Synthesizing Optimally Resilient Controllers

Daniel Neider
Max Planck Institute for Software Systems, 67663 Kaiserslautern, Germany
neider@mpi-sws.org

Alexander Weinert[1]
Reactive Systems Group, Saarland University, 66123 Saarbrücken, Germany
weinert@react.uni-saarland.de

Martin Zimmermann[2]
Reactive Systems Group, Saarland University, 66123 Saarbrücken, Germany
zimmermann@react.uni-saarland.de

Abstract

Recently, Dallal, Neider, and Tabuada studied a generalization of the classical game-theoretic model used in program synthesis, which additionally accounts for unmodeled intermittent disturbances. In this extended framework, one is interested in computing optimally resilient strategies, i.e., strategies that are resilient against as many disturbances as possible. Dallal, Neider, and Tabuada showed how to compute such strategies for safety specifications.

In this work, we compute optimally resilient strategies for a much wider range of winning conditions and show that they do not require more memory than winning strategies in the classical model. Our algorithms only have a polynomial overhead in comparison to the ones computing winning strategies. In particular, for parity conditions optimally resilient strategies are positional and can be computed in quasipolynomial time.

2012 ACM Subject Classification Theory of computation → Automata over infinite objects

Keywords and phrases Controller Synthesis, Infinite Games, Resilient Strategies, Disturbances

Digital Object Identifier 10.4230/LIPIcs.CSL.2018.34

Related Version Full version available online [18], `https://arxiv.org/abs/1709.04854`.

1 Introduction

Reactive synthesis is an exciting and promising approach to solving a crucial problem, whose importance is ever-increasing due to ubiquitous deployment of embedded systems: obtaining correct and verified controllers for safety-critical systems. Instead of an engineer programming a controller by hand and then verifying it against a formal specification, synthesis automatically constructs a correct-by-construction controller from the given specification (or reports that no such controller exists).

Typically, reactive synthesis is modeled as a two-player zero-sum game on a finite graph that is played between the system, which seeks to satisfy the specification, and its environment, which seeks to violate it. Although this model is well understood, there are still multiple obstacles to overcome before synthesis can be realistically applied in practice. These obstacles include not only the high computational complexity of the problem, but also more fundamental ones. Among the most prohibitive issues in this regard is the need

[1] Supported by the Saarbrücken Graduate School of Computer Science.
[2] Supported by the project "TriCS" (ZI 1516/1-1) of the German Research Foundation (DFG).

for a complete model of the interaction between the system and its environment, including an accurate model of the environment, the actions available to both players, as well as the effects of these actions.

This modeling task often places an insurmountable burden on engineers as the environments in which real-life controllers are intended to operate tend to be highly complex or not fully known at design time. Also, when a controller is deployed in the real world, a common source of errors is a mismatch between the controller's intended result of an action and the actual result. Such situations arise, e.g., in the presence of disturbances, when the effect of an action is not precisely known, or when the intended control action of the controller cannot be executed, e.g., when an actuator malfunctions. By a slight abuse of notation from control theory, such errors are subsumed under the generic term *disturbance* (cf. [10]).

To obtain controllers that can handle disturbances, one has to yield control over their occurrence to the environment. However, due to the antagonistic setting of the two-player zero-sum game, this would allow the environment to violate the specification by causing disturbances at will. Overcoming this requires the engineer to develop a realistic disturbance model, which is a highly complex task, as such disturbances are assumed to be rare events. Also, incorporating such a model into the game leads to a severe blowup in the size of the game, which can lead to intractability due to the high computational complexity of synthesis.

To overcome these fundamental difficulties, Dallal, Neider, and Tabuada [10] proposed a conceptually simple, yet powerful extension of infinite games termed "games with unmodeled intermittent disturbances". Such games are played similarly to classical infinite games: two players, called Player 0 and Player 1, move a token through a finite graph, whose vertices are partitioned into vertices under the control of Player 0 and Player 1, respectively; the winner is declared based on a condition on the resulting play. In contrast to classical games, however, the graph is augmented with additional *disturbance edges* that originate in vertices of Player 0 and may lead to any other vertex. Moreover, the mechanics of how Player 0 moves is modified: whenever she moves the token, her move might be overridden, and the token instead moves along a disturbance edge. This change in outcome implicitly models the occurrence of a disturbance – the intended result of the controller and the actual result differ – but it is not considered to be antagonistic. Instead, the occurrence of a disturbance is treated as a rare event without any assumptions on frequency, distribution, etc. This approach very naturally models the kind of disturbances typically occurring in control engineering [10].

As a non-technical example, consider a scenario with three siblings, Alice, Bob, and Charlie, and their father, Donald. He repeatedly asks Alice to fetch water from a well using a jug made of clay. Alice has three ways to fulfill that task: she may get the water herself or she may delegate it to either Bob or Charlie. In a simple model, the outcome of these strategies is identical: Donald's request for water is fulfilled. This is, however, unrealistic, as this model ignores the various ways that the execution of the strategies may go wrong. By modeling the situation as a game with disturbances, we obtain a more realistic model.

If Alice gets the jug herself, no disturbance can occur: she controls the outcome completely. If she delegates the task to Bob, the older of her brothers, Donald may get angry with her for not fulfilling her duties herself, which should not happen infinitely often. Finally, if she delegates the task to her younger brother Charlie, he might drop and break the jug, which would be disastrous for Alice.

These strategies can withstand different numbers of disturbances: the first strategy does not offer any possibility for disturbances, while infinitely many (a single) disturbance cause Alice to lose when using the second (the third) strategy. This model captures the intuition about Donald's and Charlie's behavior: both events occur non-antagonistically and their frequency is unknown.

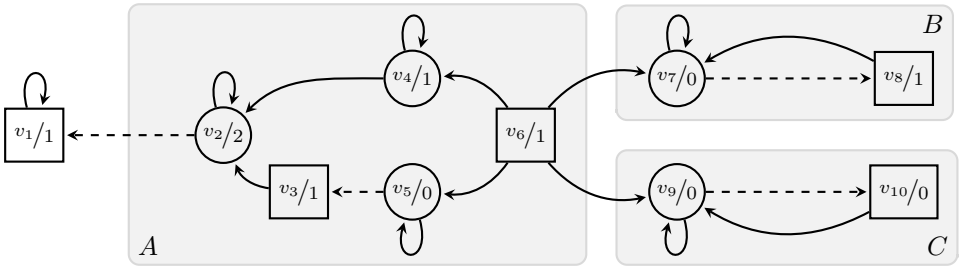

Figure 1 A (max-) parity game with disturbances. Disturbance edges are drawn as dashed arrows. Vertices are labeled with both a name and a color. Vertices under control of Player 0 are drawn as circles, while vertices under control of Player 1 are drawn as rectangles.

This non-antagonistic nature of disturbances is different from existing approaches in the literature and causes many interesting phenomena that do not occur in the classical theory of infinite graph-based games. Some of these already manifest themselves in the parity game shown in Figure 1, in which vertices are labeled with non-negative integers, so-called colors, and Player 0 wins if the highest color seen infinitely often is even. Consider, for instance, vertex v_2. In the classical setting without disturbances, Player 0 wins every play reaching v_2 by simply looping in this vertex forever (since the highest color seen infinitely often is even). However, this is no longer true in the presence of disturbances: a disturbance in v_2 causes a play to proceed to vertex v_1, from which Player 0 can no longer win. In vertex v_7, Player 0 is in a similar, yet less severe situation: she wins every play with finitely many disturbances but loses if infinitely many disturbances occur. Finally, vertex v_9 falls into a third category: from this vertex, Player 0 wins every play even if infinitely many disturbances occur. In fact, disturbances partition the set of vertices from which Player 0 can guarantee to win into three disjoint regions (indicated as shaded boxes in Figure 1): (A) vertices from which she can win if at most a fixed finite number of disturbances occur, (B) vertices from which she can win if any finite number of disturbances occurs but not if infinitely many occur, and (C) vertices from which she can win even if infinitely many disturbances occur.

