Quantitative and Approximate Monitoring

Thomas A. Henzinger
IST Austria
tah@ist.ac.at

N. Ege Saraç
IST Austria
ege.sarac@ist.ac.at

***Abstract*—In runtime verification, a monitor watches a trace of a system and, if possible, decides after observing each finite prefix whether or not the unknown infinite trace satisfies a given specification. We generalize the theory of runtime verification to monitors that attempt to estimate numerical values of quantitative trace properties (instead of attempting to conclude boolean values of trace specifications), such as maximal or average response time along a trace. Quantitative monitors are approximate: with every finite prefix, they can improve their estimate of the infinite trace's unknown property value. Consequently, quantitative monitors can be compared with regard to a precision-cost trade-off: better approximations of the property value require more monitor resources, such as states (in the case of finite-state monitors) or registers, and additional resources yield better approximations. We introduce a formal framework for quantitative and approximate monitoring, show how it conservatively generalizes the classical boolean setting for monitoring, and give several precision-cost trade-offs for monitors. For example, we prove that there are quantitative properties for which every additional register improves monitoring precision.**

I. Introduction

We provide a theoretical framework for the convergence of two recent trends in computer-aided verification. The first trend is *runtime verification* [1]. Classical verification aspires to provide a judgment about all possible runs of a system; runtime verification, or *monitoring*, provides a judgment about a single, given run. There is a trend towards monitoring because the classical "verification gap" keeps widening: while verification capabilities are increasing, system complexity is increasing more quickly, especially in the time of many-core processors, cloud computing, cyber-physical systems, and neural networks. Theoretically speaking, the paradigmatic classical verification problem is *emptiness* of the product between system and negated specification ("does some run of the given system violate the given specification?"), whereas the central runtime verification problem is *membership* ("does a given run satisfy a given specification?"). Since membership is easier to solve than emptiness, this has ramifications for specification formalisms; in particular, there is no need to restrict ourselves to ω-regular specifications or finite-state monitors. We do restrict ourselves to the *online* setting, where a monitor watches the finite prefixes of an infinite run and, with each prefix, renders a *verdict*, which could signal a satisfaction or violation of the specification, or "don't know yet."

The second trend is *quantitative verification* [2], [3]. While classical verification is boolean, in that every complete run either satisfies or violates the specification and, accordingly, the system is either correct (i.e., without a violating run) or incorrect, quantitative verification provides additional, often numerical information about runs and systems. For example, a quantitative specification may measure the probability of an event, the "response time" or the use of some other resource along a run, or by how much a run deviates from a correct run. In *quantitative runtime verification*, we wish to observe, for instance, the maximal or average response time along a given run, not across all possible runs. Quantitative verification is interesting for an important reason beyond its ability to provide non-boolean information: it may provide *approximate* results [4]. A monitor that under- or over-approximates a quantitative property may be able to do so with fewer computational resources than a monitor that computes a quantitative property's exact value. We provide a theoretical framework for *quantitative and approximate monitoring*, which allows us to formulate and prove such statements.

In boolean runtime verification frameworks, there are several different notions of *monitorability* [5], [6], [7]. Along with safety and co-safety, a well-studied definition is by [8] and [9]: after watching any finite prefix of a run, if a positive or negative verdict has not been reached already, there exists at least one continuation of the run which will allow such a verdict. This *existential* definition is popular because a *universal* definition, that on every run a positive or negative verdict will be reached eventually, is very restrictive; only boolean properties that are both safe and co-safe can be monitored universally [7]. By contrast, the existential definition covers finite boolean combinations of safety and co-safety, and more [6]. In *quantitative approximate* monitoring, however, there is less need to prefer an existential definition of monitoring because usually many approximations are available, even if some are poor. The main attention must shift, rather, to the quality—i.e., precision—of the approximation. Our quantitative framework fully generalizes the standard boolean versions of monitorability in a universal setting where monitors yield approximate results on all runs and can be compared regarding their precision and resource use. In fact, we advocate the consideration of precision-resource trade-offs as a central design criterion for monitors, which requires a formalization of monitoring in which precision-resource trade-offs can be analyzed. *Such a formalization is the main contribution of this paper.*

As an example, let us illustrate a precision-resource trade-off that occurs when using *register machines* as monitors. Consider a server that processes requests. Each trace of the server is an infinite word over the alphabet $\{req, ack, other\}$ of events. An interesting quantitative property of the server is *maximal response time*, which measures the maximal number of events before each *req* event in a trace is followed by an *ack* event. This property, denoted p_1, is a function that maps every infinite word to a value in $\mathbb{N} \cup \{\infty\}$. To construct a precise online monitor for p_1, we need two counter registers x and y and the ability to compare their values: as long as $x < y$, register x counts the current response time, and y stores the maximal response time encountered so far; if $x = y$, counting continues in y, and x is reset to 0. The output, or *verdict* value, of the monitor is always y. In this way the 2-counter monitor M_{max} generates the verdict function depicted in Figure 1.

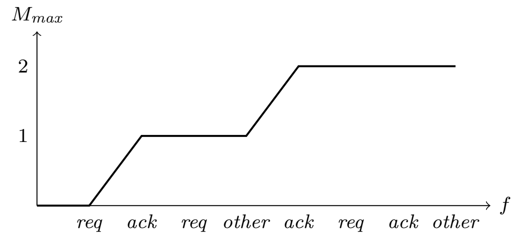

Fig. 1. Monitoring maximal response time for a trace f.

Considering the same server, one may also be interested in the *average response time* of a trace. The precise monitoring of average response time requires 3 counters and division between counter registers to generate outputs. Moreover, verdict values can fluctuate along a trace, producing a non-monotonic verdict function. Figure 2 shows the verdict function generated by a 3-register monitor M_{avg} with division.

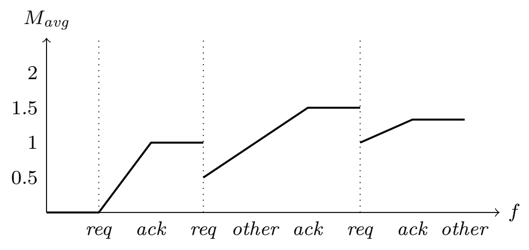

Fig. 2. Monitoring average response time for a trace f.

Now, let us consider an alphabet $\{req_1, ack_1, req_2, ack_2, other\}$ with two types of matching (req_i, ack_i) pairs. The quantitative property p_2 measures the maximal response times for both pairs: it maps every trace to an ordered pair of values from $\mathbb{N} \cup \{\infty\}$. A construction similar to the one for p_1 gives us a precise monitor that uses 4 counters. Indeed, we will show that 3 counters do not suffice to monitor p_2 precisely. However, the quantitative property p_2 can be approximately monitored with 3 counters: two counters can be used to store the maxima so far, and the third counter may track the current response time prioritizing the pair (req_1, ack_1) whenever both request types are active. This 3-counter monitor will always under-approximate the maximal response time for the (req_2, ack_2) pair. In case the resources are even scarcer, a 2-counter monitor can keep the same value as an under-approximation for both maximal response times in one counter, and use the second counter to wait sequentially for witnessing (req, ack) pairs of both types before incrementing the first counter. Just like the number of registers leads to precision-resource trade-offs for register monitors, the number of states leads to precision-resource trade-offs for finite-state monitors. For instance, a fixed number of states can encode counter values up to a certain magnitude, but can under-approximate larger values. We provide a general formal framework for quantitative and approximate monitoring which allows us to study such trade-offs for different models of monitors.

In Section II, we define quantitative properties, approximate verdict functions, and how the precision of monitors can be compared. In Section III, we give a variety of different examples and closure operations for quantitative monitoring. We also characterize the power of the important class of *monotonic* monitors by showing that, in our framework, the quantitative properties that can be monitored universally (on all traces) and precisely by monotonically increasing verdict functions are exactly the co-continuous properties on the value domain. In Section IV, we embed several variations of the boolean value domain within our quantitative framework. This allows us to characterize, within the safety-progress hierarchy [10], which boolean properties can be monitored universally and existentially; see Tables I and II. The section also connects our quantitative definitions of monitorability to the boolean definitions of [6], [7], [8], [9] and shows that our quantitative framework generalizes their popular boolean settings conservatively. Finally, in Section V, we present precision-resource trade-offs for register monitors. For this purpose, we generalize the quantitative setting of [11], [12] to approximate monitoring within our framework. In particular, we show a family of quantitative properties for which every additional counter register improves the monitoring precision.

