Acta Informatica (2018) 55:401–444
https://doi.org/10.1007/s00236-017-0301-x

ORIGINAL ARTICLE

Parity game reductions

Sjoerd Cranen[1] · **Jeroen J. A. Keiren**[2,3] · **Tim A. C. Willemse**[1]

Received: 11 April 2016 / Accepted: 8 July 2017 / Published online: 4 August 2017

Abstract Parity games play a central role in model checking and satisfiability checking. Solving parity games is computationally expensive, among others due to the size of the games, which, for model checking problems, can easily contain 10^9 vertices or beyond. Equivalence relations can be used to reduce the size of a parity game, thereby potentially alleviating part of the computational burden. We reconsider (governed) bisimulation and (governed) stuttering bisimulation, and we give detailed proofs that these relations are equivalences, have unique quotients and they approximate the winning regions of parity games. Furthermore, we present game-based characterisations of these relations. Using these characterisations our equivalences are compared to relations for parity games that can be found in the literature, such as direct simulation equivalence and delayed simulation equivalence. To complete the overview we develop coinductive characterisations of direct- and delayed simulation equivalence and we establish a lattice of equivalences for parity games.

1 Introduction

We study preorders and equivalences defined on *parity games*. Such games are turn-based graph games between two players taking turns pushing a token along the vertices of a finitely

✉ Jeroen J. A. Keiren
Jeroen.Keiren@ou.nl

Sjoerd Cranen
s.cranen@tue.nl

Tim A. C. Willemse
t.a.c.willemse@tue.nl

1 Department of Computer Science and Mathematics, Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven, The Netherlands

2 Faculty of Management, Science and Technology, Open University of the Netherlands, PO Box 2960, 6401 DL Heerlen, The Netherlands

3 Institute for Computing and Information Sciences, Radboud University, Nijmegen, The Netherlands

coloured graph. These players, called *even* and *odd*, strive to optimise the parity of the dominating colour occurring infinitely often in a play. Parity games appear in the core of various foundational results such as Rabin's proof of the decidability of a monadic second-order theory. Solving parity games is a computationally expensive but key step in many model checking algorithms [18,50,51] and synthesis and supervisory control algorithms [1,2,21].

Parity game solving enjoys a special status among combinatorial optimisation problems, being one of the rare problems in the intersection of the UP and coUP classes [35] that is not known to be in P. Despite the continued research effort directed to it, resulting in numerous algorithms for solving parity games, see, *e.g.*, [5,6,36,37,44,48–50,53,56], no polynomial time algorithm has yet been found.

Orthogonally to the algorithmic improvements, heuristics and static analyses have been devised that may speed up solving, or fully solve parity games that occur in practice [20, 32,33]. Such heuristics work particularly well for verification problems, which give rise to games with only few different priorities. In a similar vein, heuristics based on the intimate ties between temporal logics and bisimulation relations are often exploited to speed-up model checking. First minimising a state space by computing the equivalence quotient and only then analysing this quotient can be an effective strategy, see *e.g.* [38].

Given the close connection between parity game solving and model checking, a promising heuristic in this setting is to *reduce* (minimise) a game prior to solving it. Of course, this requires that the winning regions of the original game can be recovered cheaply from the winning regions of the minimised game. Moreover, minimisation makes sense only for equivalence relations that strike a favourable balance between their power to compress the game graph and the computational complexity of quotienting with respect to the equivalence relation. Indeed, in [13,42] we showed that quotienting using standard strong bisimilarity and stuttering equivalence allow to solve parity games that could not be solved otherwise. Despite the immense reductions that can be obtained for parity games encoding natural decision problems, the results were mixed and, apart from a number of cases that become solvable, there was on average no clear gain from using such relations. It should be noted that the stuttering equivalence experiments in [13,14] were conducted on the benchmarks from [40] using the Groote-Vaandrager algorithm [30] which runs in $\mathcal{O}(mn)$, where m is the number of edges and n is the number of states. A recent improvement on this algorithm, described in [31], may very well mean the scale tips in favour of using stuttering equivalence minimisation prior to solving a parity game, as experiments using this $\mathcal{O}(m \log n)$ algorithm have shown speed-ups of several orders of magnitude compared to the $\mathcal{O}(mn)$ algorithm.

Similar observations can be made for *governed bisimilarity* [39] (also known as *idempotence-identifying bisimilarity* in [42]) and *governed stuttering bisimilarity* [14], which weaken strong bisimilarity and stuttering equivalence, respectively, by taking the potentials of players into account. Quotienting for these relations relies on the claim that the relations are equivalences.

As a side-note, simulation and bisimulation relations, tailored to parity games, may lead to insights into the core of the parity game solving problem. Indeed, in *e.g.* [34], Janin relies on different types of simulations to provide uniform proofs when showing the existence of winning strategies; at the same time he suggests simulation relations may ultimately be used to solve games efficiently.

Contributions. In this paper, we revisit the notions of (governed) bisimilarity and (governed) stuttering bisimilarity for parity games from [13,14,39,42]. Parts of this paper have appeared earlier in *ibid.* In this paper we extend upon these works as follows.

We give formal proofs showing that they are indeed equivalence relations and, equally important, that they approximate the winning regions of a parity game, substantiating our claims in the aforementioned papers. Showing that the relations are indeed equivalence relations is technically rather involved, and slight oversights are easily made, see *e.g.* [3]. The added complexity of working in a setting with two players complicates matters significantly compared to, *e.g.*, the setting of labelled transition systems studied in [3].

We furthermore study how our equivalence relations are related to two other notions that have been studied in the context of parity games, *viz. direct simulation* and *delayed simulation* [23] and the latter's *even* and *odd*-biased versions. A complicating factor is the fact that these relations have only *game-based* definitions, whereas our equivalences are defined *coinductively*. We mend this by providing alternative coinductive definitions for direct simulation and delayed simulation, inspired by [45], and we show that these coincide with their game-based definitions. Likewise, we give game-based definitions for our coinductively defined relations, drawing inspiration from [8,55], thereby offering a more operational view on our relations.

Finally, we show that, contrary to (even- and odd-biased) delayed simulation equivalence, direct simulation equivalence, governed bisimilarity and governed stuttering bisimilarity have unique quotients, and, in fact, unique *smallest* quotients. As a result, one can reduce a parity game to the point that it contains no redundancy (from the perspective of the equivalence relation).

In this paper, proofs that are straightforward have been sketched or omitted. Detailed versions of these proofs can be found in the preprint [15]. The proofs of Propositions 7 and 8 are non-trivial, but for the sake of readability, the full details of these proofs have been deferred to the "Appendix".

Related work. In logic, bisimulation has been used to characterise the subfamily of first-order logic that is definable in modal logic [4], and the fragment of monadic second-order logic that is captured by the modal μ-calculus. Bisimulation and simulation-like relations, called *consistent correlations* [54] and *consistent consequence* [25] for PBESs, a fixpoint-logic based framework which is closely related to parity games, were imperative to prove the soundness of the syntax-based static analysis techniques described in [12,41,46,47]. Various simulation relations have been used successfully for minimising Büchi automata, see *e.g.* [10,19,43].

In the context of process theory, there is an abundance of different simulation and bisimulation relations, allowing to reason about the powers of different types of observers of a system's behaviour, see [27,28]. Coinductive definitions of weak behavioural equivalences such as stuttering equivalence (which is, essentially, the same as *branching bisimulation* for labelled transition systems) are commonplace, see [28] for a comprehensive overview. Typically, these definitions rely on the transitive closure of the transition relation. As argued by Namjoshi [45], local reasoning, using single steps instead of resorting to transitive closure, typically leads to simpler arguments. He therefore introduced well-founded bisimulation, a notion equivalent to stuttering bisimulation which solely relies on local reasoning by introducing a well-founded order into the relation. Still, at its basis, well-founded bisimulation only serves to show the reachability of some pair of related vertices. In our coinductive characterisation of the delayed simulation of [23], we use Namjoshi's ideas. However, we need to factor in that in delayed simulation each step on one side must be matched by exactly one step on the simulating side.

There are only a few documented attempts that provide game-based definitions for weak behavioural equivalences. Yin et al. [55] describe branching bisimulation games for normed process algebra. A game-based characterisation of divergence-blind stuttering bisimulation was provided by Bulychev et al. [8]. Neither of these definitions is easily extended to the setting of governed stuttering bisimulation for parity games. In particular, the latter definition is only sound for transition systems that are free of divergences and requires a separate preprocessing step to deal with these. Recently, in [16] we developed a game-based characterisation of branching bisimulation (with explicit divergence) that does not need such a preprocessing step. Our game-based characterisation of governed stuttering bisimulation is based on the same ideas, but requires extensions to the two-player setting of parity games.

Structure of the paper. Parity games are introduced in Sect. 2. In Sect. 3, we introduce notation that facilitates us to define preorders and equivalences on parity games and we state several basic results concerning this notation. A technical overview of the relations studied in the remainder of the paper, and how these are related is presented in Sect. 4. In Sect. 5 we study direct simulation, delayed simulation and its biased versions and in Sect. 6 we study governed bisimulation and governed stuttering bisimulation. Quotienting, for all involved equivalences that admit unique quotients, is discussed in Sect. 7 and in Sect. 8 we return to, and substantiate, the overview we presented in Sect. 4. We wrap up with conclusions and an outlook for future work in Sect. 9.

2 Parity games

A parity game is a two-player graph game, played by two players *even* and *odd* (denoted $\Diamond$ and $\Box$) on a total directed graph in which the vertices are partitioned into two sets, one for each player, and in which a natural priority is assigned to every vertex. The game is played by placing a token on some initial vertex, and if the token is on a vertex owned by player *even*, then she moves the token to a successor of the current vertex (likewise for vertices owned by *odd*). The infinite *play* that results from playing such moves indefinitely is won by player *even* if and only if the least priority on play has even parity. The game is formally defined as follows.

Definition 1 (*Parity game*) A parity game is a directed graph $(V, \to, \Omega, \mathcal{P})$, where

- V is a finite set of vertices,
- $\to \subseteq V \times V$ is a total edge relation (*i.e.*, for each $v \in V$ there is at least one $w \in V$ such that $(v, w) \in \to$),
- $\Omega : V \to \mathbb{N}$ is a priority function that assigns priorities to vertices,
- $\mathcal{P} : V \to \{\Diamond, \Box\}$ is a function assigning vertices to players.

Instead of $(v, w) \in\to$ we typically write $v \to w$, and we write $v^\bullet$ for the set $\{w \in V \mid v \to w\}$. If i is a player, then $\neg i$ denotes the opponent of i, *i.e.*, $\neg\Diamond = \Box$ and $\neg\Box = \Diamond$. The function $\mathcal{P}$ induces a partitioning of V into a set of vertices $V_\Diamond$ owned by player *even* and a set of vertices $V_\Box$ owned by player *odd*; we use $\mathcal{P}$ and $V_\Diamond$, $V_\Box$ interchangeably. The *reward order* [53] on natural numbers is defined such that $n \preccurlyeq m$ if n is even and m is odd; or n and m are even and $n \leq m$, or n and m are odd and $m \leq n$. Note that $n \prec m$ means that n is *better than* m for player *even*. Notions min and max are always used with respect to the standard ordering on natural numbers. Finally, we remark that the assumption that the edge relation is total only serves to simplify the theory described in this paper. All results can be generalised to deal with the situation in which one of the players is unable to move.

Paths. A sequence of vertices $v_0 \dots v_n$ for which $v_m \to v_{m+1}$ for all $m < n$ is a *path*. The concatenation $p_1 p_2$ of paths p_1 and p_2 is again a path, provided there is a step from the last vertex in p_1 to the first vertex in p_2. Infinite paths are defined in a similar manner. We use $p[j]$ to denote the jth vertex in a path p, counting from 0. The set of paths of length n starting in v is defined inductively for $n \geq 1$ as follows:

$$\Pi^1(v) \triangleq \{v\}$$
$$\Pi^{n+1}(v) \triangleq \{pu \mid p \in \Pi^n(v) \wedge p[n-1] \rightarrow u\}$$

The set of infinite paths starting in v is denoted $\Pi^\omega(v)$, and the set of both finite and infinite paths starting in v is defined as follows:

$$\Pi(v) \triangleq \Pi^\omega(v) \cup \bigcup_{n \in \mathbb{N}} \Pi^n(v)$$

Plays and their winners. A game starts by placing a token on some vertex $v \in V$. Players move the token indefinitely according to the following simple rule: if the token is on some vertex v, player $\mathcal{P}(v)$ moves the token to some vertex w such that $v \rightarrow w$. The result is an infinite path p in the game graph; we refer to this infinite path as a *play*. The *parity* of the lowest priority that occurs infinitely often on p defines the *winner* of the play. If this priority is even, then player $\Diamond$ wins, otherwise player $\Box$ wins.

Strategies. A *strategy* for player i is a partial function $\sigma : V^* \rightarrow V$, that is defined only for paths ending in a vertex owned by player i and determines the next vertex to be played onto. The set of strategies for player i in a game $\mathcal{G}$ is denoted $\mathbb{S}^*_{\mathcal{G},i}$, or simply $\mathbb{S}^*_i$ if $\mathcal{G}$ is clear from the context. If a strategy yields the same vertex for every pair of paths that end in the same vertex, then the strategy is said to be *memoryless*. The set of memoryless strategies for player i in a game $\mathcal{G}$ is denoted $\mathbb{S}_{\mathcal{G},i}$, abbreviated to $\mathbb{S}_i$ when $\mathcal{G}$ is clear from the context. A memoryless strategy is usually given as a partial function $\sigma : V \rightarrow V$.

A strategy $\sigma \in \mathbb{S}^*_i$ *allows* a path p of length n, denoted $\sigma \Vdash p$, if and only if for all $j < n-1$ it is the case that if σ is defined for $p[0] \dots p[j]$, then $p[j+1] = \sigma(p[0] \dots p[j])$. The definition of *allows* is extended to infinite paths in the obvious manner. We generalise the definition of Π to paths allowed by a strategy σ; formally, we define:

$$\Pi^n_\sigma(v) \triangleq \{p \in \Pi^n(v) \mid \sigma \Vdash p\}$$

The definition for infinite paths is generalised in the same way and denoted $\Pi^\omega_\sigma(v)$. By $\Pi_\sigma(v)$ we denote $\Pi^\omega_\sigma(v) \cup \bigcup_{n \in \mathbb{N}} \Pi^n_\sigma(v)$, *i.e.*, the set of all finite and infinite paths starting in v and allowed by σ.

A strategy $\sigma \in \mathbb{S}^*_i$ is said to be a *winning strategy* from a vertex v if and only if i is the winner of every infinite path allowed by σ. A vertex is won by player i if i has a winning strategy from that vertex. Likewise, a strategy $\sigma \in \mathbb{S}^*_i$ is a *winning strategy* from a set of vertices $W \subseteq V$ if σ is winning from all $v \in W$.

Solving parity games. It is well-known that parity games are determined, *i.e.* that each vertex in a game is won by exactly one player, and if a winning strategy for a player exists from a vertex, then also a memoryless strategy exists. This is summarised in the following theorem.

Theorem 1 (Memoryless determinacy [17]) *For every parity game there is a unique partition $(W_\Diamond, W_\Box)$ such that winning strategies $\sigma_\Diamond \in \mathbb{S}^*_\Diamond$ from $W_\Diamond$ and $\sigma_\Box \in \mathbb{S}^*_\Box$ from $W_\Box$ exist. Furthermore, for $i \in \{\Diamond, \Box\}$, if $\sigma_i \in \mathbb{S}^*_i$ is winning from W_i a memoryless strategy $\psi_i \in \mathbb{S}_i$ winning from W_i exists.*

The problem of *solving* a parity game is defined as the problem of computing the winning partition $(W_\Diamond, W_\Box)$ of a parity game.

3 Notation

In the remainder of this paper we frequently need to reason about the concept of a player being able to force the play towards a set of vertices. We introduce notation that facilitates such reasoning and we provide some lemmata that express basic properties of parity games in terms of this extended notation. Throughout this section, we fix a parity game $(V, \to, \Omega, \mathcal{P})$. Furthermore, we let $T, U \subseteq V$ be subsets of vertices in the game.

Given a memoryless strategy σ, we introduce a single-step relation ${}_{\sigma}\!\!\to\ \subseteq \to$ that contains only those edges allowed by σ:

$${}_{\sigma}\!\!\to\ \overset{\Delta}{=} \{(v, u) \in \to \mid \text{ if } \sigma(v) \text{ is defined then } u = \sigma(v)\}$$

In other words, ${}_{\sigma}\!\!\to$ contains those edges (v, u) from $\to$ for which $\sigma(v)$ is undefined, or $\sigma(v)$ is defined and $u = \sigma(v)$.

In line with $v \to u$, we write $v \;{}_{\sigma}\!\!\to u$ if $(v, u) \in {}_{\sigma}\!\!\to$. Abstracting from the specific strategy, we write $v \;{}_{i}\!\!\to u$ iff player i has a memoryless strategy σ such that $v \;{}_{\sigma}\!\!\to u$.

We introduce special notation to express which parts of the graph can be reached from a certain node. We use $v \mapsto_U T$ to denote that there is a finite path $v_0 \dots v_n$, for some n, such that $v = v_0$, $v_n \in T$ and for all $j < n$, $v_j \in U$. Conversely, $v \mapsto_U$ denotes the existence of an infinite path $v_0\, v_1 \dots$ for which $v = v_0$ and for all j, $v_j \in U$.

We extend this notation to restrict the reachability analysis to plays that can be enforced by a specific player. We say that strategy σ forces the play from v to T via U, denoted $v \;{}_{\sigma}\!\!\mapsto_U T$, if and only if for all plays p starting in v such that $\sigma \Vdash p$, there exists an n such that $p[n] \in T$ and $p[j] \in U$ for all $j < n$. Note that, in particular, $v \;{}_{\sigma}\!\!\mapsto_U T$ if $v \in T$. Similarly, strategy σ forces the play to diverge in U from v, denoted $v \;{}_{\sigma}\!\!\mapsto_U$, if and only if for all such plays p, $p[j] \in U$ for all j.

Finally, if we are not interested in a particular strategy, but only in the *existence* of a strategy for a player i via which certain parts of the graph are reachable from v, we replace σ by i in our notation to denote an existential quantification over memoryless strategies:

$$v \;{}_{i}\!\!\mapsto_U T \overset{\Delta}{=} \exists \sigma \in \mathbb{S}_i : v \;{}_{\sigma}\!\!\mapsto_U T \qquad\qquad v \;{}_{i}\!\!\mapsto_U \overset{\Delta}{=} \exists \sigma \in \mathbb{S}_i : v \;{}_{\sigma}\!\!\mapsto_U$$

In the following lemmas and definitions, let $v \in V$ be a vertex, $U, T, T' \subseteq V$ be sets of vertices, and i a player. The lemma below shows that rather than using memoryless strategies, one may, if needed, use arbitrary strategies when reasoning about $v \;{}_{i}\!\!\mapsto_U T$.

Lemma 1 $\exists \sigma \in \mathbb{S}_i : v \;{}_{\sigma}\!\!\mapsto_U T$ *iff* $\exists \sigma \in \mathbb{S}_i^* : v \;{}_{\sigma}\!\!\mapsto_U T$.

Proof Observe that the implication from left to right holds by definition. So assume that for some $\sigma \in \mathbb{S}_i^*$, we have $v \;{}_{\sigma}\!\!\mapsto_U T$. Note that $v \;{}_{\sigma}\!\!\mapsto_U T$ iff $v \;{}_{\sigma}\!\!\mapsto_{U \setminus T} T$. The truth value of the latter predicate does not depend on priorities of the vertices and only depends on the edges that originate in $U \setminus T$. Therefore, the truth value of this predicate will not change if we apply the following transformation to our graph:

- for all $u \notin U$, replace all outgoing edges by a single edge $u \to u$.
- set the priorities for all $u \in T$ to 0 iff $i = \Diamond$ and the priorities of all other vertices to 1 iff $i = \Diamond$.

Since $v \;{}_{\sigma}\!\!\mapsto_{U \setminus T} T$, vertex v is won by i in the resulting graph. As parity games are memoryless determined (Theorem 1), i must have a memoryless strategy to move from $U \setminus T$ to T in the resulting graph. Hence there is some $\sigma' \in \mathbb{S}_i$ such that $v \;{}_{\sigma'}\!\!\mapsto_{U \setminus T} T$ in the resulting graph, but then also $v \;{}_{\sigma'}\!\!\mapsto_{U \setminus T} T$ in the original graph, and hence also the required $v \;{}_{\sigma'}\!\!\mapsto_U T$. □

The complement of these relations is denoted by a slashed version of the corresponding arrow, *e.g.*, $\neg v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T$ can be written $v \;{}_i\!\!\not\mapsto\!\!\!\!\rightarrow_U T$. We extend the transition relation of the parity game to sets and to sets of sets in the usual way, *i.e.*, if T is a set of vertices, and $\mathcal{U}$ is a set of vertex sets, then

$$v \to T \triangleq \exists u \in T : v \to u \qquad\qquad v \to \mathcal{U} \triangleq v \to \bigcup \mathcal{U}$$

All other arrow notation is extended in the same way; if a set of sets $\mathcal{U}$ is given as a parameter, it is interpreted as the union of $\mathcal{U}$.

The notation $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T$ is closely related to the notion of *attractor sets* [44]. In essence, the attractor set ${}_U\mathit{Attr}_i(T)$ captures the subset of $U \cup T$ from which i can force the game to $T \subseteq V$, by staying within U until T is reached.

Definition 2 (*Attractor set*) We define ${}_U\mathit{Attr}_i(T)$ as $\bigcup\limits_n {}_U\mathit{Attr}_i^n(T)$ where:

$$\begin{aligned}
{}_U\mathit{Attr}_i^0(T) &\triangleq T \\
{}_U\mathit{Attr}_i^{n+1}(T) &\triangleq {}_U\mathit{Attr}_i^n(T) \\
&\cup \{v \in U \mid \mathcal{P}(v) = i \wedge \exists v' \in v^\bullet : v' \in {}_U\mathit{Attr}_i^n(T)\} \\
&\cup \{v \in U \mid \mathcal{P}(v) \neq i \wedge \forall v' \in v^\bullet : v' \in {}_U\mathit{Attr}_i^n(T)\}
\end{aligned}$$

The attractor set as defined in [44] is obtained for $U = V$. Note that *Attr* is a monotone operator; *i.e.* for $T \subseteq T'$ we have ${}_U\mathit{Attr}_i(T) \subseteq {}_U\mathit{Attr}_i(T')$. The correspondence between our 'forcing' arrow notation and the (generalised) attractor is given by the following lemma.

