Good-Enough Synthesis

Shaull Almagor and Orna Kupferman

[1] Department of Computer Science, The Technion, Israel.
[2] School of Computer Science and Engineering, The Hebrew University, Israel.

Abstract. In the classical *synthesis* problem, we are given an LTL formula ψ over sets of input and output signals, and we synthesize a system $\mathcal{T}$ that *realizes* ψ: with every input sequences x, the system associates an output sequence $\mathcal{T}(x)$ such that the generated computation $x \otimes \mathcal{T}(x)$ satisfies ψ. In practice, the requirement to satisfy the specification in all environments is often too strong, and it is common to add assumptions on the environment. We introduce and study a new type of relaxation on this requirement. In *good-enough synthesis* (GE-synthesis), the system is required to generate a satisfying computation only if one exists. Formally, an input sequence x is *hopeful* if there exists some output sequence y such that the computation $x \otimes y$ satisfies ψ, and a system GE-realizes ψ if it generates a computation that satisfies ψ on all hopeful input sequences. GE-synthesis is particularly relevant when the notion of correctness is *multi-valued* (rather than Boolean), and thus we seek systems of the highest possible quality, and when synthesizing *autonomous systems*, which interact with unexpected environments and are often only expected to do their best.
We study GE-synthesis in Boolean and multi-valued settings. In both, we suggest and solve various definitions of GE-synthesis, corresponding to different ways a designer may want to take hopefulness into account. We show that in all variants, GE-synthesis is not computationally harder than traditional synthesis, and can be implemented on top of existing tools. Our algorithms are based on careful combinations of nondeterministic and universal automata. We augment systems that GE-realize their specifications by monitors that provide satisfaction information. In the multi-valued setting, we provide both a worst-case analysis and an expectation-based one, the latter corresponding to an interaction with a stochastic environment.

1 Introduction

Synthesis is the automated construction of a system from its specification: given a specification ψ, typically by a linear temporal logic (LTL) formula over sets I and O of input and output signals, the goal is to construct a finite-state system that satisfies ψ [9, 20]. At each moment in time, the system reads an assignment, generated by the environment, to the signals in I, and responds with an assignment to the signals in O. Thus, with every input sequence, the system associates an output sequence. The system *realizes* ψ if ψ is satisfied in all the interactions of the system, with all environments [5].

In practice, the requirement to satisfy the specification in all environments is often too strong. Accordingly, it is common to add assumptions on the behavior of the environment. An assumption may be direct, say given by an LTL formula that restricts the set of possible input sequences [8], less direct, say a bound on the size of the environment [13] or other resources it uses, or conceptual, say rationality from the side of the environment, which may have its own objectives [11, 14]. We introduce and study a new type of relaxation of the requirement to satisfy the specification in all environments. The idea behind the relaxation is that if an environment is such that no system can interact with it in a way that satisfies the specification, then we cannot expect our system to succeed. In other words, the system has to satisfy the specification only when it interacts with environments in which this mission is possible. This is particularly relevant when synthesizing *autonomous systems*, which interact with unexpected environments and often replace human behavior, which is only expected to be *good enough* [28], and when the notion of correctness is multi-valued (rather than Boolean), and thus we seek *high-quality* systems.

Before we explain the relaxation formally, let us consider a simple example, and we start with the Boolean setting. Let $I = \{req\}$ and $O = \{grant\}$. Thus, the system receives requests and generates grants. Consider the specification $\psi = \mathsf{GF}(req \wedge grant) \wedge \mathsf{GF}(\neg req \wedge \neg grant)$. Clearly, ψ is not realizable, as an input sequence need not satisfy $\mathsf{GF}req$ or $\mathsf{GF}\neg req$. However, a system that always generates a grant upon (and only upon) a request, GE-*realizes* ψ, in the sense that for every input sequence, if there is some interaction with it with which ψ is satisfied, then our system generates such an interaction.

Formally, we model a system by a strategy $f : (2^I)^+ \to 2^O$, which given an input sequence $x = i_0 \cdot i_1 \cdot i_2 \cdots \in (2^I)^\omega$, generates an output sequence $f(x) = f(i_0) \cdot f(i_0 \cdot i_1) \cdot f(i_0 \cdot i_1 \cdot i_2) \cdots \in (2^O)^\omega$, inducing the computation $x \otimes f(x) = (i_0 \cup f(i_0)) \cdot (i_i \cup f(i_0 \cdot i_1)) \cdot (i_2 \cup f(i_0 \cdot i_1 \cdot i_2)) \cdots \in (2^{I \cup O})^\omega$, obtained by "merging" x and $f(x)$. In traditional realizability, a system realizes ψ if ψ is satisfied in all environments. Formally, for all input sequences $x \in (2^I)^\omega$, the computation $x \otimes f(x)$ satisfies ψ. For our new notion, we first define when an input sequence $x \in (2^I)^\omega$ is *hopeful*, namely there is an output sequence $y \in (2^O)^\omega$ such that the computation $x \otimes y$ satisfies ψ. Then, a system GE-*realizes* ψ if ψ is satisfied in all interactions with hopeful input sequences. Formally, for all $x \in (2^I)^\omega$, if x is hopeful, then the computation $x \otimes f(x)$ satisfies ψ.

Since LTL is Boolean, synthesized systems are correct, but there is no reference to their quality. This is a crucial drawback, as designers would be willing to give up manual design only if automated-synthesis algorithms return systems of comparable quality. Addressing this challenge, researchers have developed quantitative specification formalisms. For example, in [4], the input to the synthesis problem includes also Mealy machines that grade different realizing systems. In [1], the specification formalism is the multi-valued logic LTL[$\mathcal{F}$] , which augments LTL with quality operators. The satisfaction value of an LTL[$\mathcal{F}$] formula is a real value in $[0, 1]$, where the higher the value, the higher the quality in which the computation satisfies the specification. The quality operators in $\mathcal{F}$ can prioritize and weight different scenarios. The synthesis algorithm for LTL[$\mathcal{F}$] seeks systems with a highest possible satisfaction value. One can consider either a worst-case approach, where the satisfaction value of a system is the satisfaction value of its computation with the lowest satisfaction value [1], or a stochastic approach, where it is the expected satisfaction value, given a distribution of the inputs [2].

We demonstrate the effectiveness of GE-synthesis in the multi-valued setting with an example. Consider an acceleration controller of an autonomous car. Normally, the car should maintain a relatively constant speed. However, in order to optimize travel time, if a long stretch of road is visible and is identified as low-risk, the car should accelerate. Conversely, if an obstacle or some risk factor is identified, the car should decelerate. Clearly, the car cannot accelerate and decelerate at the same time. We capture this desired behavior with the following LTL[$\mathcal{F}$] formula over the inputs $\{safe, obs\}$ and outputs $\{acc, dec\}$:

$$\psi = \mathsf{G}(safe \to (acc \oplus_{\frac{2}{3}} \mathsf{X} acc)) \wedge \mathsf{G}(obs \to (dec \oplus_{\frac{3}{4}} \mathsf{X} dec)) \wedge \mathsf{G}(\neg(acc \wedge dec)).$$

Thus, in order to get satisfaction value 1, each detection of a safe stretch should be followed by an acceleration during two transactions, with a preference to the first (by the semantics of the weighted average $\oplus_\lambda$ operator, the satisfaction value of $safe \to (acc \oplus_{\frac{2}{3}} \mathsf{X} acc)$ is 1 when $safe$ is followed by two accs, $\frac{2}{3}$ when it is followed by one acc, and $\frac{1}{3}$ if it is followed by one acc with a delay), and each detection of an obstacle should be followed by a deceleration during two transactions, with a (higher) preference to the first. Clearly, ψ is not realizable with satisfaction value 1, as for some input sequences, namely those with simultaneous or successive occurrences of $safe$ and obs, it is impossible to respond with the desired patterns of acceleration or declaration. Existing frameworks for synthesis cannot handle this challenge. Indeed, we do not want to add an assumption about $safe$ and obs occurring far apart. Rather, we want our autonomous car to behave as well as possible also in problematic environments, and we want, when we evaluate the quality of a car, to take into an account the challenge posed by the environment. This is exactly what high-quality GE-synthesis does: for each input sequence, it requires the synthesized car to obtain the maximal satisfaction value that is possible for that input sequence.

We show that in the Boolean setting, GE-synthesis can be reduced to synthesis of LTL with quantification of atomic propositions [26]. Essentially, GE-synthesis of ψ amounts to synthesis of $(\exists O.\psi) \to \psi$. We show that by carefully switching between nondeterminisitc and universal automata, we can solve the GE-synthesis problem in doubly-exponential time, thus it is not harder than traditional synthesis. Also, our algorithm is *Safraless*, thus no determinization and parity games are needed [17, 15].

A drawback of GE-synthesis is that we do not actually know whether the specification is satisfied. We describe two ways to address this drawback. The first goes beyond providing satisfaction information and enables the designer to partition the specification into a *strong* component, which is guaranteed to be satisfied in all environments, and a *weak* component, which is guaranteed to be satisfied only in hopeful ones. The second way augments GE-realizing systems by "satisfaction indicators". For example, we show that when a system is lucky to interact with an environment that generates a prefix of an input sequence such that, when combined with a suitable prefix of an output sequence, the specification becomes realizable, then GE-synthesis guarantees that the system indeed responds with a suitable prefix of an output sequence. Moreover, it is easy to add to the system a monitor that detects such prefixes, thus indicating that the specification is going to be satisfied in all environments. While a naive construction of such a monitor is based on a solution of the synthesis problem, we show that since the system GE-realizes the specification, the monitor can be based on a solution of the *universal-satisfiability* problem (that is, deciding whether all input sequences are hopeful), which is much simpler. Additional monitors we suggest detect prefixes after which the satisfaction becomes valid or unsatisfiable.

We continue to the quantitative setting. We parameterize hope by a satisfaction value $v \in [0,1]$ and say that an input sequence $x \in (2^I)^\omega$ is *v-hopeful* for an LTL[$\mathcal{F}$] formula ψ if an interaction with it can generate a computation that satisfies ψ with value at least v. Formally, there is an output sequence $y \in (2^O)^\omega$ such that $[\![x \otimes y, \psi]\!] \geq v$, where for a computation $w \in (2^{I \cup O})^\omega$, we use $[\![w, \psi]\!]$ to denotes the satisfaction value of ψ in w. As we elaborate below, while the basic idea of GE-synthesis, namely "input sequences with a potential to high quality should realize this potential" is as in the Boolean setting, there are several ways to implement this idea.

We start with a worst-case approach. There, a strategy $f : (2^I)^+ \to 2^O$ GE-realizes an LTL[$\mathcal{F}$] formula ψ if for all input sequences $x \in (2^I)^\omega$, if x is v-hopeful, then $[\![x \otimes f(x), \psi]\!] \geq v$. The requirement can be applied to a threshold value or to all values $v \in [0,1]$. For example, our autonomous car controller has to achieve satisfaction value 1 in roads with no simultaneous or successive occurrences of *safe* and *obs*, and value $\frac{3}{4}$ in roads that violate the latter only with some *obs* followed by *safe*. We then argue that the situation is similar to that of *high-quality assume guarantee synthesis* [3], where richer relations between a quantitative assumption and a quantitative guarantee are of interest. In our case, the assumption is the hopefulness level of the input sequence, namely $[\![x, \exists O.\psi]\!]$, and the guarantee is the satisfaction value of the specification in the generated computation, namely $[\![x \otimes f(x), \psi]\!]$. We assume that the desired relation between the assumption and the guarantee is given by a function $\texttt{comb} : [0,1] \times [0,1] \to [0,1]$. For example, $\texttt{comb}(A, G) = \max\{1 - A, G\}$ captures implication, and $\texttt{comb}(A, G) = 1 - (A - G)$ measures the distance between the satisfaction value of the generated computation and the highest possible satisfaction value for the input sequence. When synthesizing, for example, a robot controller (e.g., vacuum cleaner) in a building, the doors to rooms are controlled by the environment, whereas the movement of the robot by the system. A measure of the performance of the robot has to take into an account both the number of "hopeful rooms", namely these with an open door, and the number of room cleaned. Note that the satisfaction value of the assumption $[\![x, \exists O.\psi]\!]$ measures the performance of a good-enough *off-line* system. Thus, using a function $\texttt{comb}(A,G)=\frac{G}{A}$, we can synthesize an *on-line* system with the best *competitive ratio* [7] (see Example 4). We show that the GE-synthesis Safraless algorithm we suggested in the Boolean setting can be extended to handle LTL[$\mathcal{F}$] formulas in all the above approaches, thus the problem can be solved in doubly-exponential time.

