

Parity Game Reductions

Sjoerd Cranen¹, Jeroen J.A. Keiren^{2,3}, and Tim A.C. Willemse¹

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

s.cranen, t.a.c.willemse@tue.nl

²*Open University of the Netherlands, Faculty of Management, Science & Technology, PO Box 2960, 6401 DL Heerlen, The Netherlands, Jeroen.Keiren@ou.nl*

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

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 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 [17, 47, 48] and synthesis and supervisory control algorithms [3, 4, 20].

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 [33] 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., [7, 8, 34, 35, 41, 45, 46, 47, 49, 52], 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 [19, 30, 31]. 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.* [36].

Given the close connection between parity game solving and model checking, a promising heuristic in this setting is to 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 [14, 39] 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, 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 [14, 15] were conducted using the Groote-Vaandrager algorithm [28] 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 [29], 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* [37] (also known as *idempotence-identifying bisimilarity* in [39] and *governed stuttering bisimilarity* [15], which weaken strong bisimilarity and stuttering equivalence, respectively, by taking the potentials of players into account. Quotienting for the latter 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.* [32], 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 [14, 15, 37, 39]. We give formal proofs 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.* [5], and the added complexity of working in a setting with two players complicates matters significantly.

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 simula-*

tion and *delayed simulation* [22] 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 [42], 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 [9, 51], 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.

Related work. In logic, bisimulation has been used to characterise the subfamily of first-order logic that is definable in modal logic [6], and which fragment of monadic second-order logic is captured by the modal μ -calculus. Bisimulation and simulation-like relations, called *consistent correlations* [50] and *consistent consequence* [24] 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 [13, 38, 43, 44]. Various simulation relations have been used successfully for minimising Büchi automata, see *e.g.* [11, 18, 40].

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 [26, 27]. 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 [27] for a comprehensive overview. Typically, these definitions rely on the transitive closure of the transition relation. As argued by Namjoshi [42], local reasoning 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 [22], 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.* describe branching bisimulation games for normed process algebra [51]. A game-based characterisation of divergence-blind stuttering bisimulation was provided by Bulychev *et al.* [9]. 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. For governed stuttering bisimulation, it is unclear how the parity game should be preprocessed, so instead we incorporate divergence into the game-based definition as a first-class citizen.

Structure of the paper. Parity games are introduced in Section 2. In Section 3, we introduce notation that facilitates us to define preorders and equival-

ences 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 Section 4. In Section 5 we study direct simulation, delayed simulation and its biased versions and in Section 6 we study governed bisimulation and governed stuttering bisimulation. Quotienting, for all involved equivalences that admit unique quotients, is discussed in Section 7 and in Section 8 we return to, and substantiate, the overview we presented in Section 4. We wrap up with conclusions and an outlook for future work in Section 9.

2 Parity Games

A parity game is a two-player graph game, played by two players *even* and *odd* (denoted \diamond and \square) 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 game is formally defined as follows.

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

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

Instead of $(v, w) \in \rightarrow$ we typically write $v \rightarrow w$, and we write v^\bullet for the set $\{w \in V \mid v \rightarrow w\}$. If i is a player, then $\neg i$ denotes the opponent of i , *i.e.*, $\neg \diamond = \square$ and $\neg \square = \diamond$. 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_\square owned by player *odd*; we use \mathcal{P} and V_\diamond, V_\square interchangeably. The *reward order* on natural numbers is defined such that $n \preceq m$ 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 $v_m \rightarrow v_{m+1}$ 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 j^{th} 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:

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

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 \square 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 consistency 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:

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

The definition for infinite paths is generalised in the same way and denoted $\Pi_\sigma^\omega(v)$. By $\Pi_\sigma(v)$ we denote $\Pi_\sigma^\omega(v) \cup \bigcup_{n \in \mathbb{N}} \Pi_\sigma^n(v)$, *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 path allowed by σ . A vertex is won by player i if i has a winning strategy from that vertex.

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 [16]). *For every parity game there is a unique partition (W_\diamond, W_\square) such that winning strategies $\sigma_\diamond \in \mathbb{S}_\diamond^*$ from W_\diamond*

and $\sigma_\square \in \mathbb{S}_\square^*$ from W_\square exist. Furthermore, 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_\square) 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 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 $(V, \rightarrow, \Omega, \mathcal{P})$. 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 $\sigma \rightarrow \subseteq \rightarrow$ that contains only those edges allowed by σ :

$$\sigma \rightarrow \triangleq \{(v, u) \mid (v, u) \in \rightarrow \text{ and if } \sigma(v) \text{ is defined } \sigma(v) = u\}$$

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

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 this 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 $\sigma \models p$, 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:

$$v_{i \mapsto_U} T \triangleq \exists \sigma \in \mathbb{S}_i : v_{\sigma \mapsto_U} T \qquad v_{i \mapsto_U} \triangleq \exists \sigma \in \mathbb{S}_i : v_{\sigma \mapsto_U}$$

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 $u \in T$, replace all outgoing edges by a single edge $u \rightarrow u$.
- set the priorities for all $u \in T$ to even iff $i = \diamond$ and the priorities of all other vertices to odd iff $i = \diamond$.

Since $v \xrightarrow{\sigma} U \setminus T$, vertex v is won by i in the resulting graph. As parity games are memoryless determined, 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 $v \xrightarrow{\sigma'} U \setminus T$ in the resulting graph, but then also $v \xrightarrow{\sigma'} U \setminus T$ in the original graph, and hence also the required $v \xrightarrow{\sigma'} U$. \square

The complement of these relations is denoted by a slashed version of the corresponding arrow, *e.g.*, $\neg v \xrightarrow{i} U$ can be written $v \not\xrightarrow{i} U$. 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

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

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 $v \xrightarrow{i} U$ is closely related to the notion of *attractor sets* [41]. To formalise this correspondence, we generalise the attractor set definition along the lines of the generalisation used for the computation of the *Until* in the alternating-time temporal logic ATL [2].

Definition 2 (Attractor set). We define ${}_{U}Attr_i(T)$ as ${}_{U}Attr_i^\omega(T)$ where:

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

The attractor set as defined in [41] is obtained for $U = V$. In essence, the attractor set ${}_{U}Attr_i(T)$ 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. Note that $Attr$ is a monotone operator; *i.e.* for $T \subseteq T'$ we have ${}_{U}Attr_i(T) \subseteq {}_{U}Attr_i(T')$. 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 $v \xrightarrow{i} U$ iff $v \in {}_{U}Attr_i(T)$.*

Proof. We first introduce some additional notation. Let $v \xrightarrow{i}^n U$ 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 $v \xrightarrow{i} U$ iff $v \xrightarrow{i}^n U$. We can now use induction to prove $u \xrightarrow{i} U$ iff $u \in {}_{U}Attr_i^n(T)$ for all u . The required property then follows.

The base case, $n = 0$, follows instantly. For $n = m + 1$, our induction hypothesis yields $u \xrightarrow{i}^m U$ iff $u \in {}_{U}Attr_i^m(T)$ for all u . We distinguish two cases: $\mathcal{P}(u) = i$ and $\mathcal{P}(u) \neq i$.

Suppose $\mathcal{P}(u) = i$, and assume $u \in {}_{U}Attr_i^{n+1}(T)$. Then, by definition, $u \in {}_{U}Attr_i^n(T)$, or, since $\mathcal{P}(u) = i$, $u \in \{v \in U \mid \exists v' \in v^\bullet : v' \in {}_{U}Attr_i^n(T)\}$. But then this is equivalent to

$$u \in {}_{U}Attr_i^n(T) \text{ or } v' \in {}_{U}Attr_i^n(T) \text{ for some } v' \in u^\bullet. \quad (*)$$

By our induction hypothesis, $(*)$ is equivalent to $u \dot{\mapsto}_U^n T$ or $v' \dot{\mapsto}_U^n T$ for some $v' \in u^\bullet$. But since i then has a strategy to play $u \rightarrow v'$, this is again equivalent to

$$u \dot{\mapsto}_U^n T \text{ or } u \dot{\mapsto}_U^{n+1} T \quad (\dagger)$$

Since $u \dot{\mapsto}_U^n T$ implies $u \dot{\mapsto}_U^{n+1} T$ and $u \dot{\mapsto}_U^{n+1} T$ implies $u \dot{\mapsto}_U^n T$ or $u \dot{\mapsto}_U^{n+1} T$, we find that (\dagger) is equivalent to the desired $u \dot{\mapsto}_U^{n+1} T$, which finishes the proof for the case $\mathcal{P}(u) = i$. The case for $\mathcal{P}(u) \neq i$ uses a similar line of reasoning; the only difference is that we do not need to identify a strategy for player i . \square

We are now ready to formalise some intuitions using our notation. One of the most basic properties we expect to hold is that a player can force the play towards some given set of vertices, or otherwise her opponent can force the play to the complement of that set.

In the following lemmas, let $v \in V$, $U, T, T' \subseteq V$ and i a player.

Lemma 3. $v \dot{\mapsto}_U T \vee v \dot{\mapsto}_U V \setminus T$.

Proof. We prove the equivalent $v \dot{\not\mapsto}_U T \implies v \dot{\mapsto}_U V \setminus T$. Assume that $v \dot{\not\mapsto}_U T$. Observe that it follows directly that $v \notin T$, and hence $v \in V \setminus T$. Therefore, we immediately find that $v \dot{\mapsto}_U V \setminus T$. \square

In a similar train of thought, we expect that if from a single vertex, each player can force play towards some target set, then the players' target sets must overlap.

Lemma 4. $v \dot{\mapsto}_U T \wedge v \dot{\mapsto}_U T' \implies \exists u \in T, u' \in T' : u = u' \vee u \in U \vee u' \in U$.

Proof. Assume $v \dot{\mapsto}_U T \wedge v \dot{\mapsto}_U T'$. Then there must be strategies $\sigma \in \mathbb{S}_i$ and $\sigma' \in \mathbb{S}_{\neg i}$ such that $v_{\sigma \mapsto U} T \wedge v_{\sigma' \mapsto U} T'$. Let σ and σ' be such, and consider a play p such that $\sigma \Vdash p$ and $\sigma' \Vdash p$. For this play, there must be m and n such that $p[m] \in T \wedge \forall j < m : p[j] \in U$, and $p[n] \in T' \wedge \forall j < n : p[j] \in U$. If $m = n$, then this witnesses $\exists u \in T, u' \in T' : u = u'$. If $m < n$, then $p[m] \in T \wedge p[m] \in U$, and if $n < m$, then $p[n] \in T' \wedge p[n] \in U$. \square

The above lemmata reason about players being able to reach sets of vertices. The following 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 5. $v \dot{\mapsto}_U \iff v \dot{\not\mapsto}_U V \setminus U$

Proof. Again, note that the truth values of $v \dot{\mapsto}_U$ and $v \dot{\not\mapsto}_U V \setminus U$ only depend on edges that originate in U , and that these truth values do not depend on priorities at all. Therefore, the truth value of these predicates will not change if we apply the following transformations to our graph:

- For all $u \in V \setminus U$, replace all outgoing edges by a single edge $u \rightarrow u$.
- Make the priorities of all vertices in U such that they are even iff $i = \diamond$, and the priorities of all other vertices odd iff $i = \diamond$.

In the resulting graph, player i wins if and only if $v \dot{\mapsto}_U$, and player $\neg i$ wins if and only if $v \dot{\not\mapsto}_U V \setminus U$. Since parity games are determined, *i.e.* v can only be won by one player, the desired result follows. \square

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 6. $(v \mapsto_U T \wedge \forall u \in T : u \mapsto_U T') \implies v \mapsto_U T'$

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

Finally, we state two results that relate a player's capabilities to reach a set of vertices to the capabilities of the vertices that are able to leave a set of vertices in a single step.

Lemma 7. *Let $S = \{u \in U \mid u^\bullet \cap T \neq \emptyset\}$ and $v \notin T$. Then $v \mapsto_U T$ implies $V_i \cap S \neq \emptyset$ or for some $u \in S$, $u^\bullet \subseteq T$.*

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

Lemma 8. *For $v \in U$, $v \mapsto_U T$ implies $v \mapsto_U T'$ whenever $T' \subseteq T$ 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 $p[k] \in U$ 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 $v \mapsto_U T'$. \square

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.

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 $V' \subseteq V$, we write $[V']_R$ for the set of equivalence classes with respect to V , *i.e.* the set $\{[v]_R \mid v \in V'\}$. The set of equivalence classes of V under R is denoted V/R , and defined as $\{[v]_R \mid v \in V\}$.

