Fixpoint Algorithms for Fair Parity/$\perp$ Games[*]

Daniel Hausmann
University of Gothenburg,
Gothenburg, Sweden

Nir Piterman
University of Gothenburg,
Gothenburg, Sweden

Irmak Sağlam
Max Planck Institute for Software
Systems (MPI-SWS), Kaiserslautern,
Germany

Anne-Kathrin Schmuck
Max Planck Institute for Software
Systems (MPI-SWS), Kaiserslautern,
Germany

1 Summary

Omega-regular games are a popular abstract modelling formalism for many core computational problems in the context of correct-by-construction synthesis of reactive software or hardware [8, 13, 4, 12, 15, 10]. However, before using synthesis techniques, the reactive software design problem at hand needs to be abstractly modelled as a two-player game. In order for the subsequently synthesized software to be 'correct-by-construction' this game graph needs to reflect all possible interactions between involved components in an abstract manner. Building such a game graph with the 'right' level of abstraction is a known severe challenge, in particular, if the synthesized software is interacting with existing components that already possess certain behavior. Here, part of the modelling challenge amounts to finding the 'right' power of both players in the resulting abstract game to ensure that winning strategies do not fail to exist due to an unnecessarily conservative overapproximation of modeling uncertainty (or the dual problem due to underapproximation).

In this context, *fairness* has been adopted as a notion to abstractly model known characteristics of the involved components in a very concise manner. *Fairness assumptions* have been used in model checking [1] and scheduler synthesis for the classical AMBA arbiter [11] or shared resource management [5], cyber-physical system design [16] and robot motion planning [9]. In all these applications, fairness is used as an *assumption* that the synthesized (or verified) component can rely on. In particular, if these assumptions are modelled by *transition fairness* over a two-player game arena $(V_\forall, V_\exists, E)$ – i.e., by a set of fair *environment* edges $E_f \subseteq E$ (i.e., with $V_\forall$ as their domain) that need to be taken infinitely often if the source vertex is seen infinitely often along a play – the resulting synthesis games can be solved efficiently [3, 14].

While most existing work has only looked at fairness as an *assumption*, all mentioned applications also naturally allow for scenarios where multiple components with intrinsically fair behavior are interacting with each other in a non-trivial manner. For example, the ability of a concurrent process to eventually free a shared resource might depend on how fair re-allocation is implemented in other threads. On an abstract level, the formal reasoning about such scenarios requires to understand how the interactive decision making of two dependent processes is influenced by intrinsic fairness constraints imposed on their decisions. Algorithmically, these synthesis questions require fairness restrictions on both players in a game. This work studies two-player games over finite graphs in which both players are restricted by

[*]The complete paper including the results given in this short abstract will be published in the conference proceedings of FoSSaCS'24. The full version is also available at: `https://arxiv.org/abs/2310.13612`

Submitted to:
SOS 2007

© D. Hausmann, N. Piterman, I. Sağlam & A-K. Schmuck
This work is licensed under the
Creative Commons Attribution License.

fairness constraints on their edges. We simply call such games *fair games*. Given a two player game graph $G = (V, E)$ and a set of fair edges $E_f \subseteq E$ a player is said to play *fair* in G if they choose an edge $e \in E_f$ infinitely often whenever the source vertex of e is visited infinitely often. Otherwise, they play *unfair*. We equip such games with two ω-regular winning conditions α and β deciding the winner of mutually-fair and mutually-unfair plays, respectively. Whenever one player plays fair and the other plays unfair, the fairly playing player wins the game. The resulting games are called *fair* α/β *games*.

In this talk we focus on the restriction of fair α/β games where α is a parity condition and $\beta = \perp$ (meaning mutually-unfair plays are losing for $\exists$-player), called *fair parity/⊥ games*.

We give two different solution algorithms for fair parity/⊥ games. The first one is based on a 3-step gadget construction inspired by [6] and the second one is a direct symbolic algorithm that is based on the gadget construction, which solves the game on the original graph.

Fair Game Arenas. A *fair game arena* $A = (V_\exists, V_\forall, E, E_f)$ consists of a set of *nodes* $V = V_\exists \uplus V_\forall$, together with an edge set $E \subseteq V \times V$ partitioned into the set $E_f \subseteq E$ of *fair edges* and $E \setminus E_f$ of *normal edges*.