We continue with an analysis of the expected performance of the system. We do so by assuming a stochastic environment, with a known distribution on the input sequences. We introduce and study two measures for high-quality GE-synthesis in a stochastic environment. In the first, termed *expected* GE-*synthesis*, all input sequences are sampled, yet the satisfaction value in each input sequence takes its hopefulness level into account, for example by a `comb` function as in the assume-guarantee setting. In the second, termed *conditional expected* GE-*synthesis*, only hopeful input sequences are sampled. For both approaches, our synthesis algorithm is based on the high-quality LTL[$\mathcal{F}$] synthesis algorithm of [2], which is based on an analysis of deterministic automata associated with the different satisfaction values of the LTL[$\mathcal{F}$] specification. Here too, the complexity stays doubly exponential. In addition, we extend the synthesized systems with guarantees for satisfaction and monitors indicating satisfaction in various satisfaction levels.

2 Preliminaries

Consider two finite sets I and O of input and output signals, respectively. For two words $x=i_0\cdot i_1\cdot i_2\cdots\in(2^I)^\omega$ and $y=o_0\cdot o_1\cdot o_2\cdots\in(2^I)^\omega$, we define $x\otimes y$ as the word in $(2^{I\cup O})^\omega$ obtained by merging x and y. Thus, $x\otimes y=(i_0\cup o_0)\cdot(i_1\cup o_1)\cdot(i_2\cup o_2)\cdots$. The definition is similar for finite x and y of the same length. For a word $w\in(2^{I\cup O})^\omega$, we use $w_{|I}$ to denote the projection of w on I. In particular, $(x\otimes y)_{|I}=x$.

A *strategy* is a function $f:(2^I)^+\to 2^O$. Intuitively, f models the interaction of a system that generates in each moment in time a letter in 2^O with an environment that generates letters in 2^I. For an input sequence $x=i_0\cdot i_1\cdot i_2\cdots\in(2^I)^\omega$, we use $f(x)$ to denote the output sequence $f(i_0)\cdot f(i_0\cdot i_1)\cdot f(i_0\cdot i_1\cdot i_2)\cdots\in(2^O)^\omega$. Then, $x\otimes f(x)\in(2^{I\cup O})^\omega$ is the *computation* of f on x. Note that the environment initiates the interaction, by inputting i_0. Of special interest are *finite-state strategies*, induced by finite state transducers. Formally, an *I/O-transducer* is $\mathcal{T}=\langle I,O,S,s_0,M,\tau\rangle$, where S is a finite set of states, $s_0\in S$ is an initial state, $M:S\times 2^I\to S$ is a transition function, and $\tau:S\to 2^O$ is a labelling function. For $x=i_0\cdot i_1\cdot i_2\cdots\in(2^I)^*$, let $M^*(x)$ be the state in S that $\mathcal{T}$ reaches after reading x. Thus is, $M^*(\epsilon)=s_0$ and for every $j\geq 0$, we have that $M^*(i_0\cdot i_1\cdot i_2\cdots i_j)=M(M^*(i_0\cdot i_1\cdot i_2\cdots i_{j-1}),i_j)$. Then, $\mathcal{T}$ induces the strategy $f_{\mathcal{T}}:(2^I)^+\to 2^O$, where for every $x\in(2^I)^+$, we have that $f_{\mathcal{T}}(x)=\tau(M^*(x))$. We use $\mathcal{T}(x)$ and $x\otimes\mathcal{T}(x)$ to denote the output sequence and the computation of $\mathcal{T}$ on x, respectively, and talk about $\mathcal{T}$ realizing a specification, referring to the strategy $f_{\mathcal{T}}$.

We specify on-going behaviors of reactive systems using the *linear temporal logic* LTL [19]. Formulas of LTL are constructed from a set AP of atomic proposition using the usual Boolean operators and temporal operators like G ("always"), F ("eventually"), X ("next time"), and U ("until"). Each LTL formula ψ defines a language $L(\psi) = \{w : w \models \psi\} \subseteq (2^{AP})^\omega$. We also use *automata on infinite words* for specifying and reasoning about on-going behaviors. We use automata with different branching modes (nondeterministic, where some run has to be accepting; universal, where all runs have to be accepting; and deterministic, where there is a single run) and different acceptance conditions (Büchi, co-Büchi, and parity). We use the three letter acronyms NBW, UCW, DPW, and DFW, to refer to nondeterministic Büchi, universal co-Büchi, deterministic parity, and deterministic finite word automata, respectively. Given an LTL formula ψ over AP, one can constructs an NBW $\mathcal{A}_\psi$ with at most $2^{O(|\psi|)}$ states such that $L(\mathcal{A}_\psi) = L(\psi)$ [27]. Constructing an NBW for $\neg\psi$ and then dualizing it, results in a UCW for $L(\psi)$, also with at most $2^{O(|\psi|)}$ states. Determinization [23] then leads to a DPW for $L(\psi)$ with at at most $2^{2^{O(|\psi|)}}$ states and index $2^{O(|\psi|)}$. For full definitions of LTL, automata, and their relation, see [12].

Consider an LTL formula ψ over $I \cup O$. We say that ψ is *realizable* if there is a finite-state strategy $f : (2^I)^+ \to 2^O$ such that for all $x \in (2^I)^\omega$, we have that $x \otimes f(x) \models \psi$. That is, the computation of f on every input sequence satisfies ψ. We say that a word $x \in (2^I)^\omega$ is *hopeful* for ψ if there is $y \in (2^O)^\omega$ such that $x \otimes y \models \psi$. Then, we say that ψ is *good-enough realizable* (GE-realizable, for short) if there is a finite-state strategy $f : (2^I)^+ \to 2^O$ such that for every $x \in (2^I)^\omega$ that is hopeful for ψ, we have that $x \otimes f(x) \models \psi$. That is, if there is some output sequence whose combination with x satisfies ψ, then the computation of f on x satisfies ψ. The LTL GE-synthesis problem is then to decide whether a given LTL formula is GE-realizable, and if so, to return a transducer that GE-realizes it. Clearly, every realizable specification is GE-realizable – by the same transducer. We say that ψ is *universally satisfiable* if all input sequences are hopeful for ψ. It is easy to see that for universally satisfiable specifications, realizability and GE-realizability coincide. On the other hand, as demonstrated in Section 1, there are specifications that are not realizable and are GE-realizable.

Example 1. Let $I = \{p\}$ and $O = \{q\}$. Consider the specification $\psi = \mathsf{GF}((\mathsf{X}p) \wedge q) \wedge \mathsf{GF}((\mathsf{X}\neg p) \wedge \neg q)$. Clearly, ψ is not realizable, as an input sequence $x \in (2^I)^\omega$ is hopeful for ψ iff $x \models \mathsf{GF}p \wedge \mathsf{GF}\neg p$. Since the system has to assign a value to q before it knowns the value of $\mathsf{X}p$, it seems that ψ is also not GE-realizable. As we show below, however, the specification ψ is GE-realizable. Intuitively, it follows from the fact that hopeful input sequences consists of alternating p-blocks and $(\neg p)$-blocks. Then, by outputting $\neg q$ in p-blocks and outputting q in $(\neg p)$-blocks, the system guarantees that each last position in a $(\neg p)$-block satisfies $q \wedge \mathsf{X}p$ and each last position in a p-block satisfies $(\neg q) \wedge \mathsf{X}p$. Formally, ψ is GE-realized by the transducer $\mathcal{T} = \langle \{p\}, \{q\}, \{s_0, s_1\}, s_0, M, \tau\rangle$, where $M(s_0, \emptyset) = M(s_1, \emptyset) = s_0$, $M(s_0, \{p\}) = M(s_1, \{p\}) = s_1$, $\tau(s_0) = \{q\}$, and $\tau(s_1) = \emptyset$. □

3 LTL Good-Enough Synthesis

Recall that a strategy $f : (2^I)^+ \to 2^O$ GE-realizes an LTL formula ψ if its computations on all hopeful input sequences satisfy ψ. Thus, for every input sequence $x \in (2^I)^\omega$, either $x \otimes y \not\models \psi$ for all $y \in (2^O)^\omega$, or $x \otimes f(x) \models \psi$. The above suggests that algorithms for solving LTL GE-synthesis involve existential and universal quantification over the behavior of output signals. The logic EQLTL extends LTL by allowing existential quantification over atomic propositions [26]. We refer here to the case the atomic propositions are the signals in $I \cup O$, and the signals in O are existentially quantified. Then, an EQLTL formula is of the form $\exists O.\psi$, and a computation $w \in (2^{I\cup O})^\omega$ satisfies $\exists O.\psi$ iff there is $y \in (2^O)^\omega$ such that $w_{|I} \otimes y \models \psi$. Dually, AQLTL extends LTL by allowing universal quantification over atomic propositions. We consider here formulas of the form $\forall O.\psi$, which are equivalent to $\neg\exists O.\neg\psi$. Indeed, a computation $w \in (2^{I\cup O})^\omega$ satisfies $\forall O.\psi$ iff for all $y \in (2^O)^\omega$, we have that $w_{|I} \otimes y \models \psi$. Note that in both the existential and universal cases, the O-component of w is ignored. Accordingly, we sometimes interpret EQLTL and AQLTL formulas with respect to input sequences $x \in (2^I)^\omega$. Also note that both EQLTL and AQLTL increase the expressive power of LTL. For example, the EQLTL formula $\exists q.q \wedge \mathsf{X}\neg q \wedge \mathsf{G}(q \leftrightarrow \mathsf{XX}q) \wedge \mathsf{G}(q \to p)$ states that p holds in all even positions of the computation, which cannot be specified in LTL [29].

Theorem 1. *The LTL GE-synthesis problem is 2EXPTIME-complete.*

Proof. We start with the upper bound. Given an LTL formula ψ over $I \cup O$, we describe an algorithm that returns a transducer $\mathcal{T}$ that GE-realizes ψ, or declares that no such transducer exists.

It is not hard to see that $\mathcal{T}$ GE-realizes ψ iff $\mathcal{T}$ realizes $\varphi = \psi \vee \forall O.\neg\psi$. Indeed, an input sequence $x \in (2^I)^\omega$ is hopeful for ψ iff $x \models \exists O.\psi$, and so the specification φ requires all hopeful input sequences to satisfy ψ. A naive construction of an NBW for φ involves a universal projection of the signals in O in an automaton for $\neg\psi$, and results in an NBW that is doubly exponential. In order to circumvent the extra exponent, we construct an NBW $\mathcal{A}_{\neg\varphi}$ for $\neg\varphi$, and then dualize it to get a UCW for φ, as follows.

Let $\mathcal{A}_{\neg\psi}$ be an NBW for $L(\neg\psi)$ and $\mathcal{A}_{\exists O.\psi}$ be an NBW for $L(\exists O.\psi)$. Thus, $\mathcal{A}_{\exists O.\psi}$ is obtained from an NBW $\mathcal{A}_\psi$ for $L(\psi)$ by existentially projecting its transitions on 2^I. In more details, if $\mathcal{A}_\psi = \langle 2^{I\cup O}, Q, Q_0, \delta, \alpha\rangle$, then $\mathcal{A}_{\exists O.\psi} = \langle 2^{I\cup O}, Q, Q_0, \delta', \alpha\rangle$, where for all $q \in Q$ and $i \cup o \in 2^{I\cup O}$, we have $\delta'(q, \sigma) = \bigcup_{o\in 2^O}\{\delta(q, (\sigma \cap I) \cup o)\}$.

