2017 Int'l Conference on Cyberphysical Systems (ICCPS)
http://www.cds.caltech.edu/~murray/papers/dlm17-iccps.pdf

Enhancing Tolerance to Unexpected Jumps in GR(1) Games

Sumanth Dathathri
California Institute of Technology
sdathath@caltech.edu

Scott C. Livingston
slivingston@cds.caltech.edu

Richard M. Murray
California Institute of Technology
murray@cds.caltech.edu

ABSTRACT

When used as part of a hybrid controller, finite-memory strategies synthesized from linear-time temporal logic (LTL) specifications rely on an accurate dynamics model in order to ensure correctness of trajectories. In the presence of uncertainty about the underlying model, there may exist unexpected trajectories that manifest as unexpected transitions under control of the strategy. While some disturbances can be captured by augmenting the dynamics model, such approaches may be conservative in that bisimulations may fail to exist for which strategies can be synthesized. In this paper, we consider games of the GR(1) fragment of LTL, and we characterize the tolerance of hybrid controllers to perturbations that appear as unexpected jumps (transitions) to states in the discrete strategy part of the controller. As a first step, we show robustness to certain unexpected transitions that occur in a finite manner, i.e., despite a certain number of unexpected jumps, the sequence of states obtained will still meet a stricter specification and hence the original specification. Additionally, we propose algorithms to improve robustness by increasing tolerance to additional disturbances. A robot gridworld example is presented to demonstrate the application of the developed ideas and also to perform empirical analysis.

CCS CONCEPTS

•**Computing methodologies** →Artificial intelligence; •**Software and its engineering** →Formal methods;

KEYWORDS

Linear Temporal Logic, Formal Methods, Synthesis, Robustness

ACM Reference format:
Sumanth Dathathri, Scott C. Livingston, and Richard M. Murray. 2017. Enhancing Tolerance to Unexpected Jumps in GR(1) Games. In *Proceedings of The 8th ACM/IEEE International Conference on Cyber-Physical Systems, Pittsburgh, PA USA, April 2017 (ICCPS),* 11 pages.
DOI: http://dx.doi.org/10.1145/3055004.3055014

1 INTRODUCTION

The ability of strategies synthesized from formal specifications to be tolerant to unexpected events such as disturbances or failures is important, especially for safety-critical applications. Reactive strategies synthesized to meet temporal logic objectives are not error resilient by default. Even with non-critical disturbances that were not accurately modeled during synthesis, no guarantees can be provided about satisfaction of the temporal-formula used for synthesis. In some cases it may not be possible to synthesize a strategy if these perturbations are fully modelled through the dynamics.

In this paper, we make progress towards enhancing the tolerance of strategies synthesized to satisfy specifications in the generalized reactivity(1) (GR(1)) fragment of linear-temporal logic (LTL) [11, 12]. GR(1) formulae are considered because they are quite expressive in terms of temporal properties captured, yet symbolic synthesis algorithms are possible with relatively low computational complexity [3, 7, 10]. The first result we show is that by refining a strategy synthesized to satisfy a GR(1) formula, a strategy that is winning against a stricter formula can be generated and is robust to certain unexpected events. Then, exploiting this tolerance, we propose multiple algorithms that combine separately synthesized strategies to form a single robust winning strategy. It is often desired that the system can recover from these unexpected events and this be done without resynthesizing the entire strategy again. In this regard, we propose an approach which lets us recover from these unexpected events without a complete resynthesis.

Understanding the response of systems to perturbations has been extensively studied in control theory and more recently, for reactive controllers and their synthesis. In [9, 13], the robustness considered is in terms of bounded input-output deviation. This relates directly to the prevalent notion in control for robustness [16], where controllers are designed to ensure bounded disturbances lead to bounded deviations from nominal-behavior for the system. In the current work, the tolerance to disturbances is in the form of satisfaction of a formula representing the desired system behavior. In [2], the effect of disturbances on system behavior is quantified and the focus on synthesizing robust systems that degrade gracefully - smallest number of system failures possible but not primarily directed at GR(1) specifications. Some existing work on notions of robustness in terms of satisfaction or violation of a formula can be found in [1, 14].

In [4], they use a similar underlying idea to the one in the current work to re-synthesize a strategy against a new GR(1) formula that is more robust. In scenarios where unforeseen perturbations occur when the controller is implemented on the hybrid system, the results presented in our paper allow for continued execution with guarantees in terms of formula satisfaction. In [15], motivated by a similar idea, they propose an approach which uses the saved results from an intermediate step during synthesis to construct sequences of control actions that can tolerate violations of environmental safety assumptions. However, the work presented here differs in that the perturbations tolerated are more general. Additional guarantees are provided, for instance, where there is a failure on the system end. This has other potential implications as discussed in Section 6. Additionally, an approach to recover from both system failures and environmental assumption violations when possible

ICCPS, Pittsburgh, PA USA

DOI: http://dx.doi.org/10.1145/3055004.3055014

is proposed that does not involve enumeration of an entire GR(1) strategy as in [15].

In summary, the main contributions of this work are the following: 1) to characterize the inherent tolerance of GR(1) strategies to unexpected events; 2) to propose approaches to refine enumerated GR(1) strategies to augment their tolerance to unexpected perturbations and prove that they satisfy a stricter temporal formula; 3) to quantify empirically the cost of augmenting the tolerance using the proposed schemes.

2 PRELIMINARIES

Let Σ be a finite set. The power set of Σ (i.e., the set of all subsets of Σ) is denoted by $\mathcal{P}(\Sigma)$. The set of all finite strings formed from concatenating elements of Σ is denoted by Σ^*, which is known as the Kleene closure [6]. The set of all countably infinite strings of Σ is Σ^ω. In this paper, a subscript notation is used, e.g., $\sigma_0\sigma_1\sigma_2\cdots\sigma_n \in \Sigma^*$, but observe that infinite strings can also be regarded as functions of the natural numbers $\mathbb{N}$ into Σ.

Let $\mathrm{AP}_{\mathrm{env}}$ be a set of input atomic propositions, and $\mathrm{AP}_{\mathrm{sys}}$ be a set of output atomic propositions such that $\mathrm{AP}_{\mathrm{env}} \cap \mathrm{AP}_{\mathrm{sys}} = \emptyset$. A state s is an assignment of `True` and `False` to the atomic propositions in $\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}}$. We use subset notation to indicate states and thus, for brevity, introduce $\Sigma = \mathcal{P}(\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}})$.

A *nondeterministic finite-memory strategy* is a pair (f, m_0) together with a finite set M, where $m_0 \in M$ and

$$f : M \times \mathcal{P}(\mathrm{AP}_{\mathrm{env}}) \to \mathcal{P}\left(M \times \mathcal{P}(\mathrm{AP}_{\mathrm{sys}})\right) \tag{1}$$

is a function. Intuitively the set M represents the memory of the strategy. At each move, a new output is given depending on the input and the current memory value. As part of the move, a memory value is selected. Since we are only concerned with finite-memory strategies in this paper, we refer to them as *strategies*. The set of input-output sequences that may occur under f is defined as

$$\mathrm{Plays}(f) = \Big\{\sigma \in \Sigma^\omega \mid \exists m \in M^\omega . \forall k \geq 0. \\ (m_{k+1}, \sigma_k \cap \mathrm{AP}_{\mathrm{sys}}) \in f(m_k, \sigma_k \cap \mathrm{AP}_{\mathrm{env}})\Big\}, \tag{2}$$

where every $m \in M^\omega$ has the same first element, m_0. Elements of $\mathrm{Plays}(f)$ are referred to as *plays*. The set of prefixes that may be extended into a play is

$$\mathrm{Pref}(f) = \{\sigma \in \Sigma^* \mid \exists \alpha \in \Sigma^\omega . \sigma\alpha \in \mathrm{Plays}(f)\}. \tag{3}$$

We describe specifications for these strategies in linear-time temporal logic (LTL) [12]. LTL formulae over propositions $(\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}})$ are evaluated over positions i in $\sigma \in \Sigma^\omega$, where $\sigma = \sigma_0\sigma_1\cdots$. In addition to the Boolean operators, the standard LTL operators $\Box$ (always), $\Diamond$ (eventually) and $\bigcirc$ (next) are used here for the specification.

A finite-memory strategy (f, m_0) is said to be

- *input-enabled* if and only if for every $\sigma^{\mathrm{env}} \in \mathcal{P}(\mathrm{AP}_{\mathrm{env}})^\omega$, there exists $\sigma \in \mathrm{Plays}(f)$ such that $\sigma^{\mathrm{env}}_k = \sigma_k \cap \mathrm{AP}_{\mathrm{env}}$ for $k \geq 0$.
- *deterministic* if and only if $|f(m,s)| \leq 1$ for all $(m,s) \in M \times \mathcal{P}(\mathrm{AP}_{\mathrm{env}})$.
- a *realization of* an LTL formula φ if and only if (f, m_0) is input-enabled and $\mathrm{Plays}(f) \subseteq L(\varphi)$, where $L(\varphi)$ is the language of φ . This is to say, for every $\sigma \in \mathrm{Plays}(f)$, $\sigma \models \varphi$.

Deterministic strategies imply a function from state sequences to memory sequences.

Remark 1. *Let (f, m_0) be a deterministic strategy. For each $\sigma \in \mathrm{Plays}(f)$, there exists a unique $m \in M^\omega$ satisfying $f(m_k, \sigma_k \cap \mathrm{AP}_{\mathrm{env}}) = \left\{(m_{k+1}, \sigma_k \cap \mathrm{AP}_{\mathrm{sys}})\right\}$ for $k \geq 0$.*

It follows from the remark that a sequence of inputs determines precisely one output sequence for a deterministic strategy.