Lemma 2 *Let $U, T \subseteq V$. Then $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T$ iff $v \in {}_U\mathit{Attr}_i(T)$.*

Proof We first introduce some additional notation. Let $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U^n T$ denote that there is a $\sigma \in \mathbb{S}_i$ such that for all $p \in \Pi_\sigma^{n+1}(v)$, there is some $m \leq n$ such that $p[m] \in T$, and $p[j] \in U$ for all $j < m$. Note that $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T$ iff there is some n such that $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U^n T$. Using induction on n, it follows that $u \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U^n T$ iff $u \in {}_U\mathit{Attr}_i^n(T)$ for all u. The required property is then a direct consequence. □

We are now ready to formalise some intuitions using our notation. The first lemma is essentially about avoiding sets of vertices: it states that if one player can force divergence within a set, then this is the same as saying that the opponent cannot force the play outside this set.

Lemma 3 $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U \iff v \;{}_{\neg i}\!\!\not\mapsto\!\!\!\!\rightarrow_U V \setminus U$

Proof Follows using a similar argument as the proof of Lemma 1. □

Next, we formalise the idea that if a player can force the play to a first set of vertices, and from there he can force the play to a second set of vertices, then he must be able to force the play to that second set.

Lemma 4 $(v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T \wedge \forall u \in T : u \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T') \implies v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T'$

Proof By Lemma 2, $\forall u \in T : u \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T'$ implies $T \subseteq {}_U\mathit{Attr}_i(T')$. By monotonicity of *Attr* we have ${}_U\mathit{Attr}_i(T) \subseteq {}_U\mathit{Attr}_i({}_U\mathit{Attr}_i(T'))$. Since ${}_U\mathit{Attr}_i({}_U\mathit{Attr}_i(T')) = {}_U\mathit{Attr}_i(T')$, we thus have ${}_U\mathit{Attr}_i(T) \subseteq {}_U\mathit{Attr}_i(T')$. By the same token, from $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T$ we find $v \in {}_U\mathit{Attr}_i(T)$. Combined, we find $v \in {}_U\mathit{Attr}_i(T')$ which, by Lemma 2, yields the desired $v \;{}_i\!\!\mapsto\!\!\!\!\rightarrow_U T'$. □

Finally, we state two results that relate a player's capabilities to reach a set of vertices T to the capabilities of the vertices that are able to leave a set of vertices U in a single step.

First, we show that if some player i can force the play from v to T via U, then either i own a vertex in U with a transition to T, or there is vertex owned by $\neg i$ of which all transitions end up in T.

Lemma 5 *Let $S = \{u \in U \mid u^\bullet \cap T \neq \emptyset\}$ and $v \notin T$. Then $v \,{}_i\!\mapsto_U T$ implies $V_i \cap S \neq \emptyset$ or for some $u \in S$, $u^\bullet \subseteq T$.*

Proof Assume $v \,{}_i\!\mapsto_U T$ and suppose $V_i \cap S = \emptyset$. Assume that for all $u \in S$, not $u^\bullet \subseteq T$. Then ${}_U Attr_i(T) = T$ follows from $S \subseteq V_{\neg i}$. Since $v \notin T$ this implies $v \notin {}_U Attr_i(T)$. By Lemma 2 we then have $v \,{}_i\!\not\mapsto_U T$. Contradiction. So there is some $u \in S$ such that $u^\bullet \subseteq T$. □

If player i can force the play from v to T via U, U and T do not overlap, and there are no transitions from U to $T \setminus T'$, then player i can also for the play from v to T' via U.

Lemma 6 *For $v \in U$, $v \,{}_i\!\mapsto_U T$ implies $v \,{}_i\!\mapsto_U T'$ whenever $T' \subseteq T$, $U \cap T = \emptyset$, and $w^\bullet \subseteq U \cup T'$ for all $w \in U$.*

Proof Choose $\sigma \in \mathbb{S}_i$ such that $v \,{}_\sigma\!\mapsto_U T$, and let p be a path for which $p[0] = v \in U$ and $\sigma \Vdash p$. Then for some $j > 0$, $p[j] \in T$ and $p[k] \in U$ (and since $U \cap T = \emptyset$, $p[k] \notin T$) for all $k < j$. Since $p[j-1]^\bullet \subseteq U \cup T'$, also $p[j] \in T'$ and therefore $v \,{}_\sigma\!\mapsto_U T'$. Thus $v \,{}_i\!\mapsto_U T'$. □

4 A lattice of parity game relations

In the rest of this paper, we study relations on parity games. We forego a formal treatment and present an overview of the studied relations and how these are related in this section. Since we study both relational (or coinductive) and game-based characterisations, we first introduce relations and (bi)simulation games.

4.1 Relations

Let R be a relation over a set V, *i.e.* $R \subseteq V \times V$. For $v, w \in V$ we write $v \; R \; w$ to denote $(v, w) \in R$. For a relation R and vertex $v \in V$ we define $v \; R \triangleq \{w \in V \mid v \; R \; w\}$, and likewise $R \; v \triangleq \{w \in V \mid w \; R \; v\}$. We also generalise this notation to sets of vertices such that, for $U \subseteq V$, $U \; R \triangleq \bigcup_{u \in U} u \; R$, and $R \; U \triangleq \bigcup_{u \in U} R \; u$.

A relation R is a *preorder* if it is reflexive and transitive. If, in addition, R is symmetric, then it is an *equivalence* relation. Note that for an equivalence relation R, and vertex $v \in V$, we have $v \; R = R \; v$. In this case we also write $[v]_R \triangleq \{v \in V \mid v \; R \; w\}$, and call this the *equivalence class* of v under R. By abuse of notation, for a subset $U \subseteq V$, we write $[U]_R$ for the set of equivalence classes with respect to V, *i.e.* the set $\{[v]_R \mid v \in U\}$. The set of equivalence classes of V under R is denoted $V_{/R}$, and defined as $\{[v]_R \mid v \in V\}$.

4.2 (Bi)simulation games

Some of the relations we consider in this paper are defined using two-player games. The games we consider are, essentially, Ehrenfeucht-Fraïssé games. For an extensive review of their applications in computer science, the interested reader is referred to [52].

Like parity games, the two-player games we use to characterise relations are instances of two-player, infinite-duration games with ω-regular winning conditions, that are played on game arenas that can be represented by graphs. Each vertex in these games is assigned to one of two players, *Spoiler* and *Duplicator*. In the game, tokens are passed from one vertex to the next in the following fashion. The player that owns the vertex on which the token currently resides pushes the token to an adjacent vertex. These moves continue as long as possible, possibly forever. The path consisting of the sequences of vertices visited by the token is called a play. The winner of the play is decided depending on the predetermined winning criterion, which differs between the games. We say that a player can win from a given vertex if she has a *strategy* such that any play with the token initially at that vertex will be won by her.

The two-player games are also memoryless determined, *i.e.*, every vertex is won by exactly one player, who also has positional winning strategy. These winning strategies can be efficiently computed while solving the game. We refer to [29] for an in-depth treatment of the underlying theory.

4.3 Introducing a lattice of equivalences

Preorders for parity games are particularly (and perhaps only) interesting if they allow one to approximate the winning regions of a game. A preorder R approximates the winning region of a game if, whenever $v\ R\ w$, and player *even* has a winning strategy from v, then she also has a winning strategy from w. For equivalence relations, this requirement is stronger and often more useful: we require that if $v\ R\ w$, then *even* has a winning strategy from v if and only if she has a winning strategy from w. The finest natural equivalence relation on V is *graph isomorphism*, denoted $\cong$.

Definition 3 (*Isomorphism*) Let $(V, \rightarrow, \Omega, \mathcal{P})$ be a parity game. Vertices $v, w \in V$ are *isomorphic*, denoted $v \cong w$ iff $\phi(v) = w$ for some bijection $\phi\colon V \rightarrow V$ that satisfies, for all $\bar{v} \in V$:

- $\Omega(\bar{v}) = \Omega(\phi(\bar{v}))$,
- $\mathcal{P}(\bar{v}) = \mathcal{P}(\phi(\bar{v}))$, and
- $\bar{v} \rightarrow \bar{v}'$ if and only if $\phi(\bar{v}) \rightarrow \phi(\bar{v}')$.

The coarsest sensible equivalence on parity games is the equivalence induced by the determinacy of parity games, *viz.* the equivalence that exactly relates those vertices won by the same player.

Definition 4 (*Winner equivalence*) Let $(V, \rightarrow, \Omega, \mathcal{P})$ be a parity game. Vertices $v, w \in V$ are *winner equivalent*, denoted $v \sim_w w$ iff v and w are won by the same player.

Deciding winner equivalence of parity games is equivalent to partitioning the vertices in the parity game into winning sets.

Winner equivalence and isomorphism are the extreme points in the lattice of equivalence relations shown in Fig. 1. Between the extremal points in the lattice of Figure 1 we list the other parity game equivalences that we study in more detail in the subsequent sections

- Strong bisimilarity ($\leftrightarroweq$) [39,42], see Sect. 6.1;
- Strong direct simulation equivalence ($\equiv_{sd}$), see Sect. 5.1;
- Direct simulation equivalence ($\equiv_d$) [23–25], see Sect. 5.1;
- Delayed simulation equivalence ($\equiv_{de}$) [23], see Sect. 5.2;
- Delayed simulation equivalence, *even*-biased ($\equiv_{de}^{e}$) [23], see Sect. 5.2.1;
- Delayed simulation equivalence, *odd*-biased ($\equiv_{de}^{o}$) [23], see Sect. 5.2.1;

$$\begin{tikzcd}
 & \cong \arrow[d, "1"] & & \\
 & \underline{\leftrightarrow} \arrow[dl, "2"'] \arrow[d, "2"] \arrow[dr, "2"] & & \\
\simeq \arrow[d, "6"'] & \underset{\cdots}{\leftrightarrow} \arrow[dl, "4"] \arrow[dr, "4"] & \equiv_{sd} \arrow[d, "3"] & \\
\underset{\cdots}{\simeq} \arrow[ddd, "7"'] & & \equiv_{d} \arrow[dl, "5"'] \arrow[dr, "5"] & \\
 & \equiv^{o}_{de} \arrow[dr, "5"] & & \equiv^{e}_{de} \arrow[dl, "5"] \\
 & & \equiv_{de} \arrow[dll, "5"] & \\
\sim_{w} & & &
\end{tikzcd}$$

1. Theorem 11;
2. Theorem 12;
3. Theorem 13;
4. Theorem 14;
5. Theorem 16;
6. Theorem 19;
7. Theorem 20.

Fig. 1 Lattice of equivalences for parity games. The numbers on the edges refer to the legend shown to the right, which in turn refers to the theorems that witness the existence of the edge

- Governed bisimilarity ($\underset{\cdots}{\leftrightarrow}$) [39,42], see Sect. 6.1;
- Stuttering bisimilarity ($\simeq$) [13], see Sect. 6.2;
- Governed stuttering bisimilarity ($\underset{\cdots}{\simeq}$) [14], see Sect. 6.2;

In the lattice, an arrow from one equivalence to the other indicates that the first equivalence is finer than the latter. The number on an arrow refers to the theorem in this paper that claims this *strictly finer-than* relation between the equivalences.

The original definitions of the equivalences listed above vary in nature. Strong-, governed-, stuttering-, and governed stuttering bisimilarity are defined coinductively, whereas direct simulation and all variations of delayed simulation are defined as simulation games. Furthermore, the direct- and delayed simulation games define a preorder, whereas the others define an equivalence relation; the preorders are lifted to equivalence relations in the standard way.

5 Direct and delayed simulation equivalence

We introduce the direct simulation preorder and the induced direct simulation equivalence in Sect. 5.1. In Sect. 5.2, we recall the delayed simulation preorder, the induced delayed simulation equivalence and two *biased* versions of the delayed simulation preorder and equivalence. Throughout these sections, we assume an arbitrary parity game $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$.

5.1 Direct simulation and direct simulation equivalence

Direct simulation is one of the most basic preorders studied for parity games. It is difficult to trace the exact origins of the definition, but it was suggested (though not formally defined)

Table 1 Allowed moves in a (bi)simulation game

Note that when playing on v the allowed moves are $v \to v'$ where $v \to v'$ is an edge in the parity game, and likewise for w

$\mathcal{P}(v)$	$\mathcal{P}(w)$	1st move	Plays on	2nd move	Plays on
$\Diamond$	$\Diamond$	S	v	D	w
$\Diamond$	$\Box$	S	v	S	w
$\Box$	$\Diamond$	D	w	D	v
$\Box$	$\Box$	S	w	D	v

in [23] and appeared earlier in the setting of alternating Büchi automata [24]. We here follow the game-based definition as given in [25].

The game is played on pairs of vertices (v, w), where *Duplicator* tries to prove that v is simulated by w, and *Spoiler* tries to disprove this fact (hence the names of the players). The game proceeds in rounds, such that either *Duplicator* or *Spoiler* plays first according to the rules in Table 1. The player who is to play first, and the vertex on which this player has to play is determined by $\mathcal{P}(v)$ and $\mathcal{P}(w)$, *i.e.* the players that own the parity game vertices that constitute the current configuration. For example, in a configuration (v, w) with $\mathcal{P}(v) = \mathcal{P}(w) = \Box$, *Spoiler* gets to play first on w by moving to some w' such that $w \to w'$ in the parity game, and *Duplicator* has to respond from v by playing to some v' such that $v \to v'$ in the parity game.

The intuition here is as follows. Starting in a pair of vertices (v, w), *Spoiler* produces, step by step, an ω-word of priorities which it runs as two 'simultaneous plays' from v and w. *Spoiler* gets to control which successor is chosen at *even* vertices reached by the play that started in v, whereas for the play that started in w, *Duplicator* controls the choice of successor for *even* vertices. This is reversed at *odd* vertices, in which case the player hands over control over the play to her opponent who then gets to choose the successor.

Since the game is total, play continues indefinitely, and *Duplicator* wins if she can always match *Spoiler*'s move with the same priority, showing that the states are equivalent. If *Duplicator* cannot always match the priority, this means *Spoiler* found a way to distinguish two vertices. The game is formally defined as follows.

Definition 5 (*Direct simulation game*) The *direct simulation game* is played on *configurations* drawn from $V \times V$, and it is played in rounds. A round of the game proceeds as follows:

1. The players move from (v, w) to (v', w') such that $v \to v'$ and $w \to w'$, according to the rules in Table 1;
2. Play continues in the next round from the newly reached position (v', w').

An infinite play $(v_0, w_0), (v_1, w_1), \dots$ is won by *Duplicator* if $\Omega(v_j) = \Omega(w_j)$ for all j, *i.e.*, *Duplicator* was able to mimic every move from *Spoiler* with a move to a vertex with equal priority. In all other cases *Spoiler* wins the play.

We say that v is *directly simulated* by w, denoted $v \sqsubseteq_d w$ whenever *Duplicator* has a winning strategy from (v, w) in the direct simulation game.

Example 1 In the parity game in Fig. 2, $v_0 \sqsubseteq_d v_1$. Observe that from (v_0, v_1), *Duplicator* can choose both successors in the direct simulation game. We do not have $v_1 \sqsubseteq_d v_0$, since from configuration (v_1, v_0), *Spoiler*'s move $v_1 \to v_3$ cannot be matched from v_0. Note that additionally we have $v_0 \sqsubseteq_d v_2$ and $v_2 \sqsubseteq_d v_0$, and, of course, $v_i \sqsubseteq_d v_i$ for all i.

Direct simulation is a preorder: reflexivity is easily seen to hold (*Duplicator* can follow a copy-cat strategy), but transitivity is more involved. In the setting of alternating Büchi

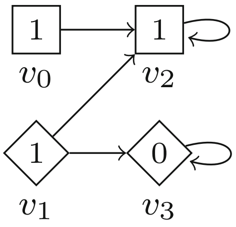


Fig. 2 Parity game with $v_0 \sqsubseteq_d v_1$, $v_0 \sqsubseteq_d v_2$, $v_2 \sqsubseteq_d v_0$, and for all v_i, $v_i \sqsubseteq_d v_i$

automata, direct simulation was shown to be transitive using strategy composition, see [22, 24]. Following essentially the same technique one can show transitivity of direct simulation for parity games. We use the direct simulation preorder to obtain direct simulation equivalence in the standard way.

Definition 6 (*Direct simulation equivalence* [23,25]) Vertices v and w are *direct simulation equivalent*, denoted $v \equiv_d w$, iff $v \sqsubseteq_d w$ and $w \sqsubseteq_d v$.

The alternative coinductive definition of direct simulation which we present next, allows for a more straightforward proof of transitivity. Our definition below was taken from [26], where it is also referred to as *governed simulation*. Intuitively, if $v \in V_\Diamond$, then for every successor $v \to v'$, player even must be able to force to some w' from w in one step, such that $v' \; R \; w'$. The case where $v \in V_\Box$ more or less directly reflects the last two cases in Table 1: if *Duplicator* owns w, she merely needs to find $v \to v'$ and $w \to w'$ such that $v' \; R \; w'$, otherwise, for every such w' she needs to find a $v \to v'$ with $v' \; R \; w'$.

Definition 7 (*Direct simulation relation* [26]) A relation $R \subseteq V \times V$ is a *direct simulation* if and only if $v \; R \; w$ implies

- $\Omega(v) = \Omega(w)$;
- if $v \in V_\Diamond$, then for each $v' \in v^\bullet$, $w \;{}_\Diamond\!\!\to v' \; R$;
- if $v \in V_\Box$, then $w \;{}_\Diamond\!\!\to v^\bullet \; R$.

We write $v \leq_d w$, if and only if there is a direct simulation relation R such that $v \; R \; w$. Likewise, we write $v \rightleftarrows_d w$ iff $v \leq_d w$ and $w \leq_d v$.

The theorem below states that the game-based and coinductive definitions of direct simulation coincide.

Theorem 2 *For all $v, w \in V$, we have*

- *$v \leq_d w$ if and only if $v \sqsubseteq_d w$, and*
- *$v \rightleftarrows_d w$ if and only if $v \equiv_d w$.*

Proof For the first part of the statement, both implications follow straightforwardly from the definitions, using a case distinction on the players of v and w. The second part then follows from the definitions of $\equiv_d$ and $\rightleftarrows_d$. □

Proposition 1 *The relations $\leq_d$ and $\rightleftarrows_d$ are a preorder and an equivalence relation, respectively. Moreover, $\leq_d$ itself is a direct simulation relation.*

Proof For transitivity, one can check that for direct simulation relations R and S, the relation $R \circ S$, defined as $v \; (R \circ S) \; w$ iff there is some u such that $v \; R \; u$ and $u \; S \; w$, is again a direct simulation relation. □

Strong direct simulation. If we impose an additional constraint on direct simulation, *viz.* we do not allow to relate vertices owned by different players, we obtain the stronger notion *strong direct simulation*. Clearly, this notion again is a preorder. We write $v \leq_{sd} w$ iff there is some strong direct simulation relation that relates v and w, and we write $v \equiv_{sd} w$ iff $v \leq_{sd} w$ and $w \leq_{sd} v$. Note that in the parity game in Fig. 2, we still have $v_0 \equiv_{sd} v_2$, but $v_0 \not\leq_{sd} v_1$.

5.2 Delayed simulation

Direct simulation equivalence is limited in its capability to relate vertices. The reason for this is that in each step of the simulation game, *Duplicator* is required to match with a move to a vertex with exactly the same priority. Following Etessami et al. [19], in [23], a more liberal notion of simulation called *delayed simulation* is considered. In this notion matching may be delayed. The idea is that in the winning condition of a play of a parity game, only the priorities that occur infinitely often are of importance. Therefore, intuitively it is allowed to delay matching a given (dominating) priority for a finite number of rounds.

The delayed simulation game is, like the direct simulation game, played on an arena consisting of configurations that contain a pair of vertices, and turns are taken according to the same rules, see Table 1. These configurations are now extended with a third parameter which is used to keep track of the dominating priority that still needs to be matched by *Duplicator*. We say that this is the *obligation* that still needs to be met by *Duplicator*.

An obligation in a delayed simulation game is either a natural number (corresponding to a priority in the game) or the symbol ✓; the latter is used to indicate the absence of obligations, signifying that *Duplicator* has matched the dominating priority played by *Spoiler*. We denote the set of obligations by K. Given two priorities and an existing obligation, a new obligation is obtained using the function $\gamma : \mathbb{N} \times \mathbb{N} \times K \to K$. Let $n, m \in \mathbb{N}$, and $k \in K \setminus \{\checkmark\}$, then γ is defined as follows:

$$\gamma(n, m, \checkmark) = \begin{cases} \checkmark & \text{if } m \preccurlyeq n \\ \min\{n, m\} & \text{otherwise} \end{cases}$$

$$\gamma(n, m, k) = \begin{cases} \checkmark & \text{if } m \preccurlyeq n \text{ and } \begin{cases} n \text{ odd and } n \leq k, \text{ or} \\ m \text{ even and } m \leq k \end{cases} \\ \min\{n, m, k\} & \text{otherwise} \end{cases}$$

For vertices $v, w \in V$ and obligations $k \in K$, we typically write $\gamma(v, w, k)$ to denote $\gamma(\Omega(v), \Omega(w), k)$.

The intuition behind this update is as follows. Either, with the priorities of the states reached after taking a step, the pending obligation is fulfilled, and the configuration reached by taking a step does itself not give rise to a new obligation, in which case the result is ✓. Otherwise the new obligation is the minimum of the priorities passed to the function, or the current obligation. This represents the dominating priority that still needs to be met by *Duplicator*. Note that there are two ways to fulfil the pending obligation. Either the first argument renders the pending obligation superfluous since it is smaller and odd (corresponding to a vertex with a dominating odd priority), or the second argument is such that it matches the pending obligation (corresponding to a vertex with a dominating even priority).