4.2 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 $\phi: V \rightarrow V$ 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 only and exactly 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 Figure 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 (\rightleftharpoons) [37, 39], see Section 6.1;
- Strong direct simulation equivalence (\equiv_{sd}), see Section 5.1;
- Direct simulation equivalence (\equiv_d) [22, 23, 24], see Section 5.1;
- Delayed simulation equivalence (\equiv_{de}) [22], see Section 5.2;
- Delayed simulation equivalence, *even*-biased (\equiv_{de}^e) [22], see Section 5.2.1;
- Delayed simulation equivalence, *odd*-biased (\equiv_{de}^o) [22], see Section 5.2.1;
- Governed bisimilarity (\rightleftharpoons^g) [37, 39], see Section 6.1;
- Stuttering bisimilarity (\simeq) [14], see Section 6.1;
- Governed stuttering bisimilarity (\simeq^g) [15], see Section 6.2;

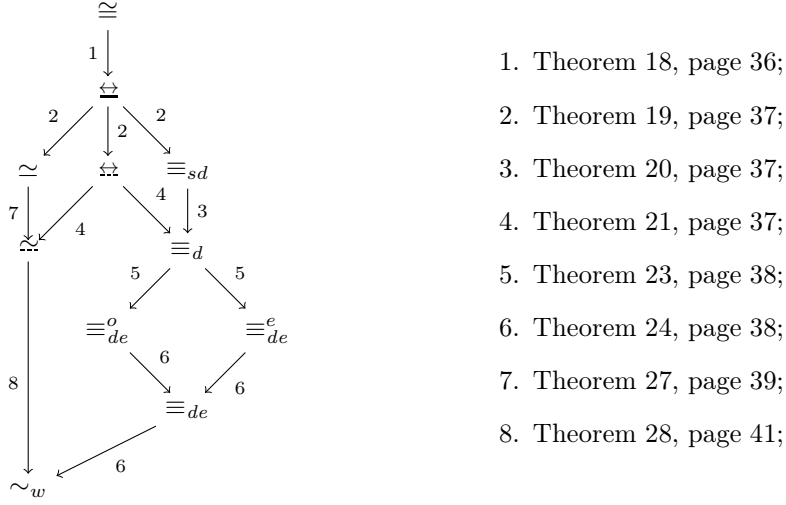


Figure 1: Lattice of equivalences for parity games. The numbers on the edges refer to the legend shown to the right, which in turn refers to the theorems that witness the existence of the edge.

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 Section 5.1. In Section 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 that $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$ is an arbitrary parity game.

5.1 Direct Simulation and Direct Simulation Equivalence

Direct simulation for parity games 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) in [22] and appeared earlier in the setting of alternating Büchi automata [23]. We here follow the game-based definition as given in [24].

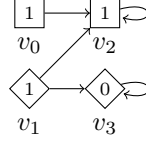


Figure 2: Parity game with $v_0 \sqsubseteq_d v_1$, $v_0 \sqsubseteq_d v_2$, $v_2 \sqsubseteq_d v_0$, and for all v_i , $v_i \sqsubseteq_d v_i$.

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) according to the rules in Table 1;
2. 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 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.

$\mathcal{P}(v)$	$\mathcal{P}(w)$	1 st move	plays on	2 nd move	plays on
\diamond	\diamond	S	v	D	w
\diamond	\square	S	v	S	w
\square	\diamond	D	w	D	v
\square	\square	S	w	D	v

Table 1: Allowed moves in a (bi)simulation game.

Example 1. In the parity game in Figure 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 $v_1 \rightarrow v_3$ cannot be matched from v_0 . Note that additionally we have $v_0 \sqsubseteq_d v_2$ and $v_2 \sqsubseteq_d v_0$.

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 automata, direct simulation was shown to be transitive using strategy composition, see [21, 23]. 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 [22, 24]). 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 [25], where it is also referred to as *governed simulation*.

Definition 7 (Direct simulation relation [25]). 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 $v' \in v^\bullet$, $w_\Diamond \rightarrow v' R$;
- if $v \in V_\Box$, then $w_\Diamond \rightarrow v^\bullet R$.

We say that vertex v is directly simulated by w , denoted $v \leq_d w$, if and only if there is a direct simulation relation R such that $v R w$.

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 $v \sqsubseteq_d w$.*

Proof. We prove both implications separately.

- \Rightarrow We prove that from a pair of vertices $v \leq_d w$ the game can always be played such that we again end up in related vertices after one round in the direct simulation game. Since $\Omega(v) = \Omega(w)$, it immediately follows that *Duplicator* has a winning strategy in the direct simulation game, and hence $v \sqsubseteq_d w$.

Let v, w be such that $v \leq_d w$. We distinguish two cases:

- $v \in V_\Diamond$. Then *Spoiler* first chooses some $v \rightarrow v'$. Since $v \leq_d w$, we know that $w_\Diamond \rightarrow v' \leq_d$, hence if $w \in V_\Diamond$, *Duplicator* can choose a successor $w \rightarrow w'$ such that $v' \leq_d w'$, and if $w \in V_\Box$, all successors $w \rightarrow w'$ that *Spoiler* may choose are such that $v' \leq_d w'$.
- $v \in V_\Box$. Then $w_\Diamond \rightarrow v^\bullet \leq_d$, so if $w \in V_\Diamond$, there exist $w \rightarrow w'$ and $v \rightarrow v'$ such that $v' \leq_d w'$, and *Duplicator* can play such that he picks those. In case $w \in V_\Box$, then for all $w \rightarrow w'$ there exists a $v \rightarrow v'$ such that $v' \leq_d w'$. Since *Spoiler* plays first on w , *Duplicator* can match with the appropriate $v \rightarrow v'$.

- \Leftarrow We prove that \sqsubseteq_d is a direct simulation relation.

Let v, w be arbitrary such that $v \sqsubseteq_d w$. Consider a winning strategy for *Duplicator* in the direct simulation game from (v, w) . Observe that $\Omega(v) = \Omega(w)$ follows trivially. We again distinguish two cases:

- $v \in V_\Diamond$. Then *Spoiler* first chooses some $v \rightarrow v'$. If $w \in V_\Diamond$, then *Duplicator* matches this with some $w \rightarrow w'$ according to his strategy. Since *Duplicator*'s strategy is winning from (v, w) , it is also winning from (v', w') , hence $v' \sqsubseteq_d w'$. If $w \in V_\Box$, any choice of successor $w \rightarrow w'$ that *Spoiler* makes is such that $v' \sqsubseteq_d w'$. In both cases we have shown $w_\Diamond \rightarrow v' \sqsubseteq_d$.
- $v \in V_\Box$. If $w \in V_\Diamond$, *Duplicator* plays both on v , say $v \rightarrow v'$, and on w , say $w \rightarrow w'$, such that $v' \sqsubseteq_d w'$. If $w \in V_\Box$, *Spoiler* plays first on w , say $w \rightarrow w'$, and given this choice, *Duplicator* responds according to his strategy with $v \rightarrow v'$ such that $v' \sqsubseteq_d w'$. Since *Duplicator*'s strategy is winning, he is able to match any choice made by *Spoiler*. In both cases the strategy witnesses $w_\Diamond \rightarrow v^\bullet \sqsubseteq_d$. \square

Given that both definitions of direct simulation coincide, we can now say that v and w are *direct simulation equivalent*, i.e. $v \equiv_d w$, if and only if $v \leq_d w$ and $w \leq_d v$.

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

Proof. 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. \square

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 a notion that resembles *alternating refinement* [1]. 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 Figure 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.* [18], in [22], 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. These configurations are now extended with a third parameter which is used to keep track of the obligation that still needs to be met by *Duplicator*.

An obligation in a delayed simulation game is either a natural number or the symbol \checkmark ; the latter is used to indicate the absence of obligations. We denote the set of obligations by K . Given two priorities and an existing obligation, a new obligation is obtained using the function $\gamma: \mathbb{N} \times \mathbb{N} \times K \rightarrow K$, where:

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

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

By abuse of notation we will typically write $\gamma(v, w, k)$, for vertices $v, w \in V$ and obligation $k \in K$, to denote $\gamma(\Omega(v), \Omega(w), k)$.

The intuition behind this update is as follows. Either, with the new priorities, the pending obligation is fulfilled, and the new configuration does not give rise to a new obligation, in which case the result is \checkmark . Otherwise the new obligation is the minimum of the priorities passed to the function, or the current

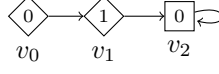


Figure 3: Parity game in which all vertices are delay simulation equivalent.

obligation. This signifies the most significant obligation 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, or the second argument is such that it matches the pending obligation.

Definition 8 (Delayed simulation game [22]). 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:

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

An infinite play $(v_0, w_0, k_0), (v_1, w_1, k_1), \dots$ 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 Figure 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 [21]; 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 [22]). 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 [42]. 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 $<$ 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 $v' \in v^{\bullet}$, $w_{\Diamond} \rightarrow \{w' \in V \mid \ell = \gamma(v', w', k) \wedge v' R^{\ell} w' \wedge (k = \checkmark \vee (v', w', \ell) < (v, w, k))\}$;
- $v \in V_{\Box}$ implies $w_{\Diamond} \rightarrow \{w' \in V \mid \exists v' \in v^{\bullet} : \ell = \gamma(v', w', k) \wedge v' R^{\ell} w' \wedge (k = \checkmark \vee (v', w', \ell) < (v, w, k))\}$.

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.

Lemma 9. 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 $v \rightarrow v'$. By definition of the well-founded delayed simulation, there is $w \rightarrow w'$ 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 $v \rightarrow v'$ and $w \rightarrow w'$, $v' R^{\gamma(v', w', k)} w'$.

- $(v, w) \in V_{\square} \times V_{\diamond}$. *Duplicator* plays twice in the delayed simulation game. According to the well-founded delayed simulation, there exist $v \rightarrow v'$ and $w \rightarrow w'$ such that $v' R^{\gamma(v', w', k)} w'$. *Duplicator* plays such moves.
- $(v, w) \in V_{\square} \times V_{\square}$. In the delayed simulation game, *Spoiler* is to move first, say $w \rightarrow w'$. From the well-founded delayed simulation, we find that for all such moves, there exists some $v \rightarrow v'$ 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 $(v', w', k') < (v, w, k)$. Since $<$ is a well-founded order, this can only be repeated finitely many times, and eventually all obligations are met. \square

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*.

Lemma 10. *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 play 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 $<$ when we set $(v', w', \ell) < (v, w, k)$ iff the height of the tree rooted in (v', w', ℓ) is less than that of the tree rooted in (v, w, k) . \square

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 $(v', w', \ell) < (v, w, k)$.*

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

Lemma 11. *For $v, w \in V$, $v \sqsubseteq_{de} w$ implies $v \leq_{de} w$.*

Proof. Let $R \subseteq V \times K \times V$ be such that $v R^k w$ if *Duplicator* wins the delay simulation game from (v, w) with obligation k . Observe that, in particular, since $v \sqsubseteq_{de} w$, we have $v R^{\gamma(v, w, \checkmark)} w$. We show that R is a well-founded delayed simulation.

For configurations (v, w) with obligation k , *Duplicator* has a winning strategy. Since the delay simulation game has a Büchi winning condition, *Duplicator* also has a memoryless winning strategy. For the remainder of the proof, fix this strategy. Observe that a well-founded order on configurations won by *Duplicator* exists, according to Lemma 10.

Next we show that this relation satisfies the transfer conditions. Let v, w, k be arbitrary, such that $v R^k w$. We distinguish four cases.

- $(v, w) \in V_{\diamond} \times V_{\diamond}$. According to Definition 8, for all moves $v \rightarrow v'$ made by *Spoiler*, *Duplicator* matches with a move $w \rightarrow w'$, and the game continues from configuration (v', w') with obligation $\ell = \gamma(v', w', k)$. Since *Duplicator*'s strategy is winning from (v, w) with obligation k , the strategy is also winning from (v', w') with obligation ℓ hence $v' R^{\ell} w'$. Furthermore, if $k \neq \checkmark$ we have $(v', w', \ell) < (v, w, k)$ according to Corollary 1.
- $(v, w) \in V_{\diamond} \times V_{\square}$. For all moves $v \rightarrow v'$ and $w \rightarrow w'$ made by *Spoiler*, the game continues from configuration (v', w') with obligation $\ell = \gamma(v', w', k)$ from which again *Duplicator*'s strategy is winning, hence $v' R^{\ell} w'$, and again we have $(v', w', \ell) < (v, w, k)$ if $k \neq \checkmark$ according to Corollary 1.
- $(v, w) \in V_{\square} \times V_{\diamond}$. *Duplicator* plays a move $w \rightarrow w'$ and $v \rightarrow v'$ and continues from (v', w') with obligation $\ell = \gamma(v', w', k)$, from which her strategy is again winning, hence $v' R^{\ell} w'$. Using the same argument as before, also $(v', w', \ell) < (v, w, k)$ if $k \neq \checkmark$ according to Corollary 1.
- $(v, w) \in V_{\square} \times V_{\square}$. For all moves $w \rightarrow w'$ made by *Spoiler*, *Duplicator*'s strategy matches with some $v \rightarrow v'$ such that *Duplicator* wins from (v', w') with obligation $\ell = \gamma(v', w', k)$. Again $(v', w', \ell) < (v, w, k)$ if $k \neq \checkmark$, hence $w \rightarrow_{\diamond} \{w' \in V \mid \exists v \rightarrow v' : \ell = \gamma(v', w', k) \wedge v' R^{\ell} w' \wedge (k = \checkmark \vee (v', w', \ell) < (v, w, k))\}$.

