, secondly, to maintain the number of occurrences of all odd priorities simultaneously during the play. The parity condition is monitored by synchronising the game graph with a small counter gadget equipped with a safety condition.

Our construction illustrates a basic design pattern for applying imperfect information as a synchronisation mechanism. Furthermore, the counting gadgets provide examples of safety games in which winning strategies require memory of exponential size. Finally, our reduction shows that the problem of solving imperfect-information games with safety conditions is EXPTIME-complete.

2 Parity games with perfect information

We first describe the model of parity games with perfect information and introduce the key properties needed for our reduction. In view of a uniform treatment of both perfect and imperfect information models, our terminology sometimes deviates from the standard literature.

2.1 Games and strategies

Let Σ be a finite alphabet of *actions*. A *game structure* with *perfect information* is a tuple $G = (L, \ell_0, \Delta)$ consisting of a finite set L of *locations* (or *positions*), a designated *initial location* $\ell_0 \in L$, and a *transition* relation $\Delta \subseteq L \times \Sigma \times L$. We assume that the transition relation is total, i.e., for every location $\ell \in L$ and every action $a \in \Sigma$, there exists at least one *a-successor* ℓ' such that $(\ell, a, \ell') \in \Delta$, and that all locations of L are reachable from ℓ_0 via transitions in Δ. Games on G are played by two players, Player 1 and Player 2, taking turns to move a token along transitions of G. Initially, the token is located at ℓ_0. The game proceeds in rounds. In every round, Player 1 first chooses an action $a \in \Sigma$, then Player 2 moves the token to an a-successor of the current location. Thus, playing the game yields an infinite sequence of locations $\pi = \ell_1\ell_2\dots$, called a *play*, such that $\ell_1 = \ell_0$ and $(\ell_i, a, \ell_{i+1}) \in \Delta$ for all $i \geq 1$. A *history* is a finite prefix $\ell_1 \dots \ell_i$ of a play. A *strategy* for Player 1 in G is a function $\sigma : L^+ \to \Sigma$ that maps histories to actions. A play $\ell_1\ell_2\dots$ is *consistent* with σ if, for every position $i \geq 1$, there is a transition $(\ell_i, a, \ell_{i+1}) \in \Delta$ with $a = \sigma(\ell_1, \dots, \ell_i)$. We denote the set of plays in G that are consistent with σ by $\mathsf{Outcome}(G, \sigma)$.

A *winning condition* for a game structure G is a set $\varphi \subseteq L^\omega$. A strategy σ for Player 1 is *winning* for the condition φ, if all plays consistent with σ are winning, i.e., $\mathsf{Outcome}(G, \sigma) \subseteq \varphi$. A *game* is a pair (G, φ) consisting of a game structure G and a matching winning condition φ. We say that Player 1 *wins* the game, if he has a winning strategy for the condition φ.

We shall consider two kinds of winning conditions. Given a set $T \subseteq L$ of target locations, the *safety* condition requires that the play stay within the set T: $\mathsf{Safe}(T) = \{\ell_1\ell_2\dots \mid \ell_i \in T \text{ for all } i \geq 1\}$. We call the elements of $L \setminus T$ *bad* locations. Given a *priority function* $\Omega : L \to \mathbb{N}$ that maps each location to a priority, the *parity* condition requires that the least priority visited infinitely often in a play be even: $\mathsf{Parity}(\Omega) = \{\ell_1\ell_2\dots \mid \liminf_{i\to\infty} \Omega(\ell_i) \text{ is even}\}$. Parity conditions can be viewed as nested combinations of safety and reachability conditions, where reachability is the dual of safety: $\mathsf{Reach}(T) = L^\omega \setminus \mathsf{Safe}(L \setminus T)$. They provide a canonical form to express all ω-regular winning conditions [16].

The algorithmic problem of *solving* a game is to decide, given a game structure G and a winning condition φ, whether Player 1 wins the game (G, φ). Safety conditions are specified by a target set, and parity conditions are specified by a priority function.