Related work. In the boolean setting, the first definition of *monitorability* [5] focused on detecting violations of a property. This definition was generalized by [8] and [9] to capture satisfactions as well. Later, instead of using a fixed, three-valued domain for monitoring, Falcone et al. [6] proposed a definition with parameterized truth domains. According to their definition, every linear-time property is monitorable in a four-valued domain where the usual "inconclusive" verdict is split into "currently true" and "currently false" verdicts. Frameworks that capture existential as well as universal modalities for monitorability were studied in a branching-time setting [13], [14].

The prevalence of LTL and ω-regular specifications in formal verification is also reflected in runtime verification [9], [15], [16]. Recently, several more expressive models have been proposed, such as register monitors [11], monitors for visibly pushdown languages [17], quantified event automata [18], and many others for monitoring data events over an infinite alphabet of observations, as surveyed in [19]. One step towards quantitative properties is the augmenting of boolean specifications with quantities, e.g., discounting and averaging modalities [20], [21], timed specifications [22], [23], or specifications that include continuous signals, particularly in the context of cyber-physical systems [24], [25]. Another prominent line of work that provides a framework for runtime verification beyond finite-state is that of Alur et al. [26], [27], [28]. Their work focuses on runtime decidability issues for boolean specifications over streams of data events, but they do not consider approximate monitoring at varying degrees of precision. Quantitative frameworks for comparing traces and implementations for the same boolean specification were studied in [29], [30]. Our approach is fundamentally different as we consider quantitative property values.

Quantitative properties, a.k.a. quantitative languages, were defined in [31]. Although such properties have been studied much in the context of probabilistic model checking [2], decision problems in verification [31], and games with quantitative objectives [32], [33], in runtime verification, we observe a gap. While some formalisms for monitoring certain quantitative properties have been proposed [12], [34], [35], to the best of our knowledge, our work is the first general semantic framework that explores what it means to monitor and approximate generic quantitative properties of traces. We believe that such a framework is needed for the systematic study of precision-resource trade-offs in runtime verification. See [36] for a discussion of why quantitative verification at runtime is needed for self-adapting systems, and [37], [38] for monitoring neural networks.

II. Definitions

Let $\Sigma = \{a, b, \ldots\}$ be a finite alphabet of observations. A *trace* is a finite or infinite sequence of observations, denoted by $s, r, t \in \Sigma^*$ or $f, g, h \in \Sigma^\omega$, respectively. For traces $w \in \Sigma^* \cup \Sigma^\omega$ and $s \in \Sigma^*$, we write $s \prec w$ (resp. $s \preceq w$) iff s is a strict (resp. non-strict) finite prefix of w, and denote by $\mathit{pref}(w)$ the set of finite prefixes of w.

A. Quantitative properties and verdict functions

A *boolean property* $P \subseteq \Sigma^\omega$ is a set of infinite traces, and a *value domain* $\mathbb{D}$ is a partially ordered set. Unless otherwise stated, we assume that $\mathbb{D}$ is a complete lattice and, whenever appropriate, we write 0, $-\infty$, and ∞ instead of $\bot$ and $\top$ for the least and greatest elements. A *quantitative property* $p : \Sigma^\omega \to \mathbb{D}$ is a function on infinite traces. A *verdict* $v : \Sigma^* \to \mathbb{D}$ is a function on finite traces such that for all infinite traces $f \in \Sigma^\omega$, the set $\{v(s) : s \in \mathit{pref}(f)\}$ of verdict values over all prefixes of f has a supremum (least upper bound) and an infimum (greatest lower bound). If $\mathbb{D}$ is a complete lattice, then these limits always exist. For an infinite trace $f \in \Sigma^\omega$, we write $v(f) = (v(s_i))_{i \in \mathbb{N}}$ for the infinite *verdict sequence* over the prefixes $s_i \prec f$ of increasing length i. We use the $\limsup$ or $\liminf$ of a verdict sequence $v(f)$ to represent the "estimate" that the verdict function v provides for a quantitative property value $p(f)$ on the infinite trace f.

Definition 1. *Let p be a quantitative property and $f \in \Sigma^\omega$ an infinite trace. A verdict function v* approximates p on f from below *(resp.* above*) iff $\limsup v(f) \leq p(f)$ (resp. $p(f) \leq \liminf v(f)$). Moreover, v* monitors p on f from below *(resp.* above*) iff the equality holds.*

B. Universal, existential, and approximate monitorability

We define three modalities of quantitative monitorability.

Definition 2. *A quantitative property p is* universally monitorable from below *(resp.* above*) iff there exists a verdict function v such that for every $f \in \Sigma^\omega$ we have that v monitors p on f from below (resp. above).*

Definition 3. *A quantitative property p is* existentially monitorable from below *(resp.* above*) iff there exists a verdict function v such that (i) for every $f \in \Sigma^\omega$ we have that v approximates p on f from below (resp. above), and (ii) for every $s \in \Sigma^*$ there exists $f \in \Sigma^\omega$ such that v monitors p on sf from below (resp. above).*

Definition 4. *A quantitative property p is* approximately monitorable from below *(resp.* above*) iff there exists a verdict function v such that for every $f \in \Sigma^\omega$ we have that v approximates p on f from below (resp. above).*

Observe that every property is trivially approximately monitorable from below or above. We demonstrate the definitions in the example below.

Example 1. *Let $\Sigma = \{\mathit{req}_1, \mathit{ack}_1, \mathit{req}_2, \mathit{ack}_2, \mathit{other}\}$ and $\mathbb{D}$ be the nonnegative integers with ∞. Consider the maximal response-time properties p_1 and p_2 over $(\mathit{req}_1, \mathit{ack}_1)$ and $(\mathit{req}_2, \mathit{ack}_2)$ pairs, respectively. For every $f \in \Sigma^\omega$, let $p(f) = \max(p_1(f), p_2(f))$. Consider the verdict v_1 that counts both response times and outputs the maximum of the two, the verdict v_2 that counts and computes the maximum only for the $(\mathit{req}_1, \mathit{ack}_1)$ pair, and the constant verdict v_3 that always outputs 0. Evidently, v_1 universally monitors p from below, and v_3 approximately monitors p from below. Moreover, v_2 existentially monitors p from below because the true maximum can only be greater, and we can extend every finite trace $s \in \Sigma^*$ with $f = \mathit{req}_1 \cdot \mathit{other}^\omega$ such that* $\limsup v_2(sf) = p(sf) = \infty$.

C. Monotonic verdict functions

Of particular interest are *monotonic* verdict functions, because the "estimates" they provide for a quantitative property value are always conservative (below or above) and can improve in quality over time. On the other hand, some properties, such as average response time, inherently require non-monotonic verdict functions for universal monitoring.

Definition 5. *A verdict function v is* monotonically increasing *(resp.* decreasing*) iff for every $s, t \in \Sigma^*$ we have $s \prec t$ implies $v(s) \leq v(t)$ (resp. $v(s) \geq v(t)$). Moreover, v is* monotonic *iff it is either monotonically increasing or monotonically decreasing. If v is monotonic or non-monotonic, then it is* unrestricted*.*

If the value domain $\mathbb{D}$ has a least and a greatest element, every monotonic verdict v that universally monitors a property p from below also universally monitors p from above. Therefore, in such cases, we say that v *universally monitors p*. In Example 1 above, the verdict v_1 is monotonically increasing and thus universally monitors p. Let v_4 be such that $v_4(s) = \infty$ if s contains a request that is not acknowledged, and $v_4(s) = v_1(s)$ otherwise. The verdict v_4 is not monotonic, but it universally monitors p from above.

D. Comparison of verdict functions

Quantitative monitoring provides a natural notion of precision for verdict functions.

Definition 6. *Let p be a quantitative property that is (universally, existentially, or approximately) monitorable from below (resp. above) by the verdict functions v_1 and v_2. The verdict v_1 is* more precise *than the verdict v_2 iff for every $f \in \Sigma^\omega$ we have $\limsup v_2(f) \leq \limsup v_1(f)$ (resp. $\liminf v_1(f) \leq \liminf v_2(f)$) and there exists $g \in \Sigma^\omega$ such that $\limsup v_2(g) < \limsup v_1(g)$ (resp. $\liminf v_1(g) < \liminf v_2(g)$). Moreover, v_1 and v_2 are* equally precise *iff for every $f \in \Sigma^\omega$ we have $\limsup v_2(f) = \limsup v_1(f)$ (resp. $\liminf v_1(f) = \liminf v_2(f)$).*

Note that for a quantitative property p, if the verdict functions v_1 and v_2 universally monitor p both from below or from above, then v_1 and v_2 are equally precise. Two monotonically increasing or monotonically decreasing verdict functions can be compared not only according to their precision but also according to their *speed*, that is, how quickly they approach the property value. This will be important if monitors have limited resources and their outputs are delayed, i.e., they affect not the current but a future verdict value.