We denote the set of $\exists$- and $\forall$-player strategies over A with Σ and Π, respectively; where strategies are defined as usual. We denote the nodes that have some fair outgoing edges with $V^{\mathsf{fair}} = \{v \in V \mid E_f(v) \neq \emptyset\}$ and set $V_i^{\mathsf{fair}} = V_i \cap V^{\mathsf{fair}}$ for $i \in \{\exists, \forall\}$. We denote all plays over A by $\mathsf{plays}(A)$ and a play starting from $v \in V$, compliant with the strategies $s \in \Sigma$ and $t \in \Pi$ by $\mathsf{play}_v(s,t)$. For a play ρ, $\mathsf{fair}_i(\rho)$ holds iff ρ is fair for player $i \in \{\exists, \forall\}$. The winning region of player i is denoted by Win_i.

Definition 1 (Fair Parity/⊥ Games)**.** A *fair game* $G = (A, Parity(\lambda), \perp)$ consists of a fair game arena A together with a parity condition given via a coloring λ of the nodes (or edges) of A. $Parity(\lambda) \subseteq \mathsf{plays}(A)$ determines the winner of mutually-fair plays, and $\perp$ determines the winner of mutually-unfair plays, indicating such plays are losing for $\exists$-player. A play that is i-fair and $(1-i)$-unfair is won by player i. Formally, in a fair parity/⊥ game $G = (A, Parity(\lambda), \perp)$, $v \in \mathsf{Win}_\exists$ iff,

$$\exists s \in \Sigma . \forall t \in \Pi . \mathsf{fair}_\exists(\mathsf{play}_v(s,t)) \wedge (\mathsf{fair}_\forall(\mathsf{play}_v(s,t)) \Rightarrow \mathsf{play}_v(s,t) \in \mathrm{Parity}(\lambda)). \tag{1}$$

1.1 Reduction to Parity Games

We show a linear reduction of fair parity/⊥ games to parity games in the case that α is a parity objective and $\beta = \top$. Our reduction works by replacing each fair node in the fair game with a 3-step parity gadget given in Figure 1. Theorem 1 states this formally.

This construction is inspired by the work of Chatterjee et al. [7] where the qualitative analysis of stochastic parity games is reduced to solving parity games.

Theorem 1. *Let* $G = (A, Parity(\lambda), \perp)$ *where* $A = (V_\exists, V_\forall, E, E_f)$ *is a fair game arena,* $V = V_\exists \uplus V_\forall$ *and* $\lambda : V \to [2k]$ *is the priority function. Then there exists a parity game* G' *on the node set* V' *with* $V \subseteq V'$ *and* $|V'| \leq n(3k+1)$ *over* $2k+1$ *priorities such that for* $i \in \{\exists, \forall\}$, $\mathsf{Win}_i(G) = \mathsf{Win}_i(G') \cap V$.

2 Fixpoint Characterization of Winning Regions

In this section, we characterize the winning regions in fair parity/⊥ games by means of fixpoint expressions. Thereby we provide an alternative, symbolic route to solve such games by computing fixpoint expressions, rather then by reducing to parity games. We start by briefly recalling details on Boolean fixpoint expressions.