A conceptually simple way of solving parity games is to provide a winning strategy for Player 1. For this purpose, strategies that depend only on the last location of the history of the play are of particular interest. A strategy σ is *memoryless* if $\sigma(\rho \cdot \ell) = \sigma(\rho' \cdot \ell)$ for all $\rho, \rho' \in L^*$. It is easy to see that, if Player 1 wins a game with safety or reachability condition, then he also has a memoryless strategy to win the game. The following fundamental result establishes that memoryless strategies are sufficient even for perfect-information games with parity conditions.

THEOREM 1.*[[6]] Player* 1 *wins a parity game with perfect information if and only if he has a memoryless winning strategy.*

A memoryless strategy σ for a game structure $G = (L, \ell_0, \Delta)$ can be represented as a substructure G_σ obtained by removing from Δ all transitions (ℓ, a, ℓ') with $a \neq \sigma(\ell)$. The plays in G_σ are then precisely the plays in $\mathsf{Outcome}(G, \sigma)$. Accordingly, for a parity condition φ, the strategy σ is winning if and only if all the plays in G_σ are winning, which amounts to saying that on each cycle in G_σ reachable from ℓ_0, the least visited priority is even. This remark provides the key argument for the transformation of parity games into safety games.

DEFINITION 2. Let $n \in \mathbb{N}$. An infinite sequence $p_1p_2\dots$ of natural numbers is *parity-n-fair* if, for every odd number r, each subsequence $p_ip_{i+1}\dots p_j$ that contains the number r more than n times also contains a number strictly smaller than r.

For a fixed game $(G, \mathsf{Parity}(\Omega))$, we say that a play π is parity-n-fair if the sequence of priorities visited by π is parity-n-fair. Notice that every parity-n-fair play satisfies the parity condition. Conversely, if G has n locations, then all plays consistent with a memoryless winning strategy σ of Player 1 in G are parity-n-fair. Indeed, every subsequence of a play consistent with σ that contains more than n occurrences of an odd priority r must follow a cycle in G_σ. As the least priority in every cycle of G_σ is even, every such subsequence also contains a priority smaller than r. According to Theorem 1, we can hence restrict our attention, without loss of generality, to strategies that enforce parity-n-fair plays.

PROPOSITION 3. *Player* 1 *wins a parity game with perfect information of size* n *if and only if he has a strategy* σ *such that every play consistent with* σ *is parity-*n*-fair.*

Let us now turn to the computational complexity of solving a parity game with perfect information G. A memoryless strategy σ for Player 1 can be guessed in linear time and we can verify in polynomial time whether σ is winning, i.e., whether the minimal priorities on all reachable cycles in G_σ are even. Thus, the problem of solving a game belongs to NP and, by the Determinacy Theorem of [6], it follows that it is in NP $\cap$ Co-NP. Hence the problem is close to polynomial time, in terms of general complexity (see also [8]). The question whether parity games can be solved in polynomial time is a major open problem. The best known deterministic algorithms have running times that are polynomial with respect to the size of the game structure, but exponential with respect to the number of different priorities (see [10, 14]).

2.2 Priority-range reduction

Due to the apparent impact of the number of priorities on the complexity of solving parity games, it would be very desirable to find efficient procedures for reducing the range of the priority function.

An explicit reduction from parity to safety games with perfect information is presented by Bernet, Janin, and Walukiewicz in [3]; it can be understood as an online-version of Jurdzinski's progress-measure algorithm for solving parity games [9]. The main ingredient of the reduction is an internal memory device consisting of a vector of counters, one for each odd priority which is maintained along the transitions of a play. Basically, the device works as follows: if the current state has priority r, all counters corresponding to priorities strictly higher than r are reset; additionally, if r is odd, the counter corresponding to r is incremented. The range of each counter is bounded by the number of locations in the game. To transform a parity game into a safety game, the memory device is synchronised with the game structure via a product operation. Finally, the safety condition requires that no counter overflow occur. Essentially, the internal memory monitors whether the current play is parity-n-fair and forces the play into a bad location when it detects that this is not the case. The correctness of this reduction is justified by arguments similar to Proposition 3. Notice, however, that to monitor a game with n states and d priorities, a memory device with $O(n^{d/2})$ many states is needed. Accordingly, this reduction from parity to safety games involves an exponential blow-up of the game structure.

3 Games with Imperfect Information

We consider a model of games with imperfect information that was originally introduced in [13]. The set of locations is partitioned into information sets indexed by *observations*.