Let $\mathcal{A}_{\neg\varphi}$ be an NBW for the intersection of $\mathcal{A}_{\neg\psi}$ and $\mathcal{A}_{\exists O.\psi}$. We can define $\mathcal{A}_{\neg\varphi}$ as the product of $\mathcal{A}_{\neg\psi}$ and $\mathcal{A}_{\exists O.\psi}$, possibly using the generalized Büchi acceptance condition (see Remark 1), thus its size is exponential in ψ. The language of $\mathcal{A}_{\neg\varphi}$ is then $\{w \in (2^{I\cup O})^\omega : w \not\models \psi \text{ and } w \models \exists O.\psi\}$. We then solve usual synthesis for the complementing UCW. Its language is $\{w \in (2^{I\cup O})^\omega : w \models \psi \text{ or } w \models \forall O.\neg\psi\}$, as required. By [17], the synthesis problem for UCW can be solved in EXPTIME, and we are done.

The lower bound follows from the 2EXPTIME-hardness of LTL realizability [22]. The hardness proof there constructs, given a 2EXPTIME Turing machine M, an LTL formula ψ that is realizable iff M accepts the empty tape. Since all input sequences are hopeful for ψ, realizability and GE-realizability coincide, and we are done. $\square$

Note that working with a UCW not only handles the universal quantification for free but also has the advantage of a Safraless synthesis algorithm – no determinization and parity games are needed [17, 15]. Also note that the algorithm we suggest in the proof of Theorem 1 can be generalized to handle specifications that are arbitrary positive Boolean combinations of EQLTL formulas.

Remark 1. [**Products and Optimizations**] Throughout the paper, we construct products of automata whose state space is $2^{cl(\psi)}$, and states correspond to maximal consistent subsets of $cl(\psi)$, possibly in the scope of an existential quantifier of O. Accordingly, the product can be minimized to include only consistent pairs. Also, since traditional-synthesis algorithms, in particular the Safraless algorithms we use, can handle automata with *generalized* Büchi and co-Büchi acceptance condition, we need only one copy of the product. $\square$

Remark 2. [**Determinancy of the GE-synthesis Game**] Determinancy of games implies that in traditional synthesis, a specification ψ is not I/O-realizable iff $\neg\psi$ is O/I-realizable This is useful, for example when we want to synthesize a transducer of a bounded size and proceed simultaneously, aiming to synthesize either a system transducer that realizes ψ or an environment transducer that realizes $\neg\psi$ [17]. For GE-synthesis, simple dualization does not hold, but we do have determinancy in the sense that $(\exists O.\psi) \rightarrow \psi$ is not I/O-realizable iff $(\exists O.\psi) \wedge \neg\psi$ is O/I-realizable. Accordingly, ψ is not GE-realizable iff the environment has a strategy that generates, for each output sequence $y \in (2^O)^\omega$, a helpful input sequence $x \in (2^I)^\omega$ such that $x \otimes y \models \neg\psi$. In Appendix A, we formalize and study this duality further. $\square$

4 Guarantees in Good-Enough Synthesis

A drawback of GE-synthesis is that we do not actually know whether the specification is satisfied. In this section we describe two ways to address this drawback. The first way goes beyond providing satisfaction information and enables the designer to partition the specification into to a *strong* component, which should be satisfied in all environments, and a *weak* component, which should be satisfied only in hopeful ones. The second way augments GE-realizing transducers by flags, raised to indicate the status of the satisfaction.

4.1 GE-Synthesis with a Guarantee

Recall that GE-realizability is suitable especially in settings where we design a system that has to do its best in all environments. GE-synthesis with a guarantee is suitable in settings where we want to make sure that some components of the specification are satisfied in all environment. Accordingly, a specification is an LTL formula $\psi = \psi_{strong} \wedge \psi_{weak}$. When we GE-*synthesize* ψ_{weak} *with guarantee* ψ_{strong}, we seek a transducer $\mathcal{T}$ that realizes ψ_{strong} and GE-realizes ψ_{weak}. Thus, for all input sequences $x \in (2^I)^\omega$, we have that $x \otimes \mathcal{T}(x) \models \psi_{strong}$, and if x is hopeful for ψ_{weak}, then $x \otimes \mathcal{T}(x) \models \psi_{strong}$.

Theorem 2. *The LTL* GE*-synthesis with guarantee problem is 2EXPTIME-complete.*

Proof. Consider an LTL formula $\psi = \psi_{strong} \wedge \psi_{weak}$ over $I \cup O$. It is not hard to see that a transducer $\mathcal{T}$ GE-realizes ψ_{weak} with guarantee ψ_{strong} iff $\mathcal{T}$ realizes $\varphi = \psi_{strong} \wedge ((\exists O.\psi_{weak}) \rightarrow \psi_{weak})$. We can then construct a UCW $\mathcal{A}_\varphi$ for $L(\varphi)$ by dualizing an NBW for its negation $\neg\psi_{strong} \vee ((\exists O.\psi_{weak}) \wedge \neg\psi_{weak})$, which can be constructed using techniques similar to those in the proof of Theorem 1. We then proceed with standard synthesis for $\mathcal{A}_\varphi$. Note that the approach is Safraless. Taking an empty (that is, `True`) guarantee, a lower bound follows from the 2EXPTIME-hardness of LTL GE-synthesis. □

4.2 Flags by a GE-Realizing Transducer

For a language $L \subseteq (2^{I\cup O})^\omega$ and a finite word $w \in (2^{I\cup O})^*$, let $L^w = \{w' \in (2^{I\cup O})^\omega : w \cdot w' \in L\}$. That is, L^w is the language of suffixes of words in L that have w as a prefix. We say that a word $w \in (2^{I\cup O})^*$ is *green for* L if L^w is realizable. Then, a word $x \in (2^I)^*$ is *green for* L if there is $y \in (2^O)^*$ such that $x \otimes y$ is green for L. When a system is lucky to interact with an environment that generates a green input sequence, we want the system to react in a way that generates a green prefix, and then realizes the specification. Formally, we say that a strategy $f : (2^I)^+ \rightarrow 2^O$ *green realizes* L if for every $x \in (2^I)^+$, if x is green for L, then $x \otimes f(x)$ is green for L.[3] [4] We say that a word $w \in (2^{I\cup O})^*$ is *light green for* L if L^w is universally satisfiable, thus all input sequences are hopeful for L^w. A word $x \in (2^I)^*$ is *light green for* L if there is $y \in (2^O)^*$ such that $x \otimes y$ is light green for L. It is not hard to see that for GE-realizable languages, green and light green coincide. Indeed, if L is universally satisfiable and GE-realizable, then L is realizable.

Theorem 3. GE*-realizability is strictly stronger than green realizability.*

Proof. We first prove that every strategy $f : (2^I)^+ \rightarrow 2^O$ that GE-realizes a specification ψ also green realizes ψ. Consider $x \in (2^I)^+$ that is green for ψ. By definition, there is $y \in (2^O)^+$ such that $L^{x\otimes y}$ is realizable. Then, for every

[3] Note that while the definition of green realization does not refer to ϵ directly, we have that ϵ is green iff L is realizable, in which case all $x \in (2^I)^*$ are green.

[4] While synthesis corresponds to finding a winning strategy for the system, green synthesis can be viewed as a subgame-perfect best-response strategy, where the system does its best in every subgame, even if it loses the overall game.

$x' \in (2^I)^\omega$, there is $y' \in (2^O)^\omega$ such that $x' \otimes y'$ in $L^{x \otimes y}$. Hence, for every $x' \in (2^I)^\omega$, we have that $x \cdot x'$ is hopeful. Therefore, as f GE-realizes ψ, we have that $(x \cdot x') \otimes f(x \cdot x') \models \psi$. Thus, $x \otimes f(x)$ is green, and so f green realizes ψ.

We continue and describe a specification that is green realizable and not GE-realizable. Let $I = \{p\}$ and $O = \{q\}$. Consider the specification $\psi = \mathsf{G}((\mathsf{X}p) \leftrightarrow q)$. Clearly, ψ is not realizable, as the system has to commit a value for q before a value for Xp is known. Likewise, no word $w \in (2^{I \cup O})^*$ is green for ψ, and so no finite input sequence $x \in (2^I)^*$ is green for ψ. Hence, every strategy (vacuously) green realizes ψ. On the other hand, for every input sequences $x \in (2^I)^\omega$ there is an output sequence $y \in (2^O)^\omega$ such that $x \otimes y \models \psi$. Thus, all input sequences are hopeful for ψ. Thus, synthesis and GE-synthesis coincide for ψ, which is not GE-realizable. □

Theorem 3 brings with it two good news. The first is that a GE-realizing transducer has the desired property of being also green realizing. The second has to do with our goal of providing the user with information about the satisfaction status, in particular raising a green flag whenever a green prefix is detected. By Theorem 3, such a flag indicates that the computation generated by our GE-realizing transducer satisfies the specification. A naive way to detect green prefixes for a specification ψ is to solve the synthesis problem for ψ by solving a game on top of a DPW $\mathcal{D}_\psi$ for ψ. The winning positions in the game are states in $\mathcal{D}_\psi$. By defining them as accepting states, we can obtain from $\mathcal{D}_\psi$ a DFW for green prefixes. Then, we run this DFW in parallel with the GE-realizing transducer, and raise the green flag whenever a green prefix is detected. This, however, requires a generation of $\mathcal{D}_\psi$ and a solution of parity games. Below we describe a much simpler way, which makes use of the fact that our transducer GE-realizes the specification.

Recall that if L is universally satisfiable and GE-realizable, then L is realizable. Accordingly, given a transducer $\mathcal{T}$ that GE-realizes ψ, we can augment it with green flags by running in parallel a DFW that detects light-green prefixes. As we argue below, constructing such a DFW only requires an application of the subset construction on top of an NBW for the existential projection of ψ on 2^I.

Lemma 1. *Given an LTL formula ψ over $I \cup O$, we can construct a DFA $\mathcal{S}$ of size $2^{2^{O(|\psi|)}}$ such that $L(\mathcal{S}) = \{x \in (2^I)^* : x$ is light green for $L(\psi)\}$.*

Proof. Let $\mathcal{A}_\psi = \langle 2^{I \cup O}, Q, \delta, Q_0, \alpha \rangle$ be an NBW for $L(\psi)$, and let $\mathcal{B}_\psi = \langle 2^I, Q, \delta', Q_0, \alpha \rangle$ be its existential projection on 2^I. Thus, for every $q \in Q$ and $i \in 2^I$, we have $\delta'(q, i) = \bigcup_{o \in 2^O} \delta(q, i \cup o)$. We define the DFW $\mathcal{S} = \langle 2^I, 2^Q, M, \{Q_0\}, F \rangle$, where M follows the subset construction of $\mathcal{B}_\psi$: for every $S \in 2^Q$ and $i \in 2^I$, we have $M(S, i) = \bigcup_{s \in S} \delta'(s, i)$. Then, $F = \{S \in 2^Q : L(\mathcal{B}_\psi^S) = (2^I)^\omega\}$. Observe that $\mathcal{S}$ rejects $x \in (2^I)^*$ iff there is $x' \in (2^I)^\omega$ such that for all $y \in (2^O)^*$ and $y' \in (2^O)^\omega$, no state in $\delta(Q_0, x \otimes y)$ accepts $x' \otimes y'$. Thus, $\mathcal{S}$ rejects x iff x is not light green, and accepts it otherwise. Note that the definition of F involves universality checking, possibly via complementation, yet no determinization is required, and the size of $\mathcal{S}$ is $2^{2^{O(|\psi|)}}$. □

Note that once we reach an accepting state in $\mathcal{S}$, we can make it an accepting loop. Indeed, once a green prefix is detected, then all prefixes that extend it are green. Accordingly, once the green flag is raised, it stays up. Also note that if an input sequence is not hopeful for ψ, then none of its prefixes is light green for ψ. The converse, however, is not true: an input sequence may be hopeful and still have no light green prefixes. For example, taking $I = \{p\}$, the input sequence $\{p\}^\omega$ is hopeful for $\mathsf{G}p$, yet none of its prefixes is green light, as it can be extended to an input sequence with $\neg p$.