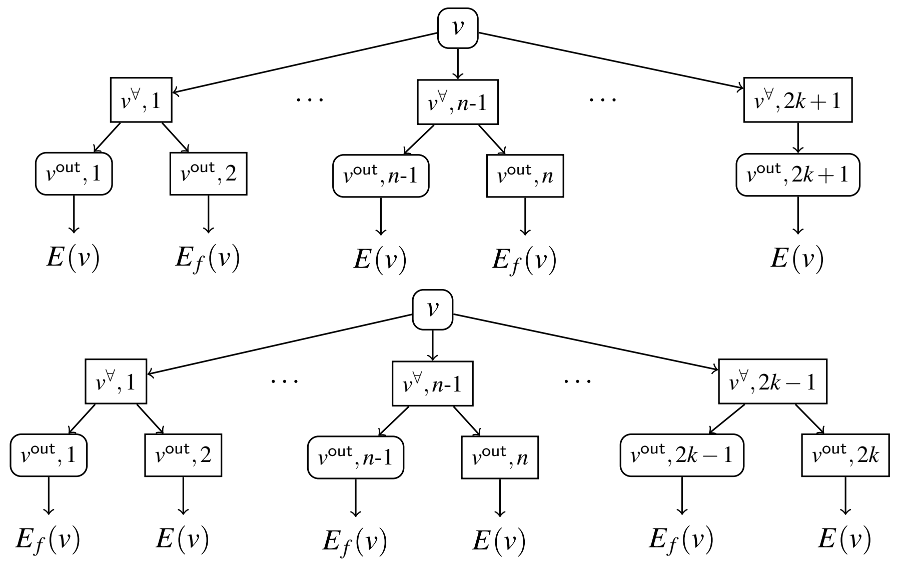

Figure 1: Gadgets for $v \in V_\exists^{\mathsf{fair}}$ (top) and $v \in V_\forall^{\mathsf{fair}}$ (bottom) in fair parity/$\perp$ games. The rectangular nodes are controlled by $\forall$-player, and the nodes with round corners are controlled by $\exists$-player. The number n is always even. Nodes (v^{out}, i) have priority i.

Fixpoint expressions and fixpoint games. Let U be a finite set, let o be a fixed number and let $f : \mathscr{P}(U)^o \to \mathscr{P}(U)$ be a monotone function, that is, assume that whenever we have sets $X_j \subseteq U$ and $Y_j \subseteq U$ such that $X_j \subseteq Y_j$ for all $1 \leq j \leq o$, then $f(X_1, \dots, X_o) \subseteq f(Y_1, \dots, Y_o)$. Then f and o induce the *fixpoint expression*

$$e = \eta_o X_o.\, \eta_{o-1} X_{o-1}.\, \dots .\, \nu X_2.\, \mu X_1.\, f(X_1, \dots, X_d) \tag{2}$$

where $\eta_i = \nu$ if i is even and $\eta_i = \mu$ if i is odd. We define the semantics of fixpoint expressions using parity games. Given a fixpoint expression e, the associated *fixpoint game* $G_e = (W_\exists, W_\forall, E, \text{Parity-edge}(\kappa))$ for the *edge-based* priority function $\kappa : E \to [o]$ is the following parity game. We put $W_\exists = U$, $W_\forall = \mathscr{P}(U)^o$. Moves and priorities are defined by

$$E(v) = \{\overline{Z} \in W_\forall \mid v \in f(\overline{Z})\} \qquad \kappa(v, \overline{Z}) = 0$$
$$E(\overline{Z}) = Z_1 \cup \ldots \cup Z_o \qquad \kappa(\overline{Z}, v) = \max\{i \mid v \in Z_i\}$$

for $v \in W_\exists$ and $\overline{Z} = (Z_1, \dots, Z_o) \in W_\forall$. Then we say that $v \in U$ is *contained* in e (denoted $v \in e$) if and only if $\exists$-player wins the node v in G_e.

Remark. The above game semantics for fixpoint expressions has been shown to be equivalent to the more traditional Knaster-Tarski semantics [2]; the cited work takes place in a more general setting and therefore uses slightly more verbose state-based parity games.

Next we present a fixpoint characterization of the winning regions in fair parity/$\perp$ games $G = (A, \text{Parity}(\lambda), \perp)$ where $A = (V_\exists, V_\forall, E, E_f)$ is a fair game arena, $V = V_\exists \cup V_\forall$ and $\lambda : V \to [2k]$ a priority function. To be able to write fixpoint expressions over such games we define monotone operators on subsets of V by putting

$$\Diamond X = \{v \in V \mid E(v) \cap X \neq \emptyset\} \qquad \Box X = \{v \in V \mid E(v) \subseteq X\}$$
$$\Diamond_f X = \{v \in V \mid E_f(v) \cap X \neq \emptyset\} \qquad \Box_f X = \{v \in V \mid E_f(v) \subseteq X\}$$

for $X \subseteq V$ and also put $\mathsf{Cpre}(X) = (V_\exists \cap \Diamond X) \cup (V_\forall \cap \Box X)$. Then $\mathsf{Cpre}(X)$ is the set of nodes from which $\exists$-player can force the game to reach a node from X in one step. Also, we define $C_i = \{v \in V \mid \lambda(v) = i\}$ for $1 \leq i \leq 2k$.