3.1 Observation-based model

In addition to the alphabet Σ of actions, we fix a finite alphabet Γ of observations. A *game structure with imperfect information* over Σ and Γ is a tuple $G = (L, \ell_0, \Delta, \gamma)$, where L, ℓ_0, Δ are defined as in the perfect-information case, and $\gamma : \Gamma \to 2^L \setminus \emptyset$ is an *observability* function that maps each observation to a nonempty set of locations such that the sets $\gamma(o)$ for $o \in \Gamma$ form a partition of L. For each location $\ell \in L$, we write $\mathsf{obs}(\ell)$ to denote the unique observation o such that $\ell \in \gamma(o)$. For an action $a \in \Sigma$ and a set of locations $s \subseteq L$, we define $\mathsf{post}_a(s) = \{\ell' \in L \mid \exists \ell \in s : (\ell, a, \ell') \in \Delta\}$.

The game on G is played in the same way as in the perfect information case, by moving a token along the transitions of G and forming an infinite play. But now, only the observation of the current location is revealed to Player 1. The effect of the uncertainty about the history of the play is formally captured by the notion of strategy.

A *strategy* for Player 1 in G is a function $\sigma : \Gamma^+ \to \Sigma$ that maps finite sequences of observations to actions. Given a play $\pi = \ell_1 \ell_2 \ldots$, we set $\mathsf{obs}(\pi) = \mathsf{obs}(\ell_1)\mathsf{obs}(\ell_2)\ldots$. We say that π is *consistent* with the strategy σ, if for every position $i \geq 1$, there is a transition $(\ell_i, a, \ell_{i+1}) \in \Delta$ with $a = \sigma(\mathsf{obs}(\ell_1) \ldots \mathsf{obs}(\ell_i))$. As before, we denote the set of plays in G that are consistent with σ by $\mathsf{Outcome}(G, \sigma)$.

Following [5], we express winning conditions in terms of observations. A winning condition for a game structure $G = (L, \ell_0, \Delta, \gamma)$ is a set $\varphi \subseteq \Gamma^\omega$ of infinite sequences of observations. A strategy σ for Player 1 is *winning* for the condition φ if $\mathsf{obs}(\pi) \in \varphi$ for all $\pi \in \mathsf{Outcome}(G, \sigma)$. The safety condition for a set $T \subseteq \Gamma$ is $\mathsf{Safe}(T) = \{o_1 o_2 \ldots \mid o_i \in T \text{ for all } i \geq 1\}$, and the parity condition for a priority function $\Omega : \Gamma \to \mathbb{N}$ is defined by $\mathsf{Parity}(\Omega) = \{o_1 o_2 \ldots \mid \liminf_{i \to \infty} p(o_i) \text{ is even}\}$.

Notice that games of perfect information correspond to the special case where $\Gamma = L$ and $\gamma(\ell) = \{\ell\}$ for all $\ell \in L$.

3.2 Reduction to perfect-information games

To solve a game with imperfect information (G, φ) over a structure $G = (L, \ell_0, \Delta, \gamma)$, the basic algorithm proposed in [13] constructs a game of perfect information $(G^{\mathsf{K}}, \varphi')$ over a game structure $G^{\mathsf{K}} = (S, s_0, \Delta')$ with the action alphabet Σ of G, such that Player 1 has a winning strategy for φ in G if and only if he has a winning strategy for φ' in G^{K}. The structure G^{K} is obtained by a *subset construction* which, intuitively, monitors the knowledge that Player 1 has about the current location of the play. The set of locations $S \subseteq 2^L \setminus \{\emptyset\}$ consists of the subsets of L reachable from the initial location $s_0 = \{\ell_0\}$ via transitions in Δ' defined by $(s_1, a, s_2) \in \Delta'$ if and only if there exists an observation $o \in \Gamma$ such that $s_2 = \mathsf{post}_a(s_1) \cap \gamma(o) \neq \emptyset$. Notice that each location in G^{K} corresponds to a unique observation in G, in the sense that for all $s \in S$, there is a unique $o \in \Gamma$ such that $s \subseteq \gamma(o)$. A bijection μ between strategies σ in G and strategies σ^{K} in G^{K} that preserves winning strategies is defined as follows. For all strategies σ in G, set $\mu(\sigma) = \sigma^{\mathsf{K}}$ such that $\sigma^{\mathsf{K}}(s_1 \ldots s_n) = \sigma(o_1 \ldots o_n)$ for all sequences $s_1 \ldots s_n$ of locations in G^{K}, where $o_1 \ldots o_n$ is the unique sequence of observations $o_1 \ldots o_n$ corresponding to $s_1 \ldots s_n$. Conversely, given a strategy σ^{K} in G^{K}, the strategy $\sigma = \mu^{-1}(\sigma^{\mathsf{K}})$ is such that for all $o_1 \ldots o_n \in \Gamma^+$, we have $\sigma(o_1 \ldots o_n) = \sigma^{\mathsf{K}}(s_1 \ldots s_n)$ where $s_1 = s_0$ and $s_{i+1} = \mathsf{post}_{a_i}(s_i) \cap \gamma(o_{i+1})$ with $a_i = \sigma^{\mathsf{K}}(s_1 \ldots s_i)$ for all $1 \leq i < n$. Observe that the plays consistent with σ in G visit the same sequences of priorities as the plays consistent with $\sigma^{\mathsf{K}} = \mu(\sigma)$ in G^{K}.