The observation above gives rise to a question that is both theoretically interesting and practically important: if Player 0 can tolerate different numbers of disturbances from different vertices, how should she play to be *resilient*[3] to as many disturbances as possible, i.e., to tolerate as many disturbances as possible but still win? Put slightly differently, disturbances induce an order on the space of winning strategies ("a winning strategy is better if it is more resilient"), and the natural problem is to compute optimally resilient winning strategies, yielding optimally resilient controllers. Note that this is in contrast to the classical theory of infinite games, where the space of winning strategies is unstructured.

Dallal, Neider, and Tabuada [10] have solved the problem of computing optimally resilient winning strategies for safety games. Their approach exploits the existence of maximally permissive winning strategies in safety games [2], which allows Player 0 to avoid "harmful" disturbance edges during a play. In games with more expressive winning conditions, however, this is no longer possible, as witnessed by vertex v_4 in the example of Figure 1: although Player 0 can avoid a disturbance edge by looping in v_4 forever, she needs to move to v_2 eventually in order to see an even color (otherwise she loses), thereby risking to lose if a

[3] We have deliberately chosen the term *resilience* so as to avoid confusion with the already highly ambiguous notions of *robustness* and *fault tolerance*.

disturbance occurs. In fact, the problem of constructing optimally resilient winning strategies for games other than safety games is still open. In this work, we solve this problem for a large class of infinite games, including parity games. In detail, our contributions are as follows.

We study the concept of *resilience*, which captures for each vertex how many disturbances need to occur for Player 0 to lose. This generalizes the notion of determinacy and allows us to derive optimally resilient winning strategies.

Our main result is an algorithm for computing the resilience of vertices and optimally resilient winning strategies. This algorithm requires the game to have a prefix-independent winning condition, to be determined, and all its subgames to be (classically) solvable. The latter two conditions are necessary, as resilience generalizes determinacy and computing optimally resilient strategies generalizes solving games. The algorithm uses solvers for the underlying game without disturbances as a subroutine, which it invokes a linear number of times on various subgames. For many winning conditions, the time complexity of our algorithm thus falls into the same complexity class as solving the original game without disturbances, e.g., we obtain a quasipolynomial algorithm for parity games with disturbances, which matches the currently best known upper bound for classical parity games.

Stated differently, if the three assumptions above are satisfied by a winning condition, then computing the resilience and optimally resilient strategies is not harder than determining winning regions and winning strategies (ignoring a polynomial overhead).

Our algorithm requires the winning condition of the game to be prefix-independent. We also show how to overcome this restriction by generalizing the classical notion of game reductions to the setting of games with disturbances. As a consequence, via reductions our algorithm can be applied to prefix-dependent winning conditions. Hence, we have generalized the original result of Dallal, Neider, and Tabuada from safety games to all games which are algorithmically solvable, in particular all ω-regular games.

Finally, we discuss further phenomena that arise in the presence of disturbances. Amongst others, we illustrate how the additional goal of avoiding disturbances whenever possible affects the memory requirements of strategies. Moreover, we raise the question of how benevolent disturbances can be leveraged to recover from losing a play. However, an in-depth investigation of these phenomena is outside the scope of this paper and left for future work.

Proofs omitted due to space restrictions are in the full version [18].

Related Work. The notion of *unmodeled intermittent disturbances* in infinite games has recently been formulated by Dallal, Neider, and Tabuada [10]. In that work, the authors also present an algorithm for computing optimally resilient strategies for safety games with disturbances, which is an extension of the classical attractor computation [14]. Due to the relatively simple nature of such games, however, this algorithm cannot easily be extended to handle more expressive winning conditions, and the approach presented in this work relies on fundamentally different ideas.

For the special case of parity games, we can also characterize vertices of finite resilience (presented in Subsection 3.1) by a reduction to finding optimal strategies in energy parity games [9], which yields the same complexity as our algorithm (though such a reduction would not distinguish between vertices of type B and type C). Also, it is unclear if and how this reduction can be extended to other winning conditions and if custom-made solutions would be required for each new class of game. By contrast, our refinement-based approach works for any class of infinite games that satisfies the mild assumptions discussed in Section 4.

Resilience is not a novel concept in the context of reactive systems synthesis. It appears, for instance, in the work by Topcu et al. [21] as well as Ehlers and Topcu [12]. A notion of resilience that is very similar to the one considered here has been proposed by Huang

et al. [15], where the game graph is augmented with so-called "error edges". However, this setting differs from the one studied in this work in various aspects. Firstly, Huang et al. work in the framework of concurrent games and model errors as being under the control of Player 1. This contrasts to the setting considered here, in which the players play in alternation and disturbances are seen as rare events rather than antagonistic to Player 0. Secondly, Huang et al. restrict themselves to safety games, whereas we consider a much broader class of infinite games. Finally, Huang et al. compute resilient strategies with respect to a fixed parameter k, thus requiring to repeat the computation for various values of k to find optimally resilient strategies. In contrast, our approach computes an optimal strategy in a single run. Hence, they consider a more general model of interaction, but only a simple winning condition, while the notion of disturbances considered here is incomparable to theirs.

Related to resilience are various notions of *fault tolerance* [1, 7, 11, 13] and *robustness* [3, 4, 5, 6, 16, 19, 20]. For instance, Brihaye et al. [7] consider quantitative games under failures, which are a generalization of sabotage games [22]. The main difference to our setting is that Brihaye et al. consider failures – embodied by a saboteur player – as antagonistic, whereas we consider disturbances as a non-antagonistic events. Moreover, solving a parity game while maintaining a cost associated with the sabotage semantics below a given threshold is ExpTime-complete, whereas our approach computes optimally resilient controllers for parity conditions in quasipolynomial time.

Besides fault tolerance, robustness in the area of reactive controller synthesis has also attracted considerable interest in the recent years, typically in settings with specifications of the form $\varphi \Rightarrow \psi$ stating that the controller needs to fulfill the guarantee ψ if the environment satisfies the assumption φ. A prominent example of such work is that of Bloem et al. [3], in which the authors understand robustness as the property that "if assumptions are violated temporarily, the system is required to recover to normal operation with as few errors as possible" and consider the synthesis of robust controllers for the GR(1) fragment of Linear Temporal Logic [6]. Other examples include quantitative synthesis [4], where robustness is defined in terms of payoffs, and the synthesis of robust controllers for cyber-physical systems [16, 19]. For a more in-depth discussion of related notions of resilience and robustness in reactive synthesis, we refer the interested reader to Dallal, Neider, and Tabuada's section on related work [10, Section I]. Moreover, a survey of a large body of work dealing with robustness in reactive synthesis has been presented by Bloem et al. [5].

2 Preliminaries

For notational convenience, we employ some ordinal notation à la von Neumann: the non-negative integers are defined inductively as $0 = \emptyset$ and $n + 1 = n \cup \{n\}$. Now, the first limit ordinal is $\omega = \{0, 1, 2, \ldots\}$, the set of the non-negative integers. The next two successor ordinals are $\omega + 1 = \omega \cup \{\omega\}$ and $\omega + 2 = \omega + 1 \cup \{\omega + 1\}$. These ordinals are ordered by set inclusion, i.e., we have $0 < 1 < 2 < \cdots < \omega < \omega + 1 < \omega + 2$. For convenience of notation, we also denote the cardinality of ω by ω.

Infinite Games with Disturbances. An *arena (with unmodeled disturbances)* $\mathcal{A} = (V, V_0, V_1, E, D)$ consists of a finite directed graph (V, E), a partition $\{V_0, V_1\}$ of V into the set of vertices V_0 of Player 0 (denoted by circles) and the set of vertices of Player 1 (denoted by squares), and a set $D \subseteq V_0 \times V$ of *disturbance edges* (denoted by dashed arrows). Note that only vertices of Player 0 have outgoing disturbance edges. We require that every vertex $v \in V$ has a successor v' with $(v, v') \in E$ to avoid finite plays.