A GR(1) formula is an LTL formula of the form

$$\Theta^{\mathrm{env}} \wedge \Box\rho^{\mathrm{env}} \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\psi_j^{\mathrm{env}}\right) \\ \implies \Theta^{\mathrm{sys}} \wedge \Box\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box\Diamond\psi_k^{\mathrm{sys}}\right), \tag{4}$$

where Θ^{env} is a state formula (i.e., without temporal operators) that is a function of $\mathrm{AP}_{\mathrm{env}}$, Θ^{sys} is a state formula that is a function of $\mathrm{AP}_{\mathrm{sys}}$, and all ψ_j^{env}, ψ_k^{sys} subformulae are functions of $\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}}$ and also without temporal operators. The subformula ρ^{env} is a function of $\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}} \cup \bigcirc\mathrm{AP}_{\mathrm{env}}$, where

$$\bigcirc\mathrm{AP}_{\mathrm{env}} = \{\bigcirc x \mid x \in \mathrm{AP}_{\mathrm{env}}\}.$$

Except for $\bigcirc$ operators appearing as subformulae from $\bigcirc\mathrm{AP}_{\mathrm{env}}$, there are no other temporal operators in ρ^{env}. Finally, ρ^{sys} is defined similarly to ρ^{env} but as a function of $\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}} \cup \bigcirc\mathrm{AP}_{\mathrm{env}} \cup \bigcirc\mathrm{AP}_{\mathrm{sys}}$.

To facilitate working with (4), and in particular the subformulae ρ^{env} and ρ^{sys}, we extend the semantics of the operator $\models$ for finite strings. For a finite string γ, by γ_{-1} we refer to the last element of γ. In other words, $\gamma_{-1} = \gamma_{|\gamma|-1}$. Let $\sigma \in \Sigma^*$. To further simplify notation, the superscripts sys and env for $s \in \Sigma$ indicate projections on to $\mathrm{AP}_{\mathrm{sys}}$ and $\mathrm{AP}_{\mathrm{env}}$ respectively. That is, given s, $s^{\mathrm{sys}} := s \cap \mathrm{AP}_{\mathrm{sys}}$ and $s^{\mathrm{env}} := s \cap \mathrm{AP}_{\mathrm{env}}$. Define

$$\sigma \models \rho \quad \iff \quad \sigma\alpha \models \rho \text{ for any } \alpha \in \Sigma^\omega, \tag{5}$$

where ρ is any Boolean formula that is a function of $\mathrm{AP}_{\mathrm{env}} \cup \mathrm{AP}_{\mathrm{sys}} \cup \bigcirc\mathrm{AP}_{\mathrm{env}} \cup \bigcirc\mathrm{AP}_{\mathrm{sys}}$. Because at most one $\bigcirc$ operator binds to each atomic proposition, it follows that only σ_0, σ_1 determine whether the formula is satisfied.

Let φ be a GR(1) formula, i.e., be of the form (4), and let (f, m_0) be a finite-memory strategy.

Definition 2. *A state $s \in \Sigma$ is said to be φ-reachable under (f, m_0) if and only if there exists $\sigma \in \mathrm{Plays}(f)$ such that for some $k \geq 0$,*

$$\sigma_k = s, \tag{6}$$

$$\sigma \models \Theta^{\mathrm{env}}, \tag{7}$$

$$\sigma_{j:(j+1)} \models \rho^{\mathrm{env}} \text{ for } j < k-1. \tag{8}$$

The set of all states that are φ-reachable under (f, m_0) is denoted by $I_\varphi(f, m_0)$.

The finite-memory strategy (f, m_0) is said to be a *strict realization* of (or to *strictly realize*) the GR(1) formula φ if it is a realization of

φ and for all $\sigma \in \text{Plays}(f)$

$$\sigma \models \Theta^{\text{env}} \implies \sigma \models \Theta^{\text{sys}}, \tag{9}$$

$$\sigma \models (\boxdot\rho^{\text{env}} \implies \boxdot\rho^{\text{sys}}). \tag{10}$$

Intuitively, strict realizability ensures that blocking of an environment liveness condition when the other assumptions are met only occurs when the system is following transition rules. Here, $\boxdot$ is the 'historically' LTL operator whose semantics are as defined in [10].

3 PROBLEM FORMULATION

3.1 LTL on state-memory pairs

A basic idea behind the methods presented in later sections is to begin with a given finite-memory strategy and construct another one that is more robust. The construction involves re-using "pieces" of the given strategy, such as memory-values and transitions. As such, the memory values occurring during each play become important for reasoning about correctness and resilience to perturbations. Thus motivated, the semantics of $\models$ for LTL as invoked in Section 2 is extended to handle memory values of strategies. Let (f, m_0) be a strategy that strictly realizes φ, and has set M of memory values. Following the notation used in the previous section, let $\Sigma = \mathcal{P}(\text{AP}_{\text{env}} \cup \text{AP}_{\text{sys}})$ be the set of game states. Define a set of state-memory pairs,

$$\bar{\Sigma} = \Sigma \times M.$$

An element $\bar{\sigma} \in \bar{\Sigma}^{\omega}$ is thus a sequence of state and memory pairs, i.e., there is $\sigma \in \Sigma^{\omega}$ and $m \in M^{\omega}$ such that for $k \geq 0$, $\bar{\sigma}_k = (\sigma_k, m_k)$. Given an LTL formula ϕ that is in terms of $\text{AP}_{\text{env}} \cup \text{AP}_{\text{sys}}$ and a variable that takes values in M, the operator $\models$ can thus be interpreted on $\bar{\sigma}$.

3.2 Memory sequences associated with plays

Because properties will be expressed in terms of sequences of states and memory values that occur during plays, a means for obtaining memory values that can occur in the strategy during a given play is needed. For this purpose, define the function Mem^f on $\text{Pref}(f) \cup \text{Plays}(f)$ as follows. Let $\sigma \in \text{Pref}(f) \cup \text{Plays}(f)$. Define $\sigma_k^{\text{sys}} := \sigma_k \cap \text{AP}_{\text{sys}}$ and $\sigma_k^{\text{env}} := \sigma_k \cap \text{AP}_{\text{env}}$. If $\sigma \in \text{Pref}(f)$, then define

$$\text{Mem}^f(\sigma) = \{m \in M^* \mid \forall k \geq 0, k < |\sigma|. \\ (m_{k+1}, \sigma_k^{\text{sys}}) \in f(m_k, \sigma_k^{\text{env}})\}. \tag{11}$$

If $\sigma \in \text{Plays}(f)$, then define

$$\text{Mem}^f(\sigma) = \{m \in M^{\omega} \mid \forall k \geq 0. \\ (m_{k+1}, \sigma_k^{\text{sys}}) \in f(m_k, \sigma_k^{\text{env}})\}. \tag{12}$$

Observe that $\text{Pref}(f) \cap \text{Plays}(f) = \emptyset$ (recall (2) and (3)), hence Mem^f is well-defined.

3.3 Problem of recovery from perturbations

Though there are many distinct notions of robustness for control systems, a common theme is tolerance to deviation from nominal plant behavior. Note that the "plant" includes actuators and sensors on the robot itself. Thus in the context of reactive synthesis, deviance can also arise on the side of the controlled system, not only the adversarial environment.

We begin by introducing a few definitions first. Define the set $I_{\varphi}^{M}(f, m_0) \subseteq \bar{\Sigma}$ as

$$I_{\varphi}^{M}(f, m_0) := \{(s, \bar{m}) \mid s \in I_{\varphi}(f, m_0), \bar{m} = m_{|\sigma s|}. \\ \sigma \in \Sigma^*, m \in M^*. \\ \sigma s \in Pref(f) \wedge \\ m \in \text{Mem}^f(\sigma s) \wedge \\ \forall k < |\sigma|. \sigma_k \in I_{\varphi}(f, m_0)\}.$$

$I_{\varphi}^{M}(f, m_0)$ contains the set of φ-reachable states along with the corresponding memory locations. Given that we have the set of φ-reachable states, we need to extract all paths to these φ-reachable states that have only φ-reachable states to get all possible combinations of memory, φ-reachable states to which the system can be perturbed to. Given a φ-reachable state s, we find a prefix (σs) that ends with this state and this prefix has only φ-reachable states (Condition: $\forall k < |\sigma|. \sigma_k \in I_{\varphi}(f, m_0)$). Once this prefix has been found, we generate the memory sequence in accordance with (f, m_0) to get the corresponding memory at this state in the strategy (Condition: $\forall k < |\sigma|. \left(m_{k+1}, (\sigma s)_{k+1} \cap \text{AP}_{\text{sys}}\right) \in f\left(m_k, (\sigma s)_{k+1} \cap \text{AP}_{\text{env}}\right)$).

Using the definitions above, a function is constructed that indicates feasible transitions (in a sense to be made precise) that are not in the given strategy f. This function is crucial for studying perturbations and applications of a given strategy that are extrinsic to its original semantics. To this end, define the set of all state-memory pairs that can be reached by some play of f, i.e.,

$$I(f) = \Big\{(s, w) \mid \exists \sigma \in \text{Plays}(f), \exists m \in \text{Mem}^f(\sigma) : \\ \exists k : s = \sigma_k \wedge w = m_{k+1}\Big\}. \tag{13}$$

Using this definition of reachable state-memory pairs, we introduce the notion of a perturbation for a strategy (f, m_0) that realizes φ.

Definition 3. *A perturbation for a controller implementing the finite-memory strategy (f, m_0) occurs when the system transitions from a state-memory pair $(s, w) \in I(f)$ to a state s' in Σ such that $ss' \not\models \rho^{\text{env}}$ or for all $w' \in M$ $(w', s' \cap \text{AP}_{\text{sys}}) \notin f(w, s' \cap \text{AP}_{\text{env}})$.*

The condition $ss' \models \neg\rho^{\text{env}}$ corresponds to the environment violating an assumption on its behavior. And $\forall w' \in M : (w', s' \cap \text{AP}_{\text{sys}}) \notin f(w, s' \cap \text{AP}_{\text{env}})$ corresponds to a disturbance during the application of an output action. This work proposes approaches that enable recovery from certain such perturbations.