Green flags provide information about satisfaction. Two additional flags of interest are related to safety and co-safety properties:

– A word $w \in (2^{I \cup O})^*$ is *red for* L if $L^w = \emptyset$. A word $x \in (2^I)^*$ is *red for* L if for all $y \in (2^O)^*$, we have that $x \otimes y$ is red for L. Thus, when the environment generates x, then no matter how the system responds, L is not satisfied.
– a word $w \in (2^{I \cup O})^*$ is *blue for* L when $L^w = (2^{I \cup O})^\omega$, and then define a word $x \in (2^I)^*$ as *blue for* L if there is $y \in (2^O)^*$ such that $x \otimes y$ is blue for L. Thus, when the environment generates x, the system can respond in a way that guarantees satisfaction no matter how the interaction continues.

A monitor that detects red and blue prefixes for L can be added to a transducer that GE-realizes L. As has been the case with the monitor for green prefixes, its construction is based on applying the subset construction on an NBW for L [16]. Also, as in the green case, once a red or blue flag is raised, it stays up. In a way analogous to green realizability, we seek a transducer that GE-realizes the specification and generates a red prefix only if all interactions generate a red prefix, and generates a blue prefix whenever this is possible. In Appendix B, we show that while GE-realization implies *red realization*, it may conflict with *blue realization*.

5 High-Quality Good-Enough Synthesis

GE-synthesis is of special interest when the satisfaction value of the specification is multi-valued, and we want to synthesize high-quality systems. We start by defining the multi-valued logic LTL[$\mathcal{F}$] , which is our multi-valued specification formalism. We then study LTL[$\mathcal{F}$] GE-synthesis, first in a worst-case approach, where the satisfaction value of a transducer is the satisfaction value of its computation with the lowest satisfaction value, and then in a stochastic approach, where it is the expected satisfaction value, given a distribution of the inputs.

5.1 The logic LTL[$\mathcal{F}$]

Let AP be a set of Boolean atomic propositions and let $\mathcal{F} \subseteq \{f : [0,1]^k \to [0,1] : k \in \mathbb{N}\}$ be a set of *quality operators*. An LTL[$\mathcal{F}$] formula is one of the following:
– `True`, `False`, or p, for $p \in AP$.
– $f(\psi_1, ..., \psi_k)$, $\mathsf{X}\psi_1$, or $\psi_1 \mathsf{U} \psi_2$, for LTL[$\mathcal{F}$] formulas $\psi_1, \ldots, \psi_k$ and a function $f \in \mathcal{F}$.

The semantics of LTL[$\mathcal{F}$] formulas is defined with respect to infinite computations over AP. For a computation $w = w_0, w_1, \ldots \in (2^{AP})^\omega$ and position $j \geq 0$, we use w^j to denote the suffix $w_j, w_{j+1}, \ldots$. The semantics maps a computation w and an LTL[$\mathcal{F}$] formula ψ to the *satisfaction value* of ψ in w, denoted $[\![w, \psi]\!]$. The satisfaction value is in $[0, 1]$ and is defined inductively as follows. [5]

- $[\![w, \mathtt{True}]\!] = 1$ and $[\![w, \mathtt{False}]\!] = 0$.
- For $p \in AP$, we have that $[\![w, p]\!] = 1$ if $p \in w_0$, and $[\![w, p]\!] = 0$ if $p \notin w_0$.
- $[\![w, f(\psi_1, ..., \psi_k)]\!] = f([\![w, \psi_1]\!], ..., [\![w, \psi_k]\!])$.
- $[\![w, \mathsf{X}\psi_1]\!] = [\![w^1, \psi_1]\!]$.
- $[\![w, \psi_1 \mathsf{U} \psi_2]\!] = \max\limits_{i \geq 0}\{\min\{[\![w^i, \psi_2]\!], \min\limits_{0 \leq j < i} [\![w^j, \psi_1]\!]\}\}$.

The logic LTL can be viewed as LTL[$\mathcal{F}$] for $\mathcal{F}$ that models the usual Boolean operators. In particular, the only possible satisfaction values are 0 and 1. For simplicity, we abbreviate common functions as described below. In addition, we introduce notations for two useful quality operators, namely factoring and weighted average. Let $x, y, \lambda \in [0, 1]$. Then,

- $\neg x = 1 - x$
- $x \vee y = \max\{x, y\}$
- $x \wedge y = \min\{x, y\}$
- $x \to y = \max\{1 - x, y\}$
- $\triangledown_\lambda x = \lambda \cdot x$
- $x \oplus_\lambda y = \lambda \cdot x + (1 - \lambda) \cdot y$

Example 2. Consider a scheduler that receives requests and generates grants, and consider the LTL[$\mathcal{F}$] formula $\psi = \psi_1 \oplus_{\frac{3}{4}} \psi_2$, with $\psi_1 = \mathsf{G}(\mathit{req} \to (\mathit{grant} \vee \triangledown_{\frac{2}{3}} \mathsf{X}\mathit{grant}))$ and $\psi_2 = \mathsf{F}\mathit{req}$. The satisfaction value of the formula ψ_1 is 1 if every request is granted immediately. If the grant is given with a delay, the satisfaction value reduces to $\frac{2}{3}$. In addition, the weighted average with ψ_2 implies that ψ_1 contributes up to $\frac{3}{4}$ of the satisfaction value of ψ: if there are no requests, and thus ψ_2 is violated, then the satisfaction of ψ_1 is vacuous, and the satisfaction value of ψ is penalized by $\frac{1}{4}$. The example demonstrates how LTL[$\mathcal{F}$] can conveniently prioritize different scenarios, as well as embody vacuity considerations in the formula. $\square$

The realizability problem for LTL[$\mathcal{F}$] is an optimization problem: For an LTL[$\mathcal{F}$] specification ψ and a transducer $\mathcal{T}$, we define the satisfaction value of ψ in $\mathcal{T}$, denoted $[\![\mathcal{T}, \psi]\!]$, by $\min\{[\![x \otimes \mathcal{T}(x), \psi]\!] : x \in (2^I)^\omega\}$, namely the satisfaction value of ψ in the worst-case. Then, the synthesis problem is to find, given ψ, a transducer that maximizes its satisfaction value. Moving to a decision problem, given ψ and a threshold value $v \in [0, 1]$, we say that ψ is *v-realizable* if there exists a transducer $\mathcal{T}$ such that $[\![\mathcal{T}, \psi]\!] \geq v$, and the synthesis problem is to find, given ψ and v, a transducer $\mathcal{T}$ that v-realizes ψ.

For an LTL[$\mathcal{F}$] formula ψ, let $V(\psi)$ be the set of possible satisfaction values of ψ in arbitrary computations. Thus, $V(\psi) = \{[\![w, \psi]\!] \ : \ w \in (2^{AP})^\omega\}$.

[5] The observant reader may be concerned by our use of max and min where sup and inf are in order. In Theorem 4 we state that there are only finitely many satisfaction values for a formula, thus the semantics is well defined.

Theorem 4. [1] *Consider an* LTL[$\mathcal{F}$] *formula* ψ.

- $|V(\psi)| \leq 2^{|\psi|}$.
- *For every predicate* $P \subseteq [0,1]$, *there exists an NBW* $\mathcal{A}_\psi^P$ *such that* $L(\mathcal{A}_\psi^P) = \{w : [\![w, \psi]\!] \in P\}$. *Furthermore,* $\mathcal{A}_\psi^P$ *has at most* $2^{O(|\psi|^2)}$ *states* [1].

As with LTL, we define the existential and universal extensions EQLTL[$\mathcal{F}$] and AQLTL[$\mathcal{F}$] of LTL[$\mathcal{F}$] . Here too, we consider the case $AP = I \cup O$, with the signals in O being quantified. Then, $[\![w, \exists O.\psi]\!] = \max_{y \in (2^O)^\omega}\{[\![w_{|I} \otimes y, \psi]\!]\}$ and $[\![w, \forall O.\psi]\!] = \min_{y \in (2^O)^\omega}\{[\![w_{|I} \otimes y, \psi]\!]\}$.

Remark 3. **[On the Semantics of** EQLTL[$\mathcal{F}$] **]** It is tempting to interpret an expression like $[\![w, \exists O.\psi]\!] \leq v$ as "there exists an output sequence y such that $[\![w_I \otimes y, \psi]\!] \leq v$". By the semantics of $\exists O.\psi$, however, $[\![w, \exists O.\psi]\!] \leq v$ actually means that $\max_{y \in (2^O)^\omega}[\![w_I \otimes y, \psi]\!] \leq v$. Thus, the correct interpretation is "for all output sequences y, we have that $[\![w_I \otimes y, \psi]\!] \leq v$". $\square$

5.2 LTL[$\mathcal{F}$] GE-Synthesis

For a value $v \in [0,1]$, we say that x is *v-hopeful for ψ* if there is $y \in (2^O)^\omega$ such that $[\![x \otimes y, \psi]\!] \geq v$. We study two variants of LTL[$\mathcal{F}$] GE-synthesis:

- In LTL[$\mathcal{F}$] GE-*synthesis with a threshold*, the input is an LTL[$\mathcal{F}$] formula ψ and a value $v \in [0,1]$, and the goal is to generate a transducer whose computation on every input sequence that is v-hopeful has satisfaction value at least v. Formally, a function $f : (2^I)^+ \to 2^O$ GE-realizes ψ with threshold v if for every $x \in (2^I)^\omega$, if x is v-hopeful, then $[\![x \otimes f(x), \psi]\!] \geq v$.
- In LTL[$\mathcal{F}$] GE-*synthesis*, the input is an LTL[$\mathcal{F}$] formula ψ, and the goal is to generate a transducer whose computation on every input sequence has the highest possible satisfaction value for this input sequence. Formally, a function $f : (2^I)^+ \to 2^O$ GE-realizes ψ if for every $x \in (2^I)^\omega$ and value $v \in [0,1]$, if x is v-hopeful, then $[\![x \otimes f(x), \psi]\!] \geq v$.

In the Boolean case, the two variants coincide, taking $v = 1$. Indeed, then, for every $x \in (2^I)^\omega$, if x is hopeful, then $x \otimes f(x)$ has to satisfy ψ. We note that GE-realization with a threshold is not monotone, in the sense that decreasing the threshold need not lead to GE-realization. Indeed, the lower is the threshold v, the more input sequences are v-helpful (see Example 3). Accordingly, we do not search for a maximal threshold, and rather may ask about a desired threshold or about GE-synthesis without a threshold.

Solving the GE-synthesis problem, a naive combination of the automata construction of Theorem 4 with the projection technique of Theorem 1, corresponds to an erroneous semantics of EQLTL[$\mathcal{F}$], as noted in Remark 3. Before describing our construction, it is helpful to state the correct (perhaps less intuitive) interpretation of existential and universal quantification in the quantitative setting:

Lemma 2. *For every* LTL[$\mathcal{F}$] *formula* ψ *and an input sequence* $x \in (2^I)^\omega$, *we have that* $[\![x, \exists O.\psi]\!] = 1 - [\![x, \forall O.\neg\psi]\!]$. *Accordingly, for every value* $v \in [0,1]$, *we have that* $[\![x, \exists O.\psi]\!] < v$ *iff* $[\![x, \forall O.\neg\psi]\!] > 1 - v$.