A *play* in $\mathcal{A}$ is an infinite sequence $\rho = (v_0, b_0)(v_1, b_1)(v_2, b_2)\cdots \in (V \times \{0,1\})^\omega$ such that $b_0 = 0$ and for all $j > 0$: $b_j = 0$ implies $(v_{j-1}, v_j) \in E$, and $b_j = 1$ implies $(v_{j-1}, v_j) \in D$. Hence, the additional bits b_j for $j > 0$ denote whether a standard or a disturbance edge has been taken to move from v_{j-1} to v_j. We say ρ starts in v_0. A play prefix $(v_0, b_0)\cdots(v_j, b_j)$ is defined similarly and ends in v_j. The number of disturbances in a play $\rho = (v_0, b_0)(v_1, b_1)(v_2, b_2)\cdots$ is $\#_D(\rho) = |\{j \in \omega \mid b_j = 1\}|$, which is either some $k \in \omega$ (if there are finitely many disturbances, namely k) or it is equal to ω (if there are infinitely many). A play ρ is *disturbance-free*, if $\#_D(\rho) = 0$.

A *game (with unmodeled disturbances)*, denoted by $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$, consists of an arena $\mathcal{A} = (V, V_0, V_1, E, D)$ and a winning condition $\mathrm{Win} \subseteq V^\omega$. A play $\rho = (v_0, b_0)(v_1, b_1)(v_2, b_2)\cdots$ is winning for Player 0, if $v_0 v_1 v_2 \cdots \in \mathrm{Win}$, otherwise it is winning for Player 1. Hence, winning is oblivious to occurrences of disturbances. A winning condition Win is prefix-independent if for all $\rho \in V^\omega$ and all $w \in V^*$ we have $\rho \in \mathrm{Win}$ if and only if $w\rho \in \mathrm{Win}$.

In examples, we often use the parity condition, the canonical ω-regular winning condition. Let $\Omega\colon V \to \omega$ be a coloring of a set V of vertices. The (max-) parity condition $\mathrm{Parity}(\Omega) = \{v_0 v_1 v_2 \cdots \in V^\omega \mid \limsup \Omega(v_0)\Omega(v_1)\Omega(v_2)\cdots \text{ is even}\}$ requires the maximal color occurring infinitely often during a play to be even. A game $(\mathcal{A}, \mathrm{Win})$ is a parity game, if $\mathrm{Win} = \mathrm{Parity}(\Omega)$ for some coloring Ω of the vertices of $\mathcal{A}$. In figures, we label a vertex v with color c by v/c.

In our proofs we make use of the safety condition $\mathrm{Safety}(U) = \{v_0 v_1 v_2 \cdots \in V^\omega \mid v_j \notin U \text{ for every } j \in \omega\}$ for a given set $U \subseteq V$ of unsafe vertices. It requires Player 0 to only visit safe vertices, i.e., Player 1 wins a play if it visits at least one unsafe vertex.

A strategy for Player $i \in \{0,1\}$ is a function $\sigma\colon V^* V_i \to V$ such that $(v_j, \sigma(v_0 \cdots v_j)) \in E$ holds for every $v_0 \cdots v_j \in V^* V_i$. A play $(v_0, b_0)(v_1, b_1)(v_2, b_2)\cdots$ is consistent with σ, if $v_{j+1} = \sigma(v_0 \cdots v_j)$ for every j with $v_j \in V_i$ and $b_{j+1} = 0$, i.e., if the next vertex is the one prescribed by the strategy unless a disturbance edge is used. A strategy σ is positional, if $\sigma(v_0 \cdots v_j) = \sigma(v_j)$ for all $v_0 \cdots v_j \in V^* V_i$.

▶ Remark. Note that a strategy σ does not have access to the bits indicating whether a disturbance occurred or not. However, this is not a restriction: let $(v_0, b_0)(v_1, b_1)(v_2, b_2)\cdots$ be a play with $b_j = 1$ for some $j > 0$. We say that this disturbance is consequential (w.r.t. σ), if $v_j \neq \sigma(v_0 \cdots v_{j-1})$, i.e., if the disturbance transition (v_{j-1}, v_j) traversed by the play did not lead to the vertex the strategy prescribed. Such consequential disturbances can be detected by comparing the actual vertex v_j to σ's output $\sigma(v_0 \cdots v_{j-1})$. On the other hand, inconsequential disturbances will just be ignored. In particular, the number of consequential disturbances is always at most the number of disturbances.

Infinite Games without Disturbances. We characterize the classical notion of infinite games, i.e., those without disturbances, (see, e.g., [14]) as a special case of games with disturbances. Let $\mathcal{G}$ be a game with vertex set V. A strategy σ for Player i in $\mathcal{G}$ is a winning strategy for her from $v \in V$, if every disturbance-free play that starts in v and that is consistent with σ is winning for Player i.

The winning region $\mathcal{W}_i(\mathcal{G})$ of Player i in $\mathcal{G}$ contains those vertices $v \in V$ from which Player i has a winning strategy. Thus, the winning regions of $\mathcal{G}$ are independent of the disturbance edges, i.e., we obtain the classical notion of infinite games. We say that Player i wins $\mathcal{G}$ from v, if $v \in \mathcal{W}_i(\mathcal{G})$. Solving a game amounts to determining its winning regions. Note that every game has disjoint winning regions. In contrast, a game is determined, if every vertex is in either winning region.

Resilient Strategies. Let $\mathcal{G}$ be a game with vertex set V and let $\alpha \in \omega + 2$. A strategy σ for Player 0 in $\mathcal{G}$ is *α-resilient from $v \in V$* if every play ρ that starts in v, that is consistent with σ, and with $\#_D(\rho) < \alpha$, is winning for Player 0. Thus, a k-resilient strategy with $k \in \omega$ is winning even under at most $k-1$ disturbances, an ω-resilient strategy is winning even under any finite number of disturbances, and an $(\omega+1)$-resilient strategy is winning even under infinitely many disturbances. Note that *every* strategy is 0-resilient, as no play has less than zero disturbances. Also, a strategy is 1-resilient from v if and only if it is winning for Player 0 from v. We define the resilience of a vertex v of $\mathcal{G}$ as

$$r_{\mathcal{G}}(v) = \sup\{\alpha \in \omega + 2 \mid \text{Player 0 has an } \alpha\text{-resilient strategy for } \mathcal{G} \text{ from } v\}.$$

Note that the definition is not antagonistic, i.e., it is not defined via strategies of Player 1. Nevertheless, due to the remarks above, resilient strategies generalize winning strategies.

▶ Remark. Let $\mathcal{G}$ be a determined game. Then, $r_{\mathcal{G}}(v) > 0$ if and only if $v \in \mathcal{W}_0(\mathcal{G})$.

A strategy σ is *optimally resilient*, if it is $r_{\mathcal{G}}(v)$-resilient from every vertex v. Every such strategy is a *uniform* winning strategy for Player 0, i.e., a strategy that is winning from every vertex in her winning region. Hence, positional optimally resilient strategies can only exist in games which have uniform positional winning strategies for Player 0. Our goal is to determine the mapping $r_{\mathcal{G}}$ and to compute an optimally resilient strategy.

3 Computing Optimally Resilient Strategies

To compute optimally resilient strategies, we first characterize the vertices of finite resilience in Subsection 3.1. All other vertices either have resilience ω or $\omega+1$. To distinguish between these possibilities, we show how to determine the vertices with resilience $\omega+1$ in Subsection 3.2. In Subsection 3.3, we show how to compute optimally resilient strategies using the results of the first two subsections.

3.1 Characterizing Vertices of Finite Resilience

Our goal in this subsection is to characterize vertices with finite resilience in a game with prefix-independent winning condition, i.e., those vertices from which Player 0 can win even under $k-1$ disturbances, but not under k disturbances, for some $k \in \omega$.

To illustrate our approach, consider the parity game in Figure 1 (on Page 3). The winning region of Player 1 only contains the vertex v_1. Thus, by Remark 2, v_1 is the only vertex with resilience zero, every other vertex has a larger resilience.