Define the function ExtTs on $I(f)$ as follows. For $(s, w) \in I(f)$,

$$\text{ExtTs}(s, w) = \{(s', w') \mid s' \in \Sigma, w' \in M. \\ (s', w') \in I_{\varphi}^{M}(f, m_0) \wedge \\ ss' \models \rho^{\text{sys}} \wedge \\ \Big((w', (s')^{\text{sys}}) \notin f(w, (s')^{\text{env}}) \vee ss' \models \neg\rho^{\text{env}}\Big)\}.$$

The function ExtTs maps a state-memory pair (s, m) in a strategy to all state-memory pairs (s', m') that occur in $I_{\varphi}^{M}(f, m_0)$ that are not immediate successors according to (f, m_0), or violate the assumption on environmental safety (Condition: $(w', s' \cap \text{AP}_{\text{sys}}) \notin f(w, s' \cap \text{AP}_{\text{env}}) \vee ss' \models \neg\rho^{\text{env}}$). An additional constraint is imposed where it is required that the transition between the state-memory

pair (s,m) and any state-memory pair $((s',m') \in \text{ExtTs}(s,m))$ to which it is mapped to does not violate ρ^{sys} i.e $ss' \models \rho^{\text{sys}}$.

3.4 Refinement of a strategy

Given φ and (f, m_0) we construct a nondeterministic strategy $(\bar{f}, m_0)$ with $\bar{f} : \mathcal{P}(\text{AP}_{\text{env}})^\omega \times M \to \mathcal{P}(\bar{\Sigma}^\omega)$. Because the strategy is nondeterministic, for a given sequence of inputs, there is a set of state-memory sequences generated.

$$\text{Plays}(\bar{f}) = \Big\{ \sigma \in \Sigma^\omega \mid \exists m \in M^\omega . \big(\forall k > 0 . (m_{k+1}, \sigma_k^{\text{sys}}) \in f(m_k, \sigma_k^{\text{env}}) \vee (\sigma_k, m_{k+1}) \in \text{ExtTs}(\sigma_{k-1}, m_k) \big) \wedge (m_1, \sigma_0^{\text{sys}}) \in f(m_0, \sigma_0^{\text{env}}) \Big\}. \quad (14)$$

Observe that the successor memory-state pair (m', s') in $\text{Plays}(\bar{f})$ depends on the current memory-state pair (m,s) and the next input $s_{input} = s' \cap \text{AP}_{\text{env}}$. By augmenting the memory-values with elements from Σ, new memory-values can be created that $\bar{f}$ can use as memory inputs. Given a memory-state pair and an input in $\mathcal{P}(\text{AP}_{\text{env}})$, $\bar{f}$ can map to an output in $\mathcal{P}(\text{AP}_{\text{sys}})$ and a memory-value in the augmented memory domain. This way $\text{Plays}(\bar{f})$ can be used to construct a strategy $\bar{f}$ (in the conventional sense) that maps from a memory value and an input to a memory value and an output.

Notice that here m is the memory sequence corresponding to σ in accordance with $\bar{f}$.

To simplify notation, for $(s,m) \in \bar{\Sigma}$ introduce the indicator formula

$$\mathbf{1}_{(s,m)} := \bigwedge_{p \in s \cup \{m\}} p \quad \wedge \bigwedge_{q \in \text{AP}_{\text{env}} \cup \text{AP}_{\text{sys}} \cup M \setminus s \cup \{m\}} \neg q,$$

where M is a finite universal set of memory values. The indicator formula $\mathbf{1}_{(s,m)}$ evaluates to `True` at $(x,y) \in \Sigma \times M$ if and only if $x = s$ and $y = m$. Define 1_{jump} as

$$1_{jump} := \bigwedge_{(s,m) \in dom(\text{ExtTs})} \Big(\mathbf{1}_{(s,m)} \rightarrow \bigcirc \bigvee_{(s',m') \in \text{ExtTs}(s,m)} \mathbf{1}_{(s',m')} \Big).$$

Proposition 4. *Let $\sigma \in \text{Plays}(\bar{f})$ and $m \in M^\omega$ such that it satisfies the conditions in* (14)*, and let $\bar{\sigma} \in \bar{\Sigma}^\omega$ such that $\forall k \geq 0 . \bar{\sigma}_k = (\sigma_k, m_{k+1})$. Then,*

$$\bar{\sigma} \models \Theta^{\text{env}} \wedge \square(\rho^{\text{env}} \vee 1_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \square\diamond \psi_j^{\text{env}} \right) \wedge \diamond\square\neg 1_{jump} \implies \Theta^{\text{sys}} \wedge \square\rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K} \square\diamond \psi_k^{\text{sys}} \right). \quad (15)$$

This proposition arises as a special case of Proposition 5 where $n = 0$ in the setup before Proposition 5. Proposition 5 proposes an approach to combine multiple strategies to provide tolerance to perturbations, when there is just one strategy the current proposition results.

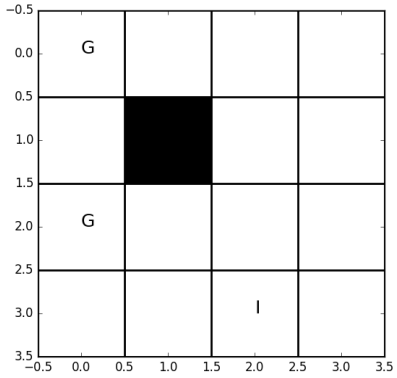


Figure 1: Gridworld setup

Intuitively, the practical significance of Proposition 4 is that, if there is a disturbance that causes an unexpected transition to some state that is φ-reachable in other plays and if there are only finitely many such disturbances, then execution of the finite-state machine can continue after an appropriate change of its internal state (memory value) and still result in a correct input-output sequence. If ρ^{env} is violated during a particular transition between a state s and its successor s', i.e, $ss' \not\models \rho^{\text{env}}$, and s' is φ-reachable, then a sequence of input-outputs that satisfies the system part of φ is still possible if there are finitely many such disturbances. It also allows for such disturbances during application of the output action to a hybrid system – where $(m', s' \cap \text{AP}_{\text{sys}}) \notin f(m, s' \cap \text{AP}_{\text{env}})$. Here m is the current memory value and $s' \cap \text{AP}_{\text{env}}$ the input and $s' \cap \text{AP}_{\text{sys}}$ the ground truth system state after the application of the output action.

However, this result not does imply that all disturbances in which the system fails to transition to the desired state or the transition rule ρ^{env} is violated can be handled. Only perturbations to φ-reachable states that do not violate ρ^{sys} are tolerated.

The added $\diamond\square\neg 1_{jump}$ segment on the assumption side in (15) ensures that the perturbations do not occur infinitely often.

4 AUGMENTING ROBUSTNESS

To illustrate the methods proposed in this section to augment robustness, we begin with a small deterministic example. Consider planar robotic motion planning on a 4×4 grid. In the example instance shown in Figure 1, the robot begins in the cell marked 'I' and has to visit the cells marked 'G' infinitely often while avoiding the shaded walls. The robot can transition to any of its non-diagonally adjacent cells.

For a given gridworld, two strategies satisfying the safety and progress requirements but differing in their initial poses are synthesized. While the robot was executing Strategy 1 (in Figure 2), a perturbation causes the robot to transition from the cell $Y = (Y_r, Y_c) = (0,1)$ with memory $m = 11$ to $(0,2)$ where Y_r refers to the row position of the robot and Y_c refers to the column position. This transition is not desired according to Strategy 1 but is still safe for the system, i.e, ρ^{sys} is still satisfied by the move of the robot from $(0,1)$ to $(0,2)$. However, the strategy fails to predict what the

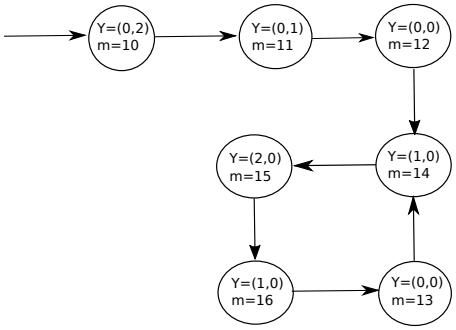
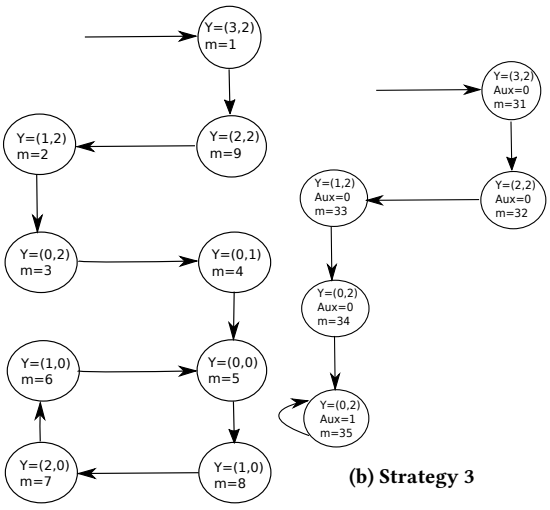


Figure 2: Strategy 1



(a) Strategy 2 **(b) Strategy 3**

Figure 3: Synthesized strategies to augment robustness

sequence of actions from there should be such that φ can be satisfied. Proposition 5 guarantees that continued execution along strategy 2 from the state $(0,2)$ with memory $m = 3$ would satisfy φ for the robot. Here, Strategy 2 (in Figure 3a) was synthesized to satisfy a specification similar to the one used to synthesize Strategy 1.

Now consider a scenario where it is not desirable to resynthesize or enumerate a new strategy. We wish to find a sequence of actions to a φ-reachable state visited by Strategy 1 such that continued execution along Strategy 1 satisfies φ. Proposition 6 guarantees the correctness of the algorithm to find such a sequence of actions. For this example, Strategy 3 (in Figure 3b) depicts one such path that was synthesized starting from $(3,2)$ reconnecting to a φ-reachable state. The reader must note that the absence of an adversary in this example makes the recovery process trivial, but solving a reachability game for general GR(1) formulae so as to reconnect to a φ-reachable state in the original strategy is more involved.

4.1 Combining Multiple Strategies

In this section, the intuition from Section 3.4 is used and a more general proposition is presented and proved. The results in this section allow for the concatenation of multiple strategies synthesized with formulae differing in the initial condition.