Definition 8 (*Delayed simulation game* [23]) A *delayed simulation game* is a game played by players *Spoiler* and *Duplicator* on an arena consisting of positions drawn from $V \times V$ and obligations taken from K. The game is played in rounds. Assuming (v, w) is the current position, and k the current obligation, a round of the game proceeds as follows:

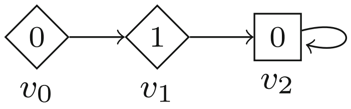


Fig. 3 Parity game in which all vertices are delayed simulation equivalent

1. *Spoiler* and *Duplicator* propose moves $v \to v'$ and $w \to w'$ according to the rules in Table 1.
2. The game continues from (v', w') with obligation $\gamma(v', w', k)$.

An infinite play $(v_0, w_0, k_0), (v_1, w_1, k_1), \ldots$ is won by *Duplicator* iff $k_j = \checkmark$ for infinitely many j. This means that *Duplicator* was always able to eventually fulfil all pending obligations. In all other cases *Spoiler* wins the game.

We say that v is *delayed simulated* by w, denoted $v \sqsubseteq_{de} w$ just whenever *Duplicator* has a winning strategy from (v, w) with obligation $\gamma(v, w, \checkmark)$ in the delayed simulation game.

Example 2 In the parity game in Fig. 3, $v_i \sqsubseteq_{de} v_j$ for all i, j. Observe that, with respect to direct simulation, vertices cannot be related to each other.

Delayed simulation is, like direct simulation, a preorder. The proof thereof is substantially more involved than the proof that direct simulation is a preorder, requiring an analysis of 24 cases, some of which are rather intricate. For details, we refer to [22]; we here only repeat this result.

Proposition 2 *The relation $\sqsubseteq_{de}$ is a preorder.*

We obtain delayed simulation equivalence in the standard way.

Definition 9 (*Delayed simulation equivalence* [23]) Vertices v and w are delayed simulation equivalent, denoted $v \equiv_{de} w$, iff $v \sqsubseteq_{de} w$ and $w \sqsubseteq_{de} v$.

Next, we give an alternative, coinductive definition for delayed simulation. Since the moves in the game for delayed simulation and direct simulation match, one may expect that such a characterisation can be obtained by a more-or-less straightforward enhancement of the direct simulation relation. This is partly true: indeed, the moves of the game are captured in a way similar to how this is done for direct simulation. However, the winning condition of delayed simulation requires that infinitely often all obligations are met. This requires 'non-local' reasoning that must somehow be captured through a coinductive argument. Meeting an obligation is typically a progress property, requiring an inductive argument rather than a coinductive argument.

To combine both aspects in a single, coinductive definition, we draw inspiration from Namjoshi's notion of well-founded bisimulation [45]. Well-founded bisimulation is a relation which is equivalent to stuttering equivalence, but which permits local reasoning by introducing a well-foundedness criterion. We use a similar well-foundedness requirement in our coinductive definition, ensuring progress is made towards fulfilling obligations. This moreover requires, as can be expected, that our coinductive relation ranges not only over pairs of vertices but also over obligations. For a relation $R \subseteq V \times K \times V$, we write $v\ R^k\ w$ if v and w are related under pending obligation k. The well-foundedness restriction thus enables us to express that $v\ R^k\ w$ holds if we can build a coinductive argument that ultimately depends on pairs of vertices v', w' related under obligation $\checkmark$; *viz.* $v'\ R^{\checkmark}\ w'$.

Definition 10 (*Well-founded delayed simulation*) A relation $R \subseteq V \times K \times V$ is a well-founded delayed simulation iff there is a well-founded order $\lessdot$ on $V \times V \times K$ such that for all $v, w \in V$ and $k \in K$ for which $v\ R^k\ w$ holds, also:

- $v \in V_\Diamond$ implies for all $v' \in v^\bullet$, $w \mathrel{{}_\Diamond{\rightarrow}} \{w' \in V \mid \ell = \gamma(v', w', k) \land v' \; R^\ell \; w' \land (k = \checkmark \lor (v', w', \ell) \lessdot (v, w, k))\}$;
- $v \in V_\Box$ implies $w \mathrel{{}_\Diamond{\rightarrow}} \{w' \in V \mid \exists v' \in v^\bullet : \ell = \gamma(v', w', k) \land v' \; R^\ell \; w' \land (k = \checkmark \lor (v', w', \ell) \lessdot (v, w, k))\}$.

Vertex v is *well-founded delayed simulated* by w, denoted $v \leq_{de} w$, iff there exists a well-founded delayed simulation R such that $v \; R^{\gamma(v,w,\checkmark)} \; w$.

In the remainder of this section we show that this definition is equivalent to the game-based definition. We first show that vertices that are related by well-founded delayed simulation are also related by delayed simulation. Essentially we show that for a pair of related vertices $v \; R^{\gamma(v,w,k)} \; w$ (where R is a well-founded delayed simulation), a winning strategy for *Duplicator* from configuration $(v, w, \gamma(v, w, k))$ in the delayed simulation game can be inferred. The proof contains two steps: first, we show that the moves in the game are made between configurations that correspond to related vertices, and second we show that when a move is made according to the induced strategy, eventually a configuration with obligation $\checkmark$ is reached, relying on the well-foundedness of the ordering $\lessdot$.

Lemma 7 *For $v, w \in V$, $v \leq_{de} w$ implies $v \sqsubseteq_{de} w$.*

Proof We prove the stronger statement that if there is a well-founded delayed simulation R such that $v \; R^{\gamma(v,w,k)} \; w$, for $k \in K$, then *Duplicator* has a strategy to win the delayed simulation game from $(v, w, \gamma(v, w, k))$. The result then follows immediately from the observation that $v \; R^{\gamma(v,w,\checkmark)} \; w$, and hence *Duplicator* wins the game from $(v, w, \gamma(v, w, \checkmark))$.

We first show that *Duplicator* has a strategy to move between positions (v, w) with obligation k for which $v \; R^k \; w$ to positions (v', w') and obligation k' for which $v' \; R^{k'} \; w'$. Assume that $v \; R^k \; w$. We distinguish four cases based on the owner of v and w.

- $(v, w) \in V_\Diamond \times V_\Diamond$. In the delayed simulation game, this corresponds to the vertex (v, w, k), in which *Spoiler* is to move first. *Spoiler* first plays an arbitrary move $v \to v'$. By definition of the well-founded delayed simulation, there is $w \to w'$ such that $v' \; R^{\gamma(v',w',k)} \; w'$; *Duplicator* matches with this w'.
- $(v, w) \in V_\Diamond \times V_\Box$. In the delayed simulation game, from position (v, w, k), *Spoiler* is to make both moves, so there is no *Duplicator* strategy to be defined. Observe that well-founded delayed simulation guarantees that for all $v \to v'$ and $w \to w'$, $v' \; R^{\gamma(v',w',k)} \; w'$.
- $(v, w) \in V_\Box \times V_\Diamond$. *Duplicator* plays twice in the delayed simulation game. According to the well-founded delayed simulation, there exist $v \to v'$ and $w \to w'$ such that $v' \; R^{\gamma(v',w',k)} \; w'$. *Duplicator* plays such moves.
- $(v, w) \in V_\Box \times V_\Box$. In the delayed simulation game, *Spoiler* is to move first, say $w \to w'$. From the well-founded delayed simulation, we find that for all such moves, there exists some $v \to v'$ such that $v' \; R^{\gamma(v',w',k)} \; w'$. *Duplicator* plays to this w'.

It remains to be shown that for all configurations (v, w, k) such that $v \; R^k \; w$, this *Duplicator*-strategy is, indeed, winning for *Duplicator*. Observe that it suffices to show that, if $k \neq \checkmark$, eventually a configuration $(v', w', \checkmark)$ is reached. This follows, since in every round in the game above moves are made from (v, w, k) to (v', w', k') such that $(v', w', k') \lessdot (v, w, k)$. Since $\lessdot$ is a well-founded order, this can only be repeated finitely many times, and eventually all obligations are met. □

Before we show the converse, we first show that a winning strategy for player *Duplicator* in the delayed simulation game induces a well-founded order on those configurations won by *Duplicator*. Essentially this is based on the observation that, since *Duplicator*'s strategy is winning from a given configuration, all partial plays starting in that configuration that follow *Duplicator*'s strategy reach a configuration with obligation $\checkmark$ in a finite number of steps.

Lemma 8 *The winning strategy for* Duplicator *in the delayed simulation game induces a well-founded order on $V \times V \times K$ for those (v, w, k) for which* Duplicator *wins position (v, w) with obligation k.*

Proof Observe that the delayed simulation game has a Büchi winning condition. Hence those configurations in the game that *Duplicator* can win, can be won using a memoryless strategy. For the remainder of the proof, fix such a winning memoryless strategy.

For each position (v, w) and obligation k that is won by *Duplicator*, we extract a finite tree from the solitaire game that is induced by *Duplicator*'s strategy by taking the (infinite) unfolding of the game starting in (v, w, k), and pruning each branch at the first node with obligation $\checkmark$. Since the strategy is *Duplicator*-winning, this tree is finite. Furthermore, if (v, w, k) appears in the tree of a different configuration, the subtree rooted in (v, w, k) in that particular subtree is identical to the tree of (v, w, k).

These trees determine a well-founded order $\lessdot$ as follows: $(v', w', \ell) \lessdot (v, w, k)$ iff the height of the tree rooted in (v', w', ℓ) is less than that of the tree rooted in (v, w, k). □

The following corollary immediately follows from the existence of the well-founded order.

Corollary 1 *In the delayed simulation game, for every position (v, w) with obligation k from which* Duplicator *has a winning strategy, if the game proceeds according to this strategy to some position (v', w') with obligation $\ell = \gamma(v', w', k)$, we have $k = \checkmark$ or $(v', w', \ell) \lessdot (v, w, k)$.*

Finally, we show that any delayed simulation is also a well-founded delayed simulation.

Lemma 9 *For $v, w \in V$, $v \sqsubseteq_{de} w$ implies $v \leq_{de} w$.*

Proof The relation $R \subseteq V \times K \times V$ defined as $v\ R^k\ w$ if *Duplicator* wins the delayed simulation game from (v, w) with obligation k is a well-founded delayed simulation. This result follows from two observations. First, a well-founded order on configurations won by *Duplicator* exists, according to Lemma 8. Second, that the transfer condition is satisfied follows from the *Duplicator*-winning strategy in the delayed simulation game, using a straightforward case distinction the players of related vertices. Furthermore, in every step, the well-founded order on configurations won by *Duplicator* decreases in every step according to Corollary 1. □

The following theorem, stating that delayed simulation and well-founded delayed simulation coincide, now follows directly.

Theorem 3 *For all $v, w \in V$ we have $v \sqsubseteq_{de} w$ if and only if $v \leq_{de} w$.*

Proof Follows immediately from Lemmata 7 and 9.

5.2.1 Biased delayed simulations.

As observed in [23], quotienting is problematic for delayed simulation: no sensible definition of quotienting appears to exist such that it guarantees that the quotient is again delayed simulation equivalent to the original game. Fritz and Wilke 'mend' this by introducing two variations (so called *biased* delayed simulations) on delayed simulation which do permit some form of quotienting although these are not unique. We briefly describe these variations below.

Even-biased delayed simulation. The even-biased delayed simulation game, and its coinductive variant, are identical to their delayed simulation and well-founded delayed simulation counterparts. The only difference lies in the update function on obligations. Given two priorities and an existing obligation, a new obligation is obtained using the update function $\gamma^e \colon \mathbb{N} \times \mathbb{N} \times K \to K$, where:

$$\gamma^e(n, m, k) = \begin{cases} k & \text{if } m \preccurlyeq n, n \text{ odd}, n \leq k, \\ & \text{ and } (m \text{ odd or } k < m) \\ \gamma(n, m, k) & \text{otherwise} \end{cases}$$

We again abbreviate $\gamma^e(\Omega(v), \Omega(w), k)$ by $\gamma^e(v, w, k)$.

Using the new update function in the delayed simulation game ensures that a pending obligation is only changed back to ✓ by a small even priority; a small odd priority does not change the obligation. We say that v is *even-biased delayed simulated* by w, denoted $v \sqsubseteq_{de}^{e} w$ iff *Duplicator* has a winning strategy from (v, w) with obligation $\gamma^e(v, w, ✓)$ in the even-biased delayed simulation game.

Likewise, we obtain *well-founded, even-biased delayed simulation* by replacing all occurrences of γ by γ^e in Definition 10. Vertex v is *well-founded, even-biased delayed simulated* by w, denoted $v \leq_{de}^{e} w$, iff there exists a well-founded, even-biased delayed simulation preorder R such that $v\ R^{\gamma^e(v,w,✓)}\ w$.

Odd-biased delayed simulation. The odd-biased delayed simulation is defined in a similar way as the even-biased delayed simulation. Instead of small even priorities leading to an update of a pending obligation, small odd priorities lead to a change in the obligation. Given two priorities and an existing obligation, a new obligation is obtained using the update function $\gamma^o \colon \Omega \times \Omega \times K \to K$, where:

$$\gamma^o(n, m, k) = \begin{cases} k & \text{if } m \preccurlyeq n, m \text{ even}, m \leq k, \\ & \text{ and } (n \text{ even or } k < n) \\ \gamma(n, m, k) & \text{otherwise} \end{cases}$$

The game-based and coinductive definitions are analogous to the even-biased version.

6 Governed bisimulation and governed stuttering bisimulation

In this section we consider essentially two notions of bisimulation for parity games, and some derived notions. First, in Sect. 6.1, we introduce governed bisimulation, which was studied under various guises in *e.g.* [25,39,42]. Governed bisimulation is, as we demonstrate in that section, closely related to direct simulation. Next, in Sect. 6.2, governed stuttering bisimulation [11,14,39] is introduced. The adjective ‘governed’ reflects that there is always one player that *governs*, or *forces* a local choice (and response to such a choice) in a play.

6.1 Governed bisimulation

Our definition of governed bisimulation, as presented below, is based on the one from [42] where it is defined in the closely related setting of *Boolean equation systems*; because of

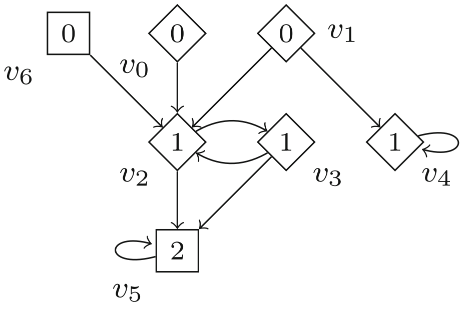


Fig. 4 Parity game in which v_2 and v_3 are governed bisimilar. Vertices v_0, v_1 and v_6 are direct simulation equivalent. Vertices v_0 and v_6 are governed bisimilar but not strong direct simulation equivalent. Vertices v_0 and v_1 are strong direct simulation equivalent, but not governed bisimilar

its capabilities to relate *conjunctive* and *disjunctive* equations, it was dubbed *idempotence identifying bisimulation*. It was rephrased for parity games in [39] and there named *governed bisimulation*.

Definition 11 (*Governed bisimulation*) A symmetric relation $R \subseteq V \times V$ is a *governed bisimulation* iff $v\ R\ w$ implies

- $\Omega(v) = \Omega(w)$;
- if $\mathcal{P}(v) \neq \mathcal{P}(w)$, then $v'\ R\ w'$ for all $v' \in v^\bullet$ and $w' \in w^\bullet$;
- for all $v' \in v^\bullet$ there is some $w' \in w^\bullet$ such that $v'\ R\ w'$.

Vertices v and w are said to be *governed bisimilar*, denoted $v \Leftrightarrow w$, if and only if there is a governed bisimulation R such that $v\ R\ w$.

Example 3 In the parity game in Fig. 4, we have for all i, $v_i \Leftrightarrow v_i$, and furthermore, $v_2 \Leftrightarrow v_3$ and $v_0 \Leftrightarrow v_6$. Observe that we have $v_0 \equiv_d v_1$, where $v_0 \leq_d v_1$ is witnessed by relation $R_1 = \{(v_i, v_i) \mid 0 \leq i \leq 5\} \cup \{(v_0, v_1)\}$, and $v_1 \leq_d v_0$ is witnessed by $R_2 = \{(v_i, v_i) \mid 0 \leq i \leq 5\} \cup (v_1, v_0), (v_4, v_2), (v_4, v_3)\}$. We do, however, not have $v_0 \Leftrightarrow v_1$, since the latter would require v_4 to be related to v_2, but from v_4 the step $v_2 \to v_5$ cannot be mimicked.

Governed bisimulation is such that vertices owned by different players can only be related whenever all their successors are related. It turns out that this is exactly what is obtained when imposing a symmetry requirement on direct simulation. As a result, we have the following proposition.

Proposition 3 *We have $v \Leftrightarrow w$ iff there is a symmetric direct simulation relation R such that $v\ R\ w$.*

As a consequence, we immediately find that governed bisimilarity is an equivalence relation.

Theorem 4 *$\Leftrightarrow$ is an equivalence relation on parity games.*

Proof Follows from combining Proposition 3 and Proposition 1. □

Additionally, the direct simulation game, extended with the possibility for *Spoiler* to switch to a symmetric position in the game play, gives the *direct bisimulation* game, as defined by Etessami et al. [19] for Büchi games.

Definition 12 (*Direct bisimulation game*) The *direct bisimulation game* is played on *configurations* drawn from $V \times V$, and it is played in rounds. A round of the game starting in (v, w) proceeds as follows:

1. *Spoiler* chooses $(u_0, u_1) \in \{(v, w), (w, v)\}$;
2. The players move from (u_0, u_1) according to the rules in Table 1
3. Play continues in the next round from the newly reached position.

An infinite play $(v_0, w_0), (v_1, w_1), \dots$ is won by *Duplicator* if $\Omega(v_j) = \Omega(w_j)$ for all j, *i.e.*, *Duplicator* was able to mimic every move from *Spoiler* with a move to a vertex with equal priority. In all other cases *Spoiler* wins the play.

We write $v \equiv_g w$ whenever *Duplicator* has a winning strategy from (v, w) in the direct bisimulation game.

The following theorem now formally establishes the correspondence between governed bisimilarity and direct bisimulation.

Theorem 5 *For all $v, w \in V$, we have $v \underline{\leftrightarrow} w$ if and only if $v \equiv_g w$.*

Proof Straightforward from the definitions, using a case distinction on the players of v and w. □

Strong Bisimulation. If we again impose the additional constraint on governed bisimulation that we do not allow to relate vertices owned by different players, we obtain a notion called *strong bisimulation* [39]. The derived notion of *strong bisimilarity*, denoted $v \underline{\leftrightarrow} w$ and defined as $v \underline{\leftrightarrow} w$ iff there is some strong bisimulation relation that relates v and w, is an equivalence relation.

6.2 Governed stuttering bisimulation

The (bi)simulation games discussed so far all have in common that the game-play proceeds in 'lock-step': *Duplicator* must match every move proposed by *Spoiler* with a proper countermove. In a sense, this ignores the fact that the parity condition is not sensitive to finite repetitions of priorities but only cares about infinite repetitions. The insensitivity of the parity condition to finite repetitions is reminiscent to the notion of *stuttering* in process theory. Indeed, as we demonstrate in what follows, governed bisimulation can be weakened such that it becomes insensitive to finite stuttering, but remains sensitive to infinite stuttering. The resulting relation is called *governed stuttering bisimulation*.

Essentially, governed stuttering bisimulation is obtained by porting stuttering equivalence for Kripke structures [7] to the setting of parity games. Intuitively, governed stuttering bisimulation requires that a move from a vertex v to v' is matched by a finite (and potentially empty) sequence of moves from w, through vertices that remain related to v, to arrive at some w' that is related to v'. In addition, every *divergent play* from a vertex v (*i.e.* a play that remains confined to a single equivalence class) should be matched with a divergent play from a related vertex w.

Details, however, are subtle. In stuttering equivalence it suffices to have the *ability* to move or diverge and match such moves or divergences with some move or a divergence. In contrast, in the parity game setting, we have to consider the abilities of a player, knowing that a single player does not control the moves from all vertices. Only moves and divergences that can be *forced* by a player count, and matching of such moves and divergences must be done through moves or divergences that the same player can force. Figure 5 illustrates some of these concepts. While in the depicted parity game there is an infinite play that passes through the two left-most vertices with priority 0, neither *even* nor *odd* can force such an infinite play. As a result, we may ignore such infinite plays, and in this sense, the abilities (for both players) from those two vertices are no different from the abilities both players have from the two right-most vertices with priority 0.

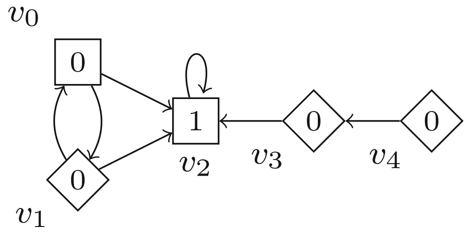


Fig. 5 Equal priorities are related by $\simeq$. Neither player can force play to visit only vertices with priority 0

The definition of governed stuttering bisimulation presented below is based on [11,14]. For our definition, we strongly rely on our notation to denote that a player is able to 'force play'. The idea is that, if from one vertex v another equivalence class $\mathcal{C}$ can be reached in a single step, then from a related vertex, the player owning v can force the play to end up in the same equivalence class $\mathcal{C}$ (irrespective of the opponent's moves). Furthermore, if a player can force the game to stay in the equivalence class of v indefinitely, than can also do so from the related vertex.

Definition 13 (*Governed stuttering bisimulation*) Let $R \subseteq V \times V$ be an equivalence relation. Then R is a *governed stuttering bisimulation* if and only if $v\ R\ w$ implies

(a) $\Omega(v) = \Omega(w)$;
(b) $v \to \mathcal{C}$ implies $w\ {}_{\mathcal{P}(v)}{\mapsto\!\!\to}_{vR}\ \mathcal{C}$, for all $\mathcal{C} \in V_{/R} \setminus \{[v]_R\}$.
(c) $v\ {}_{i}{\mapsto\!\!\to}_{R}$ implies $w\ {}_{i}{\mapsto\!\!\to}_{R}$ for $i \in \{\Diamond, \Box\}$.

Vertices v and w are *governed stuttering bisimilar*, denoted $v \simeq w$, iff a governed stuttering bisimulation R exists such that $v\ R\ w$.