In each of the cases above, the requirements for well-founded delayed simulation are satisfied, hence R is a well-founded delayed simulation relation, and $v \leq_{de} w$. \square

The following theorem, stating that delayed simulation and well-founded delay 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 9 and 11. \square

5.2.1 Biased delayed simulations.

As observed in [22], 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 $\gamma^e: \mathbb{N} \times \mathbb{N} \times K \rightarrow K$, where:

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

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 \checkmark 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 $\gamma^e(v, w, \checkmark)$ 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 $v R^{\gamma^e(v, w, \checkmark)} w$.

Odd-biased delayed simulation. 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 $\gamma^o: \Omega \times \Omega \times K \rightarrow K$, where:

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

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 Section 6.1, we introduce governed bisimulation, which was studied under various guises in *e.g.* [24, 37, 39]. Governed bisimulation is, as we demonstrate in that section, closely related to direct simulation. Next, in Section 6.2, governed stuttering bisimulation [12, 15, 37] is introduced.

6.1 Governed bisimulation

Our definition of governed bisimulation, as presented below, is based on the one from [39] where it is defined in the closely related setting of *Boolean equation*

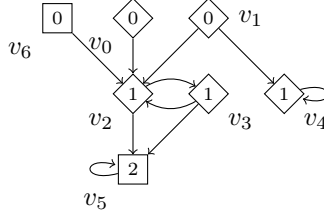


Figure 4: Parity game in which v_2 and v_3 are governed bisimilar. Vertices v_0 , v_1 and v_6 are direct simulation equivalent. Vertices v_0 and v_6 are governed bisimilar but not strong direct simulation equivalent. Vertices v_0 and v_1 are strong direct simulation equivalent, but not governed bisimilar.

systems; because of its capabilities to relate *conjunctive* and *disjunctive* equations, it was dubbed *idempotence identifying bisimulation*. It was rephrased for parity games in [37] 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 $v \dot{\sim} w$, if and only if there is a governed bisimulation R such that $v R w$.

Example 3. In the parity game in Figure 4, we have for all i , $v_i \dot{\sim} v_i$, and furthermore, $v_2 \dot{\sim} v_3$ and $v_0 \dot{\sim} v_6$. 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 $v_0 \dot{\sim} v_1$, since the latter would require v_4 to be related to v_2 , but from v_4 the step $v_2 \rightarrow v_5$ cannot be mimicked.

Governed bisimulation is such that vertices owned by different players can only be related whenever all their successors are. 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 theorem.

Theorem 4. We have $v \dot{\sim} w$ iff there is a symmetric direct simulation relation R such that $v R w$.

Proof. We prove both implications separately.

\Rightarrow Let R be a governed bisimulation, and let v, w be arbitrary such that $v R w$. Since R is symmetric, it suffices to show that R is a direct simulation.

- $\Omega(v) = \Omega(w)$ follows immediately.

- Suppose $v \in V_\diamond$. Let $v' \in v^\bullet$. In case $w \in V_\square$ we have $v' R w'$ for all $w' \in w^\bullet$. If $w \in V_\diamond$, there is some $w' \in w^\bullet$ such that $v' R w'$. Both cases lead to the desired $w_\diamond \rightarrow v' R$.
- Suppose $v \in V_\square$. We again distinguish two cases. In case $w \in V_\diamond$ we have for all $v' \in v^\bullet$ and $w' \in w^\bullet$, $v' R w'$, i.e. $w_\diamond \rightarrow v^\bullet R$. Suppose $w \in V_\square$. Pick an arbitrary $w' \in w^\bullet$. Since R is symmetric, also $w R v$. Hence, there exists a $v' \in v^\bullet$ such that $w' R v'$, which implies $v' R w'$. Thus, both cases lead to $w_\diamond \rightarrow v^\bullet R$.

\Leftarrow Let R be a symmetric direct simulation relation. Pick arbitrary v, w for which $v R w$.

- $\Omega(v) = \Omega(w)$ follows immediately.
- Suppose $\mathcal{P}(v) \neq \mathcal{P}(w)$. Because R is symmetric, we may assume, without loss of generality, that $v \in V_\diamond$ and $w \in V_\square$. Pick an arbitrary $v' \in v^\bullet$. Since R is a direct simulation, we find $w_\diamond \rightarrow v' R$. As $w \in V_\square$, we thus find that $v' R w'$ for all $w' \in w^\bullet$.
- Let $v' \in v^\bullet$. By the previous case, it suffices to consider only the case that $\mathcal{P}(v) = \mathcal{P}(w)$. Suppose $v, w \in V_\diamond$. Then $w_\diamond \rightarrow v' R$; i.e. there is some $w' \in w^\bullet$ such that $v' R w'$. Now assume $v, w \in V_\square$. By symmetry, we have $w R v$. Then we have $v_\diamond \rightarrow w^\bullet R$. Thus, for every $v'' \in w^\bullet$ there is some $w' \in w^\bullet$ such that $w' R v''$. In particular, we have $w' R v'$ for some $w' \in w^\bullet$. By symmetry, we then also have $v' R w'$ for some $w' \in w^\bullet$. \square

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

Theorem 5. \equiv is an equivalence relation on parity games.

Proof. Follows from combining Theorem 4 and Proposition 1. \square

Additionally, we immediately obtain a game-based definition for governed bisimulation: we only need to require that *Spoiler* can switch to a symmetric position in the game play.

Definition 12 (Governed bisimulation game). The *governed bisimulation 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. *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 governed bisimulation game.

Theorem 6. For all $v, w \in V$, we have $v \equiv w$ if and only if $v \equiv_g w$.

Proof. Along the lines of the proof of Theorem 2. \square

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* [37]. The derived notion of *strong bisimilarity*, denoted $v \dot{\approx} w$ and defined as $v \dot{\approx} w$ 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 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 are concerned with player’s capabilities. 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.

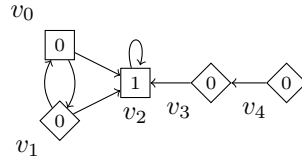


Figure 5: Equal priorities are related by \approx . Neither player can force play to visit only vertices with priority 0.

The definition of governed stuttering bisimulation presented below is based

on [12, 15]. For our definition, we strongly rely on our notation to denote that a player is able to ‘force play’.

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) $v \rightarrow \mathcal{C}$ implies $w \mathcal{P}(v) \mapsto_R \mathcal{C}$, for all $\mathcal{C} \in V/R \setminus \{[v]_R\}$.
- c) $v \mapsto_R i$ implies $w \mapsto_R i$ for $i \in \{\Diamond, \Box\}$.

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

Example 4. The parity game in Figure 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 $v_j \Diamond \mapsto_{\approx}$ nor $v_j \Box \mapsto_{\approx}$. 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 \approx . 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 governed stuttering bisimilarity is an equivalence relation that is again 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 7.

Lemma 12. *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 $v \rightarrow \mathcal{U}$ for some $\mathcal{U} \subseteq V/R \setminus \{[v]_R\}$. Suppose $u \rightarrow \mathcal{C}$ for some $\mathcal{C} \notin \mathcal{U} \cup \{[v]_R\}$. Since $v \approx u$, by Definition 13, we have $v \mathcal{P}(u) \mapsto_R \mathcal{C}$. But $v^\bullet \subseteq \bigcup \mathcal{U}$ and $\mathcal{C} \notin \mathcal{U}$ so $v \mathcal{P}(u) \not\mapsto_R \mathcal{C}$. Contradiction. \square

Theorem 7. *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) $v \mapsto_R \mathcal{C}$ iff $w \mapsto_R \mathcal{C}$ for all $i \in \{\Diamond, \Box\}, \mathcal{C} \in V/R \setminus \{[v]_R\}$;
- c) $v \mapsto_R i$ iff $w \mapsto_R i$ 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 $v \mapsto_R \mathcal{C}$ 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 $u \rightarrow \mathcal{C}$, $w \mapsto_R \mathcal{C}$ follows from Definition 13.
- Case $S \subseteq V_{\neg i}$. By Lemma 7, there is a $u \in S$ for which $u^\bullet \subseteq \mathcal{C}$ and by Lemma 12 (for $\mathcal{U} = \{\mathcal{C}\}$), $u^\bullet \setminus [v]_R \subseteq \mathcal{C}$ for all $u \in [v]_R$. Furthermore, by Lemma 5, $v \mapsto_R \mathcal{C}$ implies $v \not\mapsto_R$. Then, by Definition 13, $w \not\mapsto_R$ and by Lemma 5, $w \mapsto_R V \setminus [v]_R$. But since $u^\bullet \subseteq \mathcal{C} \cup [v]_R$ for all u , the desired $w \mapsto_R \mathcal{C}$ follows from Lemma 8. \square

While the above alternative characterisation of governed stuttering bisimulation is now fully symmetric, the restriction on the class \mathcal{C} that is considered in clause *b*) turns out to be too strong to facilitate an insightful proof that \approx is an equivalence relation. We therefore further generalise this clause such that it is phrased in terms of *sets* of classes.

A perhaps surprising side-result of this generalisation is that the divergence requirement of clause *c*) becomes superfluous. Note that this generalisation is not trivial, as $v \mapsto_R \{\mathcal{C}_1, \mathcal{C}_2\}$ is in general neither equivalent to saying that $v \mapsto_R \mathcal{C}_1$ and $v \mapsto_R \mathcal{C}_2$, nor $v \mapsto_R \mathcal{C}_1$ or $v \mapsto_R \mathcal{C}_2$.

Theorem 8. *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) $v \mapsto_R \mathcal{U}$ iff $w \mapsto_R \mathcal{U}$ for all $i \in \{\diamond, \square\}, \mathcal{U} \subseteq V_{/R} \setminus \{[v]_R\}$.

Proof. We show that the second condition is equivalent to the conjunction of the last two conditions in Theorem 7. We split the proof into an *if*-part and an *only-if*-part.

\Leftarrow The second condition from Theorem 7 is equivalent to the second condition above if we let \mathcal{U} range only over singleton sets (if $v \mapsto_R \mathcal{C}$, 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 5.

\Rightarrow Let R be a governed stuttering bisimulation and let $v, w \in V$ such that $v R w$. Assume that $v \mapsto_R \mathcal{U}$ for some $\mathcal{U} \subseteq V_{/R} \setminus \{[v]_R\}$. Let $S = \{u \in [v]_R \mid u \rightarrow \mathcal{U}\}$. By Lemma 7, either $S \cap V_i \neq \emptyset$ or there is some $u \in S$ such that $u^\bullet \subseteq \mathcal{U}$. We consider both cases separately.

- Case $S \cap V_i \neq \emptyset$. Pick some u from this set. There is a class $\mathcal{C} \in \mathcal{U}$ such that $u \rightarrow \mathcal{C}$ (in particular, $u \mapsto_R \mathcal{C}$ since $u \in V_i$). By Theorem 7 then also $w \mapsto_R \mathcal{C}$, from which $w \mapsto_R \mathcal{U}$ follows immediately.
- Case $S \cap V_i = \emptyset$. Then $u^\bullet \subseteq \mathcal{U}$ for some $u \in S$, but then, by Lemma 12, $t^\bullet \setminus [v]_R \subseteq \mathcal{U}$ for all $t \in [v]_R$. From $v \mapsto_R \mathcal{U}$ we derive, using Lemma 5, that not $v \not\mapsto_R$. By Theorem 7 it follows that not $w \not\mapsto_R$, and by Lemma 5 again $w \mapsto_R V \setminus [v]_R$. Since for all $t \in [v]_R$, $t^\bullet \subseteq [v]_R \cup \mathcal{U}$ and $\bigcup \mathcal{U} \subseteq V \setminus [v]_R$, by Lemma 8, also $w \mapsto_R \mathcal{U}$. \square

In the previous theorem, 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. This is still not sufficient

for easily proving transitivity of governed stuttering bisimulation. Therefore, in the theorem below, we introduce a final generalisation; 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.

Theorem 9. *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) $v \mapsto_{\mathcal{U}} \mathcal{T}$ iff $w \mapsto_{\mathcal{U}} \mathcal{T}$ 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 Theorem 8. We split the proof into an *if*-case and an *only-if*-part.

\Leftarrow The second condition from Theorem 8 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 $v \mapsto_{\mathcal{U}} \mathcal{T}$; under this assumption we will prove that $w \mapsto_{\mathcal{U}} \mathcal{T}$. The proof for the implication in the other direction is completely symmetric. Let $\sigma \in \mathbb{S}_i$ be such that $v \mapsto_{\sigma} \mathcal{T}$ 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 $v \mapsto \mathcal{T}$. In particular, $v \mapsto_R \mathcal{T}$ and by Theorem 8 also $w \mapsto_R \mathcal{T}$, which proves $w \mapsto_{\mathcal{U}} \mathcal{T}$.

As the induction hypothesis, assume that if $u R u'$, $u \mapsto_{\sigma} \mathcal{T}$ and the longest σ -prefix of u is shorter than the longest σ -prefix of v , then $u' \mapsto_{\mathcal{U}} \mathcal{T}$. 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 $u \mapsto_{\sigma} \mathcal{T}$. Furthermore, $v \mapsto_{\sigma} U$.