III. Monitorable Quantitative Properties

A. Examples

We provide several examples of quantitative properties and investigate their monitorability.

Example 2 (Maximal response time)**.** *Let $\Sigma = \{req, ack, other\}$ and $\mathbb{D} = \mathbb{N} \cup \{\infty\}$. Let $\mathsf{mrt} : \Sigma^* \to \mathbb{D}$ be such that $\mathsf{mrt}(s) = \infty$ if, in s, a req is followed by another req without an ack in between, it equals the maximal number m_s of observations between matching (req, ack) pairs if there is no pending request in s, and otherwise it equals $\max(m_s, n)$ where n is the current response time. For every $f \in \Sigma^\omega$, let us denote by $\mathsf{mrt}(f)$ the infinite sequence $(\mathsf{mrt}(s_i))_{i \in \mathbb{N}}$ over the prefixes $s_i \prec f$ of increasing length i. Consider the property $p(f) = \lim \mathsf{mrt}(f)$ that specifies the maximal response time of a server that can process at most one request at a time. To monitor p, we use mrt as the verdict, i.e., we let $v(s) = \mathsf{mrt}(s)$ for every $s \in \Sigma^*$. Observe that mrt is monotonically increasing, and the construction yields $\lim v(f) = p(f)$ for every $f \in \Sigma^\omega$. Therefore, the verdict v universally monitors p.*

The maximal response-time property of Example 2 is evidently infinite-state because it requires counting up to an arbitrarily large integer. However, there are finite-state approximations that improve in precision with every additional state. We say that a finite-state machine *generates* a verdict function iff, on every finite trace, the machine's output equals the verdict value, where an output is a mapping from the set of states to the value domain.

Example 3 (Approximate monitoring of maximal response time)**.** *Consider the maximal response-time property p from Example 2. Let M_k be a finite-state machine with k states, and let v_k be the verdict generated by M_k. For every $k \in \mathbb{N}$, the best the verdict v_k can do is to approximately monitor p from below, because it can only count up to some integer $m \leq k$. Suppose that we are given $k+1$ states. We can use the additional state to construct a machine M_{k+1} from M_k to generate a more precise verdict v_{k+1} as follows. We add the appropriate transitions from the states that have the output value of m to the new state, which is assigned the output $m+1$. With the additional transitions, the machine M_{k+1} can continue counting for one more step after reading a trace in which the current maximum is m. Therefore, v_{k+1} is more precise than v_k.*

Next, we define the average response-time property and present two verdict functions that illustrate another kind of precision-resource trade-off for monitors.

Example 4 (Average response time)**.** *Let $\Sigma = \{req, ack, other\}$ and $\mathbb{D} = \mathbb{R} \cup \{\infty\}$. Let $\mathsf{art} : \Sigma^* \to \mathbb{D}$ be such that $\mathsf{art}(s) = \infty$ if s contains a req followed by another req without an ack in between, it equals the average number of observations between matching (req, ack) pairs if there is no pending req in s, and otherwise it equals $\frac{n \cdot x_n + m}{n+1}$, where n is the number of acknowledged requests, x_n is the average response time for the first n requests, and m is the number of observations since the last req. For every $f \in \Sigma^\omega$, let $\mathsf{art}(f) = (\mathsf{art}(s_i))_{i \in \mathbb{N}}$ over the prefixes $s_i \prec f$ of increasing length i. Now, define $\mathsf{lim\,avg}(f) = \liminf \mathsf{art}(f)$ for every $f \in \Sigma^\omega$ [31], and let p be the quantitative property such that*

$p(f) = \mathsf{lim\,avg}(f)$. *In other words, p specifies the average response time of a server that can process at most one request at a time. To monitor p, we can use the function* art *as a verdict, i.e., let v be such that $v(s) = \mathsf{art}(s)$ for all $s \in \Sigma^*$. Intuitively, the moving average approaches to the property value as v observes longer prefixes. Therefore, by construction, for every $f \in \Sigma^\omega$, we have $\liminf v(f) = p(f)$, which means that p is universally monitorable from above by an unrestricted verdict function.*

Alternatively, we can use the monotonic verdict function v' that universally monitors the maximal response-time property in Example 2. Observe that v' existentially monitors p from above because (i) the maximal response time of a trace is greater than its average response time, and (ii) for every finite prefix s there is an extension f that contains a request that is not acknowledged, which yields $\lim v'(sf) = p(sf) = \infty$.

Boolean safety and co-safety properties can be embedded in a quantitative setting by considering their *discounted* versions [20]. We show that discounted safety and co-safety properties are universally monitorable.

Example 5 (Discounted safety and co-safety)**.** *Let p be a discounted safety property, that is, $p(f) = 1$ if f does not violate the given safety property, and $p(f) = 1 - \frac{1}{2^n}$ if the shortest violating prefix of f has length n. Similarly, let q be a discounted co-safety property: $q(f) = 0$ if f does not satisfy the given co-safety property, and $q(f) = \frac{1}{2^n}$ if the shortest satisfying prefix of f has length n. To monitor these two properties, we use verdict functions v_p and v_q that work similarly as p and q on finite traces, that is, $v_p(s) = 1$ if s is not violating for the given safety property, and $v_p(s) = 1 - \frac{1}{2^n}$ if the shortest violating prefix of s has length n; and similarly for v_q. One can easily verify that p and q are universally monitorable by v_p and v_q, respectively.*

Finally, we look at another classical value function for quantitative properties, often called *energy* values [31].

Example 6 (Energy)**.** *Let $A = (Q, \Sigma, \delta, q_0, w)$ be a deterministic finite automaton with weighted transitions, where Q is a set of states, Σ is an alphabet, $\delta \subseteq Q \times \Sigma \times Q$ is a set of transitions, q_0 is the initial state, and $w : \delta \to \mathbb{Z}$ is a weight function. Let $s = \sigma_1 \ldots \sigma_n$ be a finite trace of length n, and let $q_0 \ldots q_n$ be the corresponding run of A. We define $A(s) = \sum_{i=1}^{n} w(q_{i-1}, \sigma_i, q_i)$, where $A(\varepsilon) = 0$. Consider the value domain $\mathbb{D} = \mathbb{Z} \cup \{\infty\}$. Let p be a property such that, for every $f \in \Sigma^\omega$, we have $p(f) = k$ where k is the smallest nonnegative value that satisfies $A(s) + k \geq 0$ for every finite prefix $s \prec f$. To monitor p, we construct the following verdict function: given $s \in \Sigma^*$, let $v(s) = -\min\{A(r) \mid r \in \mathit{pref}(s)\}$. Note that v is monotonically increasing. On an infinite trace $f \in \Sigma^\omega$, if $v(f)$ approaches ∞, then f yields a negative-weight loop on A, therefore $p(f) = \infty$. Otherwise, if $v(f)$ converges to a finite value, then it is equal to $p(f)$ by construction, which means that v universally monitors p.*

B. Closure under operations on the value domain

Let $\mathbb{D}$ be a value domain and $p : \Sigma^\omega \to \mathbb{D}$ be a quantitative property. We define the *inverse of* $\mathbb{D}$, denoted $\mathbb{D}_{inv}$, as the value domain that contains the same elements as $\mathbb{D}$ with reversed ordering. Moreover, we define the *complement of* p as $\bar{p} : \Sigma^\omega \to \mathbb{D}_{inv}$ such that $\bar{p}(f) = p(f)$.

Proposition 1. *A quantitative property p is universally (resp. existentially; approximately) monitorable from below iff $\bar{p}$ is universally (resp. existentially; approximately) monitorable from above.*

If the value domain $\mathbb{D}$ is a lattice, then monitorability from below is preserved by the least upper bound (written max) and from above by greatest lower bound (written min). For all quantitative properties p and q on $\mathbb{D}$, and all infinite traces $f \in \Sigma^\omega$, let $\max(p, q)(f) = \max(p(f), q(f))$ and $\min(p, q)(f) = \min(p(f), q(f))$.