Let φ_0 be a GR(1) formula, and let $(g_0, m_0) : M_0 \times \mathcal{P}(\mathrm{AP}_{\mathrm{env}}) \to \mathcal{P}\left(M_0 \times \mathcal{P}(\mathrm{AP}_{\mathrm{sys}})\right)$ be a strategy that realizes φ_0. Let $\eta_1, \eta_2, \ldots, \eta_n$ be the additional states in Σ to which the system is likely to be perturbed to. The strategy must visit these states for the system to be able to recover after being perturbed to these states. For $i \in \{1,2,\ldots,n\}$ where $n < \infty$, define Θ_i^{env} as a Boolean formula which is `True` for a state s in Σ if and only if $s \cap \mathrm{AP}_{\mathrm{env}} = \eta_i^{\mathrm{env}}$. Similarly, define Θ_i^{sys} as a Boolean formula which is `True` for a state s in Σ if and only if $s \cap \mathrm{AP}_{\mathrm{sys}} = \eta_i^{\mathrm{sys}}$.

Then, construct a set of strategies (g_i, m_0^i) such that for each $i \in \{1,2,\ldots,n\}$, (g_i, m_0^i) strictly realizes φ_i, where

$$\varphi_i = \Theta_i^{\mathrm{env}} \wedge \Box\rho^{\mathrm{env}} \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\psi_j^{\mathrm{env}}\right) \implies \Theta_i^{\mathrm{sys}} \wedge \Box\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box\Diamond\psi_k^{\mathrm{sys}}\right). \quad (16)$$

To simplify notation, without loss of generality, let $g_i : M_i \times \mathcal{P}(\mathrm{AP}_{\mathrm{env}}) \to \mathcal{P}\left(M_i \times \mathcal{P}(\mathrm{AP}_{\mathrm{sys}})\right)$ such that $\forall i, j \in \{0,1,\ldots,n\}, i \neq j \to M_i \cap M_j = \emptyset$.

Let $\bar{M} = \bigcup_{i=0,1,2,\ldots,n} M_i$. Define a partial function $\mathrm{ExtTs} : \bar{\Sigma} \times \bar{M} \to \mathcal{P}(\bigcup_{i=0,1,2,\ldots,n} I_\varphi^M(g_i, m_0^i))$ such that for $(s,w) \in I(g_i)$ for some $i \leq n$,

$$\mathrm{ExtTs}(s,w) = \{(s',w') \mid s' \in \Sigma, w' \in M_i, i \in \{0,1,\ldots,n\}. (s',w') \in I_{\varphi_i}^M(g_i, m_0^i) \wedge ss' \models \rho^{\mathrm{sys}} \wedge \left((w', s' \cap \mathrm{AP}_{\mathrm{sys}}) \notin g_i(w, s' \cap \mathrm{AP}_{\mathrm{env}}) \vee ss' \models \neg\rho^{\mathrm{env}}\right)\}.$$

Define $\bar{1}_{jump}$ as

$$\bar{1}_{jump} := \bigwedge_{(s,m) \in dom(\mathrm{ExtTs})} \left(\mathbf{1}_{(s,m)} \to \bigcirc \bigvee_{(s',m') \in \mathrm{ExtTs}(s,m)} \mathbf{1}_{(s',m')}\right).$$

4.1.1 Robust Strategy from combining strategies. Construct a nondeterministic strategy $(\bar{g}, m_0^0)$.

$$\mathrm{Plays}(\bar{g}) = \Big\{\sigma \in \Sigma^\omega \mid \exists m \in \bar{M}^\omega . \forall k > 0 . \exists i. (m_{k+1}, \sigma_k^{\mathrm{sys}}) \in g_i(m_k, \sigma_k^{\mathrm{env}}) \vee \Big((\sigma_k, m_{k+1}) \in \mathrm{ExtTs}(\sigma_{k-1}, m_k)\Big) \wedge (m_1, \sigma_0^{\mathrm{sys}}) \in g_0(m_0^0, \sigma_0^{\mathrm{env}})\Big\}. \quad (17)$$

Consider $\sigma \in \mathrm{Plays}(\bar{g})$ and $m \in (\bar{M})^\omega$ that satisfy the conditions in (17). Let $\bar{\sigma} \in \bar{\Sigma}^\omega$ such that $\forall k \geq 0 . \bar{\sigma}_k = (\sigma_k, m_{k+1})$. Here, m is

Algorithm 1 Multi-strategy combination (Section 4.1)

Input:
- Finite-memory strategies $(g_i, m_0^i)\ \forall i \in \{0,1,2,\ldots,n\}$,
- Sequence of inputs $\sigma^{\text{env}} \in \text{AP}_{\text{env}}^{\omega}$,
- A system to which the sequence $\sigma^{\text{sys}} \in (\mathcal{P}(\text{AP}_{\text{env}}))^{\omega}$ can be applied to and its state $s \in \Sigma$ after the control action has been applied
- Set $I_{\varphi_i}^M(g_i, m_0^i)$ and function ExtTs

Output: Sequence of control actions $\sigma^{\text{sys}} \in \mathcal{P}(\text{AP}_{\text{sys}}^{\omega})$

1: memory=m_0^0, i=1, safety=1, l=0
2: (memoryNew, σ_0^{sys}) = Strategy$f$: (memory, σ_0^{env})
3: **while** (True) **do**
4: **if** $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ and safety=1 **then**
5: Choose (memoryNew, σ_i^{sys})$\in f_l$: (memory, σ_i^{env})
6: Run: SafetyCheck
7: **else if** safety=0 $\wedge$ $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ **then**
8: **if** $\exists j \leq n, m \in \bar{M}.(m, \sigma_{i-1}^{\text{env}} \cup \sigma_{i-1}^{\text{sys}}) \in I_{\varphi_j}^M(f_j, m_0^j)$ **then**
9: Choose (memoryNew,σ_i^{sys}) $\in f_j(m, \sigma_i^{\text{env}})$
10: $l = j$
11: Run: SafetyCheck
12: **else**
13: EXIT
14: **end if**
15: **else if** $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \not\models \rho^{\text{env}}$ **then**
16: **if** $\exists j \leq n, m \in \bar{M}_j, s \in \mathcal{P}(\text{AP}_{\text{sys}}).(m, \sigma_i^{\text{env}} \cup s) \in$ ExtTs($memory, \sigma_{i-1}^{\text{env}} \cup \sigma_{i-1}^{\text{sys}}$) **then**
17: (memoryNew,σ_i^{sys}) = (m, s)
18: $l = j$
19: Run: SafetyCheck
20: **else**
21: EXIT
22: **end if**
23: **end if**
24: i+=1, memory=memoryNew
25: **end while**

Algorithm 2 SafetyCheck

1: apply σ_i^{sys}, measure s
2: **if** $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})s \not\models \rho^{\text{sys}}$ **then**
3: EXIT
4: **end if**
5: **if** $(\sigma_i^{\text{env}} \sigma_i^{\text{sys}}) = s$ **then**
6: safety=1
7: **else**
8: safety=0, σ_i^{env}=$s \cap \text{AP}_{\text{env}}$, σ_i^{sys}=$s \cap \text{AP}_{\text{sys}}$
9: **end if**

the memory-sequence corresponding to σ generated in accordance with $\bar{g}$.

Proposition 5. *For all such $\bar{\sigma}$, the following holds:*

$$\bar{\sigma} \models \Theta_0^{\text{env}} \wedge \square(\rho^{\text{env}} \vee \bar{1}_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \square\diamond\psi_j^{\text{env}} \right) \wedge \diamond\square\neg\bar{1}_{jump} \implies \Theta_0^{\text{sys}} \wedge \square\rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K} \square\diamond\psi_k^{\text{sys}} \right). \quad (18)$$

A proof is provided in the appendix.

Algorithm 1 gives a formal description of the implementation of a controller on a 'plant' that exploits the nondeterminism in the strategy $\bar{g}$ to tolerate certain perturbations. Algorithm 2 measures the state of the system after the application of the output action. If ρ^{sys} is violated because of a disturbance, execution is terminated. When there is a system-actuation failure i.e $\left((w', s' \cap \text{AP}_{\text{env}}) \notin g_j(w, s' \cap \text{AP}_{\text{sys}})\right)$, it is checked if the resulting state s' corresponds to a transition that is permissible in accordance with $\bar{g}$. If so, the memory is reset to an appropriate memory such that $(s', w') \in I_{\varphi_i}^M$ and execution is continued. At a state s, if the next input $(s')^{\text{env}} \in \mathcal{P}(\text{AP}_{\text{env}})$ is such that $s(s')^{\text{env}} \models \neg\rho^{\text{env}}$ then we choose $(s')^{\text{sys}}$ as indicated by ExtTs. In this case we continue execution from a state indicated by ExtTs as continuing execution along the original strategy would give no guarantees. By choosing to deviate in accordance with Proposition 5, we are able to provide guarantees about system behavior. To summarize, for a set of strategies $\{g_i : i \leq n\}$, we proposed an algorithm that can tolerate an augmented set of perturbations finitely many times by allowing for hops between the different strategies.

Informal descriptions of several parts of Algorithm 1:

- Line 4–6: Check for successful actuation and satisfaction of assumptions on environment, and continue along original strategy.
- Line 7–14: Check for actuation failures and satisfaction of environmental assumption violation, and continue along altered path.
- Line 15–21: Check for environmental assumption violation, and continue execution along altered path.