Example 4 The parity game in Fig. 5 nicely illustrates the key properties of governed stuttering bisimulation: for v_j in $\{v_0, v_1, v_3, v_4\}$ we have neither $v_j\ {}_{\Diamond}{\mapsto\!\!\to}_{\simeq}$ nor $v_j\ {}_{\Box}{\mapsto\!\!\to}_{\simeq}$. Furthermore, for all these vertices, both players can force the game to reach vertex v_2. Therefore, all vertices with the same priorities are related by $\simeq$. Also note that the vertices with priority 0 are not related by, *e.g.*, governed bisimulation since the latter is sensitive to counting, and v_0 and v_1 can reach multiple equivalence classes.

Proving that $\simeq$ is a governed stuttering bisimulation relation is technically involved. In particular, all standard proof techniques for doing so break down or become too complex to manage. Instead of a large monolithic proof of the result, we proceed in small steps by gradually rephrasing the above definition to one that is ultimately more easily seen to be an equivalence. Our first step in this direction is to remove the asymmetry in clause (b) of the definition of governed stuttering bisimulation. Before we do so, we state a useful lemma that allows us to strengthen the conclusion of Lemma 5. Essentially, the lemma states the following. Suppose that there is a vertex v in its equivalence class such that all its outgoing edges are sure to leave the equivalence class into some set $\bigcup \mathcal{U}$. Then for all vertices u related to v, any transition leaving the equivalence class is also sure to end up in $\bigcup \mathcal{U}$.

Lemma 10 *Let R be a governed stuttering bisimulation. Let $\mathcal{U} \subseteq V_{/R} \setminus \{[v]_R\}$. If $v^\bullet \subseteq \bigcup \mathcal{U}$, then $u^\bullet \setminus [v]_R \subseteq \bigcup \mathcal{U}$ for all $u \in [v]_R$.*

Proof Let v be such that $v \to \mathcal{U}$ for some $\mathcal{U} \subseteq V_{/R} \setminus \{[v]_R\}$. Suppose $u \to \mathcal{C}$ for some $\mathcal{C} \notin \mathcal{U} \cup \{[v]_R\}$. Since $v \simeq u$, by Definition 13, we have $v\ {}_{\mathcal{P}(u)}{\mapsto\!\!\to}_{vR}\ \mathcal{C}$. But $v^\bullet \subseteq \bigcup \mathcal{U}$ and $\mathcal{C} \notin \mathcal{U}$ so $v\ {}_{\mathcal{P}(u)}{\not\mapsto\!\!\to}_{R}\ \mathcal{C}$. Contradiction. □

Now consider the following alternative for governed stuttering bisimulation. Instead of requiring that, if v leaves to another equivalence class $\mathcal{C}$ directly, than the player owning v has a strategy to do so from any related vertex, we make the second requirement symmetric. Informally, if a player i has a strategy to force the play to some equivalence class $\mathcal{C}$, then she also has a strategy to achieve the same from any related vertex.

Proposition 4 *Let $R \subseteq V \times V$ and $v, w \in V$. Then R is a governed stuttering bisimulation iff R is an equivalence relation and $v\ R\ w$ implies:*

(a) $\Omega(v) = \Omega(w)$;
(b) $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}$ *iff* $w\ {}_i{\mapsto}_{wR}\ \mathcal{C}$ *for all* $i \in \{\Diamond, \Box\}, \mathcal{C} \in V_{/R} \setminus \{[v]_R\}$;
(c) $v\ {}_i{\mapsto}_R$ *iff* $w\ {}_i{\mapsto}_R$ *for all* $i \in \{\Diamond, \Box\}$.

Proof The proof for the implication from right to left follows immediately. We focus on the implication from left to right. Assume that R is a governed stuttering bisimulation. We prove the second condition only; the other two conditions follow immediately from Definition 13 and symmetry of R. Let i be an arbitrary player and assume that $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}$ for given $v \in V$ and $\mathcal{C} \in V_{/R} \setminus \{[v]_R\}$. Let $S = \{u \in [v]_R \mid u^\bullet \cap \mathcal{C} \neq \emptyset\}$. We distinguish two cases.

– Case $V_i \cap S \neq \emptyset$. Let $u \in S \cap V_i$. Since $u \to \mathcal{C}$, $w\ {}_i{\mapsto}_{wR}\ \mathcal{C}$ follows from Definition 13.
– Case $S \subseteq V_{\neg i}$. By Lemma 5, there is a $u \in S$ for which $u^\bullet \subseteq \mathcal{C}$ and by Lemma 10 (for $\mathcal{U} = \{\mathcal{C}\}$), $u^\bullet \setminus [v]_R \subseteq \mathcal{C}$ for *all* $u \in [v]_R$. Furthermore, by Lemma 3, $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}$ implies $v\ {}_{\neg i}{\not\mapsto}_R$. Then, by Definition 13, $w\ {}_{\neg i}{\not\mapsto}_R$ and by Lemma 3, $w\ {}_i{\mapsto}_{wR}\ V \setminus [v]_R$. But since $u^\bullet \subseteq \mathcal{C} \cup [v]_R$ for all u, the desired $w\ {}_i{\mapsto}_{wR}\ \mathcal{C}$ follows from Lemma 6. □

While the above alternative characterisation of governed stuttering bisimulation is now fully symmetric, using the restriction on the class $\mathcal{C}$ that is considered in clause (b) we were still not able to give an insightful proof that $\simeq$ is an equivalence relation using this definition. We therefore further generalise this clause such that it is phrased in terms of *sets* of classes. So, if from a vertex v, player i can force the play to a set of classes $\mathcal{U}$, then she can do so from any related vertex.

A perhaps surprising side-result of this generalisation is that the divergence requirement of clause (c) becomes superfluous. This is due to the duality between diverging in a set by one player, and forcing to the complement of that set by the other as shown in Lemma 3. Note that this generalisation is not trivial, as $v\ {}_i{\mapsto}_{vR}\ \{\mathcal{C}_1, \mathcal{C}_2\}$ is in general neither equivalent to saying that $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}_1$ and $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}_2$, nor $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}_1$ or $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}_2$.

Proposition 5 *Let $R \subseteq V \times V$ and $v, w \in V$. Then R is a governed stuttering bisimulation iff R is an equivalence relation and $v\ R\ w$ implies:*

(a) $\Omega(v) = \Omega(w)$;
(b) $v\ {}_i{\mapsto}_{vR}\ \mathcal{U}$ *iff* $w\ {}_i{\mapsto}_{wR}\ \mathcal{U}$ *for all* $i \in \{\Diamond, \Box\}, \mathcal{U} \subseteq V_{/R} \setminus \{[v]_R\}$.

Proof We show that the second condition is equivalent to the conjunction of the last two conditions in Proposition 4. We split the proof into an *if*-part and an *only-if*-part.

$\Leftarrow$ The second condition from Proposition 4 is equivalent to the second condition above if we let $\mathcal{U}$ range only over singleton sets (if $v\ {}_i{\mapsto}_{vR}\ \mathcal{C}$, take $\mathcal{U} = \{\mathcal{C}\}$). The third condition is equivalent to the second condition above, where $\mathcal{U} = V_{/R} \setminus \{[v]_R\}$. This can be seen by appealing to Lemma 3.

$\Rightarrow$ Let R be a governed stuttering bisimulation and let $v, w \in V$ such that $v\ R\ w$. Assume that $v\ {}_i{\mapsto}_{vR}\ \mathcal{U}$ for some $\mathcal{U} \subseteq V_{/R} \setminus \{[v]_R\}$. Let $S = \{u \in [v]_R \mid u \to \mathcal{U}\}$. Using similar arguments as in the proof of Proposition 4, distinguishing cases on emptiness on $S \cap V_i$, we obtain that $w\ {}_i{\mapsto}_{wR}\ \mathcal{U}$.

In the previous proposition, we lifted the notion of forcing play via the current equivalence class towards a target class, to the notion of forcing a play via the current equivalence class towards a set of target classes. Towards the proving transitivity of governed stuttering bisimulation, we introduce a final generalisation in the proposition below; rather than forcing play towards a set of target classes via the current equivalence class, we now allow the play to be forced to that set via a set of equivalence classes. For this definition, we identify a set of equivalence classes $\mathcal{U}$ with the set of states $\bigcup \mathcal{U}$ to allow us to write ${}_i{\mapsto}_{\mathcal{U}}$ instead of ${}_i{\mapsto}_{\bigcup \mathcal{U}}$.

Proposition 6 *Let $R \subseteq V \times V$ and $v, w \in V$. Then R is a governed stuttering bisimulation iff R is an equivalence relation and $v \ R \ w$ implies:*

(a) $\Omega(v) = \Omega(w)$;
(b) $v \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$ *iff* $w \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$ *for all* $i \in \{\Diamond, \Box\}, \mathcal{U}, \mathcal{T} \subseteq V_{/R}$ *such that* $[v]_R \in \mathcal{U}$ *and* $[v]_R \notin \mathcal{T}$.

Proof We show that the second condition is equivalent to the second condition in Proposition 5. We split the proof into an *if*-case and an *only-if*-part.

$\Leftarrow$ The second condition from Proposition 5 is equivalent to the second condition above if we fix $\mathcal{U} = \{[v]_R\}$.

$\Rightarrow$ Let R be a governed stuttering bisimulation and let $i, v, w, \mathcal{U}$ and $\mathcal{T}$ be as described. Assume that $v \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$; under this assumption we will prove that $w \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$. The proof for the implication in the other direction is completely symmetric. Let $\sigma \in \mathbb{S}_i$ be such that $v \ {}_\sigma{\mapsto}_{\mathcal{U}} \mathcal{T}$ and consider the set of paths originating in v that are allowed by σ. All these paths must have a prefix $v \dots v', u$ such that $v, \dots, v' \notin \bigcup \mathcal{T}$ but $u \in \bigcup \mathcal{T}$. Call these prefixes the σ-prefixes of v.
We proceed by induction on the length of the longest such prefix. If the longest prefix has length 2, then all prefixes have length 2, implying that $v \ {}_i{\to} \mathcal{T}$. In particular, $v \ {}_i{\mapsto}_{vR} \mathcal{T}$ and by Proposition 5 also $w \ {}_i{\mapsto}_{wR} \mathcal{T}$, which proves $w \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$.
As the induction hypothesis, assume that if $u \ R \ u'$, $u \ {}_\sigma{\mapsto}_{\mathcal{U}} \mathcal{T}$ and the longest σ-prefix of u is shorter than the longest σ-prefix of v, then $u' \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$. Note that every σ-prefix p of v must have a first position n such that $p[n] \notin [v]_R$. Collect all these $p[n]$ in a set U, and notice that for all $u \in U$, also $u \ {}_\sigma{\mapsto}_{\mathcal{U}} \mathcal{T}$. Furthermore, $v \ {}_\sigma{\mapsto}_{vR} \ U$.
By Proposition 5, $w \ {}_i{\mapsto}_{wR} [U]_R$. Now consider an arbitrary $u' \in \bigcup [U]_R$. Because there is some $u \in U$ such that $u \ R \ u'$, its longest σ-prefix is shorter than the longest σ-prefix of v, and because $u \ {}_\sigma{\mapsto}_{\mathcal{U}} \mathcal{T}$ for such u, we can use the induction hypothesis to derive that $u' \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$.
The above in particular implies two facts: $w \ {}_i{\mapsto}_{\mathcal{U}} [U]_R$, and $u' \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$ for all $u' \in \bigcup [U]_R$. Using these, we can now apply Lemma 4 to conclude $w \ {}_i{\mapsto}_{\mathcal{U}} \mathcal{T}$. □

With this last characterisation, it is now straightforward to prove that governed stuttering bisimilarity is an equivalence relation. We do so by showing that the transitive closure of the union of two governed stuttering bisimulations R and S is again a governed stuttering bisimulation. The generalisation from classes to sets of classes allows us to view equivalence classes in $(R \cup S)^*$, *i.e.*, the transitive closure of $R \cup S$, as the union of sets of equivalence classes of R (or S), giving us an easy way to compare the effect of the second requirement of Proposition 6 on $(R \cup S)^*$ with its effect on R and S.

Theorem 6 *$\simeq$ is an equivalence relation.*

Proof We show that $(R \cup S)^*$ is a governed stuttering bisimulation if R and S are, by showing that $(R \cup S)^*$ satisfies the conditions of Proposition 6 if R and S do. If $v, w \in V$ are related under $(R \cup S)^*$, then there exists a sequence of vertices $u_0, \ldots, u_n$ such that $v\ R\ u_0\ S\ \ldots\ R\ u_n\ S\ w$ (the strict alternation between the two relations can always be achieved because R and S are reflexive). By transitivity of $=$ we then have $\Omega(v) = \Omega(w)$, so the first property is satisfied.

For the second property, assume that $v\ {}_i{\mapsto}_{\mathcal{U}}\ \mathcal{T}$ for some $i \in \{\Diamond, \Box\}$ and some $\mathcal{U}, \mathcal{T} \subseteq V_{/(R\cup S)^*}$ such that $[v]_{(R\cup S)^*} \in \mathcal{U}$ and $[v]_{(R\cup S)^*} \notin \mathcal{T}$. We need to prove that $w\ {}_i{\mapsto}_{\mathcal{U}}\ \mathcal{T}$. Note that R and S both refine $(R \cup S)^*$, so we can find sets $\mathcal{U}_R \subseteq V_{/R}$ and $\mathcal{U}_S \subseteq V_{/S}$ such that $\bigcup \mathcal{U}_R = \bigcup \mathcal{U}_S = \bigcup \mathcal{U}$. Because $v\ {}_i{\mapsto}_{\mathcal{U}}\ \mathcal{T}$, also $v\ {}_i{\mapsto}_{\mathcal{U}_R}\ \mathcal{T}$, and by Proposition 6 then $u_0\ {}_i{\mapsto}_{\mathcal{U}_R}\ \mathcal{T}$, which is equivalent to $u_0\ {}_i{\mapsto}_{\mathcal{U}_S}\ \mathcal{T}$. By a simple inductive argument we now arrive at $w\ {}_i{\mapsto}_{\mathcal{U}_S}\ \mathcal{T}$, which is equivalent to $w\ {}_i{\mapsto}_{\mathcal{U}}\ \mathcal{T}$. □

As a side-result of the proof of Theorem 6, we find that the union of *all* governed stuttering bisimulations is again a governed stuttering bisimulation, which coincides with governed stuttering bisimilarity.

In order to better understand the differences between governed stuttering bisimulation and, *e.g.* delayed simulation equivalence, we next provide a game-based characterisation of the relation. While in this new game, *Spoiler* and *Duplicator* still move according to the same rules as in the delayed simulation game, *Duplicator* now has more freedom to choose a new configuration: she can now also choose to 'roll-back' one of the proposed moves. This allows her to postpone matching a move. Of course, such moves may not be postponed indefinitely, so some additional mechanism is needed to keep track of *Duplicator*'s progress so as to prevent *Duplicator* from becoming too powerful. For this, we follow [16], and we use a system of challenges and rewards: a $\dagger$-challenge indicates *Duplicator* decided to match a move by *Spoiler* by not moving; a $\checkmark$-reward indicates *Duplicator* matched a move by *Spoiler* by making a countermove, and a challenge (k, u) taken from $\{0, 1\} \times V$ indicates that *Duplicator* is in the process of matching a move to vertex u. We let C denote the set of challenges $(\{0, 1\} \times V) \cup \{\dagger, \checkmark\}$.

Definition 14 (*Governed Stuttering bisimulation game*) The governed stuttering bisimulation game is played on an arena of configurations drawn from $(V \times V) \times C$, and it is played in rounds. A round of the game starting in a configuration $((v, w), c)$ proceeds as follows:

1. *Spoiler* chooses to play from $(u_0, u_1) \in \{(v, w), (w, v)\}$;
2. the players move from (u_0, u_1) to (t_0, t_1) according to the rules in Table 1;
3. *Duplicator* selects a new configuration drawn from the following set:

$$\begin{aligned} \{\ & ((t_0, t_1), \checkmark), \\ & ((u_0, t_1), \gamma(c, (0, t_0), v, u_0)), \\ & ((t_0, u_1), \gamma(c, (1, t_1), w, u_1))\ \} \end{aligned}$$

where update γ is defined as follows:

$$\gamma(c, c', u, t) = \begin{cases} c' & \text{if } \textit{Spoiler} \text{ played on } t,\ u = t \text{ and } c \in \{\dagger, \checkmark, c'\} \\ \checkmark & \text{if } u \neq t, \text{ or} \\ & \quad \textit{Spoiler} \text{ played on } t \text{ and } c \notin \{\dagger, \checkmark, c'\} \\ \dagger & \text{otherwise} \end{cases}$$

An infinite play $((v_0, w_0), c_0), ((v_1, w_1), c_1), \ldots$ is won by *Duplicator* iff $\Omega(v_j) = \Omega(w_j)$ for all j and $c_k = \checkmark$ for infinitely many k. *Duplicator* wins the governed stuttering bisimulation game for a position (v, w) iff she has a strategy that wins all plays starting in configuration $((v, w), \checkmark)$.

We write $v \equiv_{g,st} w$ whenever *Duplicator* wins the governed stuttering bisimulation game for position (v, w).

Observe that in the governed stuttering game, *Duplicator* earns, as explained before, a $\checkmark$ reward whenever she continues playing in the position determined at the end of step 2. However, she also earns a $\checkmark$ whenever *Spoiler* decides to drop a pending challenge or, in step 1 of a round, switch positions. The example below illustrates some of the intricacies in the game play.

Example 5 Consider the parity game depicted in Fig. 5. In this parity game, all vertices with priority 0 are related by $\simeq$. The game illustrates why *Duplicator* gains a $\checkmark$ reward whenever *Spoiler* does not respect a pending challenge. This can be seen as follows: consider the game starting in $((v_1, v_3), \checkmark)$ and suppose *Spoiler* decides to play $v_1 \rightarrow v_2$. The only suitable response by *Duplicator* is to play $v_3 \rightarrow v_4$. New configurations $((v_2, v_4), \checkmark)$ and $((v_2, v_3, \checkmark))$ are not an option for *Duplicator* since he immediately loses due to the different priorities of v_2 and v_4 or v_3 respectively. The new configuration chosen by *Duplicator* will hence be $((v_1, v_3), (0, v_2))$, challenging *Spoiler* to play $v_1 \rightarrow v_2$ again in the next round. From this configuration, if *Spoiler* indeed plays $v_1 \rightarrow v_2$, *Duplicator* can match with $v_4 \rightarrow v_2$, and play stays in $((v_2, v_2), \checkmark)$ indefinitely, leading to a win from duplicator. Now, let us consider what happens if *Spoiler* plays $v_1 \rightarrow v_0$ instead. *Spoiler* did not respect the challenge, and *Duplicator* matches with $v_4 \rightarrow v_3$, and we end up in $((v_1, v_3), \checkmark)$ again. If *Duplicator* would not have earned a $\checkmark$ reward in this case, play would have ended up in $((v_1, v_3), (0, v_0))$ instead, and, if in the next round *Spoiler* again ignores the challenge, play can alternate indefinitely between $((v_1, v_3), (0, v_0))$ and $((v_1, v_4), (0, v_2))$, which would result in a win for *Spoiler*. This is undesirable since we already observed that v_1, v_3 and v_4 are governed stuttering bisimilar.

For the remainder of this section we turn our attention to relating the above game to governed stuttering bisimulation. Our next result states that whenever vertices v, w are governed stuttering bisimilar, *Duplicator* wins all plays starting in configuration $((v, w), \checkmark)$. We sketch the main ideas behind the proof; details can be found in the “Appendix”.

Proposition 7 *For all $v, w \in V$ if $v \simeq w$ then $v \equiv_{g,st} w$.*

Proof The proof proceeds by showing that *Duplicator* has a strategy that ensures (1) that plays allowed by this strategy move along configurations of the form $((u_0, u_1), c)$ for which $u_0 \simeq u_1$ and (2) *Duplicator* never gets stuck playing according to this strategy and 3) there is a strictly decreasing measure between two consecutive non-$\checkmark$ configurations on any play allowed by this strategy. Together, this implies that *Duplicator* has a winning strategy for configurations $((v, w), \checkmark)$. □

We next establish that vertices related through the governed stuttering bisimulation game are related by governed stuttering bisimulation. A straightforward proof thereof is hampered by the fact that any purported governed stuttering bisimulation relation is, by definition, required to be an equivalence relation. However, proving that the governed stuttering bisimulation game induces an equivalence relation is rather difficult. The strategy employed to prove the stated result is to use contraposition; this requires showing that for any given pair of non-governed stuttering bisimilar vertices we can construct a strategy that is winning for *Spoiler*. Note that we can do so because the governed stuttering bisimulation game has a Büchi winning condition, which implies the game is determined. This strategy is based on a fixpoint characterisation of governed stuttering bisimilarity, given below.

Definition 15 Let $R \subseteq V \times V$ be an equivalence relation on V. The predicate transformer $\mathcal{F} : V \times V \to V \times V$ is defined as follows:

$$\begin{aligned}\mathcal{F}(R) = \{(v, w) \in R \mid \Omega(v) = \Omega(w) \wedge \forall i \in \{\Diamond, \Box\}, \mathcal{U}, \mathcal{T} \subseteq V_{/R} :\\ [v]_R \in \mathcal{U} \wedge [v]_R \notin \mathcal{T} \implies v \;{}_i{\mapsto}_{\mathcal{U}}\, \mathcal{T} \Leftrightarrow w \;{}_i{\mapsto}_{\mathcal{U}}\, \mathcal{T}\}\end{aligned}$$

The predicate transformer $\mathcal{F}$ has the following properties, both of which can be proven straightforwardly.

Lemma 11 *$\mathcal{F}(R)$ is an equivalence relation for any equivalence relation R on V.*

Lemma 12 *$\mathcal{F}$ is a monotone operator on the complete lattice of equivalence relations on V.*

Using these properties, the following corollary now follows immediately.

Corollary 2 *We have $\simeq\!\!\!\cdot\, = \nu\mathcal{F}$, where ν is the greatest fixed point.*

Proof Follows from the fact that for $R = \nu\mathcal{F}$ and $\nu\mathcal{F} = \mathcal{F}(\nu\mathcal{F})$ the definition of $\mathcal{F}$ reduces to the definition of governed stuttering bisimulation. □

We finally state our completeness result. Again, we only outline the main steps of the proof; details can be found in the "Appendix".