Proof. By definition, $[\![x, \exists O.\psi]\!] = \max_{y \in (2^O)^\omega} [\![x \otimes y, \psi]\!] = 1 - \min_{y \in (2^O)^\omega} 1 - [\![x \otimes y, \psi]\!] = 1 - \min_{y \in (2^O)^\omega} [\![x \otimes y, \neg\psi]\!] = 1 - [\![x, \forall O.\neg\psi]\!]$. Then, $[\![x, \exists O.\psi]\!] < v$ iff $1 - [\![x, \exists O.\psi]\!] > 1 - v$ iff $[\![x, \forall O.\neg\psi]\!]) > 1 - v$. □

Consider an LTL[$\mathcal{F}$] formula ψ, a value $v \in [0,1]$, and an input sequence $x \in (2^I)^\omega$. Recall that x is v-hopeful for ψ if there is $y \in (2^O)^\omega$ such that $[\![x \otimes y, \psi]\!] \geq v$. Equivalently, $[\![x, \exists O.\psi]\!] \geq v$. Indeed, $[\![x, \exists O.\psi]\!] = \max_{y \in (2^O)^\omega} [\![x \otimes y, \psi]\!]$, which is greater or equal to v iff there is $y \in (2^O)^\omega$ such that $[\![x \otimes y, \psi]\!] \geq v$. Hence, x is not v-hopeful for ψ if $[\![x, \exists O.\psi]\!] < v$. Equivalently, by Lemma 2, $[\![x, \forall O.\neg\psi]\!] > 1 - v$. Accordingly, for a strategy $f : (2^I)^+ \to 2^O$, an input sequence $x \in (2^I)^\omega$, and a value $v \in [0,1]$, we say that f is *v-good for x with respect to* ψ, if $[\![x \otimes f(x), \psi]\!] \geq v$ or $[\![x, \forall O.\neg\psi]\!] > 1 - v$.

Example 3. Let $I = \{p\}$ and $O = \{q\}$. Consider the LTL[$\mathcal{F}$] formula $\psi = (\nabla_{\frac{1}{4}} p \vee \nabla_{\frac{1}{2}} q)$. Checking for which values v a strategy f is v-good for x with respect to ψ, we examine whether $[\![x \otimes f(x), \nabla_{\frac{1}{4}} p \vee \nabla_{\frac{1}{2}} q]\!] \geq v$ or $[\![x, \forall q.\neg(\nabla_{\frac{1}{4}} p \vee \nabla_{\frac{1}{2}} q)]\!] > 1 - v$. Since ψ refers only to the first position in the computation, it is enough to examine x_0 and $f(x_0)$. For example, if $x_0 = \emptyset$ and $f(x_0) = \emptyset$, then $[\![x \otimes f(x), \nabla_{\frac{1}{4}} p \vee \nabla_{\frac{1}{2}} q]\!] = 0$, $[\![x, \exists q.\nabla_{\frac{1}{4}} p \vee \nabla_{\frac{1}{2}} q]\!] = \max\{0, \frac{1}{2}\} = \frac{1}{2}$, and $[\![x, \forall q.\neg(\nabla_{\frac{1}{4}} p \vee \nabla_{\frac{1}{2}} q)]\!] = \min\{1, 1 - \frac{1}{2}\} = \frac{1}{2}$. Hence, f is v-good for x with respect to ψ if $v = 0$ or $v > \frac{1}{2}$, thus $v \in \{0\} \cup (\frac{1}{2}, 1]$. Similarly, we have the following (see Appendix C for a detailed analysis).

- If $x_0 = \emptyset$ and $f(x_0) = \{q\}$ then f is v-good for x when $v \in [0,1]$.
- If $x_0 = \{p\}$ and $f(x_0) = \emptyset$ then f is v-good for x when $v \in [0, \frac{1}{4}] \cup (\frac{1}{2}, 1]$.
- If $x_0 = \{p\}$ and$f(x_0) = \{q\}$ then f is v-good for x when $v \in [0,1]$.

Theorem 5. *The* LTL[$\mathcal{F}$] GE*-synthesis with threshold problem is 2EXPTIME-complete.*

Proof. We start with the upper bound, and show we can adjust the upper bound described in the proof of Theorem 1 to the multi-valued setting. Given an LTL[$\mathcal{F}$] formula ψ over $I \cup O$ and a threshold $v \in [0,1]$, we describe an algorithm that returns a transducer $\mathcal{T}$ that GE-realizes ψ with threshold v, or declares that no such transducer exists.

By definition, we have that $\mathcal{T}$ GE-realizes ψ with threshold v if for every input sequence x, we have that $f_\mathcal{T}$ is v-good for x with respect to ψ. Thus, $[\![x \otimes f_\mathcal{T}(x), \psi]\!] \geq v$ or $[\![x, \forall O.\neg\psi]\!] > 1 - v$. We construct a UCW whose language is $\{w \in (2^{I \cup O})^\omega : [\![w, \psi]\!] \geq v$ or $[\![w, \forall O.\neg\psi]\!] > 1 - v\}$.

Let $\mathcal{A}_\psi^{<v}$ be an NBW for $\{w : [\![w, \psi]\!] < v\}$ and $\mathcal{A}_{\exists O.\psi}^{\geq v}$ be an NBW for $\{w : [\![w, \exists O.\psi]\!] \geq v\}$. Thus, $\mathcal{A}_{\exists O.\psi}^{\geq v}$ is obtained from an NBW $\mathcal{A}_\psi^{\geq v}$ for $\{w : [\![w, \psi]\!] \geq v\}$ by existentially projecting its transitions on 2^I. By Theorem 4, both $\mathcal{A}_\psi^{<v}$ and $\mathcal{A}_{\exists O.\psi}^{\geq v}$ are of size exponential in ψ.

Let $\mathcal{B}_\psi^v$ be an NBW for the intersection of $\mathcal{A}_\psi^{<v}$ and $\mathcal{A}_{\exists O.\psi}^{\geq v}$. The language of B_ψ^v is then $\{w \in (2^{I \cup O})^\omega : [\![w, \psi]\!] < v$ and $[\![w, \exists O.\psi]\!] \geq v\}$. We then solve usual synthesis for the complementing UCW, whose language is $\{w \in (2^{I\cup O})^\omega : [\![w,\psi]\!] \geq v \text{ or } [\![w, \forall O.\neg\psi]\!] > 1-v\}$, as required. By [17], the synthesis problem for UCW can be solved in EXPTIME.

The lower bound follows from the 2EXPTIME-hardness of LTL GE-realizability. □

Theorem 6. *The* LTL[$\mathcal{F}$] GE-*synthesis problem is 2EXPTIME-complete.*

Proof. We start with the upper bound. Given an LTL[$\mathcal{F}$] specification ψ over $I \cup O$, we describe an algorithm that returns a transducer $\mathcal{T}$ that GE-realizes ψ or declares that no such transducer exists.

As discussed above, a transducer $\mathcal{T}$ GE-realizes ψ iff for every input sequence $x \in (2^I)^\omega$ and value $v \in [0,1]$, we have that $f_{\mathcal{T}}$ is v-good for x with respect to ψ. Accordingly, we construct a UCW whose language is $\bigcap_{v\in V(\psi)}\{w \in (2^{I\cup O})^\omega : [\![w,\psi]\!] \geq v \text{ or } [\![w,\forall O.\neg\psi]\!] > 1-v\}$.

For $v \in V(\psi)$, let $\mathcal{B}^v_\psi$ be an NBW for $\{w : [\![w,\neg\psi]\!] \geq v \text{ and } [\![w,\exists O.\psi]\!] \geq v\}$, as constructed in the proof of Theorem 5, and let $\mathcal{B}$ be the union of $\mathcal{B}^v_\psi$ for all $v \in V(\psi)$. By Theorem 4, the size of $V(\psi)$ is exponential in ψ, and thus so is the size of $\mathcal{B}$. We then solve usual synthesis for the complementing UCW, whose language is as required. By [17], the synthesis problem for UCW can be solved in EXPTIME. The lower bound follows from the 2EXPTIME-hardness of LTL GE-realizability. □

Remark 4. [**Tuning Hope Down**] The quantitative setting allows the designer to tune down "satisfaction by hoplessness": rather than synthesizing $\psi \vee \forall O.\neg\psi$, we can have a factor λ and synthesize $\psi \vee \triangledown_\lambda \forall O.\neg\psi$. In Section 5.3 below we study additional ways to refer to hopefulness levels.

5.3 LTL[$\mathcal{F}$] Assume-Guarantee GE-Synthesis

In Section 5.2, we seek a transducer $\mathcal{T}$ such that for a given or for all values $v \in [0,1]$ and input sequences $x \in (2^I)^\omega$, if $[\![x,\exists O.\psi]\!] \geq v$ then $[\![x \otimes \mathcal{T}(x),\psi]\!] \geq v$. In this section we measure the quality of a transducer $\mathcal{T}$ by analyzing richer relations between $[\![x,\exists O.\psi]\!]$ and $[\![x \otimes \mathcal{T}(x),\psi]\!]$. The setting has the flavor of quantitative assume-guarantee synthesis [3]. There, the specification consists of a multi-valued assumption A, which in our case is $\exists O.\psi$, and a multi-valued guarantee G, which is our case is ψ.

There are different ways to analyze the relation between $[\![x,\exists O.\psi]\!]$ and $[\![x \otimes \mathcal{T}(x),\psi]\!]$. To this end, we assume that we are given a function `comb` : $[0,1] \times [0,1] \to [0,1]$ that given the satisfaction values of $\exists O.\psi$ and of ψ, outputs a combined satisfaction value. We assume that `comb` is decreasing in the first component and increasing in the second component. This corresponds to the intuition that a lower satisfaction value of $\exists O.\psi$ and a higher satisfaction value of ψ both yield a higher overall score. Also, since $[\![x,\exists O.\psi]\!] \geq [\![x \otimes \mathcal{T}(x),\psi]\!]$ for all $x \in (2^I)^\omega$, we assume that the first component is greater than or equal to the second. Finally, we require `comb` to be efficiently computed. Some natural `comb` functions include:

- The quantitative implication function: $\texttt{comb}(A, G) = \max\{1 - A, G\}$. This captures the quantitative notion of the implication $(\exists O.\psi) \to \psi$.
- The (negated) difference function: $\texttt{comb}(A, G) = 1 - (A - G)$. This captures how far the satisfaction value for the given computation is from the best satisfaction value. Since $A \geq G$, the range of the function is indeed $[0, 1]$.
- The ratio function, given by some normalization to $[0, 1]$ of the function $\texttt{comb}(A, G) = \frac{G}{A}$, which captures the "relative success" with respect to the best possible satisfaction value.

The choice of an appropriate comb function depends on the setting. Implication is in order when harsh environments may outweigh the actual performance of the system. For example, if our specification measures the uptime of a server in a cluster, then environments that cause very frequent power failures render the server unusable, as the overhead of reconnecting it outweighs its usefulness. In such a case, being shut down is better than continuously trying to reconnect, and so we give a higher satisfaction value for the server being down, which depends only on the environment. Then, as demonstrated with the cleaning robot in Section 1, the difference and ratio functions are fairly natural when measuring "realization of potential". We now describe a more detailed example when these measures are in order.

Example 4. Consider a controller for an elevator in an n-floor building. In each moment in time, the environment sends to the controller requests, by means of a truth assignment to $I = \{1, \ldots, n\}$, indicating the subset of floors in which the elevator is requested. Then, the controller assigns values to $O = \{up, down\}$, directing the elevator to go up, go down, or stay. The satisfaction value of the specification ψ reflects the waiting time of the request with the slowest response: it is 0 when this time is more than $2n$, and is 1 when the slowest request is granted immediately. Sure enough, there is no controller that attains satisfaction value 1 on all input sequences, and so ψ is not realizable with satisfaction value 1. Also, adding assumptions about the behavior of the environment is not of much interest. Using AG GE-realizability, we can synthesize a controller that behaves as well as possible. For example, using the difference function, we measure the performance of the controller on an input sequence $x \in (2^I)^\omega$ with respect to the best possible performance on x. Note that such a best performance needs a look-ahead on requests yet to come, which is indeed the satisfaction value of $\exists O.\psi$ in x. Thus, the assumption $[\![x, \exists O.\psi]\!]$ actually gives us the performance of a good-enough *off-line* controller. Accordingly, using the ratio function, we can synthesize a system with the best *competitive ratio* for an on-line interaction [7]. □

Given an LTL$[\mathcal{F}]$ formula ψ and a function comb, we define the GE-*AG-realization value* of ψ in a transducer $\mathcal{T}$ by $\min\{\texttt{comb}([\![x, \exists O.\psi]\!], [\![x \otimes \mathcal{T}(x), \psi]\!]) : x \in (2^I)^\omega\}$. Then, our goal in *AG* GE-*realizability* is to find, given an LTL$[\mathcal{F}]$ formula ψ and a function comb, the maximal value $v \in [0, 1]$ such that there exists a transducer $\mathcal{T}$ whose AG GE-realization value of ψ is v. The *AG* GE-*synthesis* problem is then to find such a transducer.