The construction transforms games with imperfect information into games of perfect information with the same type of winning condition [5]. For a parity condition φ defined by the priority function $\Omega : \Gamma \to \mathbb{N}$, the parity condition φ' is defined by the priority function $\Omega' : S \to \mathbb{N}$ such that $\Omega'(s) = \Omega(o)$ for all $s \in S$ and $o \in \Gamma$ such that $s \subseteq o$.

PROPOSITION 4. *[[5]] Player* 1 *wins a game with imperfect information* $(G, \mathsf{Parity}(\Omega))$ *if and only if he wins the game with perfect information* $(G^{\mathsf{K}}, \mathsf{Parity}(\Omega'))$.

4 Reduction of parity to safety games

To present our reduction from parity to safety games, let us fix a parity game with imperfect information $(G, \mathsf{Parity}(\Omega))$ with n locations and with priorities ranging from 1 to d; we set $[d] = \{1, 2, \dots, d\}$. Without loss of generality, we assume that d is even.

The game structure G^{K} obtained by the subset construction of Section 3.2 has less than 2^n locations. According to Proposition 3, we can require that a winning strategy of Player 1 in $(G^{\mathsf{K}}, \mathsf{Parity}(\Omega))$ (and thus also in $(G, \mathsf{Parity}(\Omega))$) ensures that no odd priority is visited more than 2^n times between two consecutive occurrences of lower priorities. This is a safety condition that can be checked by counting the occurrences of each odd priority in the play. If the count exceeds 2^n while no lower priority is visited, a bad location is entered.

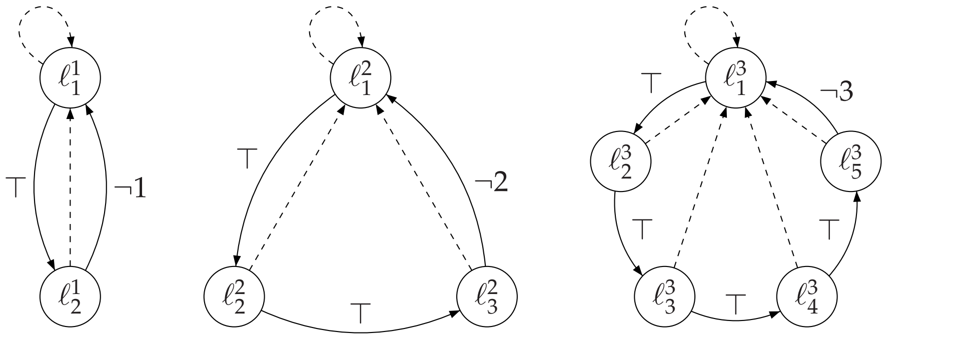
The challenge is to design counters with a bound of at least 2^n and to maintain simultaneously $d/2$ such counters, one for each odd priority, using only a polynomial number of locations.

We use a *counter gadget* to store the number of occurrences of an odd priority r. Whenever a priority smaller than r is visited, the counter is reset. For each visit to priority r, Player 1 has to increment the counter via a *click* action that he can choose from the set $[n] = \{1, 2, \dots, n\}$. The gadgets are constructed in such a way that in each step at least one click can increment the counter, until the upper bound is reached. When this happens, *all* clicks would lead to the bad location.