Proposition 2. *For all quantitative properties p and q on a lattice, if p and q are universally (resp. existentially; approximately) monitorable from below (resp. above), then the property $\max(p, q)$ (resp. $\min(p, q)$) is also universally (resp. existentially; approximately) monitorable from below (resp. above).*

Proof. Let v_p and v_q be two verdict functions that universally monitor p and q from below. Then, we have $\max(p(f), q(f)) = \max(\limsup v_p(f), \limsup v_q(f))$ for every $f \in \Sigma^\omega$. Since we assume that the domain contains a greatest element, for every $f \in \Sigma^\omega$, we also have $\max(\limsup v_p(f), \limsup v_q(f))$ equals $\limsup(\max(v_p(f), v_q(f)))$. Therefore, we can use $\max(v_p, v_q)$ as a verdict function to universally monitor $\max(p, q)$ from below the same way v_p and v_q monitor p and q. The case for min is symmetric, and the cases for existential and approximate monitoring can be proved similarly by using the fact that the domain is a lattice. ☐

Proposition 3. *For all quantitative properties p and q on a lattice, if p and q are (universally, existentially, or approximately) monitorable from below (resp. above), the property $\min(p, q)$ (resp. $\max(p, q)$) is approximately monitorable from below (resp. above).*

Proof. Let v_p and v_q be verdict functions that monitor p and q from below, therefore for every infinite trace $f \in \Sigma^\omega$ we have $\limsup v_p(f) \leq p(f)$ and $\limsup v_q(f) \leq q(f)$. Because $\limsup \min(v_p(f), v_q(f)) \leq \min(\limsup v_p(f), \limsup v_q(f))$ for every $f \in \Sigma^\omega$, we can use $\min(v_p, v_q)$ as a verdict function to approximately monitor $\min(p, q)$ from below. The case for max is dual. ☐

If $\mathbb{D}$ is a numerical value domain with addition and multiplication, such as the reals or integers, or their nonnegative subsets, then not all modalities of monitorability are preserved under these operations. For all quantitative properties p and q on $\mathbb{D}$, and all infinite traces $f \in \Sigma^\omega$, let $(p + q)(f) = p(f) + q(f)$ and $(p \cdot q)(f) = p(f) \cdot q(f)$.

Since lim sup is subadditive and submultiplicative while lim inf superadditive and supermultiplicative, one can easily conclude the following.

Proposition 4. *For all quantitative properties p and q on a numerical value domain, if p and q are (universally, existentially, or approximately) monitorable from below (resp. above), then $p + q$ and $p \cdot q$ are approximately monitorable from below (resp. above).*

However, monitorability is preserved under any monotonically increasing continuous function on value domains that are totally ordered.

Proposition 5. *Let $\mathbb{D}$ be a totally-ordered value domain. Consider a quantitative property $p : \Sigma^\omega \to \mathbb{D}$ and a monotonically increasing continuous function $\phi : \mathbb{D} \to \mathbb{D}$. If p is (universally, existentially, or approximately) monitorable from below (resp. above), then so is $\phi(p)$.*

C. Continuous quantitative properties

For this section, we assume that $\mathbb{D}$ is a complete lattice and define *continuous* and *co-continuous* properties on $\mathbb{D}$. Let p be a quantitative property and, for every $s \in \Sigma^*$, let $\nu_p(s) = \sup\{p(sf) \mid f \in \Sigma^\omega\}$. For $f \in \Sigma^\omega$, the function ν_p generates an infinite sequence $\nu_p(f) = (\nu_p(s_i))_{i\in\mathbb{N}}$ over the prefixes $s_i \prec f$ of increasing length i. Similarly, let $\mu_p(s) = \inf\{p(sf) \mid f \in \Sigma^\omega\}$ and extend it to generate infinite sequences on infinite traces.

Definition 7 ([39]). *A property p is* continuous *iff for every infinite trace $f \in \Sigma^\omega$, we have $p(f) = \lim \nu_p(f)$. Moreover, p is* co-continuous *iff $\bar{p}$ continuous, or equivalently, iff $p(f) = \lim \mu_p(f)$ for every $f \in \Sigma^\omega$.*

Intuitively, the continuous and co-continuous properties constitute well-behaved sets of properties in the sense that, to monitor them, there is no need for speculation. For example, considering a continuous property, the least upper bound can only decrease after reading longer prefixes; therefore, a verdict function monitoring such a property can simultaneously be conservative and precise. We make this connection more explicit and show that continuous and co-continuous properties satisfy the desirable property of being universally monitorable by monotonic verdict functions.

Theorem 1. *A quantitative property p is continuous iff $\bar{p}$ is universally monitorable by a monotonically increasing verdict function.*

Proof. Observe that $\bar{p}$ is universally monitorable by a monotonically increasing verdict function iff p is universally monitorable by a monotonically decreasing verdict function. For the *only if* direction, suppose p is continuous, i.e., $\lim \nu_p(f) = p(f)$ for every $f \in \Sigma^\omega$. Since ν_p is monotonically decreasing and it converges to the property value for every infinite trace, we can use it as the verdict function to universally monitor p.

Now, let v be a monotonically decreasing verdict function such that $\lim v(f) = p(f)$ for all $f \in \Sigma^\omega$. We claim that $v(s) \geq \nu_p(s)$ for all $s \in \Sigma^*$. Suppose towards contradiction that $v(s) < \nu_p(s)$ for some $s \in \Sigma^*$. Since we have either (i) $\nu_p(s) = p(sg)$ for some $g \in \Sigma^\omega$, or (ii) for every $g \in \Sigma^\omega$ there exists $h \in \Sigma^\omega$ such that $p(sg) < p(sh)$, we obtain $v(s) < p(sf)$ for some $f \in \Sigma^\omega$. It contradicts the assumption that v is a monotonically decreasing verdict which universally monitors p from below, therefore our claim is correct. Now, observe that $v(s) \geq \nu_p(s)$ for all $s \in \Sigma^*$ implies $\lim v(f) \geq \lim \nu_p(f)$ for all $f \in \Sigma^\omega$. Since v universally monitors p, we get $p(f) \geq \lim \nu_p(f)$ for all $f \in \Sigma^\omega$. By the definition of ν_p, we also know that for every property p and infinite trace $f \in \Sigma^\omega$, we have $\lim \nu_p(f) \geq p(f)$. Therefore, we conclude that $\lim \nu_p(f) = p(f)$ for all $f \in \Sigma^\omega$, i.e., p is continuous. □

Combining Theorem 1 and Definition 7, we immediately get the following characterization for the co-continuous properties.

Corollary 1. *A quantitative property p is co-continuous iff p is universally monitorable by a monotonically increasing verdict function.*

Let $\mathbb{D}$ be a numerical domain and recall the maximal response-time property from Example 2. As we discussed previously, it is universally monitorable by a monotonically increasing verdict function, and therefore co-continuous. By the same token, one can define the *minimal response-time* property, which is continuous. However, average response time, which requires a non-monotonic verdict function although it is universally monitorable from above, is neither continuous nor co-continuous. We also remark that discounted safety and co-safety properties [20] are continuous and co-continuous, respectively. In Section IV, we will discuss how these notions relate to safety and co-safety in the boolean setting.

IV. Monitoring Boolean Properties

A. Boolean monitorability as quantitative monitorability

Quantitative properties generalize boolean properties. For every boolean property $P \subseteq \Sigma^\omega$, the characteristic function $\tau_P : \Sigma^\omega \to \{\mathtt{F}, \mathtt{T}\}$ is a quantitative property, where $\tau_P(f) = \mathtt{T}$ if $f \in P$, and $\tau_P(f) = \mathtt{F}$ if $f \notin P$. Using this correspondence, we can embed the main boolean notions of monitorability within our quantitative framework. For this, we consider four different boolean value domains:

- $\mathbb{B} = \{\mathtt{F}, \mathtt{T}\}$ such that $\mathtt{F}$ and $\mathtt{T}$ are incomparable.
- $\mathbb{B}_\perp = \mathbb{B} \cup \{\perp\}$ such that $\perp < \mathtt{F}$ and $\perp < \mathtt{T}$.
- $\mathbb{B}_t = \{\mathtt{F}, \mathtt{T}\}$ such that $\mathtt{F} < \mathtt{T}$.
- $\mathbb{B}_f = \{\mathtt{F}, \mathtt{T}\}$ such that $\mathtt{T} < \mathtt{F}$.

Most work in monitorability assumes irrevocable verdicts. On the domains $\mathbb{B}$ and $\mathbb{B}_\perp$, where $\mathtt{T}$ and $\mathtt{F}$ are incomparable, the irrevocability of verdicts corresponds to monotonically increasing verdict functions. For these, positive verdicts in $\mathbb{B}_t$ and negative verdicts in $\mathbb{B}_f$ are also irrevocable. The following observations about verdict functions on boolean domains are useful as well.