Now, consider the vertex v_2, which has a disturbance edge leading into the winning region of Player 1. Due to this edge, v_2 has resilience one. The unique disturbance-free play starting in v_1 is consistent with every strategy for Player 0 and violates the winning condition. Due to prefix-independence, prepending the disturbance edge does not change the winner and consistency with every strategy for Player 0. Hence, this play witnesses that v_2 has resilience at most one, while v_2 being in Player 0's winning region yields the matching lower bound. However, v_2 is the only vertex to which this reasoning applies. Now, consider v_3: from here, Player 1 can force a play to visit v_2 using a standard edge. Thus, v_3 has resilience one as well. Again, this is the only vertex to which this reasoning is applicable.

In particular, from v_4 Player 0 can avoid reaching the vertices for which we have determined the resilience by using the self loop. However, this comes at a steep price for her: doing so results in a losing play, as the color of v_4 is odd. Thus, if she wants to have a chance at winning, she has to take a risk by moving to v_2, from which she has a 1-resilient strategy,

i.e., one that is winning if no more disturbances occur. For this reason, v_4 has resilience one as well. The same reasoning applies to v_6: Player 1 can force the play to v_4 and from there Player 0 has to take a risk by moving to v_2.

The vertices v_3, v_4, and v_6 share the property that Player 1 can either enforce a play violating the winning condition or reach a vertex with already determined finite resilience. These three vertices are the only ones currently satisfying this property. They all have resilience one since Player 1 can enforce to reach a vertex of resilience one, but he cannot enforce reaching a vertex of resilience zero. Now, we can also determine the resilience of v_5: The disturbance edge from v_5 to v_3 witnesses it being two.

Afterwards, these two arguments no longer apply to new vertices: no disturbance edge leads from a $v \in \{v_7, \dots, v_{10}\}$ to some vertex whose resilience is already determined and Player 0 has a winning strategy from each v that additionally avoids vertices whose resilience is already determined. Thus, our reasoning cannot determine their resilience. This is consistent with our goal, as all four vertices have non-finite resilience: v_7 and v_8 have resilience ω and v_9 and v_{10} have resilience $\omega + 1$. Our reasoning here cannot distinguish these two values. We solve this problem in Subsection 3.2.

We now formalize the reasoning sketched above: starting from the vertices in Player 1's winning region having resilience zero, we use a so-called disturbance update and risk update to determine all vertices of finite resilience. A disturbance update computes the resilience of vertices having a disturbance edge to a vertex whose resilience is already known (such as vertices v_2 and v_5 in the example of Figure 1). A risk update, on the other hand, determines the resilience of vertices from which either Player 1 can force a visit to a vertex with known resilience (such as vertices v_3 and v_6) or Player 0 needs to move to such a vertex in order to avoid losing (e.g., vertex v_4). To simplify our proofs, we describe both as monotone operators updating partial rankings mapping vertices to ω, which might update already defined values. We show that applying these updates in alternation eventually yields a stable ranking that indeed characterizes the vertices of finite resilience.

Throughout this section, we fix a game $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with $\mathcal{A} = (V, V_0, V_1, E, D)$ and prefix-independent $\mathrm{Win} \subseteq V^\omega$ satisfying the following condition: the game $(\mathcal{A}, \mathrm{Win} \cap \mathrm{Safety}(U))$ is determined for every $U \subseteq V$. We discuss this requirement in Section 4.

A ranking for $\mathcal{G}$ is a partial mapping $r \colon V \dashrightarrow \omega$. The domain of r is denoted by $\mathrm{dom}(r)$, its image by $\mathrm{im}(r)$. Let r and r' be two rankings. We say that r' refines r if $\mathrm{dom}(r') \supseteq \mathrm{dom}(r)$ and if $r'(v) \leq r(v)$ for all $v \in \mathrm{dom}(r)$. A ranking r is sound, if we have $r(v) = 0$ if and only if $v \in \mathcal{W}_1(\mathcal{G})$ (cf. Remark 2).

Let r be a ranking for $\mathcal{G}$. We define the ranking r' as

$$r'(v) = \min\big(\{r(v)\} \cup \{r(v') + 1 \mid v' \in \mathrm{dom}(r) \text{ and } (v, v') \in D\}\big),$$

where $\{r(v)\} = \emptyset$ if $v \notin \mathrm{dom}(r)$, and $\min \emptyset$ is undefined (causing $r'(v)$ to be undefined). We call r' the *disturbance update* of r.

▶ **Lemma 1.** *The disturbance update r' of a sound ranking r is sound and refines r.*

Again, let r be a ranking for $\mathcal{G}$. For every $k \in \mathrm{im}(r)$ let $A_k = \mathcal{W}_1(\mathcal{A}, \mathrm{Win} \cap \mathrm{Safety}(\{v \in \mathrm{dom}(r) \mid r(v) \leq k\}))$ the winning region of Player 1 in the game where he either wins by reaching a vertex v with $r(v) \leq k$ or by violating the winning condition. Now, define $r'(v) = \min\{k \mid v \in A_k\}$, where $\min \emptyset$ is again undefined. We call r' the *risk update* of r.

▶ **Lemma 2.** *The risk update r' of a sound ranking r is sound and refines r.*

Let r_0 be the unique sound ranking with domain $\mathcal{W}_1(\mathcal{G})$, i.e., r_0 maps exactly the vertices in Player 1's winning region to zero. Starting with r_0, we inductively define a sequence of rankings $(r_j)_{j\in\omega}$ such that r_j for an odd (even) $j > 0$ is the disturbance (risk) update of r_{j-1}, i.e., we alternate between disturbance and risk updates.

Due to refinement, the r_j eventually stabilize, i.e., there is some j_0 such that $r_j = r_{j_0}$ for all $j \geq j_0$. Define $r^* = r_{j_0}$. Due to r_0 being sound and by Lemma 1 and Lemma 2, each r_j, and r^* in particular, is sound. If $v \in \mathrm{dom}(r^*)$, let j_v be the minimal j with $v \in \mathrm{dom}(r_j)$; otherwise, j_v is undefined.

▶ **Lemma 3.** *If $v \in \mathrm{dom}(r^*)$, then $r_{j_v}(v) = r_j(v)$ for all $j \geq j_v$.*

Lemma 3 implies that an algorithm computing the r_j does not need to implement the definition of the two updates as presented above, but can be optimized by taking into account that a rank is never updated once set. However, for the proofs below, the definition presented above is more expedient, as it gives stronger preconditions to rely on, e.g., Lemma 1 and 2 only hold for the definition presented above.

Also, from the proof of Lemma 3, we obtain an upper bound on the maximal rank of r^*. This in turn implies that the r_j stabilize quickly, as $r_j = r_{j+1} = r_{j+2}$ implies $r_j = r^*$.

▶ **Corollary 4.** *We have $\mathrm{im}(r^*) = \{0, 1, \ldots, n\}$ for some $n < |V|$ and $r^* = r_{2|V|}$.*

The main result of this section shows that r^* characterizes the resilience of vertices of finite resilience.

▶ **Lemma 5.** *Let r^* be defined for $\mathcal{G}$ as above, and let $v \in V$.*

1. *If $v \in \mathrm{dom}(r^*)$, then $r_{\mathcal{G}}(v) = r^*(v)$.*
2. *If $v \notin \mathrm{dom}(r^*)$, then $r_{\mathcal{G}}(v) \in \{\omega, \omega + 1\}$.*

Combining Corollary 4 and Lemma 5, we obtain an upper bound on the resilience of vertices with finite resilience.

▶ **Corollary 6.** *We have $r_{\mathcal{G}}(V) \cap \omega = \{0, 1, \ldots, n\}$ for some $n < |V|$.*

3.2 Characterizing Vertices of Resilience $\omega + 1$

Our goal in this subsection is to determine the vertices of resilience $\omega + 1$, i.e., those from which Player 0 can win even under an infinite number of disturbances. Intuitively, in this setting, we give Player 1 control over the disturbance edges, as he cannot execute more than infinitely many disturbances during a play.