Proposition 8 *For all $v, w \in V$ if $v \equiv_{g,st} w$ then $v \simeq\!\!\!\cdot\, w$.*

Proof We essentially prove the contrapositive of the statement, *i.e.* for all $v, w \in V$, if $v \not\simeq\!\!\!\cdot\, w$, then also $v \not\equiv_{g,st} w$. Let $v \not\simeq\!\!\!\cdot\, w$. By Corollary 2, then also $(v, w) \notin \nu\mathcal{F}$. By the Knaster-Tarski-Kleene fixpoint approximation theorem, we thus have $(v, w) \notin \bigcap_{k \geq 1} \mathcal{F}^k(V \times V)$. Using induction, one can prove, for $R^k = \bigcap_{l \leq k} R^l$, that for all $k \geq 1$:

$$\text{\textit{Spoiler} wins the governed stuttering bisimulation game for all configurations } ((u_0, u_1), c) \text{ for which} (u_0, u_1) \notin R^k \tag{IH}$$

For the inductive case, one can construct a strategy for *Spoiler* that guarantees he never gets stuck and for which every play allowed by the strategy either (1) visits some configuration $((t_0, t_1), c')$ for which the induction hypothesis applies, or (2) is such that there are only a finite number of ✓ rewards along the play. □

Propositions 7 and 8 lead to the following theorem.

Theorem 7 *For all $v, w \in V$ we have $v \simeq\!\!\!\cdot\, w$ iff $v \equiv_{g,st} w$.*

Stuttering Bisimulation. When we impose the additional constraint on governed stuttering bisimulation that we do not allow to relate vertices owned by different players, we obtain a notion called *stuttering bisimulation* [13]. The derived notion of *stuttering bisimilarity*, denoted $v \simeq w$ and defined as $v \simeq w$ iff there is some stuttering bisimulation relation that relates v and w, is an equivalence relation.

7 Quotienting

Simulation and bisimulation equivalences are often used to reduce the size of graphs by factoring out vertices that are equivalent, *i.e.* by computing *quotient structures*. This can be particularly interesting if computationally expensive algorithms must be run on the graph: whenever the analysis such algorithms perform on the graphs are insensitive to (bi)simulation equivalence, they can be run on the smaller quotient structures instead. In our setting, the same reasoning applies: typically, parity game solving is expensive and it may therefore pay off to first compute a quotient structure and only then solve the resulting quotient structure.

In this section, we show that most of the (bi)simulation relations we studied in the previous two sections have unique quotient structures. A fundamental property of quotienting is that the resulting quotient structure of a game should again be equivalent to the original game. This requires that we lift our equivalences to relations between two different game graphs. We do so in the standard way.

Definition 16 Let $\mathcal{G}_j = (V_j, \rightarrow_j, \Omega_j, \mathcal{P}_j)$, for $j = 1, 2$, be arbitrary parity games. We say that $\mathcal{G}_1 \sim \mathcal{G}_2$, for an equivalence relation $\sim$ defined on the vertices of a parity game, whenever in the disjoint union of $\mathcal{G}_1$ and $\mathcal{G}_2$, for all $v_1 \in V_1$ there is some $v_2 \in V_2$ such that $v_1 \sim v_2$ and for all $\bar{v}_2 \in V_2$ there is some $\bar{v}_1 \in V_1$ such that $\bar{v}_1 \sim \bar{v}_2$.

7.1 Simulation equivalence quotients

Quotienting for delayed simulation equivalence is, as observed in [22,23], problematic, and only the biased versions admit some form of quotienting. However, the quotients for biased delayed simulation equivalences are not unique, see also Lemma 3.5 in [22]. We here only consider quotienting for direct simulation equivalence; for the quotients of delayed simulation equivalence we refer to the aforementioned works.

The equivalence classes of direct simulation equivalence determine the set of vertices of the quotient structure. Defining the transition relation of the quotient structure is a bit more subtle. As observed in [9], a unique quotient structure of simulation equivalence for Kripke structures exists, but requires that vertices have no transitions to a pair of vertices, one of which is sometimes referred to as a *'little brother'* of the other one (a vertex that is simulated by, but not equivalent to the other vertex). That is, transitions to non-maximal vertices are omitted.

While in the setting of Kripke structures, only transitions to *maximal* successor vertices must be retained, depending on the owner of the source vertex, in our setting we need to consider maximal or *minimal* successor vertices.

Definition 17 Let $V' \subseteq V$ be an arbitrary non-empty set of vertices. An element v is:

- *minimal among* V' iff for all $u \in V'$ for which $u \leq_d v$, also $v \leq_d u$;
- *maximal among* V' iff for all $u \in V'$ for which $v \leq_d u$, also $u \leq_d v$.

For a given vertex v, a successor $v' \in v^{\bullet}$ is in the set $\min_{\leq_d}(v)$ iff v' is minimal among $v^{\bullet}$; likewise, $v' \in v^{\bullet}$ is in the set $\max_{\leq_d}(v)$ iff v' is maximal among $v^{\bullet}$.

Since $\leq_d$ is a preorder, $\min_{\leq_d}(v)$ and $\max_{\leq_d}(v)$ are non-empty sets.

An additional complication in defining a unique quotient structure is that a single equivalence class may contain vertices owned by *even* and vertices owned by *odd*. It turns out that the owner of such equivalence classes can be chosen arbitrarily: we prove that such classes have a unique successor equivalence class. For equivalence classes with exactly one successor, we can assign a unique owner; we choose to assign such classes to player *even*.

Definition 18 (*Direct simulation equivalence quotient*) The direct simulation equivalence quotient of $(V, \to, \Omega, \mathcal{P})$ is the structure $(V_{/\equiv_d}, \to', \Omega', \mathcal{P}')$, where, for $\mathcal{C}, \mathcal{C}' \in V_{/\equiv_d}$:

- $\Omega'(\mathcal{C}) = \min\{\Omega(v) \mid v \in \mathcal{C}\}$,
- $\mathcal{P}'(\mathcal{C}) = \begin{cases} \Box & \text{if } \mathcal{C} \subseteq V_\Box \text{ and for all } u \in \mathcal{C}, |[\min_{\leq_d}(u)]_{\equiv_d}| > 1 \\ \Diamond & \text{otherwise} \end{cases}$
- $\mathcal{C} \to' \mathcal{C}'$ iff $\begin{cases} \forall v \in \mathcal{C} : \exists v' \in \min_{\leq_d}(v) : v' \in \mathcal{C}' & \text{if } \mathcal{C} \subseteq V_\Box \\ \forall v \in \mathcal{C} \cap V_\Diamond : \exists v' \in \max_{\leq_d}(v) : v' \in \mathcal{C}' & \text{otherwise} \end{cases}$

Observe that it is not obvious that $\to'$ is a total edge relation. The lemma below allows us to establish that this is the case. It establishes that for an equivalence class consisting entirely of vertices owned by *odd*(respectively, *even*), the set of minimal (respectively, maximal) successors of all vertices in this class are the same. The vertices in equivalence classes that consist of both *even* and *odd* vertices the set of minimal successors of *odd* vertices is the same as the set of maximal successors of the *even* vertices.

Lemma 13 *Let* $\mathcal{C} \in V_{/\equiv_d}$. *Then:*

- *If* $\mathcal{C} \subseteq V_\Box$ *then* $[\min_{\leq_d}(v)]_{\equiv_d} = [\min_{\leq_d}(w)]_{\equiv_d}$ *for all* $v, w \in \mathcal{C}$,
- *If* $\mathcal{C} \subseteq V_\Diamond$ *then* $[\max_{\leq_d}(v)]_{\equiv_d} = [\max_{\leq_d}(w)]_{\equiv_d}$ *for all* $v, w \in \mathcal{C}$,
- *If* $\mathcal{C} \nsubseteq V_\Box$ *and* $\mathcal{C} \nsubseteq V_\Diamond$ *then for all* $v \in \mathcal{C} \cap V_\Diamond$ *and* $w \in \mathcal{C} \cap V_\Box$ *we have* $[\max_{\leq_d}(v)]_{\equiv_d} = [\min_{\leq_d}(w)]_{\equiv_d}$.

Proof We prove the first and the third statement; the proof for the second statement is analogous to that of the first.

- Suppose $\mathcal{C} \subseteq V_\Box$. Pick $v, w \in \mathcal{C}$. Let $v' \in \min_{\leq_d}(v)$. Since $v' \in v^\bullet$ and $w \leq_d v$, we have $w' \leq_d v'$ for some $w' \in w^\bullet$. This implies that there is some $w'' \in \min_{\leq_d}(w)$ such that $w'' \leq_d v'$; for, if $w' \notin \min_{\leq_d}(w)$, then there must be some $w'' \in \min_{\leq_d}(w)$ such that $w'' \leq_d w'$. But then also $w'' \leq_d v'$.
  We next show that also $v' \leq_d w''$. Since $v \leq_d w$ and $w'' \in w^\bullet$ we have $v'' \leq_d w''$ for some $v'' \in v^\bullet$. Since $w'' \leq_d v'$ and $v'' \leq_d w''$, we have $v'' \leq_d v'$. But since $v' \in \min_{\leq_d}(v)$, this implies $v' \equiv_d v''$. But from $v' \leq_d v''$ and $v'' \leq_d w''$ we obtain $v' \leq_d w''$. Hence, $v' \equiv_d w''$ for some $w'' \in \min_{\leq_d}(w)$.
- Suppose $\mathcal{P}(v) \neq \mathcal{P}(w)$ for some $v, w \in \mathcal{C}$. Pick $v, w \in \mathcal{C}$ such that $v \in V_\Diamond$ and $w \in V_\Box$. Since $w \leq_d v$, there must be $w' \in w^\bullet$ and $v' \in v^\bullet$ such that $w' \leq_d v'$. Fix such v' and w'. Since $v \leq_d w$ we find that for all $v'' \in v^\bullet$ and $w'' \in w^\bullet$ we have $v'' \leq_d w''$. In particular, $v' \leq_d w'$. So $v' \equiv_d w'$.
  Next, since for all $w'' \in w^\bullet$ we have $v' \leq_d w''$ and $v' \equiv_d w'$, we also have $w' \leq_d w''$ for all $w'' \in w^\bullet$. But this implies $w' \in \min_{\leq_d}(w)$, and, in particular, $|[\min_{\leq_d}(w)]_{\equiv_d}| = 1$. Likewise, we deduce $v' \in \max_{\leq_d}(v)$ and $|[\max_{\leq_d}(v)]_{\equiv_d}| = 1$. We thus find $[\max_{\leq_d}(v)]_{\equiv_d} = \{[v']_{\equiv_d}\} = \{[w']_{\equiv_d}\} = [\min_{\leq_d}(w)]_{\equiv_d}$. □

As a consequence of the above lemma, we obtain the following two results, essentially confirming that the direct simulation equivalence quotient is well-defined:

Corollary 3 *Let* $(V_{/\equiv_d}, \to', \Omega', \mathcal{P}')$ *be a direct simulation equivalence quotient of some parity game* $(V, \to, \Omega, \mathcal{P})$. *Then for all* $\mathcal{C}, \mathcal{C}' \in V_{/\equiv_d}$:

- *if* $\mathcal{C} \subseteq V_\Box$ *and for some* $v \in \mathcal{C}$, $v' \in \mathcal{C}'$ *also* $v' \in \min_{\leq_d}(v)$, *then* $\mathcal{C} \to' \mathcal{C}'$.
- *if* $\mathcal{C} \cap V_\Diamond \neq \emptyset$ *and for some* $v \in \mathcal{C} \cap V_\Diamond$, $v' \in \mathcal{C}'$ *also* $v' \in \max_{\leq_d}(v)$, *then* $\mathcal{C} \to' \mathcal{C}'$.

Corollary 4 *The direct simulation reduced quotient structure associated to a parity game* $(V, \to \Omega, \mathcal{P})$ *is again a parity game.*

We next establish that the direct simulation quotient of a parity game is equivalent to the original parity game.

Proposition 9 *Let* $\mathcal{G} = (V, \to, \Omega, \mathcal{P})$ *be a parity game and* $\mathcal{G}_q = (V_{/\equiv_d}, \to', \Omega', \mathcal{P}')$ *its direct simulation quotient. Then* $\mathcal{G}_q \equiv_d \mathcal{G}$*.*

Proof $\mathcal{G}_q \leq_d \mathcal{G}$, follows from the observation that relation $H \subseteq V_{/\equiv_d} \times V$, defined as $H = \{(\mathcal{C}, v) \mid \exists w \in \mathcal{C} : w \leq_d v\}$, is a direct simulation relation. Similarly, $\mathcal{G} \leq_d \mathcal{G}_q$, follows from the fact that relation $H \subseteq V \times V_{/\equiv_d}$, given by $H = \{(v, \mathcal{C}) \mid \exists w \in \mathcal{C} : v \leq_d w\}$, is a direct simulation relation. □

From the above result it follows that quotienting is 'safe' in the sense that one can solve the quotient game and still extract the solution of the original parity game. It is not hard to see that the size of a quotient is at most as large as the original game. From a computational point of view it thus makes sense to solve the quotient game instead of the original game. We finally establish that the quotient is unique. Combined with the fact that the quotient game is at most as large as the original game we find that each parity game has a *unique, smallest quotient*. This essentially confirms that by quotienting we achieve a maximal reduction.

Theorem 8 *Let* $\mathcal{G}, \mathcal{G}'$ *be two parity games and let* $\mathcal{G}_q$ *and* $\mathcal{G}'_q$ *be their direct simulation equivalence quotients, respectively. Then* $\mathcal{G} \equiv_d \mathcal{G}'$ *iff the two structures* $\mathcal{G}_q = (V_{/\equiv_d}, \to_q, \Omega_q, \mathcal{P}_q)$ *and* $\mathcal{G}'_q = (V'_{/\equiv_d}, \to'_q, \Omega'_q, \mathcal{P}'_q)$ *are isomorphic.*

Proof The proof that isomorphism of $\mathcal{G}_q$ and $\mathcal{G}'_q$ implies $\mathcal{G} \equiv_d \mathcal{G}'$ follows essentially from Proposition 9 and that isomorphic structures are also direct simulation equivalent.

The proof that $\mathcal{G} \equiv_d \mathcal{G}'$ implies that $\mathcal{G}_q$ and $\mathcal{G}'_q$ are isomorphic structures follows the following steps. Assume that $\mathcal{G} \equiv_d \mathcal{G}'$. Let $f \subseteq V'_{/\equiv_d} \times V_{/\equiv_d}$ be defined as $(\mathcal{C}', \mathcal{C}) \in f$ iff $\mathcal{C}' \equiv_d \mathcal{C}$. Note that for all $(\mathcal{C}', \mathcal{C}) \in f$ we have $\Omega'_q(\mathcal{C}') = \Omega_q(\mathcal{C})$.

We first show that f is a total bijective function from $V'_{/\equiv_d}$ to $V_{/\equiv_d}$. For injectivity and functionality of f we reason as follows. Suppose f is not functional. Then there is some $v' \in V'$ and two $v, \bar{v} \in V$ such that $[v]_{\equiv_d} \neq [\bar{v}]_{\equiv_d}$, $([v']_{\equiv_d}, [v]_{\equiv_d}) \in f$ and $([v']_{\equiv_d}, [\bar{v}]_{\equiv_d}) \in f$. Then by definition, $v' \equiv_d v$ and $v' \equiv_d \bar{v}$. But then also $v \equiv_d \bar{v}$, contradicting that $[v]_{\equiv_d} \neq [\bar{v}]_{\equiv_d}$. So f is functional. The proof that f^{-1} is a function from $V_{/\equiv_d}$ to $V'_{/\equiv_d}$ is similar. We may therefore conclude that f is an injective function.

For surjectivity of f, we observe that by definition of $\mathcal{G} \equiv_d \mathcal{G}'$, for each $v \in V$ there is some $v' \in V'$ such that $v \equiv_d v'$. Hence, for each $[v]_{\equiv_d} \in V_{/\equiv_d}$ there is some $[v]_{\equiv_d} \in V'_{/\equiv_d}$ such that $([v']_{\equiv_d}, [v]_{\equiv_d}) \in f$. Similarly, we can show that f^{-1} is surjective and therefore f is total bijection.

We next prove that $\mathcal{P}'_q(\mathcal{C}) = \mathcal{P}_q(f(\mathcal{C}))$. Towards the contrary, assume that $\mathcal{P}'_q(\mathcal{C}) = \Box$ whereas $\mathcal{P}_q(f(\mathcal{C})) = \Diamond$ for some $\mathcal{C}$. Then $\mathcal{C} \subseteq V'_{\Box}$ and for all $v \in \mathcal{C}$ we have $|\min_{\leq_d}(v)| > 1$, and there is some $w \in f(\mathcal{C})$ satisfying either $w \in V_{\Diamond}$, or $|\min_{\leq_d}(w)| = 1$. Let $w \in f(\mathcal{C})$ be such and pick an arbitrary $v \in \mathcal{C}$. We distinguish two cases.

- Case $w \in V_{\Diamond}$. Since $w \equiv_d f(\mathcal{C}) \equiv_d \mathcal{C} \equiv_d v$ we have $w \leq_d v$ in particular. Pick an arbitrary $w' \in w^{\bullet}$. Then, since $\mathcal{P}(v) = \Box$, we have $w' \leq_d v'$ for all $v' \in v^{\bullet}$; more specifically, we have $w' \leq_d v'_1$ and $w' \leq_d v'_2$ for $v'_1, v'_2 \in \min_{\leq_d}(v)$ such that $v'_1 \not\equiv_d v'_2$. Since v'_1, v'_2 are minimal elements, we thus also have $v'_1 \leq_d w'$ and $v'_2 \leq_d w'$ and hence $v'_1 \equiv_d w'$ and $v'_2 \equiv_d w'$. But from this we obtain $v'_1 \equiv_d v'_2$. Contradiction.

- Case $|\min_{\leq_d}(w)| = 1$. Without loss of generality we may assume that $w \in V_\Box$. Since $w \equiv_d f(\mathcal{C}) \equiv_d \mathcal{C} \equiv_d v$ we also have $w \leq_d v$. Let $v'_1, v'_2 \in \min_{\leq_d}(v)$ be such that $v'_1 \not\equiv_d v'_2$. Then there must be some $w'_1, w'_2 \in w^\bullet$ such that $w'_1 \leq_d v'_1$ and $w'_2 \leq_d v'_2$. Let w'_1, w'_2 be such; without loss of generality, we may assume that w'_1 and w'_2 are minimal. Since v'_1, v'_2 are minimal, we find that $v'_1 \leq_d w'_1$ and $v'_2 \leq_d w'_2$ and hence $v'_1 \equiv_d w'_1$ and $v'_2 \equiv_d w'_2$. But because $v'_1 \not\equiv_d v'_2$ we have $w'_1 \not\equiv_d w'_2$. Since w'_1 and w'_2 are minimal we have $|[\min_{\leq_d}(w)]_{\equiv_d}| \geq 2$. Contradiction.

Hence, $\mathcal{P}'_q(\mathcal{C}) = \mathcal{P}_q(f(\mathcal{C}))$.

Finally, we prove that $\mathcal{C} \to'_q \mathcal{C}'$ iff $f(\mathcal{C}) \to_q f(\mathcal{C}')$. Suppose $\mathcal{C} \to'_q \mathcal{C}'$ but not $f(\mathcal{C}) \to_q f(\mathcal{C}')$. Assume $\mathcal{P}'_q(\mathcal{C}) = \mathcal{P}_q(f(\mathcal{C})) = \Diamond$. The case where $\mathcal{P}'_q(\mathcal{C}) = \mathcal{P}_q(f(\mathcal{C})) = \Box$ is similar. Since $\mathcal{C} \equiv_d f(\mathcal{C})$, there must be some $\mathcal{D}$ such that $f(\mathcal{C}) \to_q \mathcal{D}$ and $\mathcal{C}' \leq_d \mathcal{D}$. But then also $\mathcal{C} \to'_q \mathcal{C}''$ and $\mathcal{D} \leq_d \mathcal{C}''$ for some $\mathcal{C}''$. Then $\mathcal{C}' \leq_d \mathcal{C}''$. Distinguish two cases:

- Case $\mathcal{C}' = \mathcal{C}''$. Then $f(\mathcal{C}') = f(\mathcal{C}'') = \mathcal{D}$, contradicting our assumption that $f(\mathcal{C}) \not\to_q f(\mathcal{C}')$.
- Case $\mathcal{C}' \neq \mathcal{C}''$. Then we have $\mathcal{C} \to'_q \mathcal{C}'$ and $\mathcal{C} \to'_q \mathcal{C}''$ and $\mathcal{C}' \leq_d \mathcal{C}''$. But this means that vertices in $\mathcal{C}'$ are not maximal. Hence, $\mathcal{G}'_q$ does not have a transition $\mathcal{C} \to'_q \mathcal{C}'$. □

Corollary 5 *The direct simulation equivalence quotient of $\mathcal{G}$ is a unique (up-to isomorphism) smallest parity game direct simulation equivalent to $\mathcal{G}$.*

7.2 Governed bisimulation and governed stuttering bisimulation quotients

We first define the governed bisimulation quotient. It is essentially a simplification of the direct simulation equivalence quotient.

Definition 19 (*Governed bisimulation quotient*) The governed bisimulation quotient of $(V, \to, \Omega, \mathcal{P})$ is the structure $(V_{/\underline{\leftrightarrow}}, \to', \Omega', \mathcal{P}')$, where, for $\mathcal{C}, \mathcal{C}' \in V_{/\underline{\leftrightarrow}}$:

- $\Omega'(\mathcal{C}) = \min\{\Omega(v) \mid v \in \mathcal{C}\}$,
- $\mathcal{P}'(\mathcal{C}) = \begin{cases} \Box & \text{if } \mathcal{C} \subseteq V_\Box \text{ and for all } u \in \mathcal{C}, |[u^\bullet]_{\underline{\leftrightarrow}}| > 1 \\ \Diamond & \text{otherwise} \end{cases}$
- $\mathcal{C} \to' \mathcal{C}'$ if and only if $\forall v \in \mathcal{C} : \exists v' \in v^\bullet : v' \in \mathcal{C}'$

We next state some elementary results concerning the governed bisimulation quotient. Since the proofs of these results follow the corresponding ones we presented in the previous section, we omit their details.