Remark 1. *Let v be a verdict function on $\mathbb{B}$ or $\mathbb{B}_\bot$. If v is monotonic, it cannot switch between* T *and* F*, as these values are incomparable. Therefore, v can monitor only $\emptyset$ and Σ^ω in $\mathbb{B}$. If v is unrestricted, it can switch between* T *and* F *only finitely often, because the* $\limsup$ *and* $\liminf$ *over every infinite trace must be defined.*

We begin with the classical definition of monitorability for boolean properties [8], [9]. Let $P \subseteq \Sigma^\omega$ be a boolean property. A finite trace $s \in \Sigma^*$ *positively* (resp. *negatively*) *determines* P iff for every $f \in \Sigma^\omega$, we have $sf \in P$ (resp. $sf \notin P$). The boolean property P is *classically monitorable* iff for every $s \in \Sigma^*$, there exists $r \in \Sigma^*$ such that sr positively or negatively determines P. This definition coincides with the *persistently informative monitorability* of [7]. It is also captured by our definition of existential monitorability by monotonic verdicts on $\mathbb{B}_\bot$.

Proposition 6. *A boolean property P is classically monitorable iff τ_P is existentially monitorable from below by a monotonically increasing verdict function on $\mathbb{B}_\bot$.*

According to [7], a boolean property P is *satisfaction* (resp. *violation*) *monitorable* iff there exists a monitor that reaches a positive (resp. negative) verdict for every $f \in P$ (resp. $f \notin P$). More generally, if monitorability is parameterized by a truth domain as in [6], then violation and satisfaction monitorability correspond to monitorability over $\{\bot, \texttt{F}\}$ and $\{\bot, \texttt{T}\}$, and capture exactly the classes of safety and co-safety properties, respectively. In our framework, violation (resp. satisfaction) monitorability is equivalent to universal monitorability by monotonically increasing verdicts on $\mathbb{B}_f$ (resp. $\mathbb{B}_t$), because they require reaching an irrevocable negative (resp. positive) verdict for traces that violate (reps. satisfy) the property.

Theorem 2 ([6])**.** *A boolean property P is safe (resp. co-safe) iff τ_P is universally monitorable by a monotonically increasing verdict function on $\mathbb{B}_f$ (resp. $\mathbb{B}_t$).*

A boolean property P is *partially monitorable* according to [7] iff it is satisfaction or violation monitorable. This corresponds to parametric monitorability over the 3-valued domain $\{\bot, \texttt{T}, \texttt{F}\}$, and is equivalent to the union of safety and co-safety [6]. Due to the duality of $\mathbb{B}_f$ and $\mathbb{B}_t$, in our framework, partial monitorability corresponds to universal monitorability by monotonic verdict functions on either of these domains.

Corollary 2. *A boolean property P is safe or co-safe iff τ_P is universally monitorable by a monotonic verdict function on $\mathbb{B}_t$ (equivalently, on $\mathbb{B}_f$).*

Also defined in [7] is the notion of *complete monitorability*, which requires both satisfaction and violation monitorability. It is equivalent to our universal monitorability by monotonic verdict functions on $\mathbb{B}_\bot$, meaning that for every trace $f \in P$ we reach a positive verdict, and for every $f \notin P$, a negative verdict.

Theorem 3 ([7])**.** *A boolean property P is both safe and co-safe iff τ_P is universally monitorable by a monotonically increasing verdict function on $\mathbb{B}_\bot$.*

Based on the idea of revocable verdicts, a 4-valued domain $\{\texttt{T}, \texttt{F}, \texttt{T}_c, \texttt{F}_c\}$ is also considered in [6], where T and F are still irrevocable but the inconclusive verdict $\bot$ is split into two verdicts $\texttt{T}_c$ ("currently true") and $\texttt{F}_c$ ("currently false") for more nuanced reasoning on finite traces. In the universe of ω-regular properties, their monitorability over this domain corresponds to the class of reactivity properties of the safety-progress hierarchy [10]. In our framework, unrestricted verdict functions provide a similar effect as revocable verdicts. We will show in Theorem 7 and Example 7 that unrestricted verdict functions on $\mathbb{B}_\bot$ can existentially monitor reactivity properties and more.

Finally, two weak forms of boolean monitorability defined in [7] are *sound monitorability* and *informative monitorability.* While sound monitorability corresponds to approximate monitorability in $\mathbb{B}_\bot$, informative monitorability corresponds to approximate monitorability in $\mathbb{B}_\bot$ by monotonic verdicts but excluding the constant verdict function $\bot$.

B. Monitoring the safety-progress hierarchy

We first show that some of the modalities of quantitative monitoring are equivalent over boolean domains. Proposition 7 also indicates some limitations of flat value domains.

Proposition 7. *Let P be a boolean property and τ_P be the corresponding quantitative property. The following statements are equivalent.*

(1) τ_P is existentially monitorable from below by an unrestricted verdict function on $\mathbb{B}$.
(2) τ_P is universally monitorable from below by an unrestricted verdict function on $\mathbb{B}$.
(3) τ_P is universally monitorable from below by an unrestricted verdict function on $\mathbb{B}_\bot$.

Proof. The key observation for the proofs is that T and F are incomparable in $\mathbb{B}$ and $\mathbb{B}_\bot$, as pointed out in Remark 1.

(1) $\iff$ (2): Let τ_P be existentially monitorable from below by a verdict function v on $\mathbb{B}$. Since T and F are incomparable, for every $f \in \Sigma^\omega$, if $\limsup v(f) \leq \tau_P(f)$ then $\limsup v(f) = \tau_P(f)$ in domain $\mathbb{B}$. Therefore, v also universally monitors p from below in $\mathbb{B}$. The other direction follows from Definitions 2 and 3.

(2) $\iff$ (3): The *only if* direction follows from the fact that $\mathbb{B}_\bot$ is an extension of $\mathbb{B}$ with a least element. For the *if* direction, suppose v is a verdict function that universally monitors τ_P from below in $\mathbb{B}_\bot$. We construct a verdict function u that imitates v in $\mathbb{B}$ as follows: let $u(s) = v(s)$ if $v(s) \neq \bot$, and $u(s) = v(r)$ otherwise, where r is the longest prefix of s such that $v(r) \neq \bot$ (if there is no such prefix, assume w.l.o.g. that $u(s) = \texttt{T}$). Now, let $f \in \Sigma^\omega$ be an infinite trace, and observe that whenever $v(f)$ converges, so does $u(f)$. If $v(f)$ does not converge, then the subsequential limits must be either (i) $\bot$ and T, or (ii) $\bot$ and F. Suppose

(i) is true. Then, there exists a prefix $s \prec f$ such that for all $r \in \Sigma^*$ satisfying $sr \prec f$ we have $v(sr) = \bot$ or $v(sr) = \mathtt{T}$. If $v(s) = \mathtt{T}$, then, by construction, u always outputs $\mathtt{T}$ starting from s. Otherwise, u outputs $\mathtt{F}$ until v outputs $\mathtt{T}$ (which is bound to happen by supposition), and converges to $\mathtt{T}$ afterwards. The case for (ii) is dual. Therefore, for every $f \in \Sigma^\omega$, we have $\limsup u(f) = \limsup v(f)$, which means that u universally monitors p from below. □

Proposition 8. *For every boolean property P, we have that τ_P is existentially monitorable from below by a monotonically increasing verdict function on $\mathbb{B}_t$ iff τ_P is existentially monitorable from below by a monotonically increasing verdict function on $\mathbb{B}_f$.*

Proof. Suppose τ_P is existentially monitorable from below by a monotonically increasing verdict function v on $\mathbb{B}_t$. Consider the following verdict function: $u(s) = \mathtt{F}$ if $v(s) = \mathtt{F}$ and $\tau_P(sf) = \mathtt{F}$ for all $f \in \Sigma^\omega$; and $u(s) = \mathtt{T}$ if $v(s) = \mathtt{T}$ or $\tau_P(sf) = \mathtt{T}$ for some $f \in \Sigma^\omega$. Notice that for every $s \in \Sigma^*$, if $v(s) = \mathtt{T}$ then $\tau_P(sf) = \mathtt{T}$ for all $f \in \Sigma^\omega$, and if $v(s) = \mathtt{F}$ then $\tau_P(sf) = \mathtt{F}$ for some $f \in \Sigma^\omega$, or $v(sr) = \mathtt{T}$ for some $r \in \Sigma^*$. Therefore, we can equivalently formulate u as follows: $u(s) = \mathtt{F}$ if $\tau_P(sf) = \mathtt{F}$ for all $f \in \Sigma^\omega$; and $u(s) = \mathtt{T}$ if $\tau_P(sf) = \mathtt{T}$ for some $f \in \Sigma^\omega$. The function u is indeed monotonically increasing. Further, $\limsup u(f) = \mathtt{F}$ implies that there is $s \prec f$ such that $\tau_P(sg) = \mathtt{F}$ for every $g \in \Sigma^\omega$, which means that for every $f \in \Sigma^\omega$ we have $\limsup u(f) \leq \tau_P(f)$.