We start by solving the decision version of AG GE-realizability.

Theorem 7. *The problem of deciding, given an* LTL$[\mathcal{F}]$ *formula ψ, a function* `comb`*, and a threshold $v \in [0,1]$, whether there exists a transducer $\mathcal{T}$ whose AG* GE*-realization value of ψ is v, is 2EXPTIME-complete.*

Proof. Recall that $V(\psi)$ is the set of possible satisfaction values of ψ (and hence of $\exists O.\psi$), and that by Theorem 4, we have that $|V(\psi)| \leq 2^{|\psi|}$. Let $G_v = \{\langle v_1, v_2\rangle \in V(\psi) \times V(\psi) : \texttt{comb}(v_1, v_2) \geq v\}$. Intuitively, G is the set of satisfaction-value pairs $\langle [\![w, \exists O.\psi]\!], [\![w, \psi]\!]\rangle$ that are allowed to be generated by a transducer whose AG GE-realization value of ψ is at least v. By definition, AG GE-realization of ψ with value v coincides with realization of the language $L_v = \{w \in (2^{I \cup O})^\omega : \texttt{comb}([\![w, \exists O.\psi]\!], [\![w, \psi]\!]) \geq v\}$. By the monotonicity assumption on `comb`, for every $\langle v_1, v_2\rangle \in G_v$, we have that $\langle v_1', v_2'\rangle \in G$ for every $v_1' \leq v_1$ and $v_2' \geq v_2$. Hence, we can write

$$L_v = \bigcup_{\langle v_1, v_2\rangle \in G_v} \{w \in (2^{I \cup O})^\omega : [\![w, \exists O.\psi]\!] \leq v_1 \text{ and } [\![w, \psi]\!] \geq v_2\},$$

and proceed to construct an NBW for L_v by taking the union of NBWs $\mathcal{A}_{v_1, v_2}$ for all $\langle v_1, v_2\rangle \in G_v$, each of which is the product of NBWs $\mathcal{A}^{\leq v_1}_{\exists O.\psi}$ and $\mathcal{A}^{\geq v_2}_{\psi}$, as in the proof of Theorem 5.

Aiming to proceed Safralessly, we can also construct a UCW for L_v, as follows. First, note that by the monotonicity of `comb`, for every $\langle v_1, v_2\rangle \in V(\psi) \times V(\psi)$ we have that $\langle v_1, v_2\rangle \in G_v$ iff for every $\langle u_1, u_2\rangle \in V(\psi) \times V(\psi) \setminus G_v$, we have that $v_1 < u_1$ or $v_2 > u_2$. Hence,

$$L_v = \bigcap_{\langle u_1, u_2\rangle \in V(\psi) \times V(\psi) \setminus G_v} \{w \in (2^{I \cup O})^\omega : [\![w, \exists O.\psi]\!] < u_1 \text{ or } [\![w, \psi]\!] > u_2\},$$

and so by dualization we have

$$(2^{I \cup O})^\omega \setminus L_v = \bigcup_{\langle u_1, u_2\rangle \in V(\psi) \times V(\psi) \setminus G_v} \{w \in (2^{I \cup O})^\omega : [\![w, \exists O.\psi]\!] \geq u_1 \text{ and } [\![w, \psi]\!] \leq u_2\}.$$

Hence, we can obtain a UCW for L_v by dualizing an NBW that is the union of NBWs $\mathcal{A}_{u_1, u_2}$, for all $\langle u_1, u_2\rangle \in V(\psi) \times V(\psi) \setminus G_v$, each of which is the product of NBWs $\mathcal{A}^{\geq u_1}_{\exists O.\psi}$ and $\mathcal{A}^{\leq u_2}_{\psi}$.

Observe that in all cases, the size of the NBW is $2^{O(|\psi|)}$. Indeed, there are at most $2^{2|\psi|}$ pairs in the union, and, by Theorem 4, the size of the NBW for each pair is $2^{O(|\psi|)}$.

The lower bound follows from the 2EXPTIME-hardness of LTL GE-realizability. □

By Theorem 4, the number of possible satisfaction values for ψ is at most $2^{|\psi|}$. Thus, the number of possible values for $\texttt{comb}(A, G)$, where A and G are satisfaction values of ψ, is at most $2^{2|\psi|}$. Using binary search over the image of `comb`, we can use Theorem 7 to obtain the following.

Corollary 1. *The AG* GE*-synthesis problem can be solved in doubly-exponential time.*

Remark 5. [**GE-synthesis as a Special Case of Assume Guarantee GE-Synthesis**] The two approaches taken in Section 5.2 can be captured by an appropriate `comb` function. Indeed, for GE-synthesis with a threshold, we can use the function `comb` with $\texttt{comb}(A, G) = 1$ if $A \geq v \to G \geq v$, and $\texttt{comb}(A, G) = 0$ otherwise. For GE-synthesis (without a threshold), we can use the function `comb` with $\texttt{comb}(A, G) = 1$ if $A = G$, and $\texttt{comb}(A, G) = 0$ otherwise (recall that $A \geq G$ by definition). However, the solution described in Section 5.2 is simpler than the one described here for the general case. $\square$

5.4 LTL[$\mathcal{F}$] GE-synthesis in Stochastic Environments

The setting of LTL[$\mathcal{F}$] GE-synthesis studied in Sections 5.2 and 5.3 takes the different satisfaction values into an account, but is binary, in the sense that a specification is either (possibly AG) GE-realizable, or is not. In particular, in case the specification is not GE-realizable, synthesis algorithms only return “no”. In this section we add a quantitative measure also to the underlying realizability question. We do so by assuming a stochastic environment, with a known distribution on the inputs sequences, and analyzing the expected performance of the system.

For completeness, we remind the reader of some basics of probability theory. For a comprehensive reference see e.g., [25]. Let Σ be a finite alphabet, and let ν be some *probability distribution* over Σ^ω. For example, in the uniform distribution over $(2^I)^\omega$, the probability space is induced by sampling each letter with probability $2^{-|I|}$, corresponding to settings in which each signal in I always holds in probability $\frac{1}{2}$. We assume ν is given by a finite Markov Decision Process (MDP). That is, ν is induced by the distribution of each letter $i \in 2^I$ at each time step, determined by a finite stochastic control process that takes into account also the outputs generated by the system (see [2] for the precise model). A *random variable* is then a function $X : \Sigma^\omega \to \mathbb{R}$. When X has a finite image V, which is the case in our setting, its *expected value* is $\mathbb{E}[X] = \sum_{v \in V} v \cdot \Pr(X^{-1}(v))$. Intuitively, $\mathbb{E}[X]$ is the “average” value that X attains. Next, consider an *event* $E \subseteq \Sigma^\omega$. The *conditional expectation of* X *with respect to* E is $\mathbb{E}[X|E] = \frac{\mathbb{E}[\mathbb{1}_E X]}{\Pr(E)}$, where $\mathbb{1}_E X$ is the random variable that assigns $X(w)$ to $w \in E$ and 0 to $w \notin E$. Intuitively, $\mathbb{E}[X|E]$ is the average value that X attains when restricting to words in E, and normalizing according to the probability of E itself.

We continue and review the *high-quality synthesis problem* [2], where the GE variant is not considered. There, the environment is assumed to be stochastic and we care for the expected satisfaction value of an LTL[$\mathcal{F}$] specification in the computations of a transducer $\mathcal{T}$, assuming some given distribution on the inputs sequences. Formally, let $X_{\mathcal{T},\psi} : (2^I)^\omega \to \mathbb{R}$ be a random variable that assigns each sequence $x \in (2^I)^\omega$ of input signals with $[\![\mathcal{T}(x), \psi]\!]$. Then, when the sequences in $(2^I)^\omega$ are sampled according to a given distribution ν of $(2^I)^\omega$, we define $[\![\mathcal{T},\psi]\!]^\nu = \mathbb{E}[X_{\mathcal{T},\psi}]$. Since ν is fixed, we omit it from the notation and use $[\![\mathcal{T},\psi]\!]$ in the following.

Remark 6. **[Relating LTL GE-synthesis with LTL[$\mathcal{F}$] Synthesis in Stochastic Environments]** Given an LTL formula ψ, we can view it as an LTL[$\mathcal{F}$] formula with possible satisfaction values $\{0,1\}$, apply to it high-quality synthesis *a-la* [2], and find a transducer $\mathcal{T}$ that maximizes $\mathbb{E}[X_{\mathcal{T},\psi}]$. An interesting observation is that if $\mathcal{T}$ GE-realizes ψ, then it also maximizes $\mathbb{E}[X_{\mathcal{T},\psi}]$. Indeed, all input sequences that can contribute to the expected satisfaction value, do so. $\square$

We introduce and study two measures for high-quality synthesis in a stochastic environment. In the first, termed *expected* GE-*synthesis*, all input sequences are sampled, yet the satisfaction value in each input sequence takes its hopefulness level into account. In the second, termed *conditional expected* GE-*synthesis*, only hopeful input sequences are sampled.

We start with expected GE-synthesis. There, instead of associating each sequence $x \in (2^I)^\omega$ with $[\![x\otimes\mathcal{T}(x),\psi]\!]$, we associate it with $X^{\texttt{comb}}_{\mathcal{T},\psi} = \texttt{comb}([\![x,\exists O.\psi]\!],[\![x\otimes\mathcal{T}(x),\psi]\!]\}$, where $\texttt{comb}$ is as described in Section 5.3, thus capturing the assume-guarantee semantics of quantitative GE-synthesis. Then, we define $[\![\mathcal{T},\psi]\!]^{\texttt{comb}} = \mathbb{E}[X^{\texttt{comb}}_{\mathcal{T},\psi}]$. For example, taking $\texttt{comb}$ as implication, we have $X^{\texttt{comb}}_{\mathcal{T},\psi} = \max\{[\![x\otimes\mathcal{T}(x),\psi]\!],[\![x,\forall O.\neg\psi]\!]\}$, capturing the semantics of $(\exists O.\psi)\rightarrow\psi$.

Then, in conditional expected GE-synthesis, we consider $\exists O.\psi$ as an environment assumption, and factor it in using conditional expectation, parameterized by a threshold $v \in [0,1]$. Formally, let $\exists O.\psi \geq v$ denote the event $\{x \in (2^I)^\omega : [\![x,\exists O.\psi]\!] \geq v\}$. Then, we define $[\![\mathcal{T},\psi]\!]^{\mathrm{cond}(v)} = \mathbb{E}[X_{\mathcal{T},\psi}|\exists O.\psi \geq v]$, assuming the event $\exists O.\psi \geq v$ has a strictly positive probability.

In [2], it is shown that the high-quality synthesis problem can be solved in doubly-exponential time, also in the presence of environment assumptions. In the solution, the first step is the translation of the involved formulas to DPWs. In order to extract from [2] the results relevant to us, we describe them by means of *discrete quantitative specifications*, defined as follows. A discrete quantitative specification Ψ over $I\cup O$ is given by means of a sequence $\mathcal{A}_1,\ldots,\mathcal{A}_n$ of DPWs, with $(2^{I\cup O})^\omega = L(\mathcal{A}_1) \supseteq L(\mathcal{A}_2) \supseteq \ldots \supseteq L(\mathcal{A}_n)$, and sequence $0 \leq v_1 < \ldots < v_n \leq 1$ of values. For every $w \in (2^{I\cup O})^\omega$, the satisfaction value of w in Ψ, denoted $[\![w,\Psi]\!]$, is $\max\{v_i : w \in L(\mathcal{A}_i)\}$. We refer to n as the depth of Ψ.