4.2 Building patches to handle perturbations

This section proposes an approach to add 'patches' to a strategy that enable recovery from perturbations by finding a safe sequence of states back to the original strategy. As opposed to the previous section where complete strategies were synthesized, here only a recovery patch is synthesized. Let $(h^0, m_0^0) : M_0 \times \mathcal{P}(\text{AP}_{\text{env}}) \to \mathcal{P}(\text{AP}_{\text{sys}})(M_0 \times \mathcal{P}(\text{AP}_{\text{sys}}))$ be a finite-memory strategy that strictly realizes a GR(1) formula φ. For $i \in \{1,2,\ldots,n\}$, let η_i be a state in Σ such that $\eta_i \notin \text{I}(h^0, m_0^0)$. Define $\Theta_i^{\text{env}}, \Theta_i^{\text{sys}}$ corresponding to η_i as in Section 4.1. Define $\mathcal{T}_{reach}$ as a Boolean formula that evaluates to True at a state s in Σ if and only if $s \in I_{\varphi}(h^0)$. Define φ_{reach}^i as below:

$$\varphi_{reach}^i := \left(\Theta_i^{\text{env}} \wedge \square\rho^{\text{env}} \wedge \left(\bigwedge_{k=1}^{K} \square\diamond\psi_k^{\text{env}} \right) \right) \implies \left(\Theta_i^{\text{sys}} \wedge \square\rho^{\text{sys}} \wedge \diamond\mathcal{T}_{reach} \right).$$

For $i \leq n$, let $(h^i, m_0^i) : M_i \times \mathcal{P}(\mathrm{AP_{env}}) \to \mathcal{P}(\mathrm{AP_{sys}})(M_i \times \mathcal{P}(\mathrm{AP_{sys}}))$ be a finite-memory strategy that strictly realizes(defined as for the GR(1) specification) φ^i_{reach}. Define $\bar{M}$ as before and let $M_i \cap M_j = \emptyset$ for any $i, j \leq n$ with $i \neq j$. Note that φ^i_{reach} can be converted to a GR(1) formula by introducing an auxillary variable as in [3]. Define $I^{reach}_{\varphi^i_{reach}}$ as

$$\begin{aligned} I^{M-reach}_{\varphi^i_{reach}} := \{(s, \bar{w}) \mid s \in I_{\varphi^i_{reach}}(h^i, m_0^i), \bar{w} = m_{|\sigma s|}. \\ \sigma \in \Sigma^*, m \in M_i^*. \\ \sigma s \in \mathrm{Pref}(h^i) \wedge \\ m \in \mathrm{Mem}^{h^i}(\sigma s) \wedge \\ \forall k < |\sigma|.(\sigma_k \in I_{\varphi^i_{reach}}(h^i, m_0^i) \wedge \sigma_k \models \neg \mathcal{T}_{reach})\}. \end{aligned}$$

This definition is similar to that of I^M_φ except for the additional constraint where we require that all states in the prefix are such that they are not from $I_\varphi(h^0, m_0^0)$.

This is to allow pertubation to states in sequences before $\mathcal{T}_{reach}$ was satisfied. This is because if a sequence of states in $\mathrm{Plays}(h^i)$ (for $i > 0$) satisfies $\Theta^{\mathrm{env}}_i \wedge \Box\rho^{\mathrm{env}} \wedge (\bigwedge_{k=1}^K \Box\Diamond\psi^{\mathrm{env}}_k)$ then it satisfies $\Diamond\mathcal{T}_{reach}$. Since we are synthesizing a patch back to the original strategy, jumping to a state in $I^{M-reach}$ ensures that if the environment satisfies the assumptions on safety and liveness, we will reach a state where $\mathcal{T}_{reach}$ holds.

Next, define ExtTs_{reach} for $(s, w) \in I(h^i)$ for some $i \leq n$ as

$$\begin{aligned} \mathrm{ExtTs}_{reach}(s, w) := \{(s', w') \mid s' \in \Sigma, w' \in M_i, \\ i \in \{0, 1, \ldots, n\}. \\ (i = 0) \implies (s', w') \in I^M_\varphi(h^0, m_0^0) \wedge \\ (i > 0) \implies (s', w') \in I^{M-reach}_{\varphi^i_{reach}} \wedge \\ ss' \models \rho^{\mathrm{sys}} \wedge \\ \left((w', (s')^{\mathrm{sys}}) \notin h^i(w, (s')^{\mathrm{env}}) \vee ss' \models \neg\rho^{\mathrm{env}}\right)\}. \end{aligned}$$

This again is similar to the earlier definition of ExtTs except that we distinguish between the case where $i = 0$ and $i > 0$. Define $\bar{\bar{1}}_{jump}$ as:

$$\bar{\bar{1}}_{jump} := \bigwedge_{(s,m) \in dom(\mathrm{ExtTs}_{reach})} \left(\mathbf{1}_{(s,m)} \to \neg \bigcirc \bigvee_{(s',m') \in \mathrm{ExtTs}_{reach}(s,m)} \mathbf{1}_{(s',m')}\right).$$

4.2.1 'Patched' Robust Strategy. Construct a nondeterministic strategy $\bar{h}$ defined as below:

$$\begin{aligned} \mathrm{Plays}(\bar{h}) = \Big\{\sigma \in \Sigma^\omega \mid \exists m \in \bar{M}^\omega. \forall k > 0. \exists i. \\ (i = 0) \implies \Big((m_{k+1}, \sigma_k^{\mathrm{sys}}) \in h^i(m_k, \sigma_k^{\mathrm{env}}) \\ \vee (\sigma_k, m_{k+1}) \in \mathrm{ExtTs}_{reach}(\sigma_{k-1}, m_k)\Big) \\ \wedge \quad (i > 0) \implies \Big[(\sigma_k, m_{k+1}) \in \mathrm{ExtTs}_{reach}(\sigma_{k-1}, m_k) \\ \vee \Big(\forall w \in M_0.(w, \sigma_k) \notin I^M_\varphi(h^0, m_0^0) \wedge \\ (m_{k+1}, \sigma_k^{\mathrm{sys}}) \in h^i(m_k, \sigma_k^{\mathrm{env}})\Big)\Big] \\ \wedge (m_1, \sigma_0^{\mathrm{sys}}) \in h^0(m_0^0, \sigma_0^{\mathrm{env}})\Big\}. \quad (19) \end{aligned}$$

The definition of $\bar{h}$ is similar to $\bar{f}$ defined earlier, except for $i > 0$. The condition $(i > 0) \to \Big[(\sigma_k, m_{k+1}) \in \mathrm{ExtTs}(\sigma_{k-1}, m_k) \vee \Big(\forall w \in M_0.(w, \sigma_k) \notin I^M_\varphi(h^0, m_0^0) \wedge (m_{k+1}, \sigma_k^{\mathrm{sys}}) \in h^i(m_k, \sigma_k^{\mathrm{env}})\Big)\Big]$ ensures that unless pertubed according to ExtTs_{reach}, on reaching a state in state in $I_\varphi(h^0)$, execution is continued in accordance with the strategy h^0. Consider $\sigma \in \mathrm{Plays}(\bar{h})$ and $m \in (\bar{M})^\omega$ that satisfy the conditions in (19). Let $\bar{\sigma} \in \bar{\Sigma}^\omega$ such that $\forall k \geq 0.\bar{\sigma}_k = (\sigma_k, m_{k+1})$. Here, m is the memory-sequence corresponding to σ generated in accordance with $\bar{h}$.

Proposition 6. *For all such $\bar{\sigma}$, the following holds:*

$$\begin{aligned} \bar{\sigma} \models \Theta_0^{\mathrm{env}} \wedge \Box(\rho^{\mathrm{env}} \vee \bar{\bar{1}}_{jump}) \wedge \left(\bigwedge_{j=1}^J \Box\Diamond\psi_j^{\mathrm{env}}\right) \wedge \Diamond\Box\neg\bar{\bar{1}}_{jump} \\ \implies \Theta_0^{\mathrm{sys}} \wedge \Box\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^K \Box\Diamond\psi_k^{\mathrm{sys}}\right). \quad (20) \end{aligned}$$

A proof is provided in the appendix.

Algorithm 3 formally describes the implementation of a controller based on $\bar{h}$. The controller makes use of the nondeterminism to recover from perturbations to the patches. Execution is begun along the strategy h^0 till a perturbation sets in. The controller attempts to recover using a patch (when perturbed onto a patch) if feasible and execution is continued along the patch to a state in $I_\varphi(h^0)$. Following this, execution along the original strategy is continued. Also, note the algorithm allows for jumps to φ-reachable states in the strategy itself (as does $\bar{h}$). The strategy also allows for perturbations from one patch to another during execution of the patch itself. In the presence of an adversary, the φ-reachable states encodes the adversary's state as well as the system state.

Informally, the recovery trajectory takes the system to A when the adversary reaches B where (A,B) is a φ-reachable state. Additional informal descriptions of specific parts of Algorithm 3:

- Line 5–7: Check for successful actuation, satisfaction of environmental assumption, and for whether original strategy is being executed before continuing execution along original strategy.

Algorithm 3 Execute controller for a patched strategy

Input:
- GR(1) formula φ
- Finite-memory strategy (h^0, m_0^0) realizing φ and a set of strategies (h^l, m_0^l) for $l \in \{1, 2, \ldots, n\}$ realizing φ^i_{reach}
- Sequence of inputs $\sigma^{\text{env}} \in \mathcal{P}(\text{AP}_{\text{env}})^\omega$
- System whose state $s \in \Sigma$ is measured after a control action ($\in \mathcal{P}(\text{AP}_{\text{sys}})$) has been applied
- Sets $I_\varphi(h^0), I_0^M(h_0, m_0^0) := I_\varphi^M(h^0, m_0^0)$, $I_l^M(h^l, m_0^l) := I_{\varphi^l_{reach}}^{M-reach}(h^l, m_0^l)$ for $l \in \{1, 2, \ldots, n\}$

Output: Sequence of control actions $\sigma^{\text{sys}} \in \mathcal{P}(\text{AP}_{\text{sys}}^\omega)$