Using this notation, we define a function $\mathsf{parity} : \mathscr{P}(V)^{2k} \to \mathscr{P}(V)$ by putting

$$\mathsf{parity}(X_1, \ldots, X_{2k}) := (C_1 \cap \mathsf{Cpre}(X_1)) \cup \ldots \cup (C_k \cap \mathsf{Cpre}(X_{2k}))$$

for $(X_1, \ldots, X_{2k}) \subseteq \mathscr{P}(V)^{2k}$. This function is monotone and it is well-known (see e.g [17]) that the fixpoint induced by parity characterizes the winning region in parity games with priorities 1 through $2k$. This formula will still apply to 'normal' nodes $V^{\mathsf{n}} := V \setminus V^{\mathsf{fair}}$ in the fixpoint characterization of fair parity games.

We follow the gadget constructions from Figure 1 to define the following additional functions. For $1 \leq i < k$, we first abbreviate

$$\mathsf{Apre}_\exists(X_i, X_{i+1}) = \Diamond X_i \cap \Box_f X_{i+1} \qquad \mathsf{Apre}_\forall(X_i, X_{i+1}) = \Diamond_f X_i \cap \Box X_{i+1},$$

encoding nodes $(v^\forall, 2i)$ for $v \in V_\exists^{\mathsf{fair}}$ and $v \in V_\forall^{\mathsf{fair}}$, respectively. Then, we let $I_p = \{i \mid i \text{ odd}, p < i < 2k\}$ denote the set of odd priorities that lie strictly between p and $2k$, and put

$$\phi_{\exists,p}^{\mathsf{fair}} = \begin{cases} \bigcup_{i \in I_p} \mathsf{Apre}_\exists(X_i, X_{i+1}) \cup \Diamond X_{2k+1} & p \text{ is even} \\ \bigcup_{i \in I_p} \mathsf{Apre}_\exists(X_i, X_{i+1}) \cup \Diamond X_{2k+1} \cup \Box_f Y_p & p \text{ is odd} \end{cases}$$

and

$$\phi_{\forall,p}^{\mathsf{fair}} = \begin{cases} \bigcup_{i \in I_p} \mathsf{Apre}_\forall(X_i, X_{i+1}) & p \text{ is even} \\ \bigcup_{i \in I_p} \mathsf{Apre}_\forall(X_i, X_{i+1}) \cup \Box Y_p & p \text{ is odd} \end{cases}$$

Using this notation, the winning region for the existential player in fair parity/⊥ games with priorities 1 through $2k$ can be characterized by the fixpoint expression induced by $2k+1$ and the function χ that is defined to map $(X_1, \ldots, X_{2k+1}) \in \mathscr{P}(V)^{2k+1} \to \mathscr{P}(V)$ to the set

$$\begin{aligned} \chi(X_1, \ldots, X_{2k+1}) = & (V^{\mathsf{n}} \cap \mathsf{parity}) \cup \\ & (V_\exists^{\mathsf{fair}} \cap \bigcup_{i \in [2k+1]} C_i \cap \phi_{\exists,i}^{\mathsf{fair}}) \cup \\ & (V_\forall^{\mathsf{fair}} \cap \bigcup_{i \in [2k+1]} C_i \cap \phi_{\forall,i}^{\mathsf{fair}}) \end{aligned}$$

The full fixpoint expression then is

$$e = \mu X_{2k+1}.\, \nu X_{2k}.\, \mu X_{2k-1} \ldots \nu X_2.\, \mu X_1.\, \chi(X_1, \ldots, X_k) \tag{3}$$

That is, theorem 2 is the main result of this section.

Theorem 2. *Let $G = (A, Parity(\lambda), \bot)$ where $A = (V_\exists, V_\forall, E, E_f)$ is a fair game arena, $V = V_\exists \cup V_\forall$ and $\lambda : V \to [2k]$ is the priority function. Then the fixpoint expression given in* (3) *characterizes* $\mathsf{Win}_\exists(G)$.

References

[1] Benjamin Aminof, Thomas Ball & Orna Kupferman (2004): *Reasoning About Systems with Transition Fairness*. In Franz Baader & Andrei Voronkov, editors: *Logic for Programming, Artificial Intelligence, and Reasoning, 11th International Conference, LPAR 2004, Montevideo, Uruguay, March 14-18, 2005, Proceedings*, *Lecture Notes in Computer Science* 3452, Springer, pp. 194–208, doi:10.1007/978-3-540-32275-7_14. Available at `https://doi.org/10.1007/978-3-540-32275-7_14`.