By Theorem 8, $w \mapsto_R [U]_R$. 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 $u \mapsto_{\sigma} \mathcal{T}$ for such u , we can use the induction hypothesis to derive that $u' \mapsto_{\mathcal{U}} \mathcal{T}$.

The above in particular implies two facts: $w \mapsto_{\mathcal{U}} [U]_R$, and $u' \mapsto_{\mathcal{U}} \mathcal{T}$ for all $u' \in \bigcup [U]_R$. Using these, we can now apply Lemma 6 to conclude $w \mapsto_{\mathcal{U}} \mathcal{T}$. \square

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 $(R \cup S)^*$ 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 Theorem 9 on $(R \cup S)^*$ with its effect on R and S .

Theorem 10. \approx 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 Theorem 9 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, \dots, u_n such that $v R u_0 S \dots R u_n S w$ (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 $v \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$ for some $i \in \{\diamond, \square\}$ 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 $w \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. 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 $v \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$, also $v \dot{\mapsto}_{\mathcal{U}_R} \mathcal{T}$, and by Theorem 9 then $u_0 \dot{\mapsto}_{\mathcal{U}_R} \mathcal{T}$, which is equivalent to $u_0 \dot{\mapsto}_{\mathcal{U}_S} \mathcal{T}$. By a simple inductive argument we now arrive at $w \dot{\mapsto}_{\mathcal{U}_S} \mathcal{T}$, which is equivalent to $w \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. \square

As a side-result of the proof of Theorem 10, 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 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:

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

where update γ is defined as follows:

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

An infinite play $((v_0, w_0), c_0), ((v_1, w_1), c_1), \dots$ is won by *Duplicator* if and only if $\Omega(v_j) = \Omega(w_j)$ 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 Figure 5. In this parity game, all vertices with priority 0 are related by \approx . 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 proving that the governed stuttering bisimulation game and governed stuttering bisimulation coincide. 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 3. *For all $v, w \in V$ if $v \approx w$ 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 $u_0 \approx u_1$ 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 $((v, w), \checkmark)$. \square

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 $\mathcal{F} : V \times V \rightarrow V \times V$ is defined as follows:

$$\mathcal{F}(R) = \{ (v, w) \in R \mid \Omega(v) = \Omega(w) \wedge \forall i \in \{\Diamond, \Box\}, \mathcal{U}, \mathcal{T} \subseteq V/R : \\ [v]_R \in \mathcal{U} \wedge [v]_R \notin \mathcal{T} \implies v \xrightarrow{i} \mathcal{U} \mathcal{T} \Leftrightarrow w \xrightarrow{i} \mathcal{U} \mathcal{T} \}$$

The predicate transformer \mathcal{F} has the following properties.

Lemma 13. $\mathcal{F}(R)$ is an equivalence relation for any equivalence relation R on V .

Proof. Let R be an equivalence relation over V . Reflexivity of $\mathcal{F}(R)$ follows from the fact that R is reflexive. Symmetry of $\mathcal{F}(R)$ follows from symmetry of R and from the fact that for all $(v, w) \in R$ we have $[v]_R = [w]_R$. For transitivity, we observe that for all pairs $(u, v), (v, w) \in \mathcal{F}(R)$ and all \mathcal{U}, \mathcal{T} for which $[u]_R \in \mathcal{U}$ and $[u]_R \notin \mathcal{T}$, if $u \xrightarrow{i} \mathcal{U} \mathcal{T}$ then also $v \xrightarrow{i} \mathcal{U} \mathcal{T}$. Since $[u]_R = [v]_R$, we immediately conclude $w \xrightarrow{i} \mathcal{U} \mathcal{T}$. The implication from right to left follows from symmetric arguments. Thus, $\mathcal{F}(R)$ is an equivalence relation. \square

Lemma 14. \mathcal{F} is a monotone operator on the complete lattice of equivalence relations on V .

Proof. By the previous lemma, it follows that \mathcal{F} is an operator on the lattice of equivalence relations on V . We next show that the operator is monotone. Let R, S be arbitrary equivalences on V . Suppose $R \subseteq S$, and consider some pair $(v, w) \in \mathcal{F}(R)$. From this, it follows that $\Omega(v) = \Omega(w)$. Let $i \in \{\Diamond, \Box\}, \mathcal{U}, \mathcal{T} \subseteq V/S$, such that $[v]_S \in \mathcal{U}$ and $[v]_S \notin \mathcal{T}$. Define $\bar{\mathcal{U}}$ as the set $\{[u]_R \mid [u]_S \in \mathcal{U}\}$ and define $\bar{\mathcal{T}}$ as the set $\{[u]_R \mid [u]_S \in \mathcal{T}\}$. Since $(v, w) \in \mathcal{F}(R)$, we have:

$$v \xrightarrow{i} \bar{\mathcal{U}} \bar{\mathcal{T}} \Leftrightarrow w \xrightarrow{i} \bar{\mathcal{U}} \bar{\mathcal{T}}$$

Since $\bigcup \bar{\mathcal{U}} = \bigcup \mathcal{U}$ and $\bigcup \bar{\mathcal{T}} = \bigcup \mathcal{T}$, we immediately have:

$$v \xrightarrow{i} \mathcal{U} \mathcal{T} \Leftrightarrow w \xrightarrow{i} \mathcal{U} \mathcal{T}$$

This proves that $(v, w) \in \mathcal{F}(S)$. Thus \mathcal{F} is a monotone operator. \square

Corollary 2. We have $\nu \mathcal{F} = \nu \mathcal{F}$.

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. \square

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

Proposition 4. *For all $v, w \in V$ if $v \equiv_{g,st} w$ then $v \approx w$.*

Proof. We essentially prove the contrapositive of the statement, *i.e.* for all $v, w \in V$, if $v \not\approx w$, then also $v \not\equiv_{g,st} w$. Let $v \not\approx w$. By Corollary 2, then also $(v, w) \notin \nu\mathcal{F}$. By the 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 $k \geq 1$:

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

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 \checkmark rewards along the play. \square

Propositions 3 and 4 lead to the following theorem.

Theorem 11. *For all $v, w \in V$ we have $v \approx w$ 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* [14]. 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 the equivalences defined on game graphs to equivalences 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 [21, 22], 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 [21]. We therefore only consider quotienting for direct simulation equivalence.

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 [10], 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).

While in the setting of Kripke structures, only transitions to *maximal* successor vertices must be retained, depending on the owner of the source vertex, 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 $(V, \rightarrow, \Omega, \mathcal{P})$ is the structure $(V_{/\equiv_d}, \rightarrow', \Omega', \mathcal{P}')$, where, for $\mathcal{C}, \mathcal{C}' \in V_{/\equiv_d}$:

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

Observe that it is not obvious that \rightarrow' is a total edge relation. The lemma below allows us to establish that this is the case.

Lemma 15. *Let $\mathcal{C} \in V_{/\equiv_d}$. Then:*

- If $\mathcal{C} \subseteq V_{\square}$ 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 $\mathcal{C} \not\subseteq V_{\square}$ and $\mathcal{C} \not\subseteq V_{\diamond}$ then for all $v \in \mathcal{C} \cap V_{\diamond}$ and $w \in \mathcal{C} \cap V_{\square}$ 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 $\mathcal{C} \subseteq V_{\square}$. 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 $w \in V_{\square}$. 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 can deduce $v' \in \max_{\leq_d}(v)$ and $|[\max_{\leq_d}(v)]_{\equiv_d}| = 1$.

We thus find $[\max_{\leq_d}(v)]_{\equiv_d} = \{[v']_{\equiv_d}\} = \{[w']_{\equiv_d}\} = [\min_{\leq_d}(w)]_{\equiv_d}$. \square

As a consequence of the above lemma, we obtain the following two results:

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

- if $\mathcal{C} \subseteq V_{\square}$ and for some $v \in \mathcal{C}$, $v' \in \mathcal{C}'$ also $v' \in \min_{\leq_d}(v)$, then $\mathcal{C} \rightarrow' \mathcal{C}'$.
- 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 $\mathcal{C} \rightarrow' \mathcal{C}'$.

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

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

Theorem 12. *Let $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$ be a parity game and $\mathcal{G}_q = (V_{/\equiv_d}, \rightarrow', \Omega', \mathcal{P}')$ its direct simulation quotient. Then $\mathcal{G}_q \equiv_d \mathcal{G}$.*

Proof. We first prove $\mathcal{G}_q \leq_d \mathcal{G}$. Let $H \subseteq V_{/\equiv_d} \times V$ be the relation $H = \{(\mathcal{C}, v) \mid \exists w \in \mathcal{C} : w \leq_d v\}$. We prove H is a direct simulation relation. Let \mathcal{C}, v be arbitrary such that $(\mathcal{C}, v) \in H$. By definition, there is some $w \in \mathcal{C}$ such that $w \leq_d v$. For the remainder of this proof, we fix such a w . Clearly, $\Omega'(\mathcal{C}) = \Omega(v)$ follows directly from $w \leq_d v$ and the fact that for all $u, u' \in \mathcal{C}$ we have $\Omega(u) = \Omega(u')$. We next prove the transfer condition. For this, we distinguish four cases.

- Case $\mathcal{P}(\mathcal{C}) = \square$ and $\mathcal{P}(v) = \square$. Let $v' \in v^\bullet$. Since $v' \in v^\bullet$, also $w' \leq_d v'$ for some $w' \in w^\bullet$. Let $w^- \in \min_{\leq_d}(w)$ such that $w^- \leq_d w'$. Then by Corollary 3, $\mathcal{C} \rightarrow [w^-]_{\equiv_d}$. Moreover, by transitivity, $w^- \leq_d v'$, so we have $([w^-]_{\equiv_d}, v') \in H$, as required.
- Case $\mathcal{P}(\mathcal{C}) = \square$ and $\mathcal{P}(v) = \diamond$. Note that $\mathcal{P}(\mathcal{C}) = \square$ implies $\mathcal{C} \subseteq V_\square$; hence $\mathcal{P}(w) = \square$. But then for some $w' \in w^\bullet, v' \in v^\bullet$, we have $w' \leq_d v'$. Let v', w' be such. Then again for some $w^- \in \min_{\leq_d}(w)$ satisfying $w^- \leq_d w'$ we have $\mathcal{C} \rightarrow [w^-]_{\equiv_d}$ and by transitivity, we have $([w^-]_{\equiv_d}, v') \in H$.
- Case $\mathcal{P}(\mathcal{C}) = \diamond$ and $\mathcal{P}(v) = \diamond$. Pick $\mathcal{C}' \in \mathcal{C}^\bullet$. We must show that $(\mathcal{C}', v') \in H$ for some $v' \in v^\bullet$. Since $\mathcal{P}(\mathcal{C}) = \diamond$, there must be some $u \in \mathcal{C} \cap V_\diamond$; pick such a u . Since $u \equiv_d w \leq_d v$, for all $u' \in u^\bullet \cap \mathcal{C}'$ there is some $v' \in v^\bullet$ such that $u' \leq_d v'$. Hence, there is some $v' \in v^\bullet$ such that $(\mathcal{C}', v') \in H$.
- Case $\mathcal{P}(\mathcal{C}) = \diamond$ and $\mathcal{P}(v) = \square$. Pick $\mathcal{C}' \in \mathcal{C}^\bullet$. We must show that $(\mathcal{C}', v') \in H$ for all $v' \in v^\bullet$. Then the argument is similar to the previous case.

Next, to prove $\mathcal{G} \leq_d \mathcal{G}_q$, we show that $H \subseteq V \times V_{/\equiv_d}$, given by $H = \{(v, \mathcal{C}) \mid \exists w \in \mathcal{C} : v \leq_d w\}$, is a direct simulation relation. Let \mathcal{C}, v be arbitrary such that $(v, \mathcal{C}) \in H$. By definition, there is some $w \in \mathcal{C}$ such that $v \leq_d w$. We again fix such a w . Following similar arguments as above, $\Omega'(\mathcal{C}) = \Omega(v)$. We again prove the transfer condition by distinguishing four cases.