Next, we show that for every $s \in \Sigma^*$ there exists $f \in \Sigma^\omega$ such that $\limsup u(sf) = \tau_P(sf)$. Suppose towards contradiction that for some $s \in \Sigma^*$ every $f \in \Sigma^\omega$ gives us $\limsup u(sf) < \tau_P(sf)$, i.e., $\limsup u(sf) = \mathtt{T}$ and $\tau_P(sf) = \mathtt{F}$. Since $\limsup u(sf) = \mathtt{T}$ and u is monotonically increasing, we get $u(r) = \mathtt{T}$ for every $r \prec sf$. It means that, by construction, for every $r \prec sf$ there exists $g \in \Sigma^\omega$ such that $\tau_P(rg) = \mathtt{T}$. However, we get a contradiction since $s \prec sf$ and $\tau_P(sf) = \mathtt{F}$ for every $f \in \Sigma^\omega$ by supposition. Therefore, we conclude that u existentially monitors τ_P from below in $\mathbb{B}_f$. The *if* direction can be proved symmetrically. □

Before we relate various modalities of monitoring and boolean value domains to the rest of the safety-progress classification of boolean properties [10], we discuss how boolean safety and co-safety are special cases of continuous and co-continuous properties from Section III. Consider the value domain $\mathbb{B}_t$, let P be a safety property and τ_P be the corresponding quantitative property. Observe that for every $s \in \Sigma^*$, we have $\nu_{\tau_P}(s) = \mathtt{F}$ if s negatively determines P, and $\nu_{\tau_P}(s) = \mathtt{T}$ otherwise. Since P is safe, we also have $\tau_P(f) = \lim \nu_{\tau_P}(f)$ for every $f \in \Sigma^\omega$, which means that τ_P is continuous. Moreover, the inverse $\overline{\tau_P}$ is a co-continuous property on $\mathbb{B}_f$, and it still corresponds to the same boolean safety property. Therefore, by Theorem 1, we get that property P is safe iff $\overline{\tau_P}$ is universally monitorable by a monotonically increasing verdict function on $\mathbb{B}_f$. Similarly, co-safety properties correspond to the co-continuous properties on $\mathbb{B}_t$; and thus, they are exactly the properties that are universally monitorable by monotonically increasing verdict functions on $\mathbb{B}_t$.

Positive, finite boolean combinations of safety and co-safety properties are called *obligation* properties [10]. Every obligation property P can be expressed in a canonical conjunctive normal form $\bigcap_{i=1}^{n}(S_i \cup C_i)$ for some positive integer n, where S_i is safe and C_i is co-safe for all $1 \leq i \leq n$. Moreover, an obligation property in conjunctive normal form with $n = k$ is a *k-obligation property.*

We prove that obligation properties are universally monitorable in $\mathbb{B}$, which naturally requires finitely many switches between verdicts $\mathtt{T}$ and $\mathtt{F}$. Moreover, we establish an equivalence between the infinite hierarchy of obligation properties and a hierarchy of verdict functions on $\mathbb{B}$.

Theorem 4. *A boolean property P is a k-obligation property iff τ_P is universally monitorable by a verdict function on $\mathbb{B}$ that changes its value at most $2k$ times.*

Proof. Suppose P is a k-obligation property, in other words, $P = \bigcap_{i=1}^{k}(S_i \cup C_i)$ for some integer $k \geq 1$ where S_i is safe and C_i is co-safe for each $1 \leq i \leq k$. Consider the following verdict function: $v(s) = \mathtt{T}$ if for every $1 \leq i \leq k$ we have s does not negatively determine S_i or s positively determines C_i; and $v(s) = \mathtt{F}$ if there exists $1 \leq i \leq k$ such that s negatively determines S_i and s does not positively determine C_i. Note that if a finite trace s positively or negatively determines a boolean property, then so does sr for every finite continuation r. If P cannot be expressed as a $(k-1)$-obligation property, then there exists a sequence of finite traces $s_1 \prec r_1 \prec \ldots \prec s_k \prec r_k$, w.l.o.g., such that for every $1 \leq i \leq k$ trace s_i negatively determines S_i, does not negatively determine any S_j for $j > i$, and does not positively determine any C_j for $j \geq i$; and trace r_i positively determines C_i, does not positively determine any C_j for $j > i$, and does not negatively determine any S_j for $j > i$. This is because otherwise some safety or co-safety properties either cannot be determined, which contradicts the fact that P is an obligation property, or they are determined by the same finite traces, which contradicts the fact that P is not a $(k-1)$-obligation property. Then, the worst case for v is when P is not $(k-1)$-obligation and it reads r_k above, which forces $2k$ switches. One can verify that v always converges to the correct property value, i.e., $\lim v(f) = \tau_P(f)$ for all $f \in \Sigma^\omega$. Therefore, verdict v universally monitors τ_P in $\mathbb{B}$.

For the other direction, suppose τ_P is universally monitorable by a verdict function on $\mathbb{B}$ that changes its value at most $2k$ times, and assume towards contradiction that P is not a k-obligation property. In particular, suppose P is an m-obligation property for some $m > k$, which cannot be expressed as a k-obligation, and let $P = \bigcap_{i=1}^{m}(S_i \cup C_i)$ where S_i is safe and C_i is co-safe for each $1 \leq i \leq m$. By the same argument used above, there exist finite traces $s_1 \prec r_1 \prec \ldots \prec s_m \prec r_m$, w.l.o.g., such that for every $1 \leq i \leq m$ trace s_i negatively determines S_i, does not negatively determine any S_j for $j > i$, and does not positively determine any C_j for $j \geq i$; and trace r_i positively determines C_i, does not positively determine any C_j for $j > i$, and does not negatively determine any S_j for $j > i$. Assume w.l.o.g. that $v(\varepsilon) = \mathtt{T}$. After reading each finite trace described above, v has to switch its output because otherwise we can construct a trace f such that $\lim v(f) \neq \tau_P(f)$. But since v can only change its value $2k$ times, it immediately yields that v cannot universally monitor τ_P where P is an m-obligation property for $m > k$. Therefore, P must be a k-obligation property. ☐

The countable intersection of co-safety properties and the countable union of safety properties, i.e., so-called *response* and *persistence* properties [10], respectively, are also universally monitorable.

Theorem 5. *A boolean property P is a response property iff τ_P is universally monitorable from below by an unrestricted verdict function on $\mathbb{B}_t$.*

Proof. Suppose P is a response property, i.e., there exists a set $S \subseteq \Sigma^*$ such that for every $f \in \Sigma^\omega$ we have $f \in P$ iff infinitely many prefixes of f belong to S. Let v be a verdict function as follows: $v(s) = \mathtt{T}$ if $s \in S$, and $v(s) = \mathtt{F}$ if $s \notin S$. Now, let $f \in \Sigma^\omega$ be a trace. We have $\tau_P(f) = \mathtt{T}$ iff for every $s \prec f$ there exists $r \in \Sigma^*$ such that $sr \prec f$ and $sr \in S$ iff $\limsup v(f) = \mathtt{T}$. Therefore, v universally monitors τ_P from below in $\mathbb{B}_t$.

Now, suppose there exists a verdict function v that universally monitors τ_P from below in $\mathbb{B}_t$. Because v is a function on $\mathbb{B}_t = \{\mathtt{T}, \mathtt{F}\}$, there is a set $S \subseteq \Sigma^*$ such that $v(s) = \mathtt{T}$ for all $s \in S$, and $v(s) = \mathtt{F}$ for all $s \notin S$. Then, we get that $\limsup v(f) = \mathtt{T}$ iff for every $s \prec f$ there exists $r \in \Sigma^*$ such that $sr \prec f$ and $sr \in S$ iff $f \in P$. Observe that the set S is exactly as in the definition of a response property, therefore P is a response property. ☐

The proof for persistence properties is symmetric.

Theorem 6. *A boolean property P is a persistence property iff τ_P is universally monitorable from below by an unrestricted verdict function on $\mathbb{B}_f$.*

Positive, finite boolean combinations of response and persistence properties are called *reactivity* properties [10]. We consider existential monitorability in $\mathbb{B}_\perp$ by unrestricted verdict functions, and provide a lower bound.