Theorem 8 ([2]). *Consider a discrete quantitative specification Ψ over $I\cup O$. Let n be its depth and m be the size of the largest DPW in Ψ. For a transducer $\mathcal{T}$, let $X_\mathcal{T}$ be a random variable that assigns a word $x \in (2^I)^\omega$ with $[\![x\otimes\mathcal{T}(x),\Psi]\!]$.*
1. *We can synthesize a transducer $\mathcal{T}$ that maximizes $\mathbb{E}[X_\mathcal{T}]$ in time m^n.*
2. *Let $\mathcal{B}$ be a DPW over $(2^I)^\omega$ such that $\Pr(L(\mathcal{B})) > 0$. Then, we can synthesize a transducer $\mathcal{T}$ that maximizes $\mathbb{E}[X_\mathcal{T}|\mathcal{B}]$ in time $m^n\cdot k$, where k is the size of $\mathcal{B}$.*

We can now state the main results of this section.

Theorem 9. *Consider an* LTL[$\mathcal{F}$] *formula ψ.*

1. *Given a function* `comb`*, we can find in doubly-exponential time a transducer that maximizes* $[\![\mathcal{T},\psi]\!]^{\texttt{comb}}$.
2. *Given a threshold* $v \in [0,1]$*, we can find in doubly-exponential time a transducer that maximizes* $[\![\mathcal{T},\psi]\!]^{\text{cond}(v)}$.

Proof. Let $v_1 < v_2 < \ldots < v_n$ be the possible satisfaction values of ψ (and hence also of $\exists O.\psi$ and of $\forall O.\psi$). By Theorem 4, we have that $n \leq 2^{|\psi|}$. For each v_i, we can construct a DPW $\mathcal{D}^{\geq v_i}_{\texttt{comb}(\exists O.\psi,\psi)}$ as in Theorem 7. It is not hard to see that the discrete quantitative specification given by the DPWs $\mathcal{D}^{\geq v_i}_{\texttt{comb}(\exists O.\psi,\psi)}$ and the values v_i, for $1 \leq i \leq n$, is qual to the specification $\texttt{comb}(\exists O.\psi,\psi)$. Thus, by Theorem 8 (1), we can find a transducer that maximizes $\mathbb{E}[X_{\mathcal{T}}]$ in time $(2^{2^{O(|\psi|)}})^{2^{|\psi|}} = 2^{2^{O(|\psi|)}}$.

Next, given $v \in [0,1]$, we can check whether $\Pr(\exists O.\psi > v) > 0$, for example by converting a DPW $\mathcal{D}^{\geq v}_{\exists O.\psi}$ to an MDP, and reasoning about its Ergodic-components. Then, by Theorem 8 (2), we can find a transducer that maximizes $\mathbb{E}[X_{\mathcal{T}}|\exists O\psi > v]$, in time $(2^{2^{O(|\psi|)}})^{2^{|\psi|}} \cdot 2^{2^{O(\psi)}} = 2^{2^{O(|\psi|)}}$. □

Corollary 2. *The (possibly conditional) expected* GE*-synthesis problem for* LTL$[\mathcal{F}]$ *can be solved in doubly-exponential time.*

5.5 Guarantees in High-Quality GE-Synthesis

As in the Boolean setting, also in the high-quality one we would like to add to a GE-realizing transducer guarantees and indications about the satisfaction level. As we detail below, the quantitative setting offers many possible ways to do so.

High-Quality GE-Synthesis with Guarantees We consider specifications of the form $\psi = \psi_{strong} \wedge \psi_{weak}$, where essentially, we seek a transducer that realizes ψ_{strong} and (possibly AG) GE-realizes ψ_{weak}. Maximizing the realization value of ψ_{strong} may conflict with maximizing the GE-realization value of ψ_{weak}, and there are different ways to trade-off the two goals. Technically, in the decision-problem variant, we are given two thresholds $v_1, v_2 \in [0,1]$, and we seek a transducer $\mathcal{T}$ that realizes ψ_{strong} with value at least v_1, and GE-realizes ψ_{weak} with value at least v_2. Then, one may start, for example, by maximizing the value v_1, and then find the maximal value v_2 that may be achieved simultaneously. Alternatively, one may prefer to maximize v_2, or some other combination of v_1 and v_2. Also, it is possible to decompose ψ further, to several strong and weak components, each with its desired threshold.

The solutions in the different settings all involve a construction of a UCW $\mathcal{A}^{\geq v_1}_{\psi_{strong}}$, and its product with the automata constructed in the solutions for the different GE-synthesis variants. We thus have the following. We note that when the solution for ψ_{weak} is Safraless, we can use a UCW for ψ_{strong} to maintain a Safraless construction.

Theorem 10. *The problem of* LTL$[\mathcal{F}]$ *high-quality* GE*-synthesis with a guarantee can be solved in doubly-exponential time.*

Flags by a High-Quality GE-Realizing Transducer In the quantitative setting, we parameterized the flags raised by the GE-realizing transducer by values in $[0,1]$, indicating the announced satisfaction level. Thus, rather than talking about prefixes being green, red, or blue, we talk about them being v-green, v-red, and v-blue, for $v \in [0,1]$, which essentially means that a satisfaction value of at least v is guarantees (in green and blue flags) or is impossible (in red ones). We can think of those as "degrees" of green, red, and blue. Below we formalize this intuition and argue that even an augmentation of a transducer that GE-realizes ψ by flags for all values in $V(\psi)$ leaves the problem in doubly-exponential time.

A *quantitative language* over $2^{I \cup O}$ is $L : (2^{I \cup O})^\omega \to [0,1]$. In particular, an LTL[$\mathcal{F}$] formula ψ defines the quantitative language L_ψ, where for all $w \in (2^{I \cup O})^\omega$, we have $L_\psi(w) = [\![w, \psi]\!]$. For a quantitative language L and a word $w \in (2^{I \cup O})^*$, we define L^w as the quantitative language where for all $w' \in (2^{I \cup O})^\omega$, we have $L^w(w') = L(w \cdot w')$. For a value $v \in [0,1]$, a word $w \in (2^{I \cup O})^*$ is *v-green for L* if L^w is v-realizable. That is, there is a transducer $\mathcal{T}$ such that $[\![T, L^w]\!] \geq v$. A word $x \in (2^I)^*$ is *v-green for L* if there is $y \in (2^O)^*$ such that $x \otimes y$ is v-green for L. Thus, when the environment generates x, the system can respond in a way that would guarantee v-realizability. Finally, we say that L is *green realizable* if there is a strategy $f : (2^I)^+ \to 2^O$ that for every threshold v and for every input $x \in (2^I)^+$ that is v-green for L, we have that $x \otimes f(x)$ is v-green for L. It is not hard to see that Theorem 3 carries over to the quantitative setting:[6]

Theorem 11. *Quantitative optimal realizability is strictly stronger than quantitative green realizability. In particular, if a transducer $\mathcal{T}$ optimally realizes an* LTL[$\mathcal{F}$] *formula ψ, then $\mathcal{T}$ also green realizes ψ.*

For the detection of v-green prefixes, we parameterize by values also the notion of universal satisfiability, and say that L is v-universally-satisfiable if all input sequences are v-hopeful. Then, a word $w \in (2^{I \cup O})^*$ is *light v-green for L* if L^w is v-universally-satisfiable, and $x \in (2^I)^*$ is *light v-green for L* if there exists $y \in (2^O)^*$ such that $x \otimes y$ is light v-green for L. The following quantitative analogue of Lemma 1 can then be proved *mutatis-mutandis*: Given an LTL[$\mathcal{F}$] formula ψ over $I \cup O$ and a threshold $v \in [0,1]$, we can construct a DFA $\mathcal{S}_\psi^v$ of size $2^{2^{|\psi|}}$ such that $L(\mathcal{S}_\psi^v) = \{x \in (2^I)^* : x \text{ is light } v\text{-green for } L_\psi\}$. Now, for languages that are GE-realizable with threshold v, we have that v-green and light v-green coincide. Accordingly, we can use monitors $\mathcal{S}_\psi^v$ for all desired values $v \in [0,1]$. Note that if a transducer GE-realizes ψ with threshold v, then we are interested in a monitor for v (or lower values), and if a transducer GE-realizes ψ without a threshold, then all monitors are of potential interest. Thus, the approximation of v-green by light v-green is sound. Finally, recall that ψ has at most $2^{|\psi|}$ possible satisfaction values, and so even an execution of all monitors

[6] Recall that quantitative optimal synthesis can be parameterized by a threshold $v \in [0,1]$. Likewise, we can parameterize green realizability by a threshold. Then, optimal realizability with threshold v implies green realizability with threshold v.

is still only doubly-exponential. In fact, the monitors for the different thresholds share the same state space and differ only in the definition of accepting states.

We continue to quantitative red and blue prefixes and flags. A word $w \in (2^{I \cup O})^*$ is *v-red for L* if $L^w(w') \leq v$ for every $w' \in (2^{I \cup O})^\omega$. A word $x \in (2^I)^*$ is *v-red for L* if for all $y \in (2^O)^*$, we have that $x \otimes y$ is v-red for L. Thus, when the environment generates x, then no matter how the system responds, L is satisfied with value at most v. Similarly, a word $w \in (2^{I \cup O})^*$ is *v-blue for L* if $L^w(w') \geq v$ for every $w' \in (2^{I \cup O})^\omega$. A word $x \in (2^I)^*$ is *v-blue for L* if there is $y \in (2^O)^*$ such that $x \otimes y$ is v-blue for L. Then, when the environment generates x, the system can respond in a way that guarantees satisfaction value v no matter how the interaction continues. The study of quantitative red and blue realizability and flags proceeds with no surprises as in the Boolean setting.

6 Discussion

We introduced and solved several variants of GE-synthesis. Our complexity results are tight and show that GE-synthesis is not more complex than traditional synthesis. In practice, however, traditional synthesis algorithms do not scale well, and much research is devoted for the development of methods and heuristics for coping with the implementation challenges of synthesis. A natural future research direction is to extend these heuristics and methods for GE-synthesis. We mention here two specific examples.

Efficient synthesis algorithms have been developed for fragments of LTL [21]. Most notable is the *GR(1) fragment* [18], which supports assume-guarantee reasoning, and for which synthesis has an efficient symbolic solution. Adding existential quantification to GR(1) specifications, which is how we handled LTL GE-synthesis, is not handled by its known algorithms, and is an interesting challenge. The success of SAT-based model-checking have led to the development of SAT-based synthesis algorithms [6], where the synthesis problem is reduced to satisfiability of a QBF formula. The fact the setting already includes quantifiers suggests it can be extended to GE-synthesis. A related effort is *bounded synthesis* algorithms [24, 13], where the synthesized systems are assumed to be of a bounded size and can be represented synbolically [10].

References

1. S. Almagor, U. Boker, and O. Kupferman. Formalizing and reasoning about quality. *Journal of the ACM*, 63(3), 2016.
2. S. Almagor and O. Kupferman. High-quality synthesis against stochastic environments. In *Proc. 25th Annual Conf. of the European Association for Computer Science Logic*, volume 62 of *LIPIcs*, pages 28:1–28:17, 2016.
3. S. Almagor, O. Kupferman, J.O. Ringert, and Y. Velner. Quantitative assume guarantee synthesis. In *Proc. 29th Int. Conf. on Computer Aided Verification*, volume 10427 of *Lecture Notes in Computer Science*, pages 353–374. Springer, 2017.