- Case $\mathcal{P}(\mathcal{C}) = \diamond$ and $\mathcal{P}(v) = \diamond$. Pick $v' \in v^\bullet$. We must show that $(v', \mathcal{C}') \in H$ for some $\mathcal{C}' \in \mathcal{C}^\bullet$. We distinguish two cases.
 - Case $\mathcal{C} \subseteq V_\diamond$. Since $v \leq_d w$, also $v' \leq_d w'$ for some $w' \in w^\bullet$. Consider $w^+ \in \max_{\leq_d}(w)$ such that $w' \leq_d w^+$. Then by Corollary 3, $\mathcal{C} \rightarrow [w^+]_{\equiv_d}$. By transitivity $v' \leq_d w' \leq_d w^+$; hence $(v', [w^+]_{\equiv_d}) \in H$.
 - Case $\mathcal{C} \cap V_\square \neq \emptyset$ and $\mathcal{C} \cap V_\diamond \neq \emptyset$. Then, by Lemma 15, $\mathcal{C} \rightarrow \mathcal{C}'$ for some unique \mathcal{C}' . Fix this \mathcal{C}' . Let $u \in \mathcal{C} \cap V_\square$. Then $u' \in \mathcal{C}'$ for some $u' \in u^\bullet$. Pick such a u' . Since $u \equiv_d w$, also $v \leq_d u$. As a result, $v' \leq_d u'$. Therefore $(v', \mathcal{C}') \in H$.
- Case $\mathcal{P}(\mathcal{C}) = \diamond$ and $\mathcal{P}(v) = \square$. We must show that $(v', \mathcal{C}') \in H$ for some $v' \in v^\bullet, \mathcal{C}' \in \mathcal{C}^\bullet$. Then the argument is similar to the previous case.
- Case $\mathcal{P}(\mathcal{C}) = \square$ and $\mathcal{P}(v) = \diamond$. Let $v' \in v^\bullet$ and $\mathcal{C}' \in \mathcal{C}^\bullet$. Since $v \leq_d w$ and $\mathcal{P}(w) = \square$, we have $v' \leq_d w'$ for all $w' \in w^\bullet \cap \mathcal{C}'$. Therefore also $(v', \mathcal{C}') \in H$.
- Case $\mathcal{P}(\mathcal{C}) = \square$ and $\mathcal{P}(v) = \square$. Let $\mathcal{C}' \in \mathcal{C}^\bullet$ and let $w' \in w^\bullet \cap \mathcal{C}'$. Since $v \leq_d w$, there must be some $v' \in v^\bullet$ such that $v' \leq_d w'$. Fix this v' . Then also $(v', \mathcal{C}') \in H$. \square

We finally prove that the quotient is unique.

Theorem 13. *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 $\mathcal{G}_q = (V_{/\equiv_d}, \rightarrow_q, \Omega_q, \mathcal{P}_q)$ and $\mathcal{G}'_q = (V'_{/\equiv_d}, \rightarrow'_q, \Omega'_q, \mathcal{P}'_q)$ 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 Theorem 12 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 $\mathcal{P}'_q(\mathcal{C}) = \square$ whereas $\mathcal{P}_q(f(\mathcal{C})) = \diamond$ for some \mathcal{C} . Then $\mathcal{C} \subseteq V_{\square}$ 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 $\mathcal{P}(v) = \square$, 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 $w \in V_{\square}$. 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 $\mathcal{C} \rightarrow'_q \mathcal{C}'$ iff $f(\mathcal{C}) \rightarrow_q f(\mathcal{C}')$. Suppose $\mathcal{C} \rightarrow'_q \mathcal{C}'$ but not $f(\mathcal{C}) \rightarrow_q f(\mathcal{C}')$. Assume $\mathcal{P}'_q(\mathcal{C}) = \mathcal{P}_q(f(\mathcal{C})) = \diamond$. The case where $\mathcal{P}'_q(\mathcal{C}) = \mathcal{P}_q(f(\mathcal{C})) = \square$ is similar. Since $\mathcal{C} \equiv_d f(\mathcal{C})$, there must be some \mathcal{D} such that $f(\mathcal{C}) \rightarrow_q \mathcal{D}$ and $\mathcal{C}' \leq_d \mathcal{D}$. But then also $\mathcal{C} \rightarrow'_q \mathcal{C}''$ 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 $f(\mathcal{C}) \not\rightarrow_q f(\mathcal{C}')$.
- Case $\mathcal{C}' \neq \mathcal{C}''$. Then we have $\mathcal{C} \rightarrow'_q \mathcal{C}'$ and $\mathcal{C} \rightarrow'_q \mathcal{C}''$ 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 $\mathcal{C} \rightarrow'_q \mathcal{C}'$. \square

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

7.2 Governed Bisimulation and Governed Stuttering Bisimulation Quotients

We first define the governed bisimilarity quotients and claim some elementary results of these. We then define the governed stuttering bisimilarity quotient.

Definition 19 (Governed bisimulation quotient). The governed bisimulation quotient of $(V, \rightarrow, \Omega, \mathcal{P})$ is the structure $(V_{/\dot{\sim}}, \rightarrow', \Omega', \mathcal{P}')$, where, for $\mathcal{C}, \mathcal{C}' \in V_{/\dot{\sim}}$:

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

Theorem 14. *Let $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$ be a parity game and $\mathcal{G}_q = (V_{/\dot{\sim}}, \rightarrow', \Omega', \mathcal{P}')$ be its governed bisimulation quotient. Then $\mathcal{G} \dot{\sim} \mathcal{G}'$.*

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. \square

Theorem 15. *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 $\mathcal{G} \dot{\sim} \mathcal{G}'$ iff the two structures $\mathcal{G}_q = (V_{/\dot{\sim}}, \rightarrow_q, \Omega_q, \mathcal{P}_q)$ and $\mathcal{G}'_q = (V'_{/\dot{\sim}}, \rightarrow'_q, \Omega'_q, \mathcal{P}'_q)$ are isomorphic.*

Proof. Similar to the proof of Theorem 13. \square

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

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.

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

- $\Omega'(\mathcal{C}) = \min\{\Omega(v) \mid v \in \mathcal{C}\},$
- $\mathcal{P}'(\mathcal{C}) = \begin{cases} \diamond & \text{if for all } v \in \mathcal{C}, v \diamond \mapsto_{\approx}, \text{ or for some } v \in \mathcal{C}, \mathcal{C}' \neq \mathcal{C}, v \diamond \rightarrow \mathcal{C}' \\ \square & \text{otherwise} \end{cases}$
- $\mathcal{C} \rightarrow' \mathcal{C}'$ if and only if $\begin{cases} \exists i \in \{\diamond, \square\} : \forall v \in \mathcal{C} : v_i \mapsto_{\approx} & \text{if } \mathcal{C} = \mathcal{C}' \\ \exists i \in \{\diamond, \square\} : \forall v \in \mathcal{C} : v_i \mapsto_{\approx} \mathcal{C}' & \text{if } \mathcal{C} \neq \mathcal{C}' \end{cases}$

Theorem 16. *Let $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$ be a parity game and $\mathcal{G}_q = (V_{/\approx}, \rightarrow', \Omega', \mathcal{P}')$ be its governed stuttering bisimulation quotient. Then $\mathcal{G} \approx \mathcal{G}_q$.*

Proof. Consider the relation $R \subseteq (V \cup V_{/\approx}) \times (V \cup V_{/\approx})$, defined as follows:

$$R = \{(v, \mathcal{C}), (\mathcal{C}, v), (v, w), (\mathcal{C}, \mathcal{C}) \mid v, w \in \mathcal{C}\}$$

Then R is a governed stuttering bisimulation relation. Note that R is an equivalence relation. It thus suffices to prove that R meets the remaining conditions of governed stuttering bisimulation. We do so by proving that R is a governed stuttering bisimulation for the following cases: $v R w$, $\mathcal{C} R \mathcal{C}'$, $v R \mathcal{C}$ and $\mathcal{C} R v$. Observe that for $v, w \in V$ we have $v R w$ iff in \mathcal{G} we have $v \approx w$, and for $\mathcal{C}, \mathcal{C}' \in V_{/\approx}$ we have $\mathcal{C} R \mathcal{C}'$ iff $\mathcal{C} = \mathcal{C}'$. As a result, for these cases R is immediately a governed stuttering bisimulation relation. We therefore focus on the cases $v R \mathcal{C}$ and $\mathcal{C} R v$. Both cases are addressed separately.

Suppose that $v R \mathcal{C}$. We reason as follows:

- By definition, $\Omega(v) = \Omega'(\mathcal{C})$ as all $w \in \mathcal{C}$ are such that $\Omega(w) = \Omega(v)$.
- Suppose $v \rightarrow \mathcal{C}'$ for some $\mathcal{C}' \in V_{/R} \setminus \{[v]_R\}$. Then by definition of \approx we have $w \mathcal{P}(v) \mapsto_{\approx} \mathcal{C}'$ for all $w \in \mathcal{C}$ and therefore $\mathcal{C} \rightarrow' \mathcal{C}'$.
 - Case $\mathcal{P}(v) = \diamond$. Since for all $w \in \mathcal{C}$ we have $w \diamond \mapsto_{\approx} \mathcal{C}'$, there is some $w \in \mathcal{C}$ such that $w \diamond \rightarrow \mathcal{C}'$ and therefore $\mathcal{P}'(\mathcal{C}) = \diamond$. Because $\mathcal{C} \rightarrow' \mathcal{C}'$, also $\mathcal{C} \diamond \mapsto_R \mathcal{C}'$.
 - Case $\mathcal{P}(v) = \square$. Suppose $\mathcal{P}'(\mathcal{C}) = \square$. Since $\mathcal{C} \rightarrow' \mathcal{C}'$, also $\mathcal{C} \square \mapsto_R \mathcal{C}'$. Next, suppose $\mathcal{P}'(\mathcal{C}) = \diamond$. Then there must be some $w \in \mathcal{C}$ and some $\mathcal{C}'' \neq \mathcal{C}$ such that $w \diamond \rightarrow \mathcal{C}''$, as $w \diamond \mapsto_{\approx}$ would conflict with $w \square \mapsto_{\approx} \mathcal{C}'$. Let w be such. We also have $w \square \mapsto_{\approx} \mathcal{C}'$. This can only be the case if $\mathcal{P}(w) = \square$ and $w^\bullet \subseteq \mathcal{C}'$. But then \mathcal{C}' is the only successor of \mathcal{C} , i.e. $\mathcal{C} \rightarrow' \mathcal{C}''$ implies $\mathcal{C}'' = \mathcal{C}'$, and therefore $\mathcal{C} \square \mapsto_R \mathcal{C}'$.
- Suppose $v \mapsto_R$. Then $w \mapsto_R$ for all $w \in \mathcal{C}$. But then also $\mathcal{C} \rightarrow' \mathcal{C}$.
 - Case $i = \diamond$. Then also $\mathcal{P}'(\mathcal{C}) = \diamond$ and since $\mathcal{C} \rightarrow' \mathcal{C}$, also $\mathcal{C} \mapsto_R$.
 - Case $i = \square$. If $\mathcal{P}'(\mathcal{C}) = \square$, then, since $\mathcal{C} \rightarrow' \mathcal{C}$, also $\mathcal{C} \mapsto_R$. If $\mathcal{P}'(\mathcal{C}) = \diamond$, then also $v \diamond \mapsto_{\approx}$, since $u \diamond \rightarrow \mathcal{C}'$ for some $u \in \mathcal{C}$ and $\mathcal{C}' \neq \mathcal{C}$ contradicts $u \square \mapsto_{\approx}$. But $v \diamond \mapsto_{\approx}$ and $v \square \mapsto_{\approx}$ implies that for all \mathcal{C}' such that $\mathcal{C} \rightarrow' \mathcal{C}'$, we have $\mathcal{C}' = \mathcal{C}$. Hence, also $\mathcal{C} \mapsto_R$.

Assume that $\mathcal{C} R v$. We now reason as follows:

- $\Omega'(\mathcal{C}) = \Omega(v)$ follows from the same arguments as before.
- Suppose $\mathcal{C} \rightarrow' \mathcal{C}'$ for some $\mathcal{C}' \neq \mathcal{C}$. Then there is some $i \in \{\diamond, \square\}$ such that for all $w \in \mathcal{C}$, we have $w \mapsto_{\approx} \mathcal{C}'$. We distinguish two cases.
 - Assume $\mathcal{P}'(\mathcal{C}) = \diamond$.
 - * Case $i = \diamond$. Then $v \diamond \mapsto_{\approx}$ follows immediately since $v \in \mathcal{C}$.
 - * Case $i = \square$. Hence, for all $w \in \mathcal{C}$, we have $w \square \mapsto_{\approx} \mathcal{C}'$. Since $\mathcal{P}'(\mathcal{C}) = \diamond$, there must be some $w \in \mathcal{C}$, $\mathcal{C}'' \neq \mathcal{C}$, such that $w \diamond \rightarrow \mathcal{C}''$, since $w \diamond \mapsto_{\approx}$ contradicts $w \square \mapsto_{\approx} \mathcal{C}'$. Let w be such, and observe that $w \diamond \mapsto_{\approx} \mathcal{C}''$. This can only be the case if $\mathcal{C}'' = \mathcal{C}'$, and from $w \diamond \mapsto_{\approx} \mathcal{C}''$ and $\mathcal{C}'' = \mathcal{C}'$ we obtain $v \diamond \mapsto_{\approx} \mathcal{C}'$.

- Assume $\mathcal{P}'(\mathcal{C}) = \square$. Observe that this implies that $i = \square$, since, if $i = \diamond$, then for all $w \in \mathcal{C}$, we have $w \diamond \mapsto_{\approx} \mathcal{C}'$, which means there is some $w \in \mathcal{C}$ for which $w \diamond \rightarrow \mathcal{C}'$. This contradicts $\mathcal{P}'(\mathcal{C}) = \square$. So, $i = \square$. It then immediately follows that for all $w \in \mathcal{C}$, we have $w \square \mapsto_{\approx} \mathcal{C}'$, and in particular $v \square \mapsto_{\approx} \mathcal{C}'$.

- Assume that $\mathcal{C} \mapsto_R$. We distinguish two cases.