To each odd priority, we associate a counter gadget. In the first round of the game, Player 2 can choose a counter associated to one particular odd priority r to be tracked. This choice is not observable to Player 1. Thus, Player 1 has to ensure that *every* odd priority occurs only a bounded number of times before a lower priority is visited. This translates the parity condition (that the minimal priority seen infinitely often is even) into a safety condition (that no counter ever overflows).

4.1 Succinct counters

For each odd priority r, the counter gadget C_r is a game structure consisting of n disjoint components (numbered $1, 2, \dots, n$), one for each click. Figure 1 shows a counter gadget with 3 components. The i-th component has the shape of a loop over q_i locations, where q_i is the i-th prime number. The locations of C_r are all indistinguishable to Player 1. Therefore, we may think of a virtual game played simultaneously on all components, as if there was a token moving in each component of the gadget. The number of configurations of the tokens



Increment (solid edges)	on priority $p = r$, with any click except i on edges $\ell^i_{q_i} \to \ell^i_1$.
Reset (dashed edges)	on all priorities $p < r$.
Idle (not depicted)	on all priorities $p > r$ (self-loops).

Figure 1: Counter gadget for priority r with 3 components that counts modulo $2 \cdot 3 \cdot 5 = 30$.

in a counter is given by the primorial $q_n\# = \prod_{i=1}^{n} q_i$. Clearly, we have $q_n\# > 2^n$ whereas the number of locations in a counter is $\sum_{i=1}^{n} q_i = O(n^2 \log n)$ and thus polynomially bounded in n (cf. [2]).

The value of a counter is encoded by the position of the (virtual) tokens in each of its components. A counter can be incremented by taking, simultaneously in all components, a transition represented by a solid edge in Figure 1, it can be reset to 0 with the dashed edges, and it can idle with self-loops on each location (not drawn in the figure). The transitions of C_r are labeled by all actions $(a, p, k) \in \Sigma \times [d] \times [n]$ such that $p > r$ on all idle edges, $p < r$ on all reset edges, and $p = r$ on all increment edges, except the last edge of each component where the click k must be different from the number of the component (in Figure 1, the label $\top$ is interpreted as "for all clicks" and $\neg k$ is interpreted as "for all clicks except k", for $k \in \mathbb{N}$). Finally, we complete the transition relation, by sending all missing transitions to a sink location. Intuitively, whenever a counter is incremented, the value of the click k should be chosen (by Player 1) in such a way that every component has an enabled increment transition labeled with k, i.e., such that q_k does not divide the incremented counter value. This is always possible except when, in *all* components, the token is in the last location before completing the cycle. In the example of Figure 1, this happens after $2 \cdot 3 \cdot 5 - 1 = 29$ steps. From that moment on, Player 1 should avoid visiting priority r unless the counter is reset by a visit to a lower priority.

LEMMA 5. *Let $C_1, C_3, \ldots, C_{d-1}$ be counter gadgets, each with n components. A sequence $p_1 p_2 \ldots$ of priorities $p_i \in [d]$ is parity-($q_n\#$)-fair if and only if there exist sequences $a_1 a_2 \ldots$ of actions and $k_1 k_2 \ldots$ of clicks such that $(a_1, p_1, k_1)(a_2, p_2, k_2) \ldots$ is a play in each of the components of $C_1, \ldots, C_{d-1}$.*

4.2 Reduction

For the parity game with imperfect information $(G, \mathsf{Parity}(\Omega))$ over alphabets Σ and Γ, we construct in polynomial time a safety game with imperfect information $(G, \mathsf{Safe}(T))$ over