Proposition 10 *Let $\mathcal{G} = (V, \to, \Omega, \mathcal{P})$ be a parity game and $\mathcal{G}_q = (V_{/\underline{\leftrightarrow}}, \to', \Omega', \mathcal{P}')$ be its governed bisimulation quotient. Then $\mathcal{G} \underline{\leftrightarrow} \mathcal{G}_q$.*

Proof Follows from the fact that the relation $R = \{(v, \mathcal{C}), (\mathcal{C}, v) \mid v \in \mathcal{C}\}$, is a governed bisimulation relation. □

Theorem 9 *Let $\mathcal{G}, \mathcal{G}'$ be two parity games and let $\mathcal{G}_q$ and $\mathcal{G}'_q$ be their direct simulation equivalence quotients, respectively. Then $\mathcal{G} \underline{\leftrightarrow} \mathcal{G}'$ iff the two structures $\mathcal{G}_q = (V_{/\underline{\leftrightarrow}}, \to_q, \Omega_q, \mathcal{P}_q)$ and $\mathcal{G}'_q = (V'_{/\underline{\leftrightarrow}}, \to'_q, \Omega'_q, \mathcal{P}'_q)$ are isomorphic.*

Corollary 6 *The governed bisimulation quotient of $\mathcal{G}$ is a unique (up-to isomorphism) smallest parity game that is governed bisimilar to $\mathcal{G}$.*

As a consequence of the above results, we find that each parity game has a unique and minimal governed bisimilar parity game.

We next define the governed stuttering bisimulation quotient. It requires some subtlety to properly deal with divergences and ensure that a unique player is assigned to an equivalence class. More specifically, we cannot simply assign the player of an arbitrary vertex in an equivalence class to this equivalence class; instead, the player assigned to an equivalence class must be the one that has the ability to force play from that equivalence class.

Definition 20 (*Governed stuttering bisimulation quotient*)
The governed stuttering bisimulation quotient of $(V, \rightarrow, \Omega, \mathcal{P})$ is the structure $(V_{/\simeq}, \rightarrow', \Omega', \mathcal{P}')$, where, for $\mathcal{C}, \mathcal{C}' \in V_{/\simeq}$:

- $\Omega'(\mathcal{C}) = \min\{\Omega(v) \mid v \in \mathcal{C}\}$,
- $\mathcal{P}'(\mathcal{C}) = \begin{cases} \Diamond & \text{if for all } v \in \mathcal{C},\ v\ {}_{\Diamond}{\mapsto}_{\simeq}, \text{ or} \\ & \quad \text{for some } v \in \mathcal{C}, \mathcal{C}' \neq \mathcal{C},\ v\ {}_{\Diamond}{\rightarrow}\ \mathcal{C}' \\ \Box & \text{otherwise} \end{cases}$
- $\mathcal{C} \rightarrow' \mathcal{C}'$ if and only if

$$\begin{cases} \exists i \in \{\Diamond, \Box\} : \forall v \in \mathcal{C} : v\ {}_{i}{\mapsto}_{\simeq} & \text{if } \mathcal{C} = \mathcal{C}' \\ \exists i \in \{\Diamond, \Box\} : \forall v \in \mathcal{C} : v\ {}_{i}{\mapsto}_{v\simeq}\ \mathcal{C}' & \text{if } \mathcal{C} \neq \mathcal{C}' \end{cases}$$

The results below essentially mirror those results we presented in the previous section for the direct simulation equivalence quotient. Details of the proofs are once again omitted since they follow the exact same line of reasoning as those from the previous section.

Proposition 11 *Let $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$ be a parity game and $\mathcal{G}_q = (V_{/\simeq}, \rightarrow', \Omega', \mathcal{P}')$ be its governed stuttering bisimulation quotient. Then $\mathcal{G} \simeq \mathcal{G}_q$.*

Proof Follows from the fact that relation $R \subseteq (V \cup V_{/\simeq}) \times (V \cup V_{/\simeq})$, defined as $R = \{(v, \mathcal{C}), (\mathcal{C}, v), (v, w), (\mathcal{C}, \mathcal{C}) \mid v, w \in \mathcal{C}\}$, is a governed stuttering bisimulation. □

Theorem 10 *Let $\mathcal{G}, \mathcal{G}'$ be two parity games and let $\mathcal{G}_q$ and $\mathcal{G}'_q$ be their direct simulation equivalence quotients, respectively. Then $\mathcal{G} \simeq \mathcal{G}'$ iff the two structures $\mathcal{G}_q = (V_{/\simeq}, \rightarrow_q, \Omega_q, \mathcal{P}_q)$ and $\mathcal{G}'_q = (V'_{/\simeq}, \rightarrow'_q, \Omega'_q, \mathcal{P}'_q)$ are isomorphic.*

Corollary 7 *The governed stuttering bisimulation quotient of $\mathcal{G}$ is a unique (up-to isomorphism) smallest parity game that is governed stuttering bisimilar to $\mathcal{G}$.*

8 A comparison of discriminating power

In this section, we compare the discriminative power of each of the equivalences discussed in the preceding sections, essentially justifying the lattice we illustrated in Sect. 4. This permits us to assess the reductive power of each of the studied equivalences that admit (unique) quotienting. For each of the equivalences we show which other equivalences it strictly refines. Incomparability results are described separately.

We first focus on proving the right-hand side of the lattice we presented in Sect. 4. That is, we first compare isomorphism, strong bisimilarity and governed bisimilarity and then focus on the various simulation equivalences.

Theorem 11 *Isomorphism is strictly finer than strong bisimilarity.*

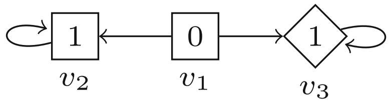
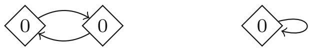


Fig. 6 Parity game which is minimal with respect to strong bisimilarity. Vertices v_2 and v_3 are governed bisimilar

Proof Clearly, every pair of isomorphic parity games is a pair of strong bisimilar parity games. Strictness follows from a standard example:



Clearly, both vertices in the left parity game are strongly bisimilar to the vertex in the right parity game, and *vice versa*. However, these vertices are not isomorphic. □

The following theorem relates strong bisimilarity to stuttering equivalence, governed bisimulation and strong direct simulation equivalence, and except for the comparison to governed bisimulation it is essentially the counterpart of the classical theorems in the setting of Kripke structures.

Theorem 12 *Strong bisimilarity is strictly finer than strong direct simulation equivalence, stuttering bisimulation equivalence and governed bisimulation equivalence.*

Proof We sketch each of the refinements:

– Every strong bisimulation relation is a direct simulation relation. Since such a relation is symmetric, every pair of parity games related via strong bisimilarity is also related via strong direct simulation equivalence. Strictness follows from the parity game in Fig. 4, in which v_0 and v_1 are strong direct simulation equivalent but not strongly bisimilar.
– Every strong bisimulation relation is a stuttering bisimulation relation, this follows directly from the definitions. Strictness follows from the parity game in Fig. 5, in which v_3 and v_4 are stuttering bisimilar, but not strongly bisimilar.
– Every strong bisimulation relation is a governed bisimulation relation, this follows directly from the definitions. Strictness follows from the parity game in Fig. 6, which is minimal modulo strong bisimilarity, but vertices v_2 and v_3 are governed bisimilar. □

We next state, without proof, a result that essentially follows by definition.

Theorem 13 *Strong direct simulation equivalence strictly refines direct simulation equivalence.*

The following refinement results follow a line of reasoning similar to the ones seen before.

Theorem 14 *Governed bisimulation equivalence strictly refines direct simulation equivalence and governed stuttering bisimulation equivalence.*

Proof We again sketch both refinements.

– Refinement follows directly from the observation that a governed bisimulation is a symmetric direct simulation. Strictness follows from examples similar to those discriminating strong bisimilarity and simulation equivalence.
– It follows from the definitions that every governed bisimulation is also a governed stuttering bisimulation. The strictness of the refinement follows from the example in Fig. 5 in which all vertices with priority 0 are governed stuttering bisimilar, but none are governed bisimilar. □

Theorem 15 *Governed bisimulation equivalence and strong direct simulation equivalence are incomparable.*

Proof This follows from the parity game in Fig. 4 in which vertices v_0 and v_6 are governed bisimilar, but not strong direct simulation equivalent. Furthermore, v_0 and v_1 are strong direct simulation equivalent, but not governed bisimilar. □

Regarding direct simulation and (biased) delayed simulations we have the following result due to Fritz and Wilke [23].

Theorem 16 *[23] Direct simulation equivalence is strictly finer than even- and odd-biased delayed simulation equivalence. Even- and odd-biased delayed simulation are incomparable, and both are strictly finer than delayed simulation equivalence. Delayed simulation equivalence in turn is strictly finer than winner equivalence.*

This completes the results underlying the right-hand side of the lattice we presented in Sect. 4.

We next focus on the left-hand side of the lattice.

Theorem 17 *Stuttering bisimilarity is incomparable to governed bisimilarity, strong direct simulation, direct simulation and all delayed simulation variations.*

Proof In the parity game in Fig. 5, v_3 and v_4 are stuttering bisimilar but they cannot be related under governed bisimilarity, (strong) direct simulation equivalence, nor any of the delayed simulation equivalences. For the other direction, consider vertices v_0, v_1 and v_6 from the parity game in Fig. 4. None of these vertices are stuttering bisimilar, whereas v_0 and v_6 are governed bisimilar, v_0 and v_1 are strong direct simulation equivalent, and all three are direct simulation equivalent, and therefore also delayed simulation equivalent. □

Theorem 18 *Governed stuttering bisimilarity is incomparable to direct simulation, strong direct simulation, and all delayed simulation variations.*

Proof Follows from the same examples as used in the proof of Theorem 17. □

Theorem 19 *Stuttering bisimilarity strictly refines governed stuttering bisimilarity.*

Proof It follows from the definitions that every stuttering bisimulation is also a governed stuttering bisimulation. Strictness follows from the parity game in Fig. 5 in which v_0 and v_1 are governed stuttering bisimilar, but not stuttering bisimilar. □

To complete the lattice, we next show that governed stuttering bisimilarity is strictly finer than winner equivalence. In order to prove this result, we must first lift the concept of governed stuttering bisimilarity to paths. Being able to reason about paths then allows us to show that every memoryless strategy of player *odd* (respectively, *even*) in the original game induces a strategy of player *odd* (respectively, *even*) in the quotient game such that all plays in the quotient game, allowed by this strategy, have a 'matching' play allowed by the strategy in the original game. If we then consider a strategy that is winning for a player in the original game, it follows that this strategy is also winning for this player in the quotient game, and *vice versa*.

We first lift governed bisimilarity from vertices to paths. Paths of length 1 are equivalent if the vertices they consist of are equivalent. If paths p and q are equivalent, then $pv \simeq q$ iff v is equivalent to the last vertex in q, and $pv \simeq qw$ iff $v \simeq w$. An infinite path p is equivalent to a path q if for all finite prefixes of p there is an equivalent prefix of q and *vice versa*.

Lemma 14 *Let $(V, \to, \mathcal{P}, \Omega)$ be a parity game, and let $(V_{/\simeq}, \to', \mathcal{P}', \Omega')$ be its quotient. Let $v \in V$, and $\mathcal{C} \in V_{/\simeq}$ such that $v \in \mathcal{C}$. For all players i, and all $\sigma \in \mathbb{S}_i$ there is some $\psi \in \mathbb{S}_i^*$ such that for all $q \in \Pi_\psi^\omega(\mathcal{C})$ there is a $p \in \Pi_\sigma^\omega(v)$ such that $p \simeq q$.*

Proof Let $\sigma \in \mathbb{S}_i$ be an arbitrary strategy for player i. Before we construct a strategy $\psi \in \mathbb{S}_i^*$ for player i in the quotient game, we introduce some auxiliary functions which help defining this strategy. Define an arbitrary complete ordering $\lessdot$ on vertices, and define the following for finite paths q starting in $\mathcal{C}$, where $\min_\lessdot \emptyset$ is defined to be $\bot$:

$$\mathsf{next}(q) = \min_{\lessdot}\{v' \in V \mid \exists p \in \Pi_\sigma(v) : p \simeq q \wedge p \,{}_\sigma\!\!\to v' \wedge pv' \not\simeq q\}$$

$$\mathsf{div}(q) = \exists p \in \Pi_\sigma^\omega(v) : p \simeq q$$

We next show that it is possible to define a strategy $\psi \in \mathbb{S}_i^*$ for finite plays $q = \mathcal{C} \dots \mathcal{C}'$ such that if $q \simeq p$ for some $p \in \Pi_\sigma(v)$, then:

$$\begin{cases} \psi(q) = \mathcal{C}' & \text{if } \mathsf{div}(q) \text{ and } \mathcal{C}' \to' \mathcal{C}' \\ \psi(q) = [\mathsf{next}(q)]_\simeq & \text{otherwise.} \end{cases}$$

Let $p \in \Pi_\sigma(v)$ be such that $q \simeq p$ for $q = \mathcal{C} \dots \mathcal{C}'$ and assume $\mathcal{P}'(\mathcal{C}') = i$. In case $\mathsf{div}(q)$ and $\mathcal{C}' \to' \mathcal{C}'$, then obviously $\psi(q)$ can be defined as $\mathcal{C}'$. We proceed to show that if $\neg\mathsf{div}(q)$ or $\mathcal{C}' \not\to' \mathcal{C}'$, then 1) $\mathsf{next}(q) \neq \bot$, and 2) we can set $\psi(q) = [\mathsf{next}(q)]_\simeq$. We show the first by distinguishing two cases:

Case $\neg\mathsf{div}(q)$. It follows straightforwardly that $\mathsf{next}(q) \neq \bot$.

Case $\mathcal{C}' \not\to' \mathcal{C}'$. Because $\mathcal{C}'$ is a vertex is a quotient graph, $\mathcal{C}' \,{}_i\!\not\mapsto_\simeq$. Consider the path $p \in \Pi_\sigma(v)$ for which $p \simeq q$, and assume that p is of the form $\bar{p}u$. Since $\bar{p}u \simeq q$, also $u \simeq \mathcal{C}'$ and hence $u \,{}_i\!\not\mapsto_\simeq$. Then by Lemma 3, $u \,{}_{\neg i}\!\mapsto_{u\simeq} V \setminus [u]_\simeq$. Let $\sigma' \in \mathbb{S}_{\neg i}$ be such that $u \,{}_{\sigma'}\!\mapsto_{u\simeq} V \setminus [u]_\simeq$ and consider the unique path $rv' \in \Pi_{\sigma'}(u)$ such that $\sigma \Vdash rv'$, $\sigma' \Vdash rv'$, $r \simeq u$ and $v' \in V \setminus [u]_\simeq$. Then $pr \in \Pi_\sigma(v)$ is such that $pr \simeq q$, $pr \,{}_\sigma\!\!\to v'$ and $prv' \not\simeq q$. Hence, $\mathsf{next}(q) \neq \bot$.

Next, we show that we can set $\psi(q) = [\mathsf{next}(q)]_\simeq$. Since $\mathsf{next}(q) \neq \bot$, there must be some $v \dots v'v'' \in \Pi_\sigma(v)$ such that $v'' = \mathsf{next}(q)$, $v \dots v' \simeq q$ and $v' \not\simeq v''$. Also, $\mathcal{C}' \,{}_{\mathcal{P}(v')}\!\mapsto_{v'\simeq} [v'']_\simeq$, since $v' \simeq \mathcal{C}'$ and $v' \to [v'']_\simeq$. As $\mathcal{C}'$ is a vertex in a quotient graph, this implies $\mathcal{C}' \,{}_{\mathcal{P}(v')}\!\!\to [v'']_\simeq$. Hence, we can set $\psi(q) = [\mathsf{next}(q)]_\simeq$.

Now that we have shown that it is always possible to define a strategy adhering to the restrictions above, let ψ be such a strategy. We show using induction on n that for all n

$$\forall q \in \Pi_\psi^n(\mathcal{C}) : \exists p \in \Pi_\sigma(v) : p \simeq q.$$

For $n = 0$, this is trivial, because $v \simeq \mathcal{C}$. For $n = m + 1$, assume as the induction hypothesis that $\forall \bar{q} \in \Pi_\psi^m(\mathcal{C}) : \exists \bar{p} \in \Pi_\sigma(v) : \bar{p} \simeq \bar{q}$. Let $q \in \Pi_\psi^n(\mathcal{C})$ and let $\mathcal{C}', \mathcal{C}'' \in V_{/\simeq}$ and $\bar{q} \in \Pi_\psi^m(\mathcal{C})$ such that $\bar{q} = \mathcal{C} \dots \mathcal{C}'$ and $q = \bar{q}\mathcal{C}''$. Distinguish cases on the player who owns $\mathcal{C}'$.

Case $\mathcal{P}'(\mathcal{C}') \neq i$. Then $\mathcal{C}'' = \psi(\bar{q})$. The induction hypothesis yields some $\bar{p} \in \Pi_\sigma(v)$ such that $\bar{p} \simeq \bar{q}$, therefore $\mathcal{C}'' = \mathcal{C}'$ if $\mathsf{div}(\bar{q})$ and $\mathcal{C}' \to' \mathcal{C}'$, and otherwise $\mathcal{C}'' = [\mathsf{next}(\bar{q})]_\simeq$. If $\mathcal{C}' = \mathcal{C}''$, then $\mathsf{div}(\bar{q})$, so there must be some $p \in \Pi_\sigma^\omega(v)$ such that $p \simeq q$ and therefore also some $p \in \Pi_\sigma(v)$ such that $p \simeq q$. If $\mathcal{C}'' = [\mathsf{next}(\bar{q})]_\simeq$, there must be some $p \in \Pi_\sigma(v)$ such that $p = p'v'$ and $p' \simeq \bar{q}$ and $v' \simeq \mathcal{C}''$. By definition, $p \simeq q$ for such p.

Case $\mathcal{P}'(\mathcal{C}') \neq i$. From the induction hypothesis, obtain a $\bar{p} \in \Pi_\sigma(v)$ such that $\bar{p} \simeq \bar{q}$.

Without loss of generality we may assume that $\bar{p}$ is finite. Note that $\mathcal{C}' \to [\mathcal{C}'']_\simeq$. We distinguish two cases.

- Case $\mathcal{C}' = \mathcal{C}''$. Then we have $\bar{p} \simeq \bar{q}\mathcal{C}''$.
- Case $\mathcal{C}' \neq \mathcal{C}''$. Let v' be the last vertex in $\bar{p}$. Because $\bar{p} \simeq \bar{q}$, also $v' \,{}_{\neg i}\!\mapsto_{v'\simeq}[\mathcal{C}'']_\simeq$. So let $\sigma' \in \mathbb{S}_{\neg i}$ be such that $v' \,{}_{\sigma'}\!\mapsto_{v'\simeq}[\mathcal{C}'']_\simeq$. Now consider an infinite path $\bar{p}p$ such that $\sigma \Vdash \bar{p}p$ and $\sigma' \Vdash \bar{p}p$. For some index $k \geq 0$, it must be the case that $p_k \simeq \mathcal{C}''$ and $p_l \simeq \mathcal{C}'$ for all $l < k$. So $\bar{p}p_0 \ldots p_k \simeq q$.

Finally, we prove that for all $q \in \Pi^\omega_\psi(\mathcal{C})$ there is a $p \in \Pi^\omega_\sigma(v)$ such that $p \simeq q$. Let $q \in \Pi^\omega_\psi(\mathcal{C})$. Then by the above, we find that there is some $p \in \Pi_\sigma(v)$. Suppose p is finite and $p = \bar{p}v'$ for some vertex v'. Since q is a path through the quotient graph, q must be of the form $\bar{q}\mathcal{C}^\omega$ for some $\mathcal{C} \simeq v'$.

- Case $\mathcal{P}'(\mathcal{C}) = i$. Then $\psi(\bar{q}\mathcal{C}) = \mathcal{C}$, and thus $\mathsf{div}(\bar{q}\mathcal{C})$ by definition of ψ. But then there must be some $p' \in \Pi^\omega_\sigma(v)$ such that $p' \simeq \bar{q}\,\mathcal{C} \simeq q$.
- Case $\mathcal{P}'(\mathcal{C}) \neq i$. Since $\mathcal{C} \to' \mathcal{C}$ we have $\mathcal{C} \,{}_{\neg i}\!\mapsto_\simeq$ and since $v' \simeq \mathcal{C}$, also $v' \,{}_{\neg i}\!\mapsto_\simeq$. Let $\sigma' \in \mathbb{S}_{\neg i}$ be such that $v' \,{}_{\neg i}\!\mapsto_\simeq$. Then there is an infinite path $p' \in \Pi^\omega_{\sigma'}(v')$ such that $\sigma \Vdash p'$ and $p' \simeq v'$. But then $\sigma \Vdash \bar{p}p'$ and $q \simeq \bar{p}p'$. □

Finally this allows us to prove the result that governed stuttering bisimilar vertices are won by the same player.

Theorem 20 *Governed stuttering bisimilarity strictly refines winner equivalence.*

Proof Let $\mathcal{G} = (V, \to, \mathcal{P}, \Omega)$ be a parity game, and let $v, w \in V$ such that $v \simeq w$. Let $(V_{/\simeq}, \to', \mathcal{P}', \Omega')$ be the governed stuttering bisimulation quotient of $\mathcal{G}$, and let $\mathcal{C} \in V_{/\simeq}$ be such that $w \simeq \mathcal{C}$. By transitivity of $\simeq$, also $v \simeq \mathcal{C}$. Now suppose that player i has a winning strategy σ from v. Then by Lemma 14, i has a strategy ψ from $\mathcal{C}$ such that for every play $q \in \Pi_\psi(\mathcal{C})$ there is a play $p \in \Pi_\sigma(v)$ such that $p \simeq q$. Because the priorities occurring infinitely often on such p and q are the same, ψ is also winning for i. If $\neg i$ had a winning strategy σ' from w, then we could repeat this argument to construct a winning strategy for $\neg i$ from $\mathcal{C}$, but this would be contrary to the fact that parity games are determined. Therefore, w must also be won by player i. □

Together with all results in this section (essentially comparing the (bi)simulations discussed in this paper) that precede this final result, we find that each of the equivalences described in this paper strictly refines winner equivalence, as also reflected by the lattice we illustrated in Sect. 4. Therefore, any pair of vertices, related by any of the equivalences described, is won by the same player.