- Case $i = \diamond$. Observe that $\mathcal{P}'(\mathcal{C}) = \diamond$. Suppose that $\mathcal{P}'(\mathcal{C}) = \square$, then $\mathcal{C} \diamond \mapsto_R$ implies that for all \mathcal{C}' such that $\mathcal{C} \rightarrow' \mathcal{C}'$ we have $\mathcal{C}' = \mathcal{C}$ and therefore, for all $w \in \mathcal{C}$ we have both $w \square \mapsto_{\approx}$ and $w \diamond \mapsto_{\approx}$. This contradicts $\mathcal{P}'(\mathcal{C}) = \square$, hence $\mathcal{P}'(\mathcal{C}) = \diamond$.

Now, towards a contradiction, assume that $w \diamond \not\mapsto_{\approx}$ for all $w \in \mathcal{C}$. Because $\mathcal{P}'(\mathcal{C}) = \diamond$, there must be some $w \in \mathcal{C}$ such that $w \diamond \rightarrow \mathcal{C}'$ for some $\mathcal{C}' \neq \mathcal{C}$. Let w be such. Since $w \diamond \not\mapsto_{\approx}$ but $\mathcal{C} \rightarrow' \mathcal{C}$, we must have $w \square \mapsto_{\approx}$. This contradicts $w \diamond \rightarrow \mathcal{C}'$. So there must be some $w \in \mathcal{C}$ such that $w \diamond \mapsto_{\approx}$. But then also $v \diamond \mapsto_{\approx}$ and therefore $v \diamond \mapsto_R$.

- Case $i = \square$. Suppose $\mathcal{P}'(\mathcal{C}') = \diamond$. Because $\mathcal{C} \square \mapsto_R$ we find that $\mathcal{C} \not\mapsto' \mathcal{C}'$ for $\mathcal{C}' \neq \mathcal{C}$. Hence, for all $w \in \mathcal{C}$ we have $w \diamond \mapsto_{\approx}$ and $w \square \mapsto_{\approx}$; in particular, $v \square \mapsto_R$.

Next, assume that $\mathcal{P}'(\mathcal{C}') = \square$. Then there must be some $w \in \mathcal{C}$ such that $w \diamond \not\mapsto_{\approx}$. Consequently $w \diamond \not\mapsto_{\approx}$ for all $w \in \mathcal{C}$. Then $\mathcal{C} \rightarrow' \mathcal{C}$ can only be because for all $w \in \mathcal{C}$ we have $w \square \mapsto_{\approx}$. In particular, $v \square \mapsto_{\approx}$ and therefore $v \square \mapsto_R$. \square

Theorem 17. *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 $\mathcal{G} \approx \mathcal{G}'$ iff the two structures $\mathcal{G}_q = (V/\approx, \rightarrow_q, \Omega_q, \mathcal{P}_q)$ and $\mathcal{G}'_q = (V'/\approx, \rightarrow'_q, \Omega'_q, \mathcal{P}'_q)$ are isomorphic.*

Proof. Again similar to the proof of Theorem 13. \square

Corollary 7. *The governed stuttering bisimulation quotient of \mathcal{G} is a unique (up-to isomorphism) 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 Section 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 Section 4. That is, we first compare isomorphism, strong bisimilarity and governed bisimilarity and then focus on the various simulation equivalences.

Theorem 18. *Isomorphism is strictly finer than strong bisimilarity.*

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

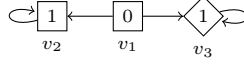


Figure 6: Parity game which is minimal with respect to strong bisimilarity. Vertices v_2 and v_3 are governed bisimilar.



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. \square

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 19. *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 Figure 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 Figure 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 Figure 6, which is minimal modulo strong bisimilarity, but vertices v_2 and v_3 are governed bisimilar. \square

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

Theorem 20. *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 21. *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 Figure 5 in which all vertices with priority 0 are governed stuttering bisimilar, but none are governed bisimilar. \square

Theorem 22. *Governed bisimulation equivalence and strong direct simulation equivalence are incomparable.*

Proof. This follows from the parity game in Figure 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. \square

To the best of our knowledge, the (elementary) result that direct simulation is strictly finer than (biased) delayed simulation has not been formally established. We therefore give a brief proof here.

Theorem 23. *Direct simulation equivalences is strictly finer than even- and odd-biased delayed simulation equivalence.*

Proof. Let $\mathcal{G} = (V, \rightarrow, \Omega, \mathcal{P})$ be a parity game, with $v, w \in V$ such that $v \equiv_d w$. Observe that $v \leq_d w$ and $w \leq_d v$. We prove that also $v \sqsubseteq_{de}^e w$; $w \sqsubseteq_{de}^e v$ and both cases for \sqsubseteq_{de}^o follow the exact same line of reasoning. As a result, we find that $v \equiv_{de}^e w$ and $v \equiv_{de}^o w$.

Observe that D has a winning strategy in the direct simulation game from (v, w) . Therefore, in each round of the game, *Duplicator* was able to mimic *Spoiler's* move by a move to a vertex with equal priority. If *Duplicator* plays the same strategy in the delayed simulation game, this gives rise to plays with obligation \checkmark in every configuration that is reached.

Strictness follows straightforwardly from the observation that the (biased) delayed simulation relations can relate vertices with different priorities, whereas direct simulation cannot. \square

The results relating (biased) delay simulations to each other and to winner equivalence were proven by Fritz and Wilke [22].

Theorem 24. *Even- and odd-biased delay simulation are incomparable, and both are strictly finer than delay simulation equivalence. Delay 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 Section 4.

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

Theorem 25. *Stuttering bisimilarity is incomparable to governed bisimilarity, strong direct simulation, direct simulation and all delayed simulation variations.*

Proof. In the parity game in Figure 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 Figure 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 delay simulation equivalent. \square

Theorem 26. *Governed stuttering bisimilarity is incomparable to strong direct simulation, direct simulation and all delayed simulation variations.*

Proof. Along the same lines as the proof of Theorem 25. \square

Theorem 27. *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 Figure 5 in which v_0 and v_1 are governed stuttering bisimilar, but not stuttering bisimilar. \square

To complete the lattice, we next move to showing 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.

Paths of length 1 are equivalent if the vertices they consist of are equivalent. If paths p and q are equivalent, then $pv \approx q$ iff v is equivalent to the last vertex in q , and $pv \approx qw$ iff $v \approx w$. 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 16. *Let $(V, \rightarrow, \mathcal{P}, \Omega)$ be a parity game, and let $(V_{/\approx}, \rightarrow', \mathcal{P}', \Omega')$ be its quotient. Let $v \in V$, and $\mathcal{C} \in V_{/\approx}$ 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 $p \approx q$.*

Proof. Define an arbitrary complete ordering $<$ on vertices, and define the following for finite paths q starting in \mathcal{C} , where $\min_{<} \emptyset$ is defined to be \perp :

$$\begin{aligned} \text{next}(q) &= \min_{<} \{v' \in V \mid \exists p \in \Pi_\sigma(v) : p \approx q \wedge p_{\sigma \rightarrow} v' \wedge pv' \not\approx q\} \\ \text{div}(q) &= \exists p \in \Pi_\sigma^\omega(v) : p \approx q \end{aligned}$$

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 $q \approx p$ for some $p \in \Pi_\sigma(v)$, then:

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

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

Case $\neg \text{div}(q)$, then it follows straightforwardly that $\text{next}(q) \neq \perp$.

Case $\mathcal{C}' \not\rightarrow' \mathcal{C}'$. Because \mathcal{C}' is a vertex is a quotient graph, $\mathcal{C}' \not\rightarrow'_{\approx}$. Consider the path $p \in \Pi_{\sigma}(v)$ for which $p \approx q$, and assume that p is of the form $\bar{p}u$. Since $\bar{p}u \approx q$, also $u \approx \mathcal{C}'$ and hence $u \not\rightarrow'_{\approx}$. Then by Lemma 5, $u \rightarrow_{\approx} V \setminus [u]_{\approx}$. Let $\sigma' \in \mathbb{S}_{\neg i}$ be such that $u \sigma' \rightarrow_{\approx} V \setminus [u]_{\approx}$ and consider the unique path $rv' \in \Pi_{\sigma'}(u)$ such that $\sigma \Vdash rv'$, $\sigma' \Vdash rv'$, $r \approx u$ and $v' \in V \setminus [u]_{\approx}$. Then $pr \in \Pi_{\sigma}(v)$ is such that $pr \approx q$, $pr \rightarrow_{\sigma} v'$ and $prv' \not\approx q$. Hence, $\text{next}(q) \neq \perp$.

Next, we show that we can set $\psi(q) = [\text{next}(q)]_{\approx}$. Since $\text{next}(q) \neq \perp$, there must be some $v \dots v'v'' \in \Pi_{\sigma}(v)$ such that $v'' = \text{next}(q)$, $v \dots v' \approx q$ and $v' \not\approx v''$. Since $v' \approx \mathcal{C}'$ and $v' \rightarrow [v'']_{\approx}$, also $\mathcal{C}' \rightarrow_{\mathcal{P}(v')} [v'']_{\approx}$. As \mathcal{C}' is a vertex in a quotient graph, this implies $\mathcal{C}' \rightarrow [v'']_{\approx}$. Hence, we can set $\psi(q) = [\text{next}(q)]_{\approx}$.

Now 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

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

For $n = 0$, this is trivial, because $v \approx \mathcal{C}$. For $n = m + 1$, assume as the induction hypothesis that $\forall \bar{q} \in \Pi_{\psi}^m(\mathcal{C}) : \exists \bar{p} \in \Pi_{\sigma}(v) : \bar{p} \approx \bar{q}$. Let $q \in \Pi_{\psi}^n(\mathcal{C})$ and let $\mathcal{C}', \mathcal{C}'' \in V_{/\approx}$ 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 $\mathcal{P}'(\mathcal{C}') = i$. Then $\mathcal{C}'' = \psi(\bar{q})$. The induction hypothesis yields some $\bar{p} \in \Pi_{\sigma}(v)$ such that $\bar{p} \approx \bar{q}$, therefore $\mathcal{C}'' = \mathcal{C}'$ if $\text{div}(\bar{q})$ and $\mathcal{C}' \rightarrow' \mathcal{C}'$, and otherwise $\mathcal{C}'' = [\text{next}(\bar{q})]_{\approx}$.

If $\mathcal{C}' = \mathcal{C}''$, then $\text{div}(\bar{q})$, so there must be some $p \in \Pi_{\sigma}^{\omega}(v)$ such that $p \approx q$ and therefore also some $p \in \Pi_{\sigma}(v)$ such that $p \approx q$.

If $\mathcal{C}'' = [\text{next}(\bar{q})]_{\approx}$, there must be some $p \in \Pi_{\sigma}(v)$ such that $p = p'v'$ and $p' \approx \bar{q}$ and $v' \approx \mathcal{C}''$. By definition, $p \approx q$ for such p .

Case $\mathcal{P}'(\mathcal{C}') \neq i$. From the induction hypothesis, obtain a $\bar{p} \in \Pi_{\sigma}(v)$ such that $\bar{p} \approx \bar{q}$. Without loss of generality we may assume that \bar{p} is finite. Note that $\mathcal{C}' \rightarrow [\mathcal{C}'']_{\approx}$. We distinguish two cases.

- Case $\mathcal{C}' = \mathcal{C}''$. Then we have $\bar{p} \approx \bar{q}\mathcal{C}''$.
- Case $\mathcal{C}' \neq \mathcal{C}''$. Let v' be the last vertex in \bar{p} . Because $\bar{p} \approx \bar{q}$, also $v' \rightarrow_{\approx} [\mathcal{C}'']_{\approx}$. So let $\sigma' \in \mathbb{S}_{\neg i}$ be such that $v' \sigma' \rightarrow_{\approx} [\mathcal{C}'']_{\approx}$. 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 $p_k \approx \mathcal{C}''$ and $p_l \approx \mathcal{C}'$ for all $l < k$. So $\bar{p}p_0 \dots p_k \approx q$.

Finally, we prove that for all $q \in \Pi_{\psi}^{\omega}(\mathcal{C})$ there is a $p \in \Pi_{\sigma}^{\omega}(v)$ such that $p \approx q$. Let $q \in \Pi_{\psi}^{\omega}(\mathcal{C})$. 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 $\mathcal{C} \approx v'$.