In the following, we prove this intuition to be correct. To this end, we transform the arena of the game so that at a Player 0 vertex, first Player 1 gets to chose whether he wants to take one of the disturbance edges and, if not, gives control to Player 0, who is then able to use a standard edge.

Given a game $\mathcal{G} = (\mathcal{A}, \mathrm{Win})$ with $\mathcal{A} = (V, V_0, V_1, E, D)$, we define the *rigged game* $\mathcal{G}_{\mathrm{rig}} = (\mathcal{A}', \mathrm{Win}')$ with $\mathcal{A}' = (V', V_0', V_1', E', D')$ such that $V' = V_0' \cup V_1'$ with $V_0' = \{\overline{v} \mid v \in V_0\}$ and $V_1' = V$ and $D' = \emptyset$. The set E' of edges is the union of the following sets:

- D: Player 1 uses a disturbance edge.
- $\{(v, \overline{v}) \mid v \in V_0\}$: Player 1 does not use a disturbance edge and yields control to Player 0.
- $\{(\overline{v}, v') \mid (v, v') \in E \text{ and } v \in V_0\}$: Player 0 has control and picks a standard edge.
- $\{(v, v') \mid (v, v') \in E \text{ and } v \in V_1\}$: Player 1 takes a standard edge.

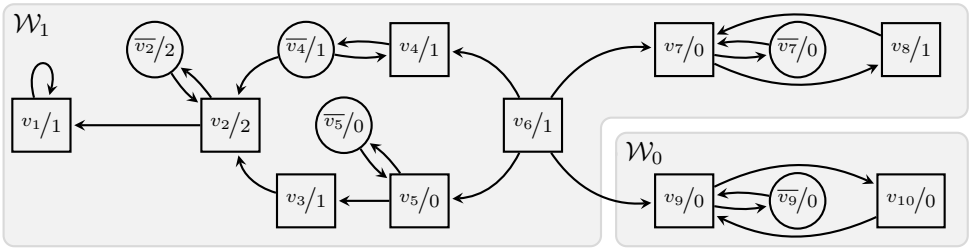

Figure 2 The rigged game obtained for the game of Figure 1.

Further, $\text{Win}' = \{\rho \in (V')^\omega \mid h(\rho) \in \text{Win}\}$ where h is the homomorphism induced by $h(v) = v$ and $h(\overline{v}) = \varepsilon$ for every $v \in V$.

Figure 2 illustrates the construction of a rigged game for the example game of Figure 1 (note that the rigged game is also a parity game in this example). Note that the winning region of Player 0 corresponds to the vertices of resilience $\omega + 1$ in the game of Figure 1.

The following lemma formalizes the observation that $\mathcal{W}_0(\mathcal{G}_{\text{rig}})$ characterizes the vertices of resilience $\omega + 1$ in $\mathcal{G}$. Note that we have no assumptions on $\mathcal{G}$ here.

▶ **Lemma 7.** *Let v be a vertex of game $\mathcal{G}$. Then, $v \in \mathcal{W}_0(\mathcal{G}_{\text{rig}})$ if and only if $r_{\mathcal{G}}(v) = \omega + 1$.*

Note that a slight extension of the rigged game also allows to characterize the vertices of resilience ω. To this end, one uses the same arena as for the rigged game, but adds to the winning condition of the rigged game all those plays during which Player 1 takes infinitely many disturbance edges. Then, Player 0 has to satisfy the original winning condition if only finitely many disturbance edges are taken by Player 1, but wins vacuously if Player 1 takes infinitely many disturbance edges. This is possible from exactly those vertices that have resilience ω. However, for our purposes, we do not need to investigate this modified rigged game. We have shown how to determine the vertices of finite resilience and those of resilience $\omega + 1$. Thus, all other vertices have resilience ω.

Furthermore, the proof of Lemma 7 also yields the preservation of positional strategies.

▶ **Corollary 8.** *Assume Player 0 has a positional winning strategy for $\mathcal{G}_{\text{rig}}$ from v. Then, Player 0 has an $(\omega + 1)$-resilient positional strategy from v.*

3.3 Computing Optimally Resilient Strategies

This subsection is concerned with computing the resilience and optimally resilient strategies. Here, we focus on positional and finite-state strategies, which are sufficient for the majority of winning conditions in the literature. Nevertheless, it is easy to see that our framework is also applicable to infinite-state strategies.

In the proof of Lemma 5, we construct strategies σ_f and σ_ω such that σ_f is $r_{\mathcal{G}}(v)$-resilient from every v with $r_{\mathcal{G}}(v) \in \omega$ and such that σ_ω is ω-resilient from every v with $r_{\mathcal{G}}(v) \geq \omega$. Both strategies are obtained by combining winning strategies for some game $(\mathcal{A}, \text{Win} \cap \text{Safety}(U))$. However, even if these winning strategies are positional, the strategies σ_f and σ_ω are in general not positional. Nonetheless, we show in the proof of Theorem 9 that such positional winning strategies and a positional one for $\mathcal{G}_{\text{rig}}$ can be combined into a single positional optimally resilient strategy.

Recall the requirements from Subsection 3.1 for a game $(\mathcal{A}, \text{Win})$: Win is prefix-independent and the game $\mathcal{G}_U$ is determined for every $U \subseteq V$, where we write $\mathcal{G}_U$ for the game $(\mathcal{A}, \text{Win} \cap \text{Safety}(U))$ for some $U \subseteq V$. To prove the results of this subsection, we

need to impose some additional effectiveness requirements: we require that each game $\mathcal{G}_U$ and the rigged game $\mathcal{G}_{\text{rig}}$ can be effectively solved. Also, we first assume that Player 0 has positional winning strategies for each of these games, which have to be effectively computable as well. We discuss the severity of these requirements in Section 4.

▶ **Theorem 9.** *Let $\mathcal{G}$ satisfy all the above requirements. Then, the resilience of $\mathcal{G}$'s vertices and a positional optimally resilient strategy can be effectively computed.*

To prove this result, we refine the following standard technique that combines positional winning strategies for games with prefix-independent winning conditions.

Assume we have a positional strategy σ_v for every vertex v in some set $W \subseteq V$ such that σ_v is winning from v. Furthermore, let R_v be the set of vertices visited by plays that start in v and are consistent with σ_v. Also, let $m(v) = \min_{\prec}\{v' \in V \mid v \in R_{v'}\}$ for some strict total ordering $\prec$ of W. Then, the positional strategy σ defined by $\sigma(v) = \sigma_{m(v)}(v)$ is winning from each $v \in W$, as along every play that starts in some $v \in W$ and is consistent with σ, the value of the function m only decreases. Thus, after it has stabilized, the remaining suffix is consistent with some strategy $\sigma_{v'}$. Hence, the suffix is winning for Player 0 and prefix-independence implies that the whole play is winning for her as well.

Here, we have to adapt this reasoning to respect the resilience of the vertices and to handle disturbance edges. Also, we have to pay attention to vertices of resilience $\omega + 1$, as plays starting in such vertices have to be winning under infinitely many disturbances.

Proof of Theorem 9. The effective computability of the resilience follows from the effectiveness requirements on $\mathcal{G}$: to compute the ranking r^*, it suffices to compute the disturbance and risk updates. The former are trivially effective while the effectiveness of the latter ones follows from our assumption. Lemma 5 shows that r^* correctly determines the resilience of all vertices with finite resilience. Finally by solving the rigged game, we also determine the resilience of the remaining vertices (Lemma 7). Again, this game can be solved by our assumption. Thus, it remains to show how to compute a positional optimally resilient strategy. To this end, we compute a positional strategy σ_v for every v satisfying the following:

- For every $v \in V$ with $r_{\mathcal{G}}(v) \in \omega \setminus \{0\}$, the strategy σ_v is winning for Player 0 from v for the game $(\mathcal{A}, \text{Win} \cap \text{Safety}(\{v' \in V \mid r_{\mathcal{G}}(v') < r_{\mathcal{G}}(v)\}))$. The existence of such a strategy has been shown in the proof of Item 1 of Lemma 5.
- For every $v \in V$ with $r_{\mathcal{G}}(v) = \omega$, the strategy σ_v is winning for Player 0 from v for the game $(\mathcal{A}, \text{Win} \cap \text{Safety}(\{v' \in V \mid r_{\mathcal{G}}(v') \in \omega\}))$. The existence of such a strategy has been shown in the proof of Item 2 of Lemma 5.
- For every $v \in V$ with $r_{\mathcal{G}}(v) = \omega + 1$, the strategy σ_v is $(\omega + 1)$-resilient from v. The existence of such a strategy follows from Corollary 8, as we assume Player 0 to win $\mathcal{G}_{\text{rig}}$ with positional strategies.
- For every $v \in V$ with $r_{\mathcal{G}}(v) = 0$, we fix an arbitrary positional strategy σ_v for Player 0.