1: memory=m_0^0
2: i=1, l=0, safety=1, reached=1
3: (memoryNew, σ_0^{sys}) = Strategyh^0 : (memory, σ_0^{env})
4: **while** (True) **do**
5: **if** $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ and safety=1 and reached=1 **then**
6: (memoryNew, σ_i^{sys}) = Strategy h^0 : (memory, σ_i^{env})
7: Run: SafetyCheck
8: **else if** $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ and safety=1 and reached=0 **then**
9: (memoryNew, σ_i^{sys}) = Strategy h^l : (memory, σ_i^{env})
10: **if** $\sigma_i^{\text{sys}} \cup \sigma_i^{\text{env}} \in I_\varphi(h^0, m_0^0)$ **then**
11: Choose (memoryNew, σ_i^{sys}) such that $(\sigma_i^{\text{sys}} \cup \sigma_i^{\text{env}}, memoryNew) \in I_0^M$
12: reached=1
13: **end if**
14: Run: SafetyCheck
15: **else if** safety=0 $\wedge$ $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ **then**
16: **if** $\exists j \leq n, m \in \bar{M}.(m, \sigma_{i-1}^{\text{env}} \cup \sigma_{i-1}^{\text{sys}}) \in I_j^M(h_j, m_0^j)$ **then**
17: Choose (memoryNew,σ_i^{sys}) $\in f_j(m, \sigma_i^{\text{env}})$
18: $l = j$, **if** $j > 0$ **then** reached=0.
19: Run: SafetyCheck
20: **else**
21: EXIT
22: **end if**
23: **else if** $(\sigma_{i-1}^{\text{env}}, \sigma_{i-1}^{\text{sys}})(\sigma_i^{\text{env}}) \not\models \rho^{\text{env}}$ **then**
24: **if** $\exists j \leq n, m \in \bar{M}_j, s \in \mathcal{P}(\text{AP}_{\text{sys}}).(m, \sigma_i^{\text{env}} \cup s) \in \text{ExtTs}_{reach}(memory, \sigma_{i-1}^{\text{env}} \cup \sigma_{i-1}^{\text{sys}})$ **then**
25: (memoryNew,σ_i^{sys}) = (m, s)
26: $l = j$, **if** $j > 0$ **then** reached=0.
27: Run: SafetyCheck
28: **else**
29: EXIT
30: **end if**
31: **end if**
32: i+=1, memory=memoryNew
33: **end while**

- Line 8–9: Check if executing a patch and update accordingly.
- Line 10–13: Check if φ-reachable state from original strategy has been reached during execution along the patch, and switch back to original strategy if condition holds.
- Line 15–22: Check for actuation failures and satisfaction of environmental assumptions, and continue along a patch with an altered memory value.
- Line 23–30: Check for environmental assumption violation, and continue execution along a patch with an altered memory value.

5 EXAMPLE IMPLEMENTATION AND ANALYSIS

Examples are implemented for the analysis of the techniques described in Section 4 for the task of planar robot motion planning in environments similar to that shown in Figure 1. The robot is required to visit a set of locations infinitely often. A moving obstacle whose motion constraints are similar to those of the robot but differing in the starting position and progress states is added to the setup.

5.1 Complexity for refinement

An empirical analysis of the computational costs involved in each of the approaches to augment robustness is presented here. The computations were performed on a 2.40GHz Quadcore machine with 16 GB of RAM. The synthesis was performed with `gr1c`[8], used in the Temporal Logic Planning (`TuLiP`) toolbox [5]. The experiment described below is repeated 50 times and the average synthesis times are presented (See Table 1).

Random 5×5 gridworlds are generated with a wall density of 0.2. The moving obstacle and robot have two different progress-locations which they visit infinitely often and different initial positions. For each of the approaches perturbation points are chosen as below:

- Multiple Strategy Approach: A single perturbation point is chosen that is not in the set of φ-reachable states visited by the initial strategy. A strategy is synthesized with this perturbed state as the initial condition and the states visited by the new strategy are stored. A new perturbation point is chosen that was not visited by the earlier strategies. And the procedure is repeated.
- Patching: The points are chosen as in the previous approach and patches are generated iteratively. However, at each iteraton only those states from the new strategy are stored that occurred before the trajectories reached the stored set of φ-reachable states.

Approach	Coverage (unique states)	Mean time for synthesis(s)
New strategy from perturbed state	145.14	0.22
Patching	173.63	2.98

Table 1: Runtimes and unique states visited in 50 trials.

The coverage i.e the number of unique states - robot, moving obstacle position combinations - visited by each strategy is also presented. It loosely characterizes the robustness for the concatenated strategies, as this count represents the number of φ-reachable states to which the system can be perturbed to. Patching is implemented iteratively, with the visited states augmented in each patch.

With iterative patching, the time for synthesis tends to decrease progressively with each patch for a given gridworld because the number of unique visited states tends to go up.

6 DISCUSSION AND CONCLUSION

This paper characterizes the inherent robustness of finite-memory strategies synthesized to satisfy GR(1) formulae and also proposes approaches to refine them to increase their tolerance to perturbations. We show that these refined strategies satisfy a stricter formula than the one used for synthesis. This tolerance is useful when the model is not exact for either the system behavior or the environment behavior. However, not all perturbations as defined in Section 3 can be tolerated. The system cannot recover from perturbations where ρ^{sys} is violated and perturbations to states that are not winning using the approaches described here.

Another application of the results presented here is in the presence of noise in measurements. Reacting to environmental events in physical systems involves measurements about the environment, and in the presence of noise, false inferences could be made under which the guarantees provided in terms of formula satisfaction would no longer hold. False inferences about an environmental event could alternatively be viewed as the system failing to apply the correct control action. Though ρ^{env} was satisfied, the false inference could result in the system applying an incorrect control action. Let $\gamma \in \mathcal{P}(\text{AP}_{\text{env}})$ be the inferred environmental event and the ground truth environmental event be $\mu \in \mathcal{P}(\text{AP}_{\text{env}})$. The system control action $s \in \mathcal{P}(\text{AP}_{\text{sys}})$ was decided based on $f(m_k, \gamma)$ where m_k is the current memory state. This can be viewed as a perturbation with $(m_{k+1}, s) \notin f(m_k, \mu)$ for any $m_{k+1} \in M$. If $s \cup \mu \in \Sigma$ is a state for which any of the above described approaches apply, the system can recover to satisfy the guarantees on system behavior.

Future Work. We plan to extend the framework built here to the case of infinitely many jumps. We also intend to develop a metric that would quantify the robustness added to a strategy through a given concatenation and prescribe approaches for refinement of strategies to make them more robust with optimal synthesis time/memory costs. Also, we plan to implement the approaches in Sections 4.1 and 4.2 using enumeration from a stored BDD computed during the original synthesis as opposed to resynthesis.

ACKNOWLEDGMENTS

This work was partially supported by United Technologies Corporation and IBM, through the industrial cyber-physical systems (iCyPhy) consortium and by STARnet, a Semiconductor Research Corporation program, sponsored by MARCO and DARPA.

REFERENCES

[1] Roderick Bloem, Krishnendu Chatterjee, Karin Greimel, Thomas A. Henzinger, and Barbara Jobstmann. 2010. *Computer Aided Verification: 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings.* Springer Berlin Heidelberg, Berlin, Heidelberg, Chapter Robustness in the Presence of Liveness, 410–424. DOI: http://dx.doi.org/10.1007/978-3-642-14295-6_36

[2] R. Bloem, K. Greimel, T. A. Henzinger, and B. Jobstmann. 2009. Synthesizing robust systems. In *Formal Methods in Computer-Aided Design, 2009. FMCAD 2009.* 85–92. DOI: http://dx.doi.org/10.1109/FMCAD.2009.5351139

[3] Roderick Bloem, Barbara Jobstmann, Nir Piterman, Amir Pnueli, and Yaniv Sa'ar. 2012. Synthesis of Reactive(1) designs. *J. Comput. System Sci.* 78 (May 2012), 911–938. Issue 3. DOI: http://dx.doi.org/10.1016/j.jcss.2011.08.007

[4] Rüdiger Ehlers and Ufuk Topcu. 2014. Resilience to Intermittent Assumption Violations in Reactive Synthesis. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control (HSCC '14).* ACM, New York, NY, USA, 203–212. DOI: http://dx.doi.org/10.1145/2562059.2562128

[5] I. Filippidis, S. Dathathri, S. C. Livingston, N. Ozay, and R. M. Murray. 2016. Control design for hybrid systems with TuLiP: The Temporal Logic Planning toolbox. In *2016 IEEE Conference on Control Applications (CCA).* 1030–1041. DOI: http://dx.doi.org/10.1109/CCA.2016.7587949

[6] John E. Hopcroft and Jeffrey D. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley.

[7] Yonit Kesten, Nir Piterman, and Amir Pnueli. 2005. Bridging the gap between fair simulation and trace inclusion. *Information and Computation* 200 (2005), 35–61. DOI: http://dx.doi.org/10.1016/j.ic.2005.01.006

[8] Scott C. Livingston. gr1c: a collection of tools for GR(1) synthesis and related activities. http://scottman.net/2012/gr1c. (????). [Online; accessed 15-March 2016].

[9] Rupak Majumdar, Elaine Render, and Paulo Tabuada. 2011. Robust Discrete Synthesis against Unspecified Disturbances. In *Hybrid Systems: Computation and Control (HSCC).*

[10] Zohar Manna and Amir Pnueli. 1990. A Hierarchy of Temporal Properties. In *(PODC '90) Proceedings of the ninth annual ACM Symposium on Principles of Distributed Computing.* 377–408. DOI: http://dx.doi.org/10.1145/93385.93442

[11] Zohar Manna and Amir Pnueli. 1992. *The Temporal Logic of Reactive and Concurrent Systems.* Springer-Verlag New York, Inc., New York, NY, USA.

[12] Amir Pnueli. 1977. The Temporal Logic of Programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (SFCS '77).* IEEE Computer Society, Washington, DC, USA, 46–57. DOI: http://dx.doi.org/10.1109/SFCS.1977.32

[13] Matthias Rungger and Paulo Tabuada. 2014. Abstracting and Refining Robustness for Cyber-physical Systems. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control (HSCC '14).* ACM, New York, NY, USA, 223–232. DOI: http://dx.doi.org/10.1145/2562059.2562133

[14] D. C. Tarraf, A. Megretski, and M. A. Dahleh. 2008. A Framework for Robust Stability of Systems Over Finite Alphabets. *IEEE Trans. Automat. Control* 53, 5 (June 2008), 1133–1146. DOI: http://dx.doi.org/10.1109/TAC.2008.923658

[15] Kai Weng Wong, Rüdiger Ehlers, and Hadas Kress-Gazit. 2014. Correct high-level robot behavior in environments with unexpected events. In *Robotics: Science and Systems Conference (RSS'14).* http://www.roboticsproceedings.org/rss10/p12.pdf

[16] Kemin Zhou, John C. Doyle, and Keith Glover. 1996. *Robust and Optimal Control.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Appendices

A PROOF OF PROPOSITION 5

Proof. Let $\sigma \in \Sigma^\omega$ and $m \in M^\omega$ be such that $(\sigma_k, m_k) = \bar{\sigma}_k$. When $\bar{\sigma} \models \Box\Diamond \bar{1}_{jump}$, the proposition holds trivially. Consider the case when $\bar{\sigma} \models \Diamond\Box\neg 1_{jump}$. This means that there are only finite instances of $\bar{\sigma}_k\bar{\sigma}_{k+1} \models \bar{1}_{jump}$ i.e in other words, only for a finite number of $k > 0, (\sigma_k, m_k) \in \text{ExtTs}(\sigma_{k-1}, m_{k-1})$. Let there be $r < \infty$ such instances i.e $\exists i_l \in \mathbb{N}$ such that $\bar{\sigma}_{i_{l-1}}\bar{\sigma}_{i_l} \models \bar{1}_{jump}$ for $l \in \{1, 2, \ldots, r\}$.