Theorem 7. *For every boolean reactivity property P, we have that τ_P is existentially monitorable from below by an unrestricted verdict function on $\mathbb{B}_\perp$.*

Proof. Suppose P is a boolean reactivity property, i.e., $P = \bigcap_{i=1}^{k}(R_i \cup P_i)$ for some $k \geq 1$ where R_i is a response and P_i is a persistence property for every $1 \leq i \leq k$. By Theorem 5, each τ_{R_i} is universally monitorable from below by a verdict function u_i on $\mathbb{B}_t$. For each $1 \leq i \leq k$, consider the verdict function v_i on $\mathbb{B}_\perp$ defined as follows: let $v_i(s) = \mathtt{T}$ if $u_i(s) = \mathtt{T}$ or s positively determines R_i, let $v_i(s) = \mathtt{F}$ if s negatively determines R_i, and $v_i(s) = \perp$ otherwise. Note that each v_i existentially monitors τ_{R_i} from below, and for every $f \in \Sigma^\omega$, if $f \in R_i$ for every $1 \leq i \leq k$, then $f \in P$. Then, we can construct the verdict v to monitor τ_P: Let $v(\varepsilon) = \mathtt{T}$ and let x be a memory for v that initially contains ε. On non-empty traces, v outputs $\perp$ until it observes a trace s such that for every $1 \leq i \leq k$ there exists r_i such that $x \prec r_i \preceq s$ and $v_i(r_i) = \mathtt{T}$. When v reads such a trace s, it outputs $\mathtt{T}$, updates x to store s, and outputs $\perp$ until the next trace that satisfies the condition above. Observe that, for every f, if $\limsup v(f) = \mathtt{T}$ then $\tau_P(f) = \mathtt{T}$; and for every s there exists f such that $\limsup v(sf) = \tau_P(sf)$ unless R_i is negatively determined for some $1 \leq i \leq k$.

If for some response component R_i is negatively determined, then we can switch to monitor corresponding persistence component instead. Consider the verdict u'_i for τ_{P_i} on $\mathbb{B}_f$ and construct v'_i on $\mathbb{B}_\perp$ as follows: let $v'_i(s) = \mathtt{F}$ if $u'_i(s) = \mathtt{F}$ or s negatively determines P_i, let $v'_i(s) = \mathtt{T}$ if s positively determines P_i, and $v'_i(s) = \perp$ otherwise. One can verify that for every f, if $\limsup v'_i(f) = \mathtt{F}$ then $\tau_P(f) = \mathtt{F}$; and for every s there exists f such that $\limsup v'_i(sf) = \tau_P(sf)$ unless P_i is positively determined. Once P_i is positively determined, we know that all possible future traces satisfy $R_i \cup P_i$. Then, we can switch to the previous procedure of monitoring the response components, excluding R_i, and repeat as many times as necessary. ☐

We now demonstrate that Theorem 7 is indeed a lower bound for the capabilities of existential monitors in $\mathbb{B}_\perp$.

Example 7. *Let $P = \bigcup_{i \in \mathbb{N}} R_i$ such that each R_i is a response property. In particular, the property P belongs to the class of $G_{\delta\sigma}$ sets in the Borel hierarchy, which strictly contains the reactivity properties. Moreover, suppose that for some $j \in \mathbb{N}$ property R_j is live. Let u be a verdict on $\mathbb{B}_t$ for τ_{R_j}. We construct a verdict v on $\mathbb{B}_\perp$ for τ_P as follows: $v(s) = \mathtt{T}$ if $u(s) = \mathtt{T}$, and $v(s) = \perp$ otherwise. Clearly, for every $f \in \Sigma^\omega$, if $f \in R_j$ then $f \in P$, and thus $\limsup v(f) \leq \tau_P(f)$. Moreover, since R_j is live, so is P, i.e., every finite trace s can be extended with some f such that $sf \in P$, and thus $\limsup v(sf) = \tau_P(sf)$. It follows that P is existentially monitorable from below by v on $\mathbb{B}_\perp$.*

The remaining combinations of monitoring modality and value domain allow us to monitor every boolean property.

Theorem 8. *For every boolean property P, we have that τ_P is existentially monitorable from below by a monotonically increasing verdict function on $\mathbb{B}_t$.*

Proof. Let v be a verdict function such that $v(s) = \mathtt{T}$ if s positively determines P, and $v(s) = \mathtt{F}$ otherwise. Observe that v is indeed monotonically increasing in $\mathbb{B}_t$, and $\limsup v(f) \leq \tau_P(f)$ for every $f \in \Sigma^\omega$. Let $s \in \Sigma^*$ be an arbitrary trace. If $v(s) = \mathtt{T}$, then $\tau_P(sf) = \mathtt{T}$ for every continuation $f \in \Sigma^\omega$; otherwise, there exists some $f \in \Sigma^\omega$

such that $\tau_P(sf) = \mathtt{F}$. It implies that for every $s \in \Sigma^*$ there exists $f \in \Sigma^\omega$ such that $\limsup v(sf) = \tau_P(sf)$. Therefore, τ_P is existentially monitorable from below by v. $\square$

Combining Proposition 8 and Theorem 8, we carry this result over to domain $\mathbb{B}_f$.

Corollary 3. *For every boolean property P, we have that τ_P is existentially monitorable from below by a monotonically increasing verdict function on $\mathbb{B}_f$.*

Tables I and II summarize the results of this section. The classes of safety, co-safety, obligation, response, persistence, reactivity, and classically monitorable boolean properties are denoted by Safe, CoSafe, Obl, Resp, Pers, React, and Mon, respectively. We note that the upper bound for unrestricted existential monitors on $\mathbb{B}_\perp$ is an open problem.

TABLE I
Correspondence between classes of boolean properties and universal monitorability.

	Universally monitorable from below	
$\mathbb{D}$	Monotonically increasing	Unrestricted verdict
$\mathbb{B}$	$\emptyset$ or Σ^ω (Rem. 1)	Obl (Thm. 4)
$\mathbb{B}_\perp$	Safe $\cap$ CoSafe (Thm. 3)	Obl (Prop. 7 + Thm. 4)
$\mathbb{B}_t$	CoSafe (Thm. 2)	Resp (Thm. 5)
$\mathbb{B}_f$	Safe (Thm. 2)	Pers (Thm. 6)

TABLE II
Correspondence between classes of boolean properties and existential monitorability.

	Existentially monitorable from below	
$\mathbb{D}$	Monotonically increasing	Unrestricted verdict
$\mathbb{B}$	$\emptyset$ or Σ^ω (Rem. 1)	Obl (Prop. 7 + Thm. 4)
$\mathbb{B}_\perp$	Mon (Prop.. 6)	at least Mon $\cup$ React (Thm. 7)
$\mathbb{B}_t$	any $P \subseteq \Sigma^\omega$ (Thm. 8)	any $P \subseteq \Sigma^\omega$ (Thm. 8)
$\mathbb{B}_f$	any $P \subseteq \Sigma^\omega$ (Cor. 3)	any $P \subseteq \Sigma^\omega$ (Cor. 3)

We conclude the section with a simple example that demonstrates the concept of precision in the context of boolean properties.

Example 8. *Let $P = \Diamond(a \vee b \vee c)$, where $\Diamond$ is the* eventually *operator [40], and τ_P be the corresponding quantitative property. Consider the following verdict functions on $\mathbb{B}_t$:*

- $v_a(s) = \mathtt{T}$ *iff s contains a,*
- $v_{ab}(s) = \mathtt{T}$ *iff s contains a or b,*
- $v_{bc}(s) = \mathtt{T}$ *iff s contains b or c,*
- $v_{abc}(s) = \mathtt{T}$ *iff s contains a or b or c.*

All these verdict functions are monotonic. Moreover, functions v_a, v_{ab}, and v_{bc} existentially monitor τ_P from below while v_{abc} monitors universally.

Observe that v_{ab} is more precise than v_a, because for every finite prefix s that yields $v_a(s) = \mathtt{T}$, we also get $v_{ab}(s) = \mathtt{T}$, but not vice versa, considering the traces that contain b but not a. However, we cannot compare v_{ab} and v_{bc}, as for every $s \prec a^\omega$, we have $v_{bc}(s) \leq v_{ab}(s)$ and $\limsup v_{bc}(a^\omega) \leq \limsup v_{ab}(a^\omega)$, and vice versa for c^ω. Finally, v_{abc} is the most precise among these verdicts as it universally monitors τ_P.

V. Approximate Register Monitors

A. Verdicts generated by register machines

In this section, we follow [12] to define *register machines* as a model for generating an output stream that represents a verdict sequence for monitoring quantitative properties. We consider a set of integer-valued *registers* denoted X. A *valuation* $\mathsf{v} : X \to \mathbb{Z}$ is a mapping from the set of registers to integers. An *update* is a function from valuations to valuations, and a *test* is a function from valuations to $\mathbb{B}$. The set of updates over X is denoted by $\Gamma(X)$, and the set of tests by $\Phi(X)$. We describe updates and tests over X using integer- and boolean-valued expressions, called *instructions*.