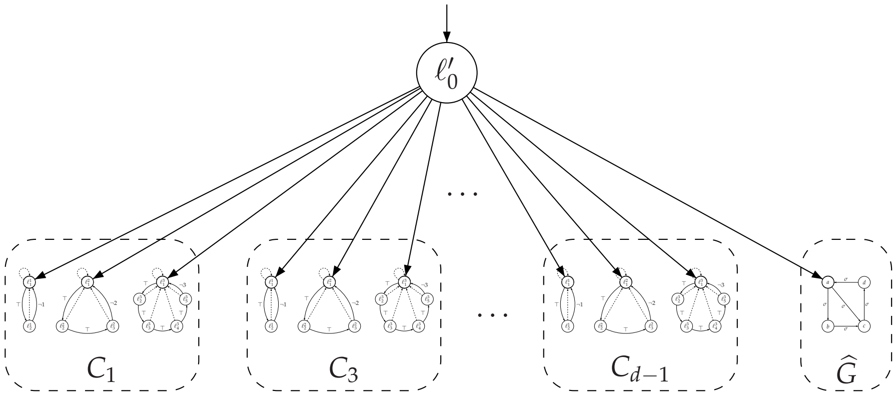


Figure 2: Reduction overview.

extended alphabets Σ' and Γ' such that Player 1 wins $(G, \mathsf{Parity}(\Omega))$ if and only if he wins $(G', \mathsf{Safe}(T))$. The set T contains all locations of G' except a designated sink location. The game structure G' consists of an initial location ℓ_0' from which there is an outgoing transition to the initial location of each of the n components of each counter gadget $C_1, \ldots, C_{d-1}$ and to the initial location of a modified copy $\widehat{G}$ of G, as in Figure 2.

The game structure $\widehat{G}$ enriches the set Σ of actions to synchronise with the counter gadgets. The locations of $\widehat{G}$ are those of G and a fresh location with odd priority. For each transition (ℓ, a, ℓ') in G, there are transitions $(\ell, (a, p, k), \ell')$ for $p = \Omega(\mathsf{obs}(\ell))$ and for all $1 \leq k \leq n$. Hence, the set of actions of $\widehat{G}$ is $\Sigma' = \Sigma \times [d] \times [n]$. We complete the transition relation of $\widehat{G}$ by sending all missing transitions to the fresh location from which Player 1 cannot win. The game $\widehat{G}$ is equivalent to G, as the strategies of Player 1 in G have access to the observation of the current location (and therefore also to its priority) and can thus be translated into equivalent strategies for $\widehat{G}$ by simply choosing the priority $p = \Omega(\mathsf{obs}(\ell))$ of the current location ℓ for the second component of the indicated action (the third component is intended for synchronisation with the clicks and does not matter in $\widehat{G}$).

The observations in G' are the same as in G, that is, $\Gamma' = \Gamma$. However, the observability function γ' of G' is defined for all $o \in \Gamma$ by $\gamma'(o) = \gamma(o) \cup L_C$ where L_C is the set of all locations of the counter gadgets. This defines overlapping observations, but we can construct in polynomial time an equivalent safety game with partitioning observations (*cf.* [5, page 7]).

PROPOSITION 6. *The problem of solving a parity game with imperfect information can be reduced in polynomial time to the problem of solving a safety game with imperfect information.*

Proof. We show that Player 1 wins the game $(G, \mathsf{Parity}(\Omega))$ if and only if he wins the game $(G', \mathsf{Safe}(T))$.

First, let us assume that Player 1 wins $(G', \mathsf{Safe}(T))$ and let us fix a winning strategy σ' in G'. We construct a strategy σ in G such that for all $\rho \in \Gamma^+$, we have $\sigma(\rho) = a$ if $\sigma'(\rho) = (a, p, k)$ for a priority p and a click k. Now we claim that σ is winning in G. To show this, we argue that for all odd priorities r, if r occurs infinitely often in a path π of G_σ, then a smaller priority $p < r$ also occurs infinitely often in π. Towards a contradiction, assume that an odd priority r occurs infinitely often on a path π of G_σ, whereas all priorities lower than r

occur only finitely often. In particular, this implies that π is not a parity-$q_n\#$-fair path. By Lemma 5 it follows that σ', which agrees with σ on the first component (on actions $a \in \Sigma$), cannot avoid an overflow of the counter C_r leading the play to the sink state. Hence, σ' is not a winning strategy in G'.