Furthermore, we fix a strict linear order $\prec$ on V such that $v \prec v'$ implies $r_{\mathcal{G}}(v) \leq r_{\mathcal{G}}(v')$, i.e., we order the vertices by ascending resilience. For $v \in V$ with $r_{\mathcal{G}}(v) \neq \omega + 1$, let R_v be the vertices reachable via disturbance-free plays that start in v and are consistent with σ_v. On the other hand, for $v \in V$ with $r_{\mathcal{G}}(v) = \omega + 1$, let R_v be the set of vertices reachable via plays with arbitrarily many disturbances that start in v and are consistent with σ_v.

We claim $R_v \subseteq \{v' \in V \mid r_{\mathcal{G}}(v') \geq r_{\mathcal{G}}(v)\}$ for every $v \in V$ $(*)$. For v with $r_{\mathcal{G}}(v) \neq \omega + 1$ this follows immediately from the choice of σ_v. Thus, let v with $r_{\mathcal{G}}(v) = \omega + 1$. Assume σ_v reaches a vertex v' of resilience $r_{\mathcal{G}}(v') \neq \omega + 1$. Then, there exists a play ρ' starting in v' that is consistent with σ_v, has less than $\omega + 1$ many disturbances and is losing for Player 0.

Thus the play obtained by first taking the play prefix to v' and then appending ρ' without its first vertex yields a play starting in v, consistent with σ_v, but losing for Player 0. This play witnesses that σ_v is not $(\omega+1)$-resilient from v, which yields the desired contradiction.

Let $m\colon V \to V$ be given as $m(v) = \min_{\prec}\{v' \in V \mid v \in R_{v'}\}$ and define the positional strategy σ as $\sigma(v) = \sigma_{m(v)}(v)$. By our assumptions, σ can be effectively computed. It remains to show that it is optimally resilient.

To this end, we apply the following two properties of edges (v, v') that may appear during a play that is consistent with σ, i.e., we either have $v \in V_0$ and $\sigma(v) = v'$ (which implies $(v, v') \in E$), or $v \in V_1$ and $(v, v') \in E$, or $v \in V_0$ and $(v, v') \in D$:

1. If $(v, v') \in E$, then we have $r_{\mathcal{G}}(v) \leq r_{\mathcal{G}}(v')$ and $m(v) \geq m(v')$. The first property follows from minimality of $m(v)$ and $(*)$ while the second follows from the definition of R_v.
2. If $(v, v') \in D$, then we distinguish several subcases, which all follow immediately from the definition of resilience:
   - If $r_{\mathcal{G}}(v) \in \omega$, then $r_{\mathcal{G}}(v') \geq r_{\mathcal{G}}(v) - 1$.
   - If $r_{\mathcal{G}}(v) = \omega$, then $r_{\mathcal{G}}(v') = \omega$, and
   - If $r_{\mathcal{G}}(v) = \omega + 1$, then $r_{\mathcal{G}}(v') = \omega + 1$ and $m(v) \geq m(v')$ (here, the second property follows from the definition of R_v for v with $r_{\mathcal{G}}(v) = \omega + 1$, which takes disturbance edges into account).

Now, consider a play $\rho = (v_0, b_0)(v_1, b_1)(v_2, b_2)\cdots$ that is consistent with σ. If $r_{\mathcal{G}}(v_0) = 0$ then we have nothing to show, as every strategy is 0-resilient from v.

Now, assume $r_{\mathcal{G}}(v_0) \in \omega \setminus \{0\}$. We have to show that if ρ has less than $r_{\mathcal{G}}(v_0)$ disturbances, then it is winning for Player 0. An inductive application of the above properties shows that in that case the last disturbance edge leads to a vertex of non-zero resilience. Furthermore, as the values $m(v_j)$ are only decreasing afterwards, they have to stabilize at some later point. Hence, there is some suffix of ρ that starts in some v' with non-zero resilience and that is consistent with the strategy $\sigma_{v'}$. Thus, the suffix is winning for Player 0 by the choice of $\sigma_{v'}$ and prefix-independence implies that ρ is winning for her as well.

Next, assume $r_{\mathcal{G}}(v_0) = \omega$. We have to show that if ρ has a finite number of disturbances, then it is winning for Player 0. Again, an inductive application of the above properties shows that in that case the last disturbance edge leads to a vertex of resilience ω or $\omega + 1$. Afterwards, the values $m(v_j)$ stabilize again. Hence, there is some suffix of ρ that starts in some v' with non-zero resilience and that is consistent with the strategy $\sigma_{v'}$. Thus, the suffix is winning for Player 0 by the choice of $\sigma_{v'}$ and prefix-independence implies that ρ is winning for her as well.

Finally, assume $r_{\mathcal{G}}(v_0) = \omega + 1$. Then, the above properties imply that ρ only visits vertices with resilience $\omega + 1$ and that the values $m(v_j)$ eventually stabilize. Hence, there is a suffix of ρ that is consistent with some $(\omega + 1)$-resilient strategy $\sigma_{v'}$, where v' is the first vertex of the suffix. Hence, the suffix is winning for Player 0, no matter how many disturbances occurred. This again implies that ρ is winning for her as well. ◀

The algorithm determining the vertices' resilience and a positional optimally resilient strategy first computes r^* and the winner of the rigged game. This yields the resilience of $\mathcal{G}$'s vertices. Furthermore, the strategy is obtained by combining winning strategies for the games $\mathcal{G}_U$ and for the rigged game as explained above.

Next, we analyze the complexity of the algorithm sketched above in some more detail. The inductive definition of the r_j can be turned into an algorithm computing r^* (using the results of Lemma 3 to optimize the naive implementation), which has to solve $\mathcal{O}(|V|)$ many games (and compute winning strategies for some of them) with winning condition $\mathrm{Win} \cap \mathrm{Safety}(U)$.

Furthermore, the rigged game, which is of size $\mathcal{O}(|V|)$, has to be solved and winning strategies have to be determined. Thus, the overall complexity is in general dominated by the complexity of solving these tasks.

We explicitly state one complexity result for the important case of parity games, using the fact that each of these games is then a parity game as well. Also, we use a quasipolynomial time algorithms for solving parity games [8] to solve the games $\mathcal{G}_U$ and $\mathcal{G}_{\text{rig}}$.

▶ **Theorem 10.** *Optimally resilient strategies in parity games are positional and can be computed in quasipolynomial time.*

Using similar arguments, one can also analyze games where positional strategies do not suffice. As above, assume $\mathcal{G}$ satisfies the same assumptions on determinacy and effectiveness, but only require that Player 0 has finite-state winning strategies[4] for each game with winning condition $(\mathcal{A}, \text{Win} \cap \text{Safety}(U))$ and for the rigged game $\mathcal{G}_{\text{rig}}$. Then, one can show that she has a finite-state optimally resilient strategy. In fact, by reusing memory states, one can construct an optimally resilient strategy that it is not larger than any constituent strategy.