- Case $\mathcal{P}'(\mathcal{C}) = i$. Then $\psi(\bar{q}\mathcal{C}) = \mathcal{C}$, and thus $\text{div}(\bar{q}\mathcal{C})$ by definition of ψ . But then there must be some $p' \in \Pi_{\sigma}^{\omega}(v)$ such that $p' \approx \bar{q}\mathcal{C} \approx q$.
- Case $\mathcal{P}'(\mathcal{C}) \neq i$. Since $\mathcal{C} \rightarrow' \mathcal{C}$ we have $\mathcal{C} \rightarrow_{\approx}$ and since $v' \approx \mathcal{C}$, also $v' \rightarrow_{\approx}$. Let $\sigma' \in \mathbb{S}_{\neg i}$ be such that $v' \sigma' \rightarrow_{\approx}$. Then there is an infinite path $p' \in \Pi_{\sigma'}^{\omega}(v')$ such that $\sigma \Vdash p'$ and $p' \approx v'$. But then $\sigma \bar{p}p'$ and $q \approx \bar{p}p'$. \square

Theorem 28. *Governed stuttering bisimilarity strictly refines winner equivalence.*

Proof. Let $\mathcal{G} = (V, \rightarrow, \mathcal{P}, \Omega)$ be a parity game, and let $v, w \in V$ such that $v \approx w$. Let $(V_{/\approx}, \rightarrow', \mathcal{P}', \Omega')$ be the governed stuttering bisimulation quotient of \mathcal{G} , and let $\mathcal{C} \in V_{/\approx}$ be such that $w \approx \mathcal{C}$. By transitivity of \approx , also $v \approx \mathcal{C}$. Now suppose that player i has a winning strategy σ from v . Then by Lemma 16, 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 $p \approx q$. 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 . \square

9 Conclusion

Preorders and equivalences for parity games have been studied on a number of occasions, see [12, 14, 15, 21, 22, 32, 37]. 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 [15, 37] 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; 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 [29]. 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 [29]. 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.

References

- [1] R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In *Proc. CONCUR'98*, volume 1466 of *LNCS*, pages 163–178. Springer, 1998.
- [2] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, September 2002.
- [3] A. Arnold, A. Vincent, and I. Walukiewicz. Games for synthesis of controllers with partial observation. *Theor. Comput. Sci.*, 303(1):7–34, 2003.
- [4] A. Arnold and I. Walukiewicz. Nondeterministic controllers of non-deterministic processes. In *Logic and Automata*, volume 2 of *Texts in Logic and Games*, pages 29–52. Amsterdam University Press, 2008.
- [5] T. Basten. Branching bisimilarity is an equivalence indeed! *Inf. Proc. Let.*, 58(3):141–147, 1996.
- [6] J. van Benthem. Correspondence theory. In D. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic*, volume II, pages 167–248. 1984.
- [7] H. Björklund, S. Sandberg, and S.G. Vorobyov. A discrete subexponential algorithm for parity games. In *Proc. STACS'03*, volume 2607 of *LNCS*, pages 663–674. Springer, 2003.
- [8] H. Björklund and S.G. Vorobyov. Combinatorial structure and randomized subexponential algorithms for infinite games. *Theor. Comput. Sci.*, 349(3):347–360, 2005.
- [9] P.E. Bulychev, I.V. Konnov, and V.A. Zakharov. 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.
- [10] D. Bustan and O. Grumberg. Simulation-based minimization. *ACM Trans. Comput. Log.*, 4(2):181–206, apr 2003.
- [11] L. Clemente. Büchi automata can have smaller quotients. In *IC-ALP'11*, volume 6756 of *Lecture Notes in Computer Science*, pages 258–270. Springer, 2011.
- [12] S. Cranen. *Getting the point: obtaining and understanding fixpoints in model checking*. PhD thesis, Eindhoven University of Technology, Eindhoven, 2015.
- [13] S. Cranen, M. Gazda, J.W. Wesselink, and T.A.C. Willemse. Abstraction in Fixpoint Logic. *ACM Trans. Comput. Logic*, 16(4):29:1–29:39, 2015.
- [14] S. Cranen, J.J.A. Keiren, and T.A.C. Willemse. Stuttering mostly speeds up solving parity games. In *Proc. NFM'11*, volume 6617 of *LNCS*, pages 207–221. Springer, 2011.
- [15] S. Cranen, J.J.A. Keiren, and T.A.C. Willemse. A cure for stuttering parity games. In *Proc. ICTAC'12*, volume 7521 of *LNCS*, pages 198–212. Springer, 2012.

- [16] E.A. Emerson and C.S. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. FOCS'91*, pages 368–377. IEEE Computer Society, 1991.
- [17] E.A. Emerson, C.S. Jutla, and A.P. Sistla. On model checking for the μ -calculus and its fragments. *Theor. Comput. Sci.*, 258(1-2):491–522, May 2001.
- [18] K. Etessami, Th. Wilke, and R.A. Schuller. Fair simulation relations, parity games, and state space reduction for büchi automata. *SIAM J. Comput.*, 34(5):1159–1175, 2005.
- [19] O. Friedmann and M. Lange. Solving parity games in practice. In *Proc. ATVA'09*, volume 5799 of *LNCS*, pages 182–196. Springer, 2009.
- [20] O. Friedmann and M. Lange. Deciding the unguarded modal μ -calculus. *Journal of Applied Non-Classical Logics*, 23(4):353–371, 2013.
- [21] C. Fritz. *Simulation-Based Simplification of omega-Automata*. PhD thesis, Christian-Albrechts-Universität zu Kiel, 2005.
- [22] C. Fritz and T. Wilke. Simulation relations for alternating parity automata and parity games. In *Proc. DLT'06*, volume 4036 of *LNCS*, pages 59–70. Springer, 2006.
- [23] C. Fritz and Th. Wilke. State space reductions for alternating büchi automata. In *Proc. FSTTCS'02*, volume 2556 of *LNCS*, pages 157–168. Springer, 2002.
- [24] M.W. Gazda and T.A.C. Willemse. Consistent consequence for boolean equation systems. In *Proc. SOFSEM'12*, volume 7147 of *LNCS*, pages 277–288. Springer, 2012.
- [25] M.W. Gazda and T.A.C. Willemse. On parity game preorders and the logic of matching plays. In *Proc. SOFSEM'16*, volume 9587 of *LNCS*, pages 277–289. Springer, 2016.
- [26] R.J. van Glabbeek. The linear time - branching time spectrum. In *Proc. CONCUR '90*, volume 458 of *LNCS*, pages 278–297. Springer, 1990.
- [27] R.J. van Glabbeek. The linear time - branching time spectrum II. In *Proc. CONCUR'93*, volume 715 of *LNCS*, pages 66–81. Springer, 1993.
- [28] J.F. Groote and F.W. Vaandrager. An efficient algorithm for branching bisimulation and stuttering equivalence. In *Proc. ICALP'90*, volume 443 of *LNCS*, pages 626–638. Springer, 1990.
- [29] J.F. Groote and A. Wijs. An $O(m \log n)$ algorithm for stuttering equivalence and branching bisimulation. Technical report, January 2016. arXiv: 1601.01478.
- [30] M. Huth, J.H.-P. Kuo, and N. Piterman. Fatal attractors in parity games. In *Proc. FOSSACS'13*, volume 7794 of *LNCS*, pages 34–49. Springer, 2013.
- [31] M. Huth, J.H.-P. Kuo, and N. Piterman. Static analysis of parity games: Alternating reachability under parity. In *Semantics, Logics, and Calculi*, volume 9560 of *LNCS*, pages 159–177. Springer, 2016.

- [32] D. Janin. A contribution to formal methods: games, logic and automata, December 2005. Habilitation thesis.
- [33] M. Jurdziński. Deciding the winner in parity games is in $UP \cap co-UP$. *Inf. Proc. Let.*, 68(3):119–124, 1998.
- [34] M. Jurdziński. Small progress measures for solving parity games. In *Proc. STACS'00*, volume 1770 of *LNCS*, pages 290–301. Springer, 2000.
- [35] M. Jurdziński, M. Paterson, and U. Zwick. A Deterministic Subexponential Algorithm for Solving Parity Games. In *Proc. SODA'06*, pages 117–123. ACM/SIAM, 2006.
- [36] J.P. Katoen, T. Kemna, I.S. Zapreev, and D.N. Jansen. Bisimulation minimisation mostly speeds up probabilistic model checking. In *Proc. TACAS'07*, volume 4424 of *LNCS*, pages 76–92. Springer, 2007.
- [37] J.J.A. Keiren. *Advanced Reduction Techniques for Model Checking*. PhD thesis, Eindhoven University of Technology, 2013.
- [38] J.J.A. Keiren, J.W. Wesselink, and T.A.C. Willemse. Liveness analysis for parameterised Boolean equation systems. In *Proc. ATVA'14*, volume 8837 of *LNCS*, pages 219–234. Springer, 2014.
- [39] J.J.A. Keiren and T.A.C. Willemse. Bisimulation Minimisations for Boolean Equation Systems. In *Proc. HVC'09*, volume 6405 of *LNCS*. Springer, 2011.
- [40] R. Mayr and L. Clemente. Advanced automata minimization. In *POPL'13*, pages 63–74. ACM, 2013.
- [41] R. McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993.
- [42] K.S. Namjoshi. A simple characterization of stuttering bisimulation. In *Proc. FSTTCS'97*, volume 1346 of *LNCS*, pages 284–296. Springer, 1997.
- [43] S. Orzan, J.W. Wesselink, and T.A.C. Willemse. Static analysis techniques for parameterised Boolean equation systems. In *Proc. TACAS'09*, volume 5505 of *LNCS*, pages 230–245. Springer, 2009.
- [44] S. Orzan and T.A.C. Willemse. Invariants for Parameterised Boolean Equation Systems. *Theor. Comput. Sci.*, 411(11-13):1338–1371, 2010.
- [45] V. Petersson and S.G. Vorobyov. A Randomized Subexponential Algorithm for Parity Games. *Nordic Journal of Computing*, 8(3):324–345, 2001.
- [46] S. Schewe. Solving parity games in big steps. In *Proc. FSTTCS'07*, volume 4855 of *LNCS*, pages 449–460. Springer, 2007.
- [47] P. Stevens and C. Stirling. Practical model checking using games. In *Proc. TACAS'98*, volume 1384 of *LNCS*, pages 85–101. Springer, 1998.
- [48] C. Stirling. Bisimulation, modal logic and model checking games. *Logic Journal of IGPL*, 7(1):103–124, 1999.

- [49] J. Vöge and M. Jurdziński. A discrete strategy improvement algorithm for solving parity games. In *Proc. CAV'00*, volume 1855 of *LNCS*, pages 202–215. Springer, 2000.
- [50] T.A.C. Willemse. Consistent correlations for parameterised Boolean equation systems with applications in correctness proofs for manipulations. In *Proc. CONCUR'10*, volume 6269 of *LNCS*, pages 584–598. Springer, 2010.
- [51] Q. Yin, Y. Fu, C. He, M. Huang, and X. Tao. Branching bisimilarity checking for PRS. In *Proc. ICALP'14*, volume 8573 of *LNCS*, pages 363–374. Springer, 2014.
- [52] W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theor. Comput. Sci.*, 200(1-2):135 – 183, 1998.

A Detailed proofs of Propositions 3 and 4

Before we address Propositions 3 and 4, we first repeat the definition of the variant function we will use in the proof of Proposition 3 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 \approx for a configuration $((u_0, u_1), c)$ in the governed stuttering bisimulation game as follows:

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

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

$$\begin{aligned} \text{expel}(u_0, u_1) &\triangleq \text{dist}_{\neg \mathcal{P}(u_0), [u_0]_{\approx}}(u_1, V \setminus [u_0]_{\approx}) \\ \text{exit}(u_0, u_1, u_2) &\triangleq \text{dist}_{\mathcal{P}(u_0), [u_0]_{\approx}}(u_1, [u_2]_{\approx}) \end{aligned}$$

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

$$\text{dist}_{i,U}(v, T) \triangleq \begin{cases} \min\{n \mid v \in {}_U\text{Attr}_i^n(T)\} & \text{if } v \xrightarrow{i}_U T \\ \infty & \text{otherwise} \end{cases}$$

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 17. For $u, v \in V$, $(u \approx v \wedge m(u, v, \dagger) = (m_0, m_1)) \implies m_0 > 0$.

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 $u_0 \approx u_1$, we have $\text{expel}(u_0, u_1) > 0$ since $u_1 \notin V \setminus [u_0]_{\approx}$. The result then immediately follows. \square

Lemma 18. *Let $U, T \subseteq V$, such that $U \cap T = \emptyset$ and let $u \in U$. For all players i , if $u \mapsto_U T$ then $\text{dist}_{i,U}(u, T) > \min\{\text{dist}_{i,U}(v, T) \mid u \rightarrow v \wedge v \in U \cup T\}$.*

Proof. Assume $u \mapsto_U T$ and let $n = \text{dist}_{i,U}(u, T)$. Hence $u \in {}_U\text{Attr}_i^n(T)$ and $u \notin {}_U\text{Attr}_i^{n-1}(T)$. 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 $u \notin {}_U\text{Attr}_i^{n-1}(T)$, we have $\exists v \in u^\bullet : v \in {}_U\text{Attr}_i^{n-1}(T)$. Let v be such, and observe that $\text{dist}_{i,U}(v, T) \leq n - 1 < n$. The result then follows immediately.
- $\mathcal{P}(u) \neq i$. As n is such that $u \notin {}_U\text{Attr}_i^{n-1}(T)$, we have $\forall v \in u^\bullet : v \in {}_U\text{Attr}_i^{n-1}(T)$. Hence $\forall v \in V : u \rightarrow v \implies \text{dist}_{i,U}(v, T) \leq n - 1 < n$. Again the result follows immediately. \square

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

Lemma 19. *Let $U, T \subseteq V$, such that $U \cap T = \emptyset$ and $u \in U \cap V_{\neg i}$. Then $u \mapsto_U T$ implies $\text{dist}_{i,U}(u, T) > \max\{\text{dist}_{i,U}(v, T) \mid u \rightarrow v \wedge v \in U \cup T\}$.*

Proof. Let $u \in U \cap V_{\neg i}$ such that $u \mapsto_U T$. Suppose $n = \text{dist}_{i,U}(u, T)$. Then $u \in {}_U\text{Attr}_i^n(T)$ and $u \notin {}_U\text{Attr}_i^{n-1}(T)$. Since $u \notin V_i$, $\forall v \in u^\bullet : v \in {}_U\text{Attr}_i^{n-1}(T)$, hence $\forall v \in u^\bullet : \text{dist}_{i,U}(v, T) \leq n - 1 < n$; furthermore, such v are in $U \cup T$, and the result follows immediately. \square

Proposition 5. *For all $v, w \in V$ if $v \approx w$ then $v \equiv_{g, st} w$.*

Proof. We prove for all governed stuttering bisimilar vertices $v \approx w$ 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 $((u_0, u_1), c)$:

- $u_0 \approx u_1$,
- $c = (j, t)$ implies $u_j \not\approx t$,
- $c = (0, u)$ implies $(u_0, u_1) \in V_{\diamond} \times V$,
- $c = (1, u)$ implies $(u_0, u_1) \in V \times V_{\square}$.