9 Conclusion

Preorders and equivalences for parity games have been studied on a number of occasions, see [11,13,14,22,23,34,39]. A major motivation for some of these is that they provide the prospect of simplifying games prior to solving them. In this paper, we reconsidered several of the parity game relations previously defined by us, *viz.* (governed) bisimulation and (governed) stuttering bisimulation. More specifically, we gave detailed proofs showing that our relations are equivalences, they have unique parity game quotients and they approximate the winning regions of parity games. Furthermore, we showed that our coinductively defined equivalence relations admit game-based definitions; the latter facilitated the comparison of our equivalences to the game-based definitions of relations for parity games found in the literature. For the latter relations, we additionally gave coinductive definitions. Finally, we showed that, unlike *e.g.* delayed simulation or any of its biased versions, our equivalence relations give rise to unique quotients.

There are several natural continuations of this research. First, the experiments that were conducted in [14,39] showed that parity games that could not be solved become solvable by preprocessing the games using an $\mathcal{O}(mn)$ stuttering bisimulation minimisation algorithm or an $\mathcal{O}(mn^2)$ governed stuttering bisimulation minimisation algorithm. While the average reduction achieved by governed stuttering bisimulation of parity games encoding typical practical decision problems exceeded 80%, see [39], the overall gain in speed otherwise was not significant. It would be worthwhile to establish whether this is still true when using the $\mathcal{O}(m \log n)$ stuttering equivalence minimisation algorithm of [31]. Moreover, it would be interesting to see whether the $\mathcal{O}(mn^2)$ time complexity of governed stuttering bisimulation can be reduced using ideas from [31]. Similarly, we believe that our coinductive rephrasing of delayed simulation will help to devise a more efficient algorithm for computing it, using a partition refinement approach.

Finally, an interesting line of investigation is to see whether the incomparable notions of governed stuttering bisimulation and delayed simulation equivalence can be married. Given that we have established game-based and coinductive definitions for both relations, defining such a relation now seems within reach. The resulting relation would be closer to winning equivalence and perhaps even shed light on ways to efficiently solve parity games in general.

Acknowledgements The authors would like to thank the reviewers for their comments and suggestions, which helped to improve this paper.

Appendix: Detailed proofs of propositions 7 and 8

Before we address Propositions 7 and 8, we first state the definition of the variant function we will use in the proof of Proposition 7 and we state three lemmata that characterise properties of this variant function.

Definition 21 (*Governed stuttering bisimulation game measure*) We define a measure with respect to $\simeq$ for a configuration $((u_0, u_1), c)$ in the governed stuttering bisimulation game as follows:

$$m(u_0, u_1, c) \triangleq \begin{cases} (0, 0) & \text{if } c = \checkmark \\ (\infty, 0) & \text{if } c = \dagger \wedge \exists v_0 \in u_0^\bullet, v_1 \in u_1^\bullet : v_0 \simeq u_0 \wedge v_1 \simeq u_1 \\ (\mathit{expel}(u_j, u_{1-j}), 0) & \text{if } c = \dagger \wedge \forall v \in u_j^\bullet : u_j \not\simeq v \\ (0, \mathit{exit}(u_j, u_{1-j}, t)) & \text{if } c = (j, t) \end{cases}$$

where $\mathit{expel}(u_0, u_1)$ denotes the number of steps before $\mathcal{P}(u_0)$'s opponent is forced from $[u_0]_\simeq$ and $\mathit{exit}(u_0, u_1, u_2)$ denotes the number of steps it takes for $\mathcal{P}(u_0)$ to force play to

$[u_2]_{\simeq}$ from u_1. Formally, we have:

$$\begin{aligned} expel(u_0, u_1) &\overset{\Delta}{=} dist_{\neg\mathcal{P}(u_0),[u_0]_{\simeq}}(u_1, V \setminus [u_0]_{\simeq}) \\ exit(u_0, u_1, u_2) &\overset{\Delta}{=} dist_{\mathcal{P}(u_0),[u_0]_{\simeq}}(u_1, [u_2]_{\simeq}) \end{aligned}$$

where for $U, T \subseteq V$ and $v \in U$:

$$dist_{i,U}(v, T) \overset{\Delta}{=} \begin{cases} \min\{n \mid v \in {}_U Attr_i^n(T)\} & \text{if } v \; {}_i{\mapsto}_U \; T \\ \infty & \text{otherwise} \end{cases}$$

Measures are ordered lexicographically, *i.e.* $(m_0, m_1) < (n_0, n_1)$ iff $m_0 < n_0 \vee (m_0 = n_0 \wedge m_1 < n_1)$.

We first prove some basic properties for the function m.

Lemma 15 *For $u, v \in V$, $(u \simeq v \wedge m(u, v, \dagger) = (m_0, m_1)) \implies m_0 > 0$.*

Proof First, observe that (apart from $c = \dagger$), the conditions in the second and third clause of the definition of m are complementary. Furthermore observe that, for all u_0, u_1 such that $u_0 \simeq u_1$, we have $expel(u_0, u_1) > 0$ since $u_1 \notin V \setminus [u_0]_{\simeq}$. The result then immediately follows. □

Lemma 16 *Let $U, T \subseteq V$, such that $U \cap T = \emptyset$ and let $u \in U$. For all players i, if $u \; {}_i{\mapsto}_U \; T$ then $dist_{i,U}(u, T) > \min\{dist_{i,U}(v, T) \mid u \to v \wedge v \in U \cup T\}$.*

Proof Assume $u \; {}_i{\mapsto}_U \; T$ and let $n = dist_{i,U}(u, T)$. Hence $u \in {}_U Attr_i^n(T)$ and $u \notin {}_U Attr_i^{n-1}(T)$. Observe that $n > 0$ since $u \in U$ and $U \cap T = \emptyset$. We proceed by a case distinction on $\mathcal{P}(u)$.

- $\mathcal{P}(u) = i$. Since n is such that $u \notin {}_U Attr_i^{n-1}(T)$, we have $\exists v \in u^\bullet : v \in {}_U Attr_i^{n-1}(T)$. Let v be such, and observe that $dist_{i,U}(v, T) \le n - 1 < n$. The result then follows immediately.
- $\mathcal{P}(u) \neq i$. As n is such that $u \notin {}_U Attr_i^{n-1}(T)$, we have $\forall v \in u^\bullet : v \in {}_U Attr_i^{n-1}(T)$. Hence $\forall v \in V : u \to v \implies dist_{i,U}(v, T) \le n - 1 < n$. Again the result follows immediately. □

We also prove the following stronger result in case u is owned by the opponent.

Lemma 17 *Let $U, T \subseteq V$, such that $U \cap T = \emptyset$ and $u \in U \cap V_{\neg i}$. Then $u \; {}_i{\mapsto}_U \; T$ implies $dist_{i,U}(u, T) > \max\{dist_{i,U}(v, T) \mid u \to v \wedge v \in U \cup T\}$.*

Proof Let $u \in U \cap V_{\neg i}$ such that $u \; {}_i{\mapsto}_U \; T$. Suppose $n = dist_{i,U}(u, T)$. Then $u \in {}_U Attr_i^n(T)$ and $u \notin {}_U Attr_i^{n-1}(T)$. Since $u \notin V_i$, $\forall v \in u^\bullet : v \in {}_U Attr_i^{n-1}(T)$, hence $\forall v \in u^\bullet : dist_{i,U}(v, T) \le n - 1 < n$; furthermore, such v are in $U \cup T$, and the result follows immediately. □

Proposition 7 *For all $v, w \in V$ if $v \simeq w$ then $v \equiv_{g,st} w$.*

Proof We prove for all governed stuttering bisimilar vertices $v \simeq w$ that there is a *Duplicator* winning strategy in the governed stuttering bisimulation game from configuration $((v, w), \checkmark)$.

We show this by constructing a *Duplicator*-strategy that moves between governed stuttering bisimilar vertices, and that makes sure that from every configuration $((v, w), c)$, within a finite number of steps another configuration $((v', w'), \checkmark)$ is reached. As a consequence, the *Duplicator*-strategy is such that it passes through configurations with reward $\checkmark$ infinitely often, hence the *Duplicator* strategy is winning.

Formally, we preserve the invariant Φ which is the conjunction of the following for configurations $((u_0, u_1), c)$:

- $u_0 \mathrel{\underset{\cdots}{\sim}} u_1$,
- $c = (j, t)$ implies $u_j \mathrel{\not\underset{\cdots}{\sim}} t$,
- $c = (0, u)$ implies $(u_0, u_1) \in V_{\Diamond} \times V$,
- $c = (1, u)$ implies $(u_0, u_1) \in V \times V_{\Box}$.

In addition, we prove that from every configuration $((u_0, u_1), c)$, within a finite number of steps a configuration $((u_0', u_1'), \checkmark)$ is reached by showing that m is a variant function. That is, if, in a round, we move from configuration $((u_0, u_1), c)$ to configuration $((u_0', u_1'), c')$ with $c \neq \checkmark$ and $c' \neq \checkmark$, then $m(u_0, u_1, c) > m(u_0', u_1', c')$.

From these two observations, the result immediately follows. Note that initially we are in a configuration $((v, w), \checkmark)$; hence Φ is satisfied trivially. Suppose the game has reached a configuration $((u_0, u_1), c)$ satisfying Φ. In step 1 of the round, *Spoiler* chooses to play from (t_0, t_1), taken from (u_0, u_1) or (u_1, u_0). We remark that if *Spoiler* decides to play from (u_1, u_0), then, regardless of step 2, any pending challenge or $\dagger$ will be replaced by a $\checkmark$ at the end of step 3. For this case, we therefore do not need to argue that m decreases.

We distinguish cases based on which player can force a divergence in the coinductive definition and consider *Duplicator*'s options in step 2 and 3 of the round, and prove that *Duplicator* can always arrive in a new configuration $((t_0', t_1'), c')$ that satisfies Φ and for which, if $c' \neq \checkmark$ and $c \neq \checkmark$, $m(t_0, t_1, c) > m(t_0', t_1', c')$.

- $t_0 \;{}_{\Diamond}{\mapsto}_{\underset{\cdots}{\sim}}$ and $t_0 \;{}_{\Box}{\mapsto}_{\underset{\cdots}{\sim}}$. This case is trivial, as in that case exactly one (reachable) equivalence class exists.
- $t_0 \;{}_{\Diamond}{\mapsto}_{\underset{\cdots}{\sim}}$ and $t_0 \;{}_{\Box}{\not\mapsto}_{\underset{\cdots}{\sim}}$. Since $t_0 \mathrel{\underset{\cdots}{\sim}} t_1$ also $t_1 \;{}_{\Diamond}{\mapsto}_{\underset{\cdots}{\sim}}$. We distinguish cases based on the owners of the vertices.
  - $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \Diamond$. *Spoiler* plays $t_0 \to w_0$.
    - Case there is some $w_1 \in t_1^{\bullet}$ such that $w_0 \mathrel{\underset{\cdots}{\sim}} w_1$. Then *Duplicator* plays to such a w_1. The new configuration is $((w_0, w_1), \checkmark)$.
    - Case there is no $w_1 \in t_1^{\bullet}$ such that $w_0 \mathrel{\underset{\cdots}{\sim}} w_1$. Then *Duplicator* plays to a $w_1 \in t_1^{\bullet}$ for which $w_1 \mathrel{\underset{\cdots}{\sim}} t_1$ with minimal $m(t_0, w_1, (0, w_0))$; the existence of a $w_1 \mathrel{\underset{\cdots}{\sim}} t_1$ follows from $t_1 \;{}_{\Diamond}{\mapsto}_{\underset{\cdots}{\sim}}$.
      New configuration: if $c \in \{\checkmark, \dagger, (0, w_0)\}$ and $u_0 = t_0$ then the new configuration is $((t_0, w_1), (0, w_0))$, and else it is $((t_0, w_1), \checkmark)$.
      Progress: we demonstrate $m(t_0, w_1, (0, w_0)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (0, w_0)\}$. If $c = \dagger$ this follows from Lemma 15. In case $c = (0, w_0)$, this follows from Lemmata 16 and 17.
  - $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \Box$. *Spoiler* plays $t_1 \to w_1$. Since $t_1 \;{}_{\Diamond}{\mapsto}_{\underset{\cdots}{\sim}}$, all $w_1 \in t_1^{\bullet}$ satisfy $t_1 \mathrel{\underset{\cdots}{\sim}} w_1$. The same holds for all $w_0 \in t_0^{\bullet}$. *Duplicator* can thus play arbitrary $t_0 \to w_0$.
    New configuration: $((w_0, w_1), \checkmark)$
  - $\mathcal{P}(t_0) = \Diamond$, $\mathcal{P}(t_1) = \Box$. *Spoiler* plays $t_0 \to w_0$ and $t_1 \to w_1$. Since $t_1 \;{}_{\Diamond}{\mapsto}_{\underset{\cdots}{\sim}}$, also $w_1 \mathrel{\underset{\cdots}{\sim}} t_1$. We distinguish two further cases.
    - Case $w_0 \mathrel{\underset{\cdots}{\sim}} w_1$.
      New configuration: $((w_0, w_1), \checkmark)$.
    - Case $w_0 \mathrel{\not\underset{\cdots}{\sim}} t_0$.
      New configuration: if $c \in \{\checkmark, \dagger, (0, w_0)\}$ and $u_0 = t_0$ then the new configuration

is $((t_0, w_1), (0, w_0))$; else the new configuration is $((t_0, w_1), \checkmark)$. Observe that $t_0 \simeq w_1$.
Progress: we demonstrate $m(t_0, w_1, (0, w_0)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (0, w_0)\}$. If $c = \dagger$ this follows from Lemma 15. In case $c = (0, w_0)$, this follows from Lemmata 16 and 17.

- $\mathcal{P}(t_0) = \Box$, $\mathcal{P}(t_1) = \Diamond$. *Duplicator* plays $t_1 \to w_1$ such that $w_1 \simeq t_1$ and $t_0 \to w_0$. Such a w_1 exists because $t_1 \,{}_{\Diamond}{\mapsto}_{\simeq}$.
New configuration: $((w_0, w_1), \checkmark)$.

– $t_0 \,{}_{\Box}{\mapsto}_{\simeq}$ and $t_0 \,{}_{\Diamond}{\not\mapsto}_{\simeq}$. So, as before, $t_1 \,{}_{\Box}{\mapsto}_{\simeq}$. This case is dual to the previous one.
– $t_0 \,{}_{\Diamond}{\not\mapsto}_{\simeq}$ and $t_0 \,{}_{\Box}{\not\mapsto}_{\simeq}$. We consider the owners of t_0 and t_1.

- $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \Diamond$. *Spoiler* plays $t_0 \to w_0$. We distinguish two cases.
  - Case there is some $w_1 \in t_1^\bullet$ for which $w_0 \simeq w_1$. *Duplicator* plays $t_1 \to w_1$ such that $w_0 \simeq w_1$.
  New configuration: $((w_0, w_1), \checkmark)$.
  - Case there is no $w_1 \in t_1^\bullet$ for which $w_0 \simeq w_1$.
    · Case $t_0 \simeq w_0$. Then for all $w_1 \in t_1^\bullet$, $w_1 \not\simeq t_1$.
    New configuration: $((w_0, t_1), \dagger)$ if $c \in \{\checkmark, \dagger\}$ and $u_0 = t_0$; else $((w_0, t_1), \checkmark)$.
    Progress: $m(t_0, w_1, \dagger) < m(t_0, t_1, \dagger)$ follows from Lemma 17.
    · Case $t_0 \not\simeq w_0$. In this case *Duplicator* plays $t_1 \to w_1$ such that $t_1 \simeq w_1$ and $m(t_0, w_0, (0, w_0))$ is minimal.
    New configuration: if $c \in \{\checkmark, \dagger, (0, w_0)\}$ and $u_0 = t_0$ then the new configuration is $((t_0, w_1), (0, w_0))$ and else $((t_0, w_1), \checkmark)$.
    Progress: we must show $m(t_0, w_1, (0, w_0)) < m(t_0, t_1, \dagger)$ for $c \in \{\dagger, (0, w_0)\}$. In case $c = \dagger$ this follows from Lemma 15. In case $c = (0, w_0)$, this follows from Lemmata 16 and 17.
- $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \Box$. *Spoiler* plays $t_1 \to w_1$.
  - Case there is some $w_0 \in t_0^\bullet$ for which $w_0 \simeq w_1$. *Duplicator* plays $t_0 \to w_0$ such that $w_0 \simeq w_1$.
  New configuration: $((w_0, w_1), \checkmark)$.
  - Case there is no $w_0 \in t_0^\bullet$ for which $w_0 \simeq w_1$.
    · Case $t_1 \simeq w_1$. Then for all $w_0 \in t_0^\bullet$, $w_0 \not\simeq t_0$. *Duplicator* plays some arbitrary $t_0 \to w_0$.
    New configuration: $((t_0, w_1), \dagger)$ if $c \in \{\checkmark, \dagger\}$ and $u_1 = t_1$; else $((t_0, w_1), \checkmark)$.
    Progress: $m(w_0, t_1, \dagger) < m(t_0, t_1, \dagger)$ follows from Lemma 17.
    · Case $t_1 \not\simeq w_1$. In this case *Duplicator* plays $t_0 \to w_0$ such that $t_0 \simeq w_0$ and $m(w_0, t_0, (1, w_1))$ is minimal.
    New configuration: if $c \in \{\checkmark, \dagger, (1, w_1)\}$ and $u_1 = t_1$ the new configuration is $((w_0, t_1), (1, w_1))$ and else it is $((w_0, t_1), \checkmark)$.
    Progress: we must show $m(w_0, t_1, (1, w_1)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (1, w_1)\}$. In case $c = \dagger$ this follows from Lemma 15. In case $c = (1, w_1)$, this follows from Lemmata 16 and 17.
- $\mathcal{P}(t_0) = \Diamond$, $\mathcal{P}(t_1) = \Box$. *Spoiler* plays $t_0 \to w_0$ and $t_1 \to w_1$. In case $w_0 \not\simeq w_1$ then either $t_0 \simeq w_0$ or $t_1 \simeq w_1$. We distinguish three cases:
  - Case $w_0 \simeq w_1$.
  New configuration: $((w_0, w_1), \checkmark)$.

- Case $w_0 \not\simeq w_1$ and $t_0 \simeq w_0$.
New configuration: $((w_0, t_1), (1, w_1))$ if $c \in \{\checkmark, \dagger, (1, w_1)\}$ and $u_1 = t_1$; else $((w_0, t_1), \checkmark)$.
Progress: we must show $m(w_0, t_1, (1, w_1)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (1, w_1)\}$. In case $c = \dagger$ this follows from Lemma 15. In case $c = (1, w_1)$ this follows from Lemmata 16 and 17.
- Case $w_0 \not\simeq w_1$ and $t_1 \simeq w_1$.
New configuration: $((t_0, w_1), (0, w_0))$ if $c \in \{\checkmark, \dagger, (0, w_0)\}$ and $u_0 = t_0$; else $((t_0, w_1), \checkmark)$.
Progress: we must show $m(t_0, w_1, (0, w_0)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (0, w_0)\}$. In case $c = \dagger$ this follows from Lemma 15. In case $c = (0, w_0)$ this follows from Lemmata 16 and 17.

– $\mathcal{P}(t_0) = \Box, \mathcal{P}(t_1) = \Diamond$.

- Case there are $w_0 \in t_0^\bullet$ and $w_1 \in t_1^\bullet$ such that $w_0 \simeq w_1$. Then *Duplicator* plays to such w_0 and w_1.
New configuration: $((w_0, w_1), \checkmark)$.
- Case there are no $w_0 \in t_0^\bullet$ and $w_1 \in t_1^\bullet$ such that $w_0 \simeq w_1$.
  · case there is some $w_0 \in t_0^\bullet$ such that $w_0 \simeq t_0$. Then *Duplicator* plays to w_0 that is such while minimising $m(w_0, t_1, \dagger)$.
  New configuration: $((w_0, t_1), \dagger)$ if $u_1 = t_1$; else it is $((w_0, t_1), \checkmark)$.
  Progress: we first show that $u_1 = t_1$ implies $c \in \{\dagger, \checkmark\}$. Towards a contradiction, assume $c = (0, t)$ for some t. By our invariant, this implies $(t_0, t_1) \in V_\Diamond \times V$. Since $(u_0, u_1) \in \{(t_0, t_1), (t_1, t_0)\}$ and $u_1 = t_1$ we have $u_0 = t_0$. But then both $u_0 \in V_\Box$ and $u_0 \in V_\Diamond$. Contradiction. Towards another contradiction, assume $c = (1, t)$ for some t. By our invariant, this implies $(t_0, t_1) \in V \times V_\Box$. This contradicts $u_1 = t_1$ since $\mathcal{P}(u_1) = \Diamond$. It therefore suffices to show $m(w_0, t_1, \dagger) < m(t_0, t_1, \dagger)$. This follows from the fact that we minimised $m(w_0, t_1, \dagger)$ and Lemmata 16 and 17.
  · case there is some $w_1 \in t_1^\bullet$ such that $w_1 \simeq t_1$. Then *Duplicator* plays to w_1 that is such while minimising $m(t_0, w_1, \dagger)$.
  New configuration: $((t_0, w_1), \dagger)$ if $u_0 = t_0$; else it is $((t_0, w_1), \checkmark)$.
  Progress: using arguments, similar to those in the previous case, it follows that $c \in \{\dagger, \checkmark\}$. It therefore suffices to show $m(t_0, w_1, \dagger) < m(t_0, t_1, \dagger)$. This follows from the fact that we minimised $m(t_0, w_1, \dagger)$ and Lemmata 16 and 17. □

We next focus on proving that every pair of vertices related through the governed stuttering bisimulation game are in fact governed stuttering bisimilar.