For the converse, assume that Player 1 wins $(G, \mathsf{Parity}(\Omega))$. Then, there exists a winning strategy σ for Player 1 in G ensuring that every path in G_σ is parity-2^n-fair, by Proposition 3 and via the bijection μ between strategies of G and G^{K} defined in Section 3.2. Therefore, each path of G_σ, can visit at most $2^n < q_n\#$ times an odd priority r without visiting a smaller priority. Hence, each counter C_r is reset before reaching the maximal value $q_n\#$. The winning strategy σ can therefore be extended to a winning strategy in G' by prescribing (a, r, k) whenever σ prescribes a, where r is the priority of the current observation, and k is a click allowed in the corresponding counter C_r (i.e., such that q_i is not a divisor of the number of visits to priority r since the last visit to a smaller priority). In this way, the sink location of the counters is never reached in G' and thus, the strategy σ is winning. ■

If we view the counter gadgets as individual games, we obtain a family of examples of safety games with an exponential lower bound for the memory size of a winning strategy.

COROLLARY 7. *There exists a family $(G_n, \mathsf{Safe}(T_n))_{n \in \mathbb{N}}$ of safety games with imperfect information where Player 1 wins, such that each game G_n is of size polynomial in n, whereas every winning strategy in G_n requires memory of size at least exponential in n.*

The problem of solving reachability and parity games of imperfect information is known to be EXPTIME-complete [13, 5]. However, the question about a matching lower bound for the complexity of safety games remained open. We can now settle this question as a direct consequence of Proposition 6.

COROLLARY 8. *The problem of solving safety games with imperfect information is EXPTIME-complete.*

References

[1] A. ARNOLD, *The μ-calculus alternation-depth is strict on binary trees*, Inf. Théorique et Applications, 33 (1999), pp. 329–339.

[2] E. BACH AND J. SHALLIT, *Algorithmic Number Theory, Vol. 1: Efficient Algorithms*, MIT Press, 1996.

[3] J. BERNET, D. JANIN, AND I. WALUKIEWICZ, *Permissive strategies: from parity games to safety games*, Inf. Théorique et Applications, 36 (2002), pp. 261–275.

[4] J. BRADFIELD, *The modal μ-calculus alternation hierarchy is strict*, Theoretical Computer Science, 195 (1998), pp. 133–153.

[5] K. CHATTERJEE, L. DOYEN, T. A. HENZINGER, AND J.-F. RASKIN, *Algorithms for omega-regular games of incomplete information*, Logical Methods in Computer Science, 3 (2007).

[6] E. A. EMERSON AND C. S. JUTLA, *Tree automata, mu-calculus and determinacy*, in Proc. of FoCS 1991, IEEE, 1991, pp. 368–377.

[7] E. GRÄDEL, W. THOMAS, AND T. WILKE, eds., *Automata, Logics, and Infinite Games*, LNCS 2500, Springer-Verlag, 2002.

[8] M. JURDZIŃSKI, *Deciding the winner in parity games is in UP ∩ co-UP*, Information Processing Letters, 68 (1998), pp. 119–124.

[9] M. JURDZIŃSKI, *Small progress measures for solving parity games*, in Proc. of STACS: Theor. Aspects of Comp. Sc., LNCS 1770, Springer, 2000, pp. 290–301.

[10] M. JURDZIŃSKI, M. PATERSON, AND U. ZWICK, *A deterministic subexponential algorithm for solving parity games*, in Proc. of SODA: Symp. on Discrete Algorithms, ACM Press, 2006, pp. 117–123.

[11] G. LENZI, *A hierarchy theorem for the mu-calculus*, in Proc. of ICALP: Automata, Languages and Programming, LNCS 1099, Springer, 1996, pp. 87–97.

[12] D. NIWINSKI AND H. SEIDL, *On distributive fixed-point expressions*, Inf. Théorique et Applications, 33 (1999), pp. 427–446.

[13] J. REIF, *The complexity of two-player games of incomplete information*, Journal of Computer and System Sciences, 29 (1984), pp. 274–301.

[14] S. SCHEWE, *Solving parity games in big steps*, in Proc. of FSTTCS: Foundations of Software Tech. and Theor. Comp. Sc., LNCS 4855, Springer, 2007, pp. 449–460.

[15] H. SEIDL, *Fast and simple nested fixpoints*, Information Processing Letters, 59 (1996), pp. 303–308.

[16] W. THOMAS, *Languages, automata, and logic*, Handbook of Formal Languages, 3 (1997), pp. 389–455.