In addition, we prove that from every configuration $((u_0, u_1), c)$, a configuration $((u'_0, u'_1), \checkmark)$ is reached within a finite number of steps 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 inductive 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 $c \neq \checkmark$, $m(t_0, t_1, c) > m(t'_0, t'_1, c')$.

- $t_0 \diamond \mapsto_{\approx}$ and $t_0 \square \mapsto_{\approx}$. This case is trivial, as in that case exactly one (reachable) equivalence class exists.
- $t_0 \diamond \mapsto_{\approx}$ and $t_0 \square \not\mapsto_{\approx}$. Since $t_0 \approx t_1$ also $t_1 \diamond \mapsto_{\approx}$. We distinguish cases based on the owners of the vertices.
 - $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \diamond$. *Spoiler* plays $t_0 \rightarrow w_0$.
 - * Case there is some $w_1 \in t_1^\bullet$ such that $w_0 \approx w_1$. 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 $w_0 \approx w_1$. Then *Duplicator* plays to a $w_1 \in t_1^\bullet$ for which $w_1 \approx t_1$ with minimal $m(t_0, w_1, (0, w_0))$; the existence of a $w_1 \approx t_1$ follows from $t_1 \diamond \mapsto_{\approx}$.
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 demonstrate $m(t_0, w_1, (0, w_0)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (0, w_0)\}$. In case $c = \dagger$ this follows from Lemma 17. In case $c = (0, w_0)$, this follows from Lemmata 18 and 19.
 - $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \square$. *Spoiler* plays $t_1 \rightarrow w_1$. Since $t_1 \diamond \mapsto_{\approx}$, all $w_1 \in t_1^\bullet$ satisfy $t_1 \approx w_1$. The same holds for all $w_0 \in t_0^\bullet$. *Duplicator* can thus play arbitrary $t_0 \rightarrow w_0$.
New configuration: $((w_0, w_1), \checkmark)$
 - $\mathcal{P}(t_0) = \diamond$, $\mathcal{P}(t_1) = \square$. *Spoiler* plays $t_0 \rightarrow w_0$ and $t_1 \rightarrow w_1$. Since $t_1 \diamond \mapsto_{\approx}$, also $w_1 \approx t_1$. We distinguish two further cases.
 - * Case $w_0 \approx w_1$.
New configuration: $((w_0, w_1), \checkmark)$.
 - * Case $w_0 \not\approx t_0$.
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 $t_0 \approx w_1$.
Progress: we demonstrate $m(t_0, w_1, (0, w_0)) < m(t_0, t_1, c)$ for $c \in \{\dagger, (0, w_0)\}$. In case $c = \dagger$ this follows from Lemma 17. In case $c = (0, w_0)$, this follows from Lemmata 18 and 19.

– $\mathcal{P}(t_0) = \square, \mathcal{P}(t_1) = \diamond$. *Duplicator* plays $t_1 \rightarrow w_1$ such that $w_1 \simeq t_1$ and $t_0 \rightarrow w_0$. Such a w_1 exists because $t_1 \diamond \mapsto \simeq$.

New configuration: $((w_0, w_1), \checkmark)$.

• $t_0 \square \mapsto \simeq$ and $t_0 \diamond \not\mapsto \simeq$. So, as before, $t_1 \square \mapsto \simeq$. This case is dual to the previous one.

• $t_0 \diamond \not\mapsto \simeq$ and $t_0 \square \not\mapsto \simeq$. We consider the owners of t_0 and t_1 .

– $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \diamond$. *Spoiler* plays $t_0 \rightarrow w_0$. We distinguish two cases.

* Case there is some $w_1 \in t_1^\bullet$ for which $w_0 \simeq w_1$. *Duplicator* plays $t_1 \rightarrow w_1$ 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 $u_0 = t_0$; otherwise $((w_0, t_1), \checkmark)$.

Progress: we must show $m(t_0, w_1, \dagger) < m(t_0, t_1, \dagger)$. This follows from Lemma 19.

• Case $t_0 \not\simeq w_0$. *Duplicator* plays $t_1 \rightarrow w_1$ 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 configuration $((t_0, w_1), (0, w_0))$ 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 17. In case $c = (0, w_0)$, this follows from Lemmata 18 and 19.

– $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \square$. *Spoiler* plays $t_1 \rightarrow w_1$.

* Case there is some $w_0 \in t_0^\bullet$ for which $w_0 \simeq w_1$. *Duplicator* plays $t_0 \rightarrow w_0$ 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 $t_0 \rightarrow w_0$.

New configuration: $((t_0, w_1), \dagger)$ if $c \in \{\checkmark, \dagger\}$ and $u_1 = t_1$; otherwise $((t_0, w_1), \checkmark)$.

Progress: we must show $m(w_0, t_1, \dagger) < m(t_0, t_1, \dagger)$. This follows from Lemma 19.

• Case $t_1 \not\simeq w_1$. *Duplicator* plays $t_0 \rightarrow w_0$ 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 $u_1 = t_1$ then configuration $((w_0, t_1), (1, w_1))$ and 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 17. In case $c = (1, w_1)$, this follows from Lemmata 18 and 19.

– $\mathcal{P}(t_0) = \diamond, \mathcal{P}(t_1) = \square$. *Spoiler* plays $t_0 \rightarrow w_0$ and $t_1 \rightarrow w_1$. 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 $w_0 \not\approx w_1$ and $t_0 \approx w_0$.
New configuration: $((w_0, t_1), (1, w_1))$ if $c \in \{\checkmark, \dagger, (1, w_1)\}$ and $u_1 = t_1$; otherwise $((w_0, t_1), \checkmark)$.
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 17. In case $c = (1, w_1)$ this follows from Lemmata 18 and 19.
 - * Case $w_0 \not\approx w_1$ and $t_1 \approx w_1$.
New configuration: $((t_0, w_1), (0, w_0))$ if $c \in \{\checkmark, \dagger, (0, w_0)\}$ and $u_0 = t_0$; otherwise $((t_0, w_1), \checkmark)$.
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 17. In case $c = (0, w_0)$ this follows from Lemmata 18 and 19.
- $\mathcal{P}(t_0) = \square, \mathcal{P}(t_1) = \diamond$.
- * Case there are $w_0 \in t_0^\bullet$ and $w_1 \in t_1^\bullet$ such that $w_0 \approx w_1$. 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 $w_0 \approx w_1$.
 - case there is some $w_0 \in t_0^\bullet$ such that $w_0 \approx t_0$. 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 $((w_0, t_1), \checkmark)$.
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 $u_0 \in V_\square$ and $u_0 \in V_\diamond$. Contradiction. Towards another contradiction, assume $c = (1, t)$ for some t . By our invariant, this implies $(t_0, t_1) \in V \times V_\square$. 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 18 and 19.
 - case there is some $w_1 \in t_1^\bullet$ such that $w_1 \approx t_1$. 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 $((t_0, w_1), \checkmark)$.
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 18 and 19. \square

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

Proposition 6. *For all $v, w \in V$ if $v \equiv_{g, st} w$ then $v \approx w$.*

Proof. We prove the contrapositive of the statement, *i.e.* for all $v, w \in V$, if $v \not\approx w$, then also $v \not\equiv_{g, st} w$. Let $v \not\approx w$. By Corollary 2, then also $(v, w) \notin \nu\mathcal{F}$. By the Tarski-Kleene fixpoint approximation theorem, we thus have $(v, w) \notin \bigcap_{k \geq 1} \mathcal{F}^k(V \times V)$. 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 $k \geq 1$:

(IH)

Spoiler wins the governed stuttering bisimulation game
for all configurations $((u_0, u_1), c)$ for which $(u_0, u_1) \notin R^k$

- 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 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 $i \in \{\diamond, \square\}$ and $\mathcal{U}, \mathcal{T} \subseteq V_{/R^k}$ for which

$$[v]_{R^k} \in \mathcal{U} \setminus \mathcal{T} \text{ but not } v \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T} \Leftrightarrow w \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}. \quad (*)$$

Let $i, \mathcal{U}, \mathcal{T}$ be such that $(*)$. We focus on the case $i = \diamond$; the case that $i = \square$ is fully dual. Assume that $v \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$ and not $w \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$; the case in which not $v \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$ but $w \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$ is symmetric. Note that we can assume that $\mathcal{T} \cap \mathcal{U} = \emptyset$, as $v \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$ iff $v \xrightarrow{i \mapsto \mathcal{U} \setminus \mathcal{T}} \mathcal{T}$ 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 $v \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$. 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 :

$$\begin{aligned} &\text{If } [t_0]_{R^k} = [t_1]_{R^k} \text{ then} \\ &\begin{cases} t_0 \xrightarrow{\sigma \mapsto \mathcal{U}} \mathcal{T} \text{ but not } t_1 \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T} \\ c = (0, t) \text{ implies } t_0 \in V_\diamond \text{ and } t_0 \xrightarrow{\sigma} t \\ c = (1, t) \text{ implies } t_1 \in V_\square, t \in t_1^\bullet \text{ and not } t \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T} \end{cases} \end{aligned}$$

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 $t_0 \xrightarrow{\sigma \mapsto \mathcal{U}} \mathcal{T}$, *Spoiler* proposes to move from t_0 to $\sigma(t_0)$. *Duplicator* proposes $u_1 \in t_1^\bullet$. Observe that not $u_1 \xrightarrow{i \mapsto \mathcal{U}} \mathcal{T}$. *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 $\mathcal{P}(t_0) = \mathcal{P}(t_1) = \square$. *Spoiler* proposes to move from t_1 to u_1 such that not $u_1 \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. Such u_1 exists. *Duplicator* proposes $u_0 \in t_0^\bullet$. Observe that $t_0 \sigma \rightarrow u_0$. *Duplicator* then 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 $\mathcal{P}(t_0) = \diamond, \mathcal{P}(t_1) = \square$. Since $t_0 \sigma \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$ *Spoiler* proposes to move from t_0 to $\sigma(t_0)$ and from t_1 to u_1 such that not $u_1 \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. Note that such u_1 exists. *Duplicator* then 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 $\mathcal{P}(t_0) = \square, \mathcal{P}(t_1) = \diamond$. *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 $u_1 \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. *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 $t_0 \sigma \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$ 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 $u_1 \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. *Duplicator* then proposes to continue in: $((t, u_1), \checkmark)$, $((t_0, u_1), (0, t))$, or $((t, t_1), \dagger)$.
 - Case $\mathcal{P}(t_1) = \square$. *Spoiler* proposes to move from t_0 to t and from t_1 to u_1 such that not $u_1 \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$. Such u_1 exists. *Duplicator* then 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 $t_1 \sigma \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$ and $\mathcal{P}(t_1) = \square$. 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* then proposes to continue in: $((\sigma(t_0), t), \checkmark)$, $((t_0, t), \checkmark)$, or $((\sigma(t_0), t_1), (1, t))$.
 - Case $\mathcal{P}(t_0) = \square$. *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* then 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. $t_1 \sigma \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$ but not $t_0 \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$;
2. $c = (0, t)$ implies $t_0 \notin V_\diamond$ or not $t_0 \sigma \rightarrow t$;
3. $c = (1, t)$ implies $t_1 \notin V_\square$, $t \notin t_1^\bullet$, or $t \dot{\mapsto}_{\mathcal{U}} \mathcal{T}$.

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 $(*)$ 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 $(*)$ 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 $(v_0, w_0) (v_1, w_1) (v_2, w_2) \dots$ 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 \checkmark '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 \checkmark rewards when π reaches a configuration containing a position with u . But then *Duplicator* earns only a finite number of \checkmark rewards along π , contradicting the assumption that *Duplicator* wins π .

Therefore, *Spoiler* has a strategy to win any configuration $((u_0, u_1), c)$ for which $(u_0, u_1) \notin R^k$. \square