Proposition 8 *For all $v, w \in V$ if $v \equiv_{g,st} w$ then $v \simeq w$.*

Proof We prove the contrapositive of the statement, *i.e.* for all $v, w \in V$, if $v \not\simeq w$, then also $v \not\equiv_{g,st} w$. Let $v \not\simeq w$. By Corollary 2, then also $(v, w) \notin \nu\mathcal{F}$. By the Knaster-Tarski-Kleene fixpoint approximation theorem, we thus have $(v, w) \notin \bigcap\limits_{k \geq 1} \mathcal{F}^k(V \times V)$. Let R^k denote the relation $\mathcal{F}^k(V \times V)$; *i.e.*, R^k is the relation obtained by applying the operator $\mathcal{F}$ k-times. Note that because of monotonicity, $R^k = \bigcap_{l \leq k} R^l$. We next prove, using induction, that for all $k \geq 1$:

$$\begin{array}{l}\textit{Spoiler} \text{ wins the governed stuttering bisimulation game} \\ \text{for all configurations} ((u_0, u_1), c) \text{ for which} (u_0, u_1) \notin R^k\end{array} \tag{IH}$$

- Base case $k = 1$. Observe that $R^1 = \{(v, w) \in V \times V \mid \Omega(v) = \Omega(w)\}$. *Spoiler* wins the governed stuttering bisimulation game for all configurations $((u_0, u_1), c)$ satisfying $(u_0, u_1) \notin R^1$: all plays starting in such a configuration trivially violate *Duplicator*'s winning condition.
- Inductive step. Assume that the statement holds for some $k \geq 1$. Pick an arbitrary position (u_0, u_1) for which $(u_0, u_1) \notin R^{k+1}$ and let c be an arbitrary challenge/reward. We must show that *Spoiler* wins the governed stuttering bisimulation game for these. Recall that we have $R^{k+1} \subseteq R^k$.

If $(u_0, u_1) \notin R^k$, then by the induction hypothesis, *Spoiler* wins the governed stuttering bisimulation game from the configuration $((u_0, u_1), c)$. Observe that by definition of $\mathcal{F}$, we have for all $(v, w) \in R^k \setminus R^{k+1}$ that there are $i \in \{\Diamond, \Box\}$ and $\mathcal{U}, \mathcal{T} \subseteq V_{/R^k}$ for which

$$[v]_{R^k} \in \mathcal{U} \setminus \mathcal{T} \text{ but not } v \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T} \Leftrightarrow w \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}. \tag{$\star$}$$

Let $i, \mathcal{U}, \mathcal{T}$ be such that ($\star$). We focus on the case $i = \Diamond$; the case that $i = \Box$ is fully dual. Assume that $v \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ and not $w \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$; the case in which not $v \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ but $w \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ is symmetric. Note that we can assume that $\mathcal{T} \cap \mathcal{U} = \emptyset$, as $v \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ iff $v \;{}_i\!\!\mapsto_{\mathcal{U}\setminus\mathcal{T}} \mathcal{T}$ for any $\mathcal{U}, \mathcal{T}$. We may therefore also simplify $\mathcal{U} \setminus \mathcal{T}$ to $\mathcal{U}$.

Let $\sigma \in \mathbb{S}_i$ be the (memoryless) strategy underlying $v \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. Using σ, we construct a winning strategy for *Spoiler* for configuration $((v, w), c)$. We first show that *Spoiler* can invariantly move between configurations $((t_0, t_1), c)$ that satisfy the following property Φ:

$$\text{If } [t_0]_{R^k} = [t_1]_{R^k} \text{ then} \begin{cases} t_0 \;{}_\sigma\!\!\mapsto_{\mathcal{U}} \mathcal{T} \text{ but not } t_1 \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T} \\ c = (0, t) \text{ implies } t_0 \in V_\Diamond \text{ and } t_0 \;{}_\sigma\!\!\to t \\ c = (1, t) \text{ implies } t_1 \in V_\Box, t \in t_1^\bullet \text{ and not } t \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T} \end{cases}$$

Let $((t_0, t_1), c)$ be a configuration for which Φ holds. For all such configurations *Spoiler*'s move in step 1 of a round is to play from (t_0, t_1); *i.e. Spoiler* does not switch positions. We distinguish three main cases, showing that *Duplicator* has no other option than to choose a new configuration that satisfies Φ.

1. Case $c \in \{\dagger, \checkmark\}$. We furthermore distinguish cases based on the players of t_0 and t_1.
   - Case $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \Diamond$. Since $t_0 \;{}_\sigma\!\!\mapsto_{\mathcal{U}} \mathcal{T}$, *Spoiler* proposes to move from t_0 to $\sigma(t_0)$. *Duplicator* proposes $u_1 \in t_1^\bullet$. Observe that not $u_1 \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. *Duplicator* then can propose to continue in: $((\sigma(t_0), u_1), \checkmark)$, $((t_0, u_1), (0, \sigma(t_0)))$, or $((\sigma(t_0), t_1), \dagger)$. Clearly, all new configurations satisfy Φ.
   - Case $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \Box$. *Spoiler* proposes to move from t_1 to u_1 such that not $u_1 \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. Such u_1 exists. *Duplicator* proposes $u_0 \in t_0^\bullet$. Observe that $t_0 \;{}_\sigma\!\!\to u_0$. *Duplicator* proposes to continue in: $((u_0, u_1), \checkmark)$, $((t_0, u_1), \dagger)$, or $((u_0, t_1), (1, u_1))$. All new configurations satisfy Φ.
   - Case $\mathcal{P}(t_0) = \Diamond$, $\mathcal{P}(t_1) = \Box$. Since $t_0 \;{}_\sigma\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ *Spoiler* proposes to move from t_0 to $\sigma(t_0)$ and from t_1 to u_1 such that not $u_1 \;{}_i\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. Note that such u_1 exists. *Duplicator* proposes to continue in: $((\sigma(t_0), u_1), \checkmark)$, $((t_0, u_1), (0, \sigma(t_0)))$, or $((\sigma(t_0), t_1), (1, u_1))$. Again, all new configurations satisfy Φ.

- Case $\mathcal{P}(t_0) = \Box$, $\mathcal{P}(t_1) = \Diamond$. *Duplicator* proposes to move from t_1 to u_1 and from t_0 to u_0. Since $i = \Diamond$, we have $t_0 \,{}_{\sigma}\!\!\rightarrow u_0$ and because of Φ, we have not $u_1 \,{}_{i}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. *Duplicator* then proposes to continue in: $((u_0, u_1), \checkmark)$, $((t_0, u_1), \dagger)$, or $((u_0, t_1), \dagger)$. All new configurations satisfy Φ.

2. Case $c = (0, t)$. Because of Φ, we have $t_0 \,{}_{\sigma}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ and $\mathcal{P}(t_0) = \Diamond$. Then *Spoiler* plays from configuration (t_0, t_1). We furthermore distinguish cases based on the owner of t_1.

   - Case $\mathcal{P}(t_1) = \Diamond$. *Spoiler* proposes to move from t_0 to t. *Duplicator* proposes $u_1 \in t_1^\bullet$. Observe that not $u_1 \,{}_{i}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. *Duplicator* proposes to continue in on of the following configurations: $((t, u_1), \checkmark)$, $((t_0, u_1), (0, t))$, or $((t, t_1), \dagger)$
   - Case $\mathcal{P}(t_1) = \Box$. *Spoiler* proposes to move from t_0 to t and from t_1 to u_1 such that not $u_1 \,{}_{i}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. Such u_1 exists. *Duplicator* proposes to continue in: $((t, u_1), \checkmark)$, $((t_0, u_1), (0, t))$, or $((t, t_1), \checkmark)$

   In both cases, the new rounds satisfy Φ.
3. Case $c = (1, t)$. Because of Φ, we have not $t_1 \,{}_{\sigma}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ and $\mathcal{P}(t_1) = \Box$. Then *Spoiler* plays from configuration (t_0, t_1). We furthermore distinguish cases based on the owner of t_0.

   - Case $\mathcal{P}(t_0) = \Diamond$. *Spoiler* proposes to move from t_0 to $\sigma(t_0)$ and from t_1 to t. *Duplicator* proposes to continue in: $((\sigma(t_0), t), \checkmark)$, $((t_0, t), \checkmark)$, or $((\sigma(t_0), t_1), (1, t))$.
   - Case $\mathcal{P}(t_0) = \Box$. *Spoiler* proposes to move from t_1 to t. *Duplicator* proposes $u_0 \in t_0^\bullet$. Observe that $t_0 \,{}_{\sigma}\!\!\rightarrow u_0$. *Duplicator* proposes to continue in: $((u_0, t), \checkmark)$, $((t_0, t), \dagger)$, or $((u_0, t_1), (1, t))$.

We next observe that for any (t_0, t_1) for which t_0, t_1 meet the premiss of Φ, but not the conclusion, *Spoiler* can, in a single round, move to a configuration that either does not meet Φ's premiss or to one that meets Φ's conclusion. More specifically, suppose that $[t_0]_{R^k} = [t_1]_{R^k}$ but one of the following holds:

1. $t_1 \,{}_{\sigma}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$ but not $t_0 \,{}_{i}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$;
2. $c = (0, t)$ implies $t_0 \notin V_\Diamond$ or not $t_0 \,{}_{\sigma}\!\!\rightarrow t$;
3. $c = (1, t)$ implies $t_1 \notin V_\Box$, $t \notin t_1^\bullet$, or $t \,{}_{i}\!\!\mapsto_{\mathcal{U}} \mathcal{T}$.

Whenever we are in case 1, *Spoiler* switches positions in step 1 of a round and follows the strategy outlined above. Whenever we are in case 2 or 3, *Spoiler* drops challenge c in step 1 and plays as if $c \in \{\dagger, \checkmark\}$. In all three cases, *Duplicator* is rewarded a $\checkmark$ as the new challenge at the end of the round and Φ holds trivially.

Summarising, we find that for configurations $((t_0, t_1), c)$ for which both ($\star$) and Φ hold, *Spoiler* can move to another configuration that either meets Φ or is such that Φ's premiss is violated. For configurations $((t_0, t_1), c)$ for which ($\star$) but not Φ holds, *Spoiler* can move in a single round to a configuration for which she can henceforth maintain Φ as an invariant or for which Φ's premiss is violated.

We finally argue that when *Spoiler* plays according to the above strategy, she wins all plays. Observe that we only need to show this for all infinite plays that pass only through positions (u, t) for which $(u, t) \in R^k$; for those plays that at some point pass along a position $(u, t) \notin R^k$, our induction hypothesis yields a winning strategy for *Spoiler*.

Let $(v_0, w_0)\ (v_1, w_1)\ (v_2, w_2) \ldots$ be an infinite sequence of positions on an infinite play π that is allowed by *Spoiler*'s strategy, such that for all l, $[v_l]_{R^k} = [w_l]_{R^k}$. Towards a contradiction, assume that *Duplicator* wins π. Observe that $[v_l]_{R^k} = [w_l]_{R^k}$ implies that $\Omega(v_l) = \Omega(w_l)$ for all $(v_l, w_l) \in R^k \setminus R^{k+1}$; therefore we can only arrive at a contradiction by showing that *Duplicator* earns a finite number of $\checkmark$ rewards along π. By invariant Φ, for all positions (v_l, w_l), for $l \geq 1$, we have $v_l \,{}_\sigma\!\!\mapsto_{\mathcal{U}} \mathcal{T}$. Let $\delta(v_l, w_l)$ denote the length of the longest path from v_l to reach $\mathcal{T}$ when playing according to σ. Note that δ is finite and decreases along the positions in π, but never reaches 0, as all vertices remain in $\mathcal{U}$. This means that for some m, we have $\delta(v_m, w_m) = \delta(v_n, w_n)$ for all $n \geq m$. Fix this m. Moreover, there must be some u such that:

$$u \,{}_\sigma\!\!\mapsto_{\mathcal{U}} \mathcal{T} \wedge \forall n \geq m : \forall (v_n, w_n) \in \pi : v_n = u$$

But this means that, once *Spoiler*'s strategy reaches the position containing u, all remaining ✓'s earned by *Duplicator* must be due to *Spoiler* switching positions or discarding a challenge in step 1 of each new round. As we explained, *Spoiler* switches positions and/or drops a challenge only in the first round when starting in a configuration that does not satisfy Φ; she never does so afterwards. Therefore, *Duplicator* earns no ✓ rewards when π reaches a configuration containing a position with u. But then *Duplicator* earns only a finite number of ✓ rewards along π, contradicting the assumption that *Duplicator* wins π. Therefore, *Spoiler* has a strategy to win from any configuration $((u_0, u_1), c)$ for which $(u_0, u_1) \notin R^k$. □

References

1. Arnold, A., Vincent, A., Walukiewicz, I.: Games for synthesis of controllers with partial observation. Theor. Comput. Sci. **303**(1), 7–34 (2003)
2. Arnold, A., Walukiewicz, I.: Nondeterministic controllers of nondeterministic processes. Logic and Automata. Volume 2 of Texts in Logic and Games, pp. 29–52. Amsterdam University Press, Amsterdam (2008)
3. Basten, T.: Branching bisimilarity is an equivalence indeed!. Inf. Proc. Let. **58**(3), 141–147 (1996)
4. van Benthem, J.: Correspondence theory. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Philosophical Logic volume II, pp. 167–248. Springer, Dordrecht (1984)
5. Björklund, H., Sandberg, S., Vorobyov, S.G.: A discrete subexponential algorithm for parity games. In: Proceedings STACS'03 volume 2607 of LNCS, pp. 663–674. Springer (2003)
6. Björklund, H., Vorobyov, S.G.: Combinatorial structure and randomized subexponential algorithms for infinite games. Theor. Comput. Sci. **349**(3), 347–360 (2005)
7. Browne, M.C., Clarke, E.M., Grumberg, O.: Characterizing finite Kripke structures in propositional temporal logic. TCS **59**, 115–131 (1988)
8. Bulychev, P.E., Konnov, I.V., Zakharov, V.A.: Computing (bi)simulation relations preserving CTL*-X for ordinary and fair kripke structures. Institute for System Programming, Russian Academy of Sciences, Mathematical Methods and Algorithms, 12 (2007)
9. Bustan, D., Grumberg, O.: Simulation-based minimization. ACM Trans. Comput. Log. **4**(2), 181–206 (2003)
10. Clemente, L.: Büchi automata can have smaller quotients. In: ICALP'11, volume 6756 of Lecture Notes in Computer Science, pp. 258–270. Springer (2011)
11. Cranen, S.: Getting the point: obtaining and understanding fixpoints in model checking. PhD thesis, Eindhoven University of Technology. Eindhoven (2015)
12. Cranen, S., Gazda, M., Wesselink, J.W., Willemse, T.A.C.: Abstraction in fixpoint logic. ACM Trans. Comput. Logic **16**(4), 29:1–29:39 (2015)
13. Cranen, S., Keiren, J.J.A., Willemse, T.A.C.: Stuttering mostly speeds up solving parity games. In: Proceedings of NFM'11 volume 6617 of LNCS, pp. 207–221. Springer (2011)
14. Cranen, S., Keiren, J.J.A., Willemse., T.A.C.: A cure for stuttering parity games. In: Proceedings of ICTAC'12, volume 7521 of LNCS, pp. 198–212. Springer (2012)
15. Cranen, S., Keiren, J.J.A., Willemse, T.A.C.: Parity game reductions (2016), arXiv:1603.06422
16. de Frutos Escrig, D., Keiren, J.J.A., Willemse, T.A.C.: Branching bisimulation games. In: Proceedings of FORTE'16, (2016). doi:10.1007/978-3-319-39570-8_10
17. Emerson, E.A., Jutla, C.S.: Tree automata, mu-calculus and determinacy. In: Proceedings of FOCS'91. IEEE Computer Society, pp. 368–377 (1991)
18. Emerson, E.A., Jutla, C.S., Sistla, A.P.: On model checking for the μ-calculus and its fragments. Theor. Comput. Sci. **258**(1–2), 491–522 (2001)

19. Etessami, K., Wilke, Th., Schuller, R.A.: Fair simulation relations, parity games, and state space reduction for büchi automata. SIAM J. Comput. **34**(5), 1159–1175 (2005)
20. Friedmann, O., Lange, M.: Solving parity games in practice. In: Proceedings of ATVA'09, volume 5799 of LNCS. Springer, pp. 182–196 (2009)
21. Friedmann, O., Lange, M.: Deciding the unguarded modal μ-calculus. J. Appl. Non-Class. Log. **23**(4), 353–371 (2013)
22. Fritz, C.: Simulation-Based Simplification of omega-Automata. PhD thesis, Christian-Albrechts-Universität zu Kiel, (2005)
23. Fritz, C., Wilke. T., Simulation relations for alternating parity automata and parity games. In: Proceedings of DLT'06, volume 4036 of LNCS, pp. 59–70. Springer (2006)
24. Fritz, C., Wilke, Th.: State space reductions for alternating büchi automata. In: Proceedings of FSTTCS'02, volume 2556 of LNCS, pp. 157–168. Springer (2002)
25. Gazda, M.W., Willemse, T.A.C.: Consistent consequence for boolean equation systems. In: Proceedings of SOFSEM'12, volume 7147 of LNCS, pp. 277–288. Springer (2012)
26. Gazda, M.W., Willemse, T.A.C.: On parity game preorders and the logic of matching plays. In: Proceedings of SOFSEM'16, volume 9587 of LNCS, pp. 277–289. Springer (2016)
27. Glabbeek, van R.J.: The linear time—branching time spectrum. In: Proceedings of CONCUR '90, volume 458 of LNCS, pp. 278–297. Springer (1990)
28. Glabbeek, van R.J.: The linear time—branching time spectrum II. In: Proceedings of CONCUR'93, volume 715 of LNCS, pages 66–81. Springer (1993)
29. Grädel, E., Thomas, W., Wilke, T., (eds) Automata Logics, and Infinite Games, volume 2500 of LNCS. Springer (2002)
30. Groote, J.F., Vaandrager, F.W.: An efficient algorithm for branching bisimulation and stuttering equivalence. In: Proceedings of ICALP'90, volume 443 of LNCS, pp. 626–638. Springer (1990)
31. Groote, J.F., Jansen, D.N., Keiren, J.J.A., Wijs, A.: An O(mlogn) algorithm for computing stuttering equivalence and branching bisimulation. ACM Trans. Comput. Log. **18**(2), 13:1–3:34 (2017). doi:10.1145/3060140
32. Huth, M., Kuo, J.H.-P., Piterman, N.: Fatal attractors in parity games. In: Proceedings of FOSSACS'13, volume 7794 of LNCS, pp. 34–49. Springer (2013)
33. Huth, M., Kuo, J.H.-P., Piterman, N., Static analysis of parity games: alternating reachability under parity. In: Semantics, Logics, and Calculi, volume 9560 of LNCS, pp. 159–177. Springer (2016)
34. Janin, D.: A contribution to formal methods: games, logic and automata, December (2005). Habilitation thesis
35. Jurdziński, M.: Deciding the winner in parity games is in UP $\cap$ co-UP. Inf. Proc. Let. **68**(3), 119–124 (1998)
36. Jurdziński, M.: Small progress measures for solving parity games. In: Proceedings of STACS'00, volume 1770 of LNCS, pp. 290–301. Springer (2000)
37. Jurdziński, M., Paterson, M., Zwick, U.: A Deterministic Subexponential Algorithm for Solving Parity Games. In: Proceedings of SODA'06, pp. 117–123. ACM/SIAM (2006)
38. Katoen, J.P., Kemna, T., Zapreev, I.S., Jansen, D.N.: Bisimulation minimisation mostly speeds up probabilistic model checking. In: Proceedings of TACAS'07 volume 4424 of LNCS, pp. 76–92. Springer (2007)
39. Keiren, J.J.A.: Advanced Reduction Techniques for Model Checking. PhD thesis, Eindhoven University of Technology, (2013)
40. Keiren, J.J.A.: Benchmarks for parity games. In: Proceedings of FSEN'15, volume 9392 of LNCS, pp. 126–142. Springer (2015)
41. Keiren, J.J.A., Wesselink, J.W., Willemse, T.A.C.: Liveness analysis for parameterised Boolean equation systems. In: Proceedings of ATVA'14 volume 8837 of LNCS, pp. 219–234. Springer (2014)
42. Keiren, J.J.A., Willemse, T.A.C.: Bisimulation Minimisations for Boolean Equation Systems. In: Proceedings of HVC'09, volume 6405 of LNCS. Springer (2011)
43. Mayr, R., Clemente, L.: Advanced automata minimization. In: POPL'13, pp. 63–74. ACM (2013)
44. McNaughton, R.: Infinite games played on finite graphs. Ann. Pure Appl. Log. **65**(2), 149–184 (1993)
45. Namjoshi, K.S.: A simple characterization of stuttering bisimulation. In: Proceedings of FSTTCS'97, volume 1346 of LNCS, pp. 284–296. Springer (1997)
46. Orzan, S., Wesselink, J.W., Willemse, T.A.C.: Static analysis techniques for parameterised Boolean equation systems. In: Proceedings of TACAS'09, volume 5505 of LNCS, pp. 230–245. Springer (2009)
47. Orzan, S., Willemse, T.A.C.: Invariants for parameterised boolean equation systems. Theor. Comput. Sci. **411**(11–13), 1338–1371 (2010)
48. Petersson, V., Vorobyov, S.G.: A randomized subexponential algorithm for parity games. Nordic J. Comput. **8**(3), 324–345 (2001)

49. Schewe, S.: Solving parity games in big steps. In: Proceedings of FSTTCS'07, volume 4855 of LNCS, pp. 449–460. Springer (2007)
50. Stevens, P., Stirling, C.: Practical model checking using games. In: Proceedings of TACAS'98, volume 1384 of LNCS, pp. 85–101. Springer (1998)
51. Stirling, C.: Bisimulation, modal logic and model checking games. Log. J. IGPL **7**(1), 103–124 (1999)
52. Thomas, W.: On the Ehrenfeucht-Fraïssé game in theoretical computer science. In: Proceedings of TAPSOFT'93, volume 668 of LNCS, pp. 559–568. Springer (1993)
53. Vöge, J., Jurdziński, M.: A discrete strategy improvement algorithm for solving parity games. In: Proceedings of CAV'00, volume 1855 of LNCS, pp. 202–215. Springer (2000)
54. Willemse, T.A.C., Consistent correlations for parameterised Boolean equation systems with applications in correctness proofs for manipulations. In: Proceedings of CONCUR'10, volume 6269 of LNCS, pp. 584–598. Springer (2010)
55. Yin, Q., Fu, Y., He, C., Huang, M.,Tao, X.: Branching bisimilarity checking for PRS. In :Proceedings of ICALP'14, volume 8573 of LNCS, pp. 363–374. Springer (2014)
56. Zielonka, W.: Infinite games on finitely coloured graphs with applications to automata on infinite trees. Theor. Comput. Sci. **200**(1–2), 135–183 (1998)