4 Discussion

In this section, we discuss the assumptions required to be able to compute positional (finite-state) optimally resilient strategies with the algorithm presented in Section 3. To this end, fix a game $\mathcal{G} = (\mathcal{A}, \text{Win})$ with vertex set V and recall that $\mathcal{G}_{\text{rig}}$ is the corresponding rigged game and that we defined $\mathcal{G}_U = (\mathcal{A}, \text{Win} \cap \text{Safety}(U))$ for $U \subseteq V$. Now, the assumptions on $\mathcal{G}$ for Theorem 9 to hold are as follows: (1) Every game $\mathcal{G}_U$ is determined. (2) Player 0 has a positional winning strategy from every vertex in her winning regions in the $\mathcal{G}_U$ and in the game $\mathcal{G}_{\text{rig}}$. (3) Each $\mathcal{G}_U$ and the game $\mathcal{G}_{\text{rig}}$ can be effectively solved and positional winning strategies can be effectively computed for each such game. (4) Win is prefix-independent.

First, consider the determinacy assumption. It is straightforward to show $\mathcal{W}_0(\mathcal{G}_U) = \mathcal{W}_0(\mathcal{A} \setminus W, \text{Win} \cap (V \setminus W)^\omega)$ with $W = \mathcal{W}_1(\mathcal{A}, \text{Safety}(U))$. Thus, one can first determine and then remove the winning region of Player 1 in the safety game and then solve the subgame of $\mathcal{G}$ played in Player 0's winning region of the safety game. Thus, all subgames of $\mathcal{G}$ being determined suffices for our determinacy requirement being satisfied. The winning conditions one typically studies, e.g., parity and in fact all Borel ones [17], satisfy this property.

The next requirement concerns the existence of positional (finite-state) winning strategies for the games $\mathcal{G}_U$ and $\mathcal{G}_{\text{rig}}$. For the $\mathcal{G}_U$, this requirement is satisfied if Player 0 has positional (finite-state) winning strategies for all subgames of $\mathcal{G}$. As every positional (finite-state) optimally resilient strategy is also a winning strategy in a certain subgame, this condition is necessary. Now, consider $\mathcal{G}_{\text{rig}}$, whose winning condition can be written as $h^{-1}(\text{Win})$ for the homomorphism h from Subsection 3.2. The winning conditions one typically studies, e.g., the Borel ones, are closed w.r.t. such supersequences. If $\mathcal{G}$ is from a class of winning conditions that allows for positional (finite-state) winning strategies for Player 0, then this class typically also contains $\mathcal{G}_{\text{rig}}$. Also, the assumption on the effective solvability and computability of positional (finite-state) strategies is obviously necessary, as we solve a more general problem when determining optimally resilient strategies.

Finally, let us consider prefix-independence. If the winning condition is not prefix-independent, the algorithm presented in Section 3 does not compute the resilience of vertices correctly anymore. As an example, consider the family $\mathcal{G}_k = (\mathcal{A}, \text{Win}_k)$ of games shown

[4] A finite state strategy is implemented by a finite automaton that processes play prefixes and outputs vertices to move to. See the full version [18] for a formal definition.

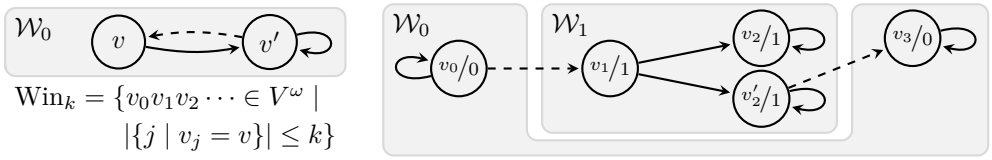

Figure 3 Left: Counterexample to the correctness of the computation of resilience for games with prefix-dependent winning conditions. Right: Intuitively, moving from v_1 to v_2' is preferable for Player 0, as it allows her to possibly "recover" from a first fault with the "help" of a second one.

on the left-hand side of Figure 3. In $\mathcal{G}_k$, it is Player 0's goal to avoid more than k visits to v. Such a visit only occurs via a disturbance or if the initial vertex is v. Hence, we have $r_{\mathcal{G}_k}(v) = k$ and $r_{\mathcal{G}_k}(v') = k+1$. Applying the algorithm from Section 3, however, the initial ranking function r_0 has an empty domain, since we have $\mathcal{W}_1(\mathcal{G}_k) = \emptyset$. Thus, the computation of the r_j immediately stabilizes, yielding r^* with empty domain. This is a counterexample to the generalization of Lemma 5 to prefix-dependent winning conditions.

Nevertheless, one can still leverage the algorithm from Section 3 in order to compute the resilience of a wide range of games with prefix-dependent winning conditions. To this end, we extend the framework of game reductions to games with disturbances, in such a way that the existence of α-resilient strategies is preserved. Using this framework shows that Player 0 has a finite-state optimally resilient strategy in every game with ω-regular winning condition. Due to space restrictions, the details are spelled out in the full version [18]. Here, we just state the main result.

▶ **Theorem 11.** *Let a game $\mathcal{G}$ be reducible to a game $\mathcal{G}'$ with prefix-independent winning condition, which can be effectively computed from $\mathcal{G}$, and satisfies the assumptions from Section 3 (with finite-state strategies). Then, the resilience of $\mathcal{G}$'s vertices and an optimally resilient finite-state strategy can be effectively computed.*

5 Outlook

We have developed a fine-grained view on the quality of strategies: instead of evaluating whether or not a strategy is winning, we compute its resilience against intermittent disturbances. While this measure of quality allows constructing "better" strategies than the distinction between winning and losing strategies, there remain aspects of optimality that are not captured in our notion of resilience. In this section we discuss these aspects and give examples of games in which there are crucial differences between optimally resilient strategies. In further research, we aim to synthesize optimal strategies with respect to these criteria.

As a first example, consider the parity game shown on the right-hand side of Figure 3. Vertices v_0 and v_3 have resilience 1 and $\omega + 1$, respectively, while vertices v_1, v_2, and v_2' have resilience 0. Player 0's only choice consists of moving to v_2 or to v_2' from v_1. Let σ and σ' be strategies for Player 0 that always move to v_2 and v_2' from v_1, respectively. Both strategies are optimally resilient. Hence, the algorithm from Section 3 may yield either one, depending on the underlying parity game solver used. Intuitively, however, σ' is preferable for Player 0, as a play prefix ending in v_2' may proceed to her winning region if a single disturbance occurs. All plays encountering v_2 at some point, however, are losing for her. Hence, another interesting avenue for further research is to study **how to recover from losing**, i.e., how to construct strategies that leverage disturbances in order to leave Player 1's winning region. For safety games, this has been addressed by Dallal, Neider, and Tabuada [10].

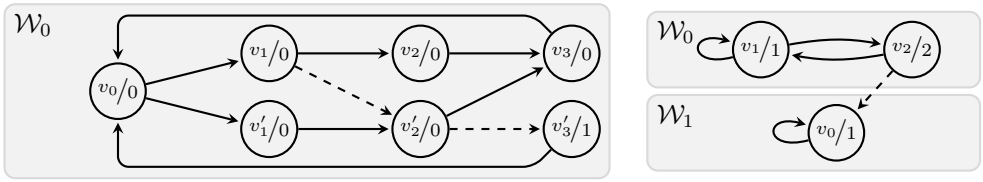

Figure 4 Left: Moving to v_1 from v_0 allows Player 0 to minimize visits to odd colors, while moving to v_1' allows her to minimize the occurrence of disturbances. Right: Additional memory allows Player 0 to remain in v_1 longer and longer, thus decreasing the potential for disturbances.

The previous example shows that Player 0 can still make "meaningful" choices even if the play has moved outside her winning region. The game $\mathcal{G}$ shown in the left-hand side of Figure 4 demonstrates that she can do so as well when remaining in vertices of resilience ω. Every vertex in $\mathcal{G}$ has resilience ω, since every play with finitely many disturbances eventually remains in vertices of color 0. Moreover, the only choice to be made by Player 0 is whether to move to vertex v_1 or to vertex v_1' from vertex v_0. Let σ and σ' be positional strategies that implement the former and the latter choice, respectively.