[2] Paolo Baldan, Barbara König, Christina Mika-Michalski & Tommaso Padoan (2019): *Fixpoint games on continuous lattices*. *Proc. ACM Program. Lang.* 3(POPL), pp. 26:1–26:29, doi:10.1145/3290339. Available at `https://doi.org/10.1145/3290339`.

[3] Tamajit Banerjee, Rupak Majumdar, Kaushik Mallik, Anne-Kathrin Schmuck & Sadegh Soudjani (2023): *Fast Symbolic Algorithms for Omega-Regular Games under Strong Transition Fairness*. *TheoretiCS* 2, doi:10.46298/theoretics.23.4. Available at `https://doi.org/10.46298/theoretics.23.4`.

[4] J.R. Büchi & L.H. Landweber (1969): *Solving Sequential Conditions by Finite-State Strategies*. *Trans. Amer. Math. Soc.* 138, pp. 295–311.

[5] Krishnendu Chatterjee, Luca de Alfaro, Marco Faella, Rupak Majumdar & Vishwanath Raman (2013): *Code aware resource management*. *Formal Methods Syst. Des.* 42(2), pp. 146–174, doi:10.1007/s10703-012-0170-4. Available at `https://doi.org/10.1007/s10703-012-0170-4`.

[6] Krishnendu Chatterjee, Marcin Jurdziński & Thomas A Henzinger (2003): *Simple stochastic parity games*. In: *International Workshop on Computer Science Logic*, Springer, pp. 100–113.

[7] Krishnendu Chatterjee, Marcin Jurdzinski & Tom Henzinger (2003): *Simple Stochastic Parity Games*. In: *In Proceedings of the International Conference for Computer Science Logic (CSL)*, pp. 100–113. Available at `http://chess.eecs.berkeley.edu/pubs/729.html`.

[8] Alonzo Church (1963): *Application of recursive arithmetic to the problem of circuit synthesis*. *Journal of Symbolic Logic* 28(4).

[9] Nicolás D'Ippolito, Natalia Rodríguez & Sebastian Sardiña (2018): *Fully Observable Non-deterministic Planning as Assumption-Based Reactive Synthesis*. *J. Artif. Intell. Res.* 61, pp. 593–621, doi:10.1613/jair.5562. Available at `https://doi.org/10.1613/jair.5562`.

[10] Philipp J. Meyer, Salomon Sickert & Michael Luttenberger (2018): *Strix: Explicit Reactive Synthesis Strikes Back!* In: *30th International Conference on Computer Aided Verification*, *Lecture Notes in Computer Science* 10981, Springer, pp. 578–586.

[11] Nir Piterman, Amir Pnueli & Yaniv Sa'ar (2006): *Synthesis of Reactive(1) Designs*. In: *Proceedings of the 7th International Conference on Verification, Model Checking, and Abstract Interpretation*, VMCAI'06, Springer-Verlag, Berlin, Heidelberg, p. 364–380.

[12] A. Pnueli & R. Rosner (1988): *A Framework for the Synthesis of Reactive Modules*. In: *Proc. Intl. Conf. on Concurrency: Concurrency 88*, *Lecture Notes in Computer Science* 335, pp. 4–17.

[13] M.O. Rabin (1969): *Decidability of Second Order Theories and Automata on Infinite Trees*. *Trans. Amer. Math. Soc.* 141, pp. 1–35.

[14] Irmak Sağlam & Anne-Kathrin Schmuck (2023): *Solving Odd-Fair Parity Games*. In: *42th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*. (to appear).

[15] S. Schewe & B. Finkbeiner (2006): *Bounded Synthesis*. In: *4th Int. Symp. on Automated Technology for Verification and Analysis*, *Lecture Notes in Computer Science* 4218, Springer, pp. 245–259.

[16] John G Thistle & RP Malhamé (1998): *Control of ω-automata under state fairness assumptions*. *Systems & control letters* 33(4), pp. 265–274.

[17] Igor Walukiewicz (2002): *Monadic second-order logic on tree-like structures*. *Theor. Comput. Sci.* 275(1-2), pp. 311–346, doi:10.1016/S0304-3975(01)00185-2. Available at `https://doi.org/10.1016/S0304-3975(01)00185-2`.