4. R. Bloem, K. Chatterjee, T. Henzinger, and B. Jobstmann. Better quality in synthesis through quantitative objectives. In *Proc. 21st Int. Conf. on Computer Aided Verification*, volume 5643 of *Lecture Notes in Computer Science*, pages 140–156. Springer, 2009.
5. R. Bloem, K. Chatterjee, and B. Jobstmann. Graph games and reactive synthesis. In *Handbook of Model Checking.*, pages 921–962. Springer, 2018.
6. R. Bloem, U. Egly, P. Klampfl, R. Könighofer, and F. Lonsing. Sat-based methods for circuit synthesis. In *Proc. 14th Int. Conf. on Formal Methods in Computer-Aided Design*, pages 31–34. IEEE, 2014.
7. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.
8. K. Chatterjee, T. Henzinger, and B. Jobstmann. Environment assumptions for synthesis. In *Proc. 19th Int. Conf. on Concurrency Theory*, volume 5201 of *Lecture Notes in Computer Science*, pages 147–161. Springer, 2008.
9. A. Church. Logic, arithmetics, and automata. In *Proc. Int. Congress of Mathematicians, 1962*, pages 23–35. Institut Mittag-Leffler, 1963.
10. R. Ehlers. Symbolic bounded synthesis. In *Proc. 22nd Int. Conf. on Computer Aided Verification*, volume 6174 of *Lecture Notes in Computer Science*, pages 365–379. Springer, 2010.
11. D. Fisman, O. Kupferman, and Y. Lustig. Rational synthesis. In *Proc. 16th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, volume 6015 of *Lecture Notes in Computer Science*, pages 190–204. Springer, 2010.
12. O. Kupferman. Automata theory and model checking. In *Handbook of Model Checking*, pages 107–151. Springer, 2018.
13. O. Kupferman, Y. Lustig, M.Y. Vardi, and M. Yannakakis. Temporal synthesis for bounded systems and environments. In *Proc. 28th Symp. on Theoretical Aspects of Computer Science*, pages 615–626, 2011.
14. O. Kupferman, G. Perelli, and M.Y. Vardi. Synthesis with rational environments. *Annals of Mathematics and Artificial Intelligence*, 78(1):3–20, 2016.
15. O. Kupferman, N. Piterman, and M.Y. Vardi. Safraless compositional synthesis. In *Proc. 18th Int. Conf. on Computer Aided Verification*, volume 4144 of *Lecture Notes in Computer Science*, pages 31–44. Springer, 2006.
16. O. Kupferman and M.Y. Vardi. Model checking of safety properties. *Formal Methods in System Design*, 19(3):291–314, 2001.
17. O. Kupferman and M.Y. Vardi. Safraless decision procedures. In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, pages 531–540, 2005.
18. N. Piterman, A. Pnueli, and Y. Saar. Synthesis of reactive(1) designs. In *Proc. 7th Int. Conf. on Verification, Model Checking, and Abstract Interpretation*, volume 3855 of *Lecture Notes in Computer Science*, pages 364–380. Springer, 2006.
19. A. Pnueli. The temporal semantics of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.
20. A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Proc. 16th ACM Symp. on Principles of Programming Languages*, pages 179–190, 1989.
21. P. Madhusudan R. Alur, S. La Torre. Playing games with boxes and diamonds. In *Proc. 14th Int. Conf. on Concurrency Theory*, volume 2761 of *Lecture Notes in Computer Science*, pages 127–141. Springer, 2003.
22. R. Rosner. *Modular Synthesis of Reactive Systems.* PhD thesis, Weizmann Institute of Science, 1992.
23. S. Safra. On the complexity of ω-automata. In *Proc. 29th IEEE Symp. on Foundations of Computer Science*, pages 319–327, 1988.

24. S. Schewe and B. Finkbeiner. Bounded synthesis. In *5th Int. Symp. on Automated Technology for Verification and Analysis*, volume 4762 of *Lecture Notes in Computer Science*, pages 474–488. Springer, 2007.
25. Ross Sheldon. *A first course in probability.* Pearson Education India, 2002.
26. A.P. Sistla, M.Y. Vardi, and P. Wolper. The complementation problem for Büchi automata with applications to temporal logic. *Theoretical Computer Science*, 49:217–237, 1987.
27. M.Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.
28. D. W. Winnicott. *Playing and Reality.* Penguin, 1971.
29. P. Wolper. Temporal logic can be more expressive. In *Proc. 22nd IEEE Symp. on Foundations of Computer Science*, pages 340–348, 1981.

A Counterstrategies in GE-Synthesis

An environment strategy is $g : (2^O)^* \to 2^I$. For an output sequence $y = o_0 \cdot o_1 \cdot o_2 \cdots \in (2^O)^\omega$, we use $g(y)$ to denote the input sequence $g(\epsilon) \cdot g(o_0) \cdot g(o_0 \cdot o_1) \cdot g(o_0 \cdot o_1 \cdot o_2) \cdots \in (2^I)^\omega$. Then, $g(y) \otimes y \in (2^{I \cup O})^\omega$ is the *computation* of g on y. Note that the setting is not completely dual to that of a strategy of the system, as in both cases the environment initiates the interaction, and thus $g(\epsilon)$ is prepended. We say that a specification ψ is O/I-realizable if there is an environment strategy $g : (2^O)^* \to 2^I$ such that for all output sequences $y \in (2^O)^\omega$, the computation of g on y satisfies ψ.

Determinancy of games implies that in traditional synthesis, a specification ψ is not I/O-realizable iff $\neg\psi$ is O/I-realizable. For GE-synthesis, simple dualization does not hold, but we do have determinancy in the following sense. We say that an environment strategy $g : (2^O)^* \to 2^I$ GE-*realizes* ψ if all $y \in (2^I)^\omega$ are hopeful with respect to $\neg\psi$ and g realizes ψ.

Theorem 12. *For every specification ψ, we have that ψ is GE-realizable by the system iff $\neg\psi$ is not GE realizable by the environment.*

Proof. By determinancy of games, we have that $\psi \vee \forall O.\neg\psi$ is I/O-realizable iff $\neg\psi \wedge \exists O.\psi$ is not O/I-realizable. Hence, ψ is GE realizable by the system iff $\psi \vee \forall O.\neg\psi$ is I/O-realizable iff $\neg\psi \wedge \exists O.\psi$ is not O/I-realizable iff $\neg\psi$ is not GE realizable by the environment. □

By Theorem 12, we can solve the synthesis problem for the dual setting in 2EXPTIME by constructing an automaton for $L = \{w : w \models \neg\psi \wedge \exists O.\psi\}$. We note that while a construction of an NBW for L follows the same lines as the construction in the proof of Theorem 1, the problem of generating a UCW, aiming to proceed Safralessly, is open.

B Red and Blue Flags

Consider a language $L \subseteq (2^{I \cup O})^\omega$. We say that a strategy $f : (2^I)^* \to 2^O$ *red realizes* L if for every $x \in (2^I)^*$, if x is not red for L, then $x \otimes f(x)$ is not red for L. Then, f *blue realizes* L if for every $x \in (2^I)^*$, if x is blue for L, then $x \otimes f(x)$ is blue for L.

Theorem 13. GE *realization implies red realization and may conflict with imply blue realization. Neither red nor blue realization imply* GE *realization.*

Proof. We first prove that every $f : (2^I)^+ \to 2^O$ that GE realizes ψ also red realizes ψ. Consider $x \in (2^I)^+$ that is not red for ψ. By definition, there is $y \in (2^O)^+$ such that $L^{x \otimes y} \neq \emptyset$. Then, there is $x' \in (2^I)^\omega$ and $y' \in (2^O)^\omega$ such that $x' \otimes y'$ in $L^{x \otimes y}$. Hence, $x \cdot x'$ is hopeful for ψ. Therefore, as f GE realizes ψ, we have that $(x \cdot x') \otimes f(x \cdot x') \models \psi$. Thus, $x \otimes f(x)$ is not red, and so f red realizes ψ.

We continue to an example where GE realization conflicts with blue realization. Let $I = \{p\}$ and $O = \{q\}$, and consider the specification $\psi = (q \wedge \mathsf{X}(p \vee \psi_1)) \vee (\neg q \wedge \mathsf{X}\psi_2)$, where ψ_1 is not GE realizable and ψ_2 is GE realizable. The input sequence $\emptyset \cdot \{p\}$ is blue, as $\emptyset \cdot \{p\} \otimes \{q\} \cdot \emptyset = \{q\} \cdot \{p\}$ is blue for ψ. The specification ψ is GE realizable: a strategy $f : (2^I)^+ \to 2^O$ that GE realizes ψ has $f(i_0) = \emptyset$, and then proceed to GE realize ψ_2. Yet, every strategy f that GE realizes ψ does not blue realizes it. Indeed, the strategy cannot have $f(i_0) = \{q\}$, as then an environment that responds (in the second input) with $\emptyset$ forces f to GE realize ψ_1. Hence, $f(i_0)$ must be $\emptyset$, but

$$\emptyset \cdot \{p\} \otimes \emptyset \cdot f(\emptyset \cdot \{p\}) = \emptyset \cdot (\{p\} \cup f(\emptyset \cdot \{p\})),$$

which, as a word in $(2^{I \cup O})^*$, is not blue for ψ.

Now, for the second claim, we use the same specification we used in the proof of Theorem 3. Let $I = \{p\}$ and $O = \{q\}$, and consider the specification $\psi = \mathsf{GF}((\mathsf{X}p) \wedge q) \wedge \mathsf{GF}((\mathsf{X}\neg p) \wedge \neg q)$. While ψ is not GE realizable, no prefix is red or blue for it, and thus all strategies red and blue realize it. $\square$

C Examples of v-Good Strategies

We analyze in detail a more generalized version of Example 3.

Let $I = \{p\}$ and $O = \{q\}$. Consider the LTL[$\mathcal{F}$] formula $\psi = (\triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q)$. Checking for which values v a strategy f is v-good for x with respect to ψ, we examine whether $[\![x \otimes f(x), \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] \geq v$ or $[\![x, \forall q. \neg(\triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q)]\!] > 1 - v$. Since ψ refers only to the first position in the computation, we can distinguish between four possible cases:

- $x_0 = \emptyset$ and $f(x_0) = \emptyset$. Then, $[\![x \otimes f(x), \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = 0$, $[\![x, \exists q. \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \max\{0, \lambda_2\} = \lambda_2$, and $[\![x, \forall q. \neg(\triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q)]\!] = \min\{1, 1 - \lambda_2\} = 1 - \lambda_2$. Hence, f is v-good for x with respect to ψ if $v = 0$ or $v > \lambda_2$, thus $v \in \{0\} \cup (\lambda_2, 1)$
- $x_0 = \emptyset$ and $f(x_0) = \{q\}$. Then, $[\![x \otimes f(x), \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \lambda_2$, $[\![x, \exists q. \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \lambda_2$, and $[\![x, \forall q. \neg(\triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q)]\!] = 1 - \lambda_2$. Hence, f is v-good for x with respect to ψ if $v \leq \lambda_2$ or $1 - v < 1 - \lambda_2$, thus for all $v \in [0, 1]$.
- $x_0 = \{p\}$ and $f(x_0) = \emptyset$. Then, $[\![x \otimes f(x), \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \lambda_1$, $[\![x, \exists q. \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \max\{\lambda_1, \lambda_2\}$, and $[\![x, \forall q. \neg(\triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q)]\!] = \min\{1 - \lambda_1, 1 - \lambda_2\}$. Hence, f is v-good for x with respect to ψ if $v \leq \lambda_1$ or $1 - v < \min\{1 - \lambda_1, 1 - \lambda_2\}$, thus $v \in [0, \lambda_1] \cup (\max\{\lambda_1, \lambda_2\}, 1]$. When $\lambda_1 < \lambda_2$, this means that $v \in [0, \lambda_1] \cup (\lambda_2, 1]$. When $\lambda_1 \geq \lambda_2$, this means that $v \in [0, 1]$.
- $x_0 = \{p\}$ and $f(x_0) = \{q\}$. Then, $[\![x \otimes f(x), \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \max\{\lambda_1, \lambda_2\}$, $[\![x, \exists q. \triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q]\!] = \max\{\lambda_1, \lambda_2\}$, and $[\![x, \forall q. \neg(\triangledown_{\lambda_1} p \vee \triangledown_{\lambda_2} q)]\!] = \min\{1 - \lambda_1, 1 - \lambda_2\}$. Hence, f is v-good for x with respect to ψ if $v \leq \max\{\lambda_1, \lambda_2\}$ or $1 - v < \min\{1 - \lambda_1, 1 - \lambda_2\}$, thus for all $v \in [0, 1]$.