First consider a scenario in which visiting an odd color models the occurrence of some undesirable event, e.g., that a request has not been answered. In this case, Player 0 should aim to prevent visits to v_3' in $\mathcal{G}$, the only vertex of odd color. Hence, the strategy σ should be more desirable for her, as it requires two disturbances in direct succession in order to visit to v_3'. When playing consistently with σ', however, a single disturbance suffices to visit v_3'.

On the other hand, consider a setting in which Player 0's goal is to avoid the occurrence of disturbances. In that case, σ' is preferable over σ, as it allows for fewer situations in which disturbances may occur, since no disturbances are possible from vertices v_2 and v_3.

Note that the goals of minimizing visits to vertices of odd color and minimizing the occurrence of disturbances are not contradictory: if both events are undesirable, it may be optimal for Player 0 to combine the strategies σ and σ'. In general, it is interesting to study how to **how to best brace for a finite number of disturbances**.

Recall that, due to Theorem 10, optimally resilient strategies for parity games do not require memory. In contrast, the game shown on the right-hand side of Figure 4 demonstrates that additional memory can serve to further improve such strategies. Any strategy for Player 0 that does not stay in v_1 from some point onwards is optimally resilient. However, every visit to v_2 risks a disturbance occurring, which would lead the play into a losing sink for Player 0. Hence, it is in her best interest to remain in vertex v_1 for as long as possible, thus minimizing the possibility for disturbances to occur. This behavior does, however, require memory to implement, as Player 0 needs to count the visits to v_1 in order to not remain in that state ad infinitum. Thus, for each optimally resilient strategy σ with finite memory there exists another optimally resilient strategy that uses more memory, but visits v_2 more rarely than σ, reducing the possibilities for disturbances to occur. Hence, it is interesting to study **how to balance avoiding disturbances with satisfying the winning condition.** This is particularly interesting if there is some cost assigned to disturbances.

Finally, another important and interesting aspect, which falls outside the scope of this paper, is to provide general guidelines and best practices on how to model synthesis problems by games with disturbances. We will address these problems in future research.

6 Conclusion

We presented an algorithm for computing optimally resilient strategies in games with disturbances to any game that satisfies some mild (and necessary) assumptions. Thereby, we have vastly generalized the work of Dallal, Neider, and Tabuada, who only considered safety games. Furthermore, we showed that optimally resilient strategies are typically of the same size as classical winning strategies. Finally, we have illustrated numerous novel phenomena that appear in the setting with disturbances but not in the classical one. Studying these phenomena is a very promising direction of future work.

References

1 Paul C. Attie, Anish Arora, and E. Allen Emerson. Synthesis of fault-tolerant concurrent programs. *ACM Trans. Program. Lang. Syst.*, 26(1):125–185, 2004. doi:10.1145/963778.963782.

2 Julien Bernet, David Janin, and Igor Walukiewicz. Permissive strategies: from parity games to safety games. *ITA*, 36(3):261–275, 2002. doi:10.1051/ita:2002013.

3 Roderick Bloem, Krishnendu Chatterjee, Karin Greimel, Thomas A. Henzinger, Georg Hofferek, Barbara Jobstmann, Bettina Könighofer, and Robert Könighofer. Synthesizing robust systems. *Acta Inf.*, 51(3-4):193–220, 2014. doi:10.1007/s00236-013-0191-5.

4 Roderick Bloem, Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Better quality in synthesis through quantitative objectives. In Ahmed Bouajjani and Oded Maler, editors, *CAV 2009*, volume 5643 of *LNCS*, pages 140–156. Springer, 2009. doi:10.1007/978-3-642-02658-4_14.

5 Roderick Bloem, Rüdiger Ehlers, Swen Jacobs, and Robert Könighofer. How to handle assumptions in synthesis. In Krishnendu Chatterjee, Rüdiger Ehlers, and Susmit Jha, editors, *SYNT 2014*, volume 157 of *EPTCS*, pages 34–50, 2014. doi:10.4204/EPTCS.157.7.

6 Roderick Bloem, Barbara Jobstmann, Nir Piterman, Amir Pnueli, and Yaniv Sa'ar. Synthesis of Reactive(1) designs. *J. Comput. Syst. Sci.*, 78(3):911–938, 2012. doi:10.1016/j.jcss.2011.08.007.

7 Thomas Brihaye, Gilles Geeraerts, Axel Haddad, Benjamin Monmege, Guillermo A. Pérez, and Gabriel Renault. Quantitative games under failures. In *FSTTCS 2015*, volume 45 of *LIPIcs*, pages 293–306. Schloss Dagstuhl - LZI, 2015. doi:10.4230/LIPIcs.FSTTCS.2015.293.

8 Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *STOC 2017*, pages 252–263. ACM, 2017. doi:10.1145/3055399.3055409.

9 Krishnendu Chatterjee and Laurent Doyen. Energy parity games. *Theor. Comput. Sci.*, 458:49–60, 2012. doi:10.1016/j.tcs.2012.07.038.

10 Eric Dallal, Daniel Neider, and Paulo Tabuada. Synthesis of safety controllers robust to unmodeled intermittent disturbances. In *CDC 2016*, pages 7425–7430. IEEE, 2016. doi:10.1109/CDC.2016.7799416.

11 Ali Ebnenasir, Sandeep S. Kulkarni, and Anish Arora. FTSyn: a framework for automatic synthesis of fault-tolerance. *STTT*, 10(5):455–471, 2008. doi:10.1007/s10009-008-0083-0.

12 Rüdiger Ehlers and Ufuk Topcu. Resilience to intermittent assumption violations in reactive synthesis. In Martin Fränzle and John Lygeros, editors, *HSCC 2014*, pages 203–212. ACM, 2014. doi:10.1145/2562059.2562128.

13 Alain Girault and Éric Rutten. Automating the addition of fault tolerance with discrete controller synthesis. *Form. Meth. in Sys. Des.*, 35(2):190–225, 2009. doi:10.1007/s10703-009-0084-y.

14 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *LNCS*. Springer, 2002. doi:10.1007/3-540-36387-4.

15 Chung-Hao Huang, Doron A. Peled, Sven Schewe, and Farn Wang. A game-theoretic foundation for the maximum software resilience against dense errors. *IEEE Trans. Software Eng.*, 42(7):605–622, 2016. doi:10.1109/TSE.2015.2510001.

16 Rupak Majumdar, Elaine Render, and Paulo Tabuada. A theory of robust omega-regular software synthesis. *ACM Trans. Embedded Comput. Syst.*, 13(3):48:1–48:27, 2013. doi:10.1145/2539036.2539044.

17 Donald A. Martin. Borel determinacy. *Annals of Mathematics*, 102:363–371, 1975.

18 Daniel Neider, Alexander Weinert, and Martin Zimmermann. Synthesizing optimally resilient controllers. *arXiv*, 1709.04854, 2017. URL: https://arxiv.org/abs/1709.04854.

19 Paulo Tabuada, Sina Yamac Caliskan, Matthias Rungger, and Rupak Majumdar. Towards robustness for cyber-physical systems. *IEEE Trans. Automat. Contr.*, 59(12):3151–3163, 2014. doi:10.1109/TAC.2014.2351632.

20 Paulo Tabuada and Daniel Neider. Robust linear temporal logic. In *CSL 2016*, volume 62 of *LIPIcs*, pages 10:1–10:21. Schloss Dagstuhl - LZI, 2016. doi:10.4230/LIPIcs.CSL.2016.10.

21 Ufuk Topcu, Necmiye Ozay, Jun Liu, and Richard M. Murray. On synthesizing robust discrete controllers under modeling uncertainty. In Thao Dang and Ian M. Mitchell, editors, *HSCC 2012*, pages 85–94. ACM, 2012. doi:10.1145/2185632.2185648.

22 Johan van Benthem. An essay on sabotage and obstruction. In *Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday*, volume 2605 of *LNCS*, pages 268–276. Springer, 2005. doi:10.1007/978-3-540-32254-2_16.