Without loss of generality, assume $0 < i_1 < i_2 < \cdots < i_r$. $\forall l \in \{1, 2, \ldots r-1\} \exists q_l \in \{0, 1, 2, \ldots, n\}$ such that $(m_i, \sigma_i \cap \text{AP}_{\text{sys}}) = g_{q_l}(m_{i-1}, \sigma_i \cap \text{AP}_{\text{env}})$ for $i \in \{i_j, i_{j+1}, \ldots, i_{j+1} - 2\}$. Otherwise at some i between i_j and $i_{j+1} - 2$, $\bar{\sigma}_i\bar{\sigma}_{i+1} \models \bar{1}_{jump}$ – contradicting our assumption. Similarly, $\forall i \in \{0, 1, \ldots, i_1 - 2\}, (m_i, \sigma_i \cap \text{AP}_{\text{sys}}) = g_0(m_{i-1}, \sigma_i \cap \text{AP}_{\text{env}})$.

$\forall l \in \{1, 2, \ldots, r-1\}, (\sigma_{i_l}, m_{i_l}) \in \text{ExtTs}(\sigma_{i_{l-1}}, m_{i_{l-1}})$ by the definition of $\bar{1}_{jump}$. From the definition of ExtTs we can conclude that $(\sigma_{i_l}, m_{i_l}) \in I_{\varphi_{q_l}}$ where $m_{i_l} \in M_{q_l}$ i.e σ_{i_l} is φ_{q_l}-reachable. For $l \in \{1, 2, \ldots, r\}$ by definition of φ_i-reachability and the discussion above, there exists $\beta^l \in \text{Plays}(g_{q_l})$ and k_l such that $\sigma_{i_l} = \beta^l_{k_l}$, $\beta^l \models \Theta_0^{\text{env}}$, and

$$\beta^l_{j:(j+1)} \models \rho^{\text{env}} \text{ for } j < k - 1, \tag{21}$$

and $\forall l \leq r-1$

$$\beta^l_{k_l+w} = \sigma_{i_l+w} \text{ for } 0 \leq w < i_{l+1} - i_l, \tag{22}$$

For $l = r$

$$\beta^l_{k_l+w} = \sigma_{i_l+w} \text{ for } 0 \leq w, \tag{23}$$

since $\bar{\sigma}_{i_r:} \models \neg\Box\bar{1}_{jump}$.

We write $\tau^l\xi^l\alpha^l = \beta^l$ by taking $\tau^l_j = \beta^l_j$ for $0 \leq j < k$. For $l \leq r-1$, $\xi^l_w = \beta^l_{k_l+w}$ where $0 \leq w < i_{l+1}-i_l$, and $\alpha^l_{j-k_l-i_{l+1}+i_l} = \beta^l_j$ for $j \geq k_l + i_{l+1} - i_l$. For $l = r$, $(\xi^r\alpha^r)_{j-k_r} = \beta_j$ for $j \geq k_r$. To conclude the setup, we argued that $\sigma = \tau^0\xi^1\xi^2\ldots\xi^r\alpha^r$ where $\tau^0 \in \text{Pref}(g_0)$. Combine with m as in constructing $\bar{\sigma}$ to have $\bar{\sigma} = \bar{\tau}^0\bar{\xi}^1\bar{\xi}^2\ldots\bar{\xi}^n\bar{\alpha}^r$.

We want to show that

$$\bar{\tau}^0\bar{\xi}^1\bar{\xi}^2\ldots\bar{\xi}^k\bar{\xi}^{k+1}\ldots\bar{\xi}^n\bar{\alpha}^n \models \Theta^{\text{env}}_0 \wedge \Box(\rho^{\text{env}} \vee \bar{1}_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\psi^{\text{env}}_j\right) \implies \Theta^{\text{sys}}_0\Box\rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K}\Box\Diamond\psi^{\text{sys}}_k\right). \tag{24}$$

This is equivalent to at least one of the following subformulae being satisfied: $\neg\Theta^{\text{env}}_0$, $\Diamond(\neg\rho^{\text{env}} \wedge \neg\bar{1}_{jump})$, $\Diamond\Box\neg\psi^{\text{env}}_j$ for some j, or

$$\Theta^{\text{sys}}_0 \wedge \Box\rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K}\Box\Diamond\psi^{\text{sys}}_k\right). \tag{25}$$

Since $\tau^0 \in \text{Pref}(g_0)$ by hypothesis, there exists $\gamma \in \Sigma^\omega$ such that $\tau^0\gamma \in \text{Plays}(g_0)$. Also by hypothesis, (g_0, m_0) realizes φ_0, i.e., $\text{Plays}(g_0) \subseteq L(\varphi_0)$, hence $\tau^0\gamma \models \varphi_0$. Note that $|\tau^0| \geq 1$ since $(m_1, \tau^0 \cap \text{AP}_{\text{env}}) = g_0(m_0, \tau^1 \cap \text{AP}_{\text{sys}})$ by the definition of Plays $\bar{f}$. $\tau^0\gamma \models \neg\Theta^{\text{env}}_0$ if and only if $\bar{\sigma} \models \neg\Theta^{\text{env}}_0$. Thus, if $\tau^0\gamma \models \neg\Theta^{\text{env}}_0$, then (24) holds directly. Otherwise (i.e., if $\tau^0\gamma \models \Theta^{\text{env}}_0$), consider the subformula $\Diamond(\neg\rho^{\text{env}} \wedge \neg\bar{1}_{jump})$.

For all $k < |\tau^0| - 1, \bar{\sigma}_{k:k+1} \models \neg\bar{1}_{jump}$ by construction. Also, for $k \geq \bar{k} = |\tau^0\xi^1\xi^2\ldots\xi^k\xi^{k+1}\ldots\xi^r|, \bar{\alpha}^n_{k-\bar{k},k+1-\bar{k}} \models \neg\bar{1}_{jump}$ and $\bar{\xi}^r_{-1}\bar{\alpha}^r_0 \models \neg\bar{1}_{jump}$. From the decomposition given above, $\bar{\tau}^0_{-1}\bar{\xi}^1_0 \models \bar{1}_{jump}$ and $\bar{\xi}^l_{-1}\bar{\xi}^{l+1}_0 \models \bar{1}_{jump}$ for $l \leq r-1$. For this case, we can conclude

$$\bar{\sigma} \models \Diamond(\neg\rho^{\text{env}} \wedge \neg\bar{1}_{jump}) \implies (\tau^0\gamma \models \Diamond\neg\rho^{\text{env}}) \vee (\xi^1\xi^2\ldots\xi^k\xi^{k+1}\ldots\xi^r\alpha^r \models \Diamond\neg\rho^{\text{env}}).$$

If $\tau^0\gamma \models \Diamond\neg\rho^{\text{env}}$, then there is a minimum k such that $\sigma_{k:}\gamma \models \neg\rho^{\text{env}}$ or $\gamma_{(k-|\sigma|):} \models \neg\rho^{\text{env}}$. If $k < |\sigma| - 1$, then $\bar{\sigma} \models \Diamond(\neg\rho^{\text{env}} \wedge \neg\bar{1}_{jump})$ (recall $\bar{\sigma}_{k:k+1} \models \neg 1_{jump}$). Then (24) is satisfied.

From the definition of ExtTs and $\bar{1}_{jump}$ we have

$$\tau^0_{-1}\xi^1_0 \models \rho^{\text{sys}}, \tag{26}$$

and

$$\xi^l_{-1}\xi^{l+1}_0 \models \rho^{\text{sys}} \forall l \in 1, 2, 3, \ldots, r-1, \tag{27}$$

which we will refer to later while addressing the final case. The other case in which $\bar{\sigma} \models \Diamond(\neg\rho^{\text{env}} \wedge \neg\bar{1}_{jump})$ is if $\xi^l \models \Diamond\neg\rho^{\text{env}}$ for some l or $\xi^r\alpha^r \models \Diamond\neg\rho^{\text{env}}$ (recall $\bar{\tau}^0_{-1}\xi^1_0 \models \bar{1}_{jump}$ and $\xi^l_{-1}\xi^{l+1}_0 \models \bar{1}_{jump}$ for $l \leq r-1$). If for any $d < |\xi^l| - 1$ for $l < r$,

$$\xi^l_d\xi^l_{d+1} \models \neg\rho^{\text{env}},$$

then (24) is directly satisfied. If $\xi^r\alpha^r \models \Diamond\neg\rho^{\text{env}}$ then again (24) holds.

Otherwise, suppose that $\bar{\sigma} \models \Diamond\Box\neg\psi^{\text{env}}_j$ for some j. From the semantics of LTL and the fact that ψ^{env}_j contains no temporal operators, this implies $\alpha^n \models \Diamond\Box\neg\psi^{\text{env}}_j \leftrightarrow \bar{\sigma} \models \Diamond\Box\neg\psi^{\text{env}}_j$. In this case where $\alpha^n \models \Diamond\Box\neg\psi^{\text{env}}_j$, (24) again holds.

We now consider the final case where $\bar{\sigma} \models \Theta^{\text{env}}_0 \wedge \Diamond\Box(\rho^{\text{env}} \vee \bar{1}_{jump}) \wedge \left(\bigwedge^J_{j=1}\Box\Diamond\psi^{\text{env}}_j\right)$. By φ_{q_r}-reachability, $\tau^r_{d,d+1} \models \rho^{\text{env}}$ for all $d < |\tau^r| - 1$. And, $\xi^r\alpha^r \models \Box\rho^{\text{env}}$ as argued for this case (otherwise (24) would directly hold). Thus, $\tau^r\xi^r\alpha^r \models \Box\rho^{\text{env}}$. Recall that $\tau^r\xi^r\alpha^r \models \Theta^{\text{env}}_{q_l}$. Because $\tau^r\xi^r\alpha^r \in \text{Plays}(f_{q_r})$ and $\tau^r\xi^r\alpha^r \models \Theta^{\text{env}}_r$, if neither $\tau^r\xi^r\alpha^r \models \Diamond\neg\rho^{\text{env}}$ nor $\Diamond\Box\neg\psi^{\text{env}}_j$ for any j, it must be that $\tau^r\xi^r\alpha^r$ satisfies $\Theta^{\text{sys}}_{q_r} \wedge \Box\rho^{\text{sys}} \wedge \left(\bigwedge^K_{k=1}\Box\Diamond\psi^{\text{sys}}_k\right)$.

By φ_{q_l}-reachability of ξ^l_0 and strict-realizability, $\xi^l_{d,d+1} \models \rho^{\text{sys}}$ $\forall d < |\xi^l| - 1$ for $l \leq r-1$. From (26) and (27), it follows that

$$\tau^0_{-1}\xi^1\xi^2\ldots\xi^r\alpha^r \models \Box\rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K}\Box\Diamond\psi^{\text{sys}}_k\right).$$

Recall the suffix γ such that $\tau^0\gamma \in \text{Plays}(g_0)$. (g_0, m_0) strictly realizes φ_0, therefore if $\tau^0\gamma \models \Theta^{\text{env}}_0$, it must be that $\tau^0\gamma \models \Theta^{\text{sys}}_0$. $\bar{\sigma} \models \Theta^{\text{sys}}_0$ since $\bar{\sigma} \models \Theta^{\text{env}}_0$ (9). Furthermore, because in this case we are assuming there is no $0 \leq k < |\tau^0| - 1$ such that $\tau^0_{k:(k+1)} \models \neg\rho^{\text{env}}$ (otherwise we would have (24) hold directly), it follows from strict realizability (cf. (10)) that for $0 \leq k < |\tau^0| - 1$, $\tau^0_{k:(k+1)} \models \rho^{\text{sys}}$, and therefore the condition in the proposition for $\bar{\sigma}$ holds.

□

B PROOF OF PROPOSITION 6

Proof. Let $\sigma \in \Sigma^\omega$ and $m \in M^\omega$ be such that $(\sigma_k, m_k) = \bar{\sigma}_k$. Repeating the arguments in Section 4.1, we only need to reason about the case where $\bar{\sigma} \models \Diamond\Box\neg\bar{\bar{1}}_{jump}$. In this case, as before, let there be $r < \infty$ instances in which $\bar{\bar{1}}_{jump}$ holds. Let these instances be i_l for $l \leq r$ such that $0 < i_1 < i_2 < \cdots < i_r$. By definition, $\bar{\sigma}_{i_l-1}\bar{\sigma}_{i_l} \models \bar{\bar{1}}_{jump} \implies \bar{\sigma}_{i_l} \in \text{ExtTs}_{reach}(\bar{\sigma}_{i_l-1})$. Reasoning as before, for $l \in \{1, 2, \ldots, n\}$, $\exists q_l$ such that $\beta^l \in \text{Plays}(h_{q_l})$ and k_l such that $\sigma_{i_l} = \beta^l_{k_l}, \beta^l \models \Theta^{\text{env}}_0$ and conditions (21),(22),(23) hold.

As before, we write $\tau^l\xi^l\alpha^l = \beta^l$. Then, repeating the arguments from earlier decompose $\bar{\sigma}$ as $\bar{\sigma} = \bar{\tau}^0\bar{\xi}^1\ldots\bar{\xi}^r\bar{\alpha}^r$ where $\tau^0 \in \text{Pref}(h^0)$. Note that the decomposition was such that for $l \in \{1, 2\ldots, r\}$, $\xi^l_0 \in I^{M-reach}_{\varphi^{q_l}_{reach}}$ holds if $q_l \geq 1$ and $\xi^l \in I^M_\varphi$ if $q_l = 0$. That is to say each ξ^l_0 is $\varphi^{q_l}_{reach}$-reachable or φ-reachable, as the case may be. We want to show (24).

Reasoning about the other trivial cases as before and rejecting them, consider the case when

$$\bar{\sigma} \models \Theta^{\text{env}}_0 \wedge \Box(\rho^{\text{env}} \vee \bar{\bar{1}}_{jump}) \wedge \left(\bigwedge_{j=1}^{J}\Box\Diamond\psi^{\text{env}}_j\right).$$

$\bar{\sigma} \models \Theta^{\text{env}}_0 \leftrightarrow \bar{\tau}^0 \models \Theta^{\text{env}}_0$. $\tau^0 \in \text{Pref}(h^0) \rightarrow \exists\gamma \in \Sigma^\omega.\ \tau^0\gamma \in \text{Plays}(h^0)$. By strict realizability, we have $\tau^0\gamma \models \Theta^{\text{env}}_0 \rightarrow \tau\gamma \models \Theta^{\text{sys}}_0$. Therefore, $\bar{\sigma} \models \Theta^{\text{sys}}_0$.

For $2 \leq l \leq r-1$, we have $\xi_0^l \xi_0^{l+1} \models \rho^{\text{sys}}$ from the definition of $\bar{\bar{1}}_{jump}$ and ExtTs_{reach}. We also have $\tau_{-1}^0 \xi_0^1 \models \rho^{\text{sys}}$. When $\bar{\sigma} \models \square(\rho^{\text{env}} \vee \bar{\bar{1}}_{jump})$, by strict realizability and $\varphi_{reach}^{q_l} - realizability$ (or φ-reachability as the case maybe) for $l \leq r-1$, we have $\xi_{d,d+1}^l \models \rho^{\text{sys}}\ \forall d < |\xi^l| - 1$ since $\xi_{d:d+1}^l \models \rho^{\text{env}} \forall d < |\xi^l| - 1$. Also, we have $\tau_{d:d+1}^0 \models \rho^{\text{sys}}$ for $\forall d < |\tau^0| - 1$ since $\tau_{d:d+1}^0 \models \rho^{\text{env}} \forall d < |\tau^0| - 1$.

Consider $\tau^r \xi^r \alpha^r \in \text{Plays}(h^{q_r})$. If $q_r > 0$, h^{q_r} strictly realizes $\varphi_{reach}^{q_r}$ and ξ_0^r is $\varphi_{reach}^{q_r}$-reachable. If $q_r = 0$, h^0 strictly-realizes φ and ξ_0^r is φ-reachable. Therefore, if $\xi^r \alpha^r \models \square\rho^{\text{env}}$ we have $\tau^r \xi^r \alpha^r \models \square\rho^{\text{env}}$. By strict realizability, we have $\tau^r \xi^r \alpha^r \models \square\rho^{\text{sys}}$.

To conclude, we showed $\bar{\sigma} \models \Theta_0^{\text{env}} \wedge \square(\rho^{\text{env}} \vee \bar{\bar{1}}_{jump}) \rightarrow \bar{\sigma} \models \Theta_0^{\text{sys}} \wedge \square\rho^{\text{sys}}$. Now only showing liveness remains. If $q_r > 0$, we have $\tau^r \xi^r \alpha^r \models \diamond\mathcal{T}_{reach}$ or $\tau^r \xi^r \alpha^r$ satisfies $\neg\Theta_0^{\text{env}}$ or $\diamond\neg\rho^{\text{env}}$ or $\diamond\square\neg\psi_j^{\text{env}}$ for some j since h^{q_r} realizes $\varphi_{reach}^{q_r}$. But, as an assumption we have $\bar{\sigma} \models \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{env}}\right)$ and by the semantics of LTL we have $\alpha^r \models \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{env}}\right)$. And, $\alpha^r \models \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{env}}\right) \leftrightarrow \tau^r \xi^r \alpha^r \models \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{env}}\right)$. By $\varphi_{reach}^{q_r}$-reachability of ξ_0^r, we have $\tau^r \xi^r \alpha^r \models \Theta_{q_r}^{\text{env}}$. And, above we reasoned $\tau^r \xi^r \alpha^r \models \square\rho^{\text{env}}$ for this case. Therefore, it must be the case that $\tau^r \xi^r \alpha^r \models \diamond\mathcal{T}_{reach}$. Observe that if $s \in I_\varphi(h^0, m_0^0)$ then $\exists w \in M^0.(s,w) \in I_\varphi^M(h^0, w_0^0)$. $\tau^r \xi^r \alpha^r \models \diamond\mathcal{T}_{reach}$ leads to a contradiction as by the definition of $\text{Plays}(\bar{h})$, we should switch strategies when we reach a state s that satisfies $\mathcal{T}_{reach}$. Therefore, $q_r = 0$ when $\bar{\sigma} \models \Theta_0^{\text{env}} \wedge \square(\rho^{\text{env}} \vee \bar{\bar{1}}_{jump}) \wedge \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{env}}\right)$.

By strict realizability and φ-reachability, we reasoned that $\tau^r \xi^r \alpha^r \models \Theta_0^{\text{env}} \wedge \square\rho^{\text{env}} \rightarrow \bar{\sigma} \models \Theta_0^{\text{sys}} \wedge \square\rho^{\text{sys}}$. Additionally $\tau^r \xi^r \gamma^r \models \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{env}}\right)$, by which we have $\tau^r \xi^r \alpha^r \models \Theta_0^{\text{sys}} \wedge \square\rho^{\text{sys}} \wedge \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{sys}}\right)$ (since $q_r = 0$ and from the definition of (h^0, m_0^0) realizing φ). Therefore, we have $\bar{\sigma} \models \Theta_0^{\text{sys}} \wedge \square\rho^{\text{sys}} \wedge \left(\bigwedge_{j=1}^J \square\diamond\psi_j^{\text{sys}}\right)$, hence proving the proposition. □