Definition 8. *A (deterministic)* register machine *is a tuple $M = (X, Q, \Sigma, \Delta, q_0, \mathbb{D}, \lambda)$, where X is a finite set of registers, Q is a finite set of states, Σ is a finite alphabet, $\Delta \subseteq Q \times \Sigma \times \Phi(X) \times \Gamma(X) \times Q$ is a set of edges, $q_0 \in Q$ is the initial state. $\mathbb{D}$ is an output value domain, and $\lambda : Q \times \mathbb{Z}^{|X|} \to \mathbb{D}$ is an output function. Moreover, for every state $q \in Q$, letter $\sigma \in \Sigma$, and valuation v, there is exactly one outgoing edge $(q, \sigma, \phi, \gamma, q') \in \Delta$ with $\mathsf{v} \models \phi$.*

Let $M = (X, Q, \Sigma, \Delta, q_0, \mathbb{D}, \lambda)$ be a register machine. A pair consisting of a state $q \in Q$ and a valuation $\mathsf{v} : X \to \mathbb{Z}$ constitute a *configuration* of M. The initial configuration (q_0, v_0) of M is such that $\mathsf{v}_0(x) = 0$ for every $x \in X$. Between two configurations of M, the *transition* relation is defined by $(q, \mathsf{v}) \xrightarrow{\sigma} (q', \mathsf{v}')$ iff there exists an edge $(q, \sigma, \phi, \gamma, q') \in \Delta$ such that $\mathsf{v} \models \phi$ and $\mathsf{v}' = \gamma(\mathsf{v})$. On an infinite word $f = \sigma_1\sigma_2 \ldots$, the machine M produces an infinite sequence of transitions $(q_0, \mathsf{v}_0) \xrightarrow{\sigma_1} (q_1, \mathsf{v}_1) \xrightarrow{\sigma_2} \cdots$, and an infinite *output sequence* $(\lambda(q_i, \mathsf{v}_i))_{i \in \mathbb{N}}$.

Definition 9. *A register machine $M = (X, Q, \Sigma, \Delta, q_0, \mathbb{D}, \lambda)$* generates *the verdict function $v : \Sigma^* \to \mathbb{D}$ iff, for every finite trace $s \in \Sigma^*$, the machine M after reading s reaches a configuration (q, v) such that $\lambda(q, \mathsf{v}) = v(s)$.*

We mainly focus on a simple form of register machines which can only increment, reset, and compare registers. The according instruction set is denoted by $\langle 0, +1, \geq \rangle$, and equivalent to the instruction set $\langle 0, +1, -1, \geq 0 \rangle$, as was shown in [11].

Definition 10. *A* counter machine *is a register machine with the instructions $x \leftarrow 0$, $x \leftarrow x + 1$, and $x \geq y$ for registers $x, y \in X$, and an output function that in every state outputs 0, ∞, or one of the register values. A verdict function v is a k-*counter verdict function *iff v is generated by a counter machine M with k registers. A quantitative property p is k-*counter monitorable *iff there is a k-counter verdict function that monitors p.*

Note that we can use the various modalities of monitoring defined in Section II. For instance, a property p is existentially k-counter monitorable from below iff p is

k-counter monitorable and the witnessing verdict function existentially monitors p from below.

One can also define *extended counter machines* with generic output functions. For example, a verdict function generated by an extended 3-counter machine (with an output function that can perform division) can universally monitor the average response-time property from above, as demonstrated in Example 4.

We remark that our model of register machines is more general than register transducer models operating over uninterpreted infinite alphabets, which typically cannot count (see, e.g., [41]).

B. Precision-resource trade-offs for register machines

In the following example, we illustrate how the arithmetic operations of register machines can play a role in precision-resource trade-offs for monitoring.

Example 9 (Adders versus counters)**.** *Let $\Sigma = \{a, b\}$ and $\mathbb{D} = \mathbb{N} \cup \{\infty\}$. Let p be a property such that $p(f) = 2^n$, where n is the length of the longest uninterrupted sequence of a's in $f \in \Sigma^\omega$. Consider a 2-register machine M with the following instructions: $x \leftarrow 1$, $x \leftarrow x + y$, and $x \geq y$ for $x, y \in X$. When M starts reading a segment of a's, it resets one of its registers, say x, to 1 and doubles its value after each a. After the segment ends, it compares the value of x with the other register, say y, and stores the maximum in y, which determines the output value. This way M can generate v_{add} such that $v_{add}(s) = 2^n$, where n is the length of the longest uninterrupted sequence of a's in $s \in \Sigma^*$. Verdict v_{add} is monotonically increasing and it universally monitors p. Now, suppose that we have, instead, a verdict v_{count} that is generated by a 2-counter machine. Since the counter values can only grow linearly, we can have $v_{count}(s) = 2n$ for n as above. Although it grows much slower, v_{count} existentially monitors p from below, because the extension a^ω yields $\limsup v_{count}(sa^\omega) = p(sa^\omega) = \infty$ for every $s \in \Sigma^*$. Since v_{add} universally monitors p, it is clearly more precise than v_{count}.*

Recall the two-pair maximal response-time property from Section I. We can generalize this property to give an example for a precision-resource trade-off on the number of counter registers that are available for monitoring.

Example 10 (Counter machine)**.** *Let $k \in \mathbb{N}$ and let $\Sigma_k = \{req_1, ack_1, \dots, req_k, ack_k, other\}$. The k-pair maximal response-time property $p : \Sigma^\omega \to (\mathbb{N} \cup \{\infty\})^k$ specifies the maximal response times for all (req, ack) pairs in Σ_k. More explicitly, for every $1 \leq i \leq k$, let p_i specify the maximal response-time for pair (req_i, ack_i) as in Example 2, and let $p(f) = (p_1(f), \dots, p_k(f))$ for every $f \in \Sigma^\omega$. As hinted in Section I, there is a $2k$-counter verdict function v_{2k} which simply combines the 2-counter verdict functions u_i that universally monitor p_i for all $1 \leq i \leq k$. Specifically, $v_{2k}(s) = (u_1(s), \dots, u_k(s))$ for every $s \in \Sigma^*$.*

Observe that a $(k + 1)$-counter verdict function v_{k+1} cannot universally monitor p, because whenever it reads a trace that contains more than one active request, it needs to either ignore some active requests and process only one of them, or forget the maximal response time for some pairs and use those counters to process the active requests. However, it can existentially monitor p from below, because every finite trace can be extended with a continuation f in which all previously active requests are acknowledged and the true maxima occur in f one by one. If the server has at most one active request at any given time, then $k + 1$ counters suffice for universal monitoring. This is because it only needs to use one register to process the current response time while storing the maxima in the remaining k counters. Let p' be the variant of p under the assumption of no simultaneous requests, and suppose we have a $(\frac{k}{2} + 1)$-counter verdict function $v_{\frac{k}{2}+1}$. By the same reason that v_{k+1} cannot universally monitor p, the function $v_{\frac{k}{2}+1}$ cannot universally monitor p'. However, for every odd number $1 \leq i < k$, we can assign one counter to store $\max(u_i(s), u_{i+1}(s))$, which provides an over-approximation for either p_i or p_{i+1} while being precise for the other. Overall, although it can provide a meaningful approximation, the function $v_{\frac{k}{2}+1}$ is less precise than v_{k+1}.

The following theorems generalize this example.

Theorem 9. *For every $k > 1$, there exists a quantitative property p_k such that p_k is universally monitorable by a monotonically decreasing k-counter verdict function v_k. Moreover, for every $\ell < k$ and every monotonically decreasing ℓ-counter verdict function v_ℓ that approximately monitors p_k from below (resp. above), there exists a monotonically decreasing $(\ell + 1)$-counter verdict function $v_{\ell+1}$ that approximately monitors p_k from below (resp. above) such that $v_{\ell+1}$ is more precise than v_ℓ.*

Proof. For convenience, we consider the $\langle 0, +1, -1, \geq 0\rangle$ variant of counter machines. Let $\Sigma_k = \{1, \dots, k\}$. For every $s \in \Sigma_k^*$ and $i \in \Sigma_k$ denote by $|s|_i$ the number of occurrences of the letter i in s. Consider the boolean safety property $P_k = \{f \in \Sigma_k^\omega \mid \forall 1 \leq i < k : \forall s \prec f : |s|_i \geq |s|_{i+1}\}$, and let p_k be as follows: $p_k(f) = \infty$ if $f \in P_k$, and $p_k(f) = |r|$ otherwise, where r is the shortest prefix of f that negatively determines P_k. We construct a verdict function v_k as follows: $v_k(s) = \infty$ if s does not negatively determine P_k, and $v_k(s) = |r|$ otherwise, where r is the shortest prefix of s that negatively determines P_k. The verdict v_k is monotonically decreasing and it can be generated by a k-counter machine where, for every $1 \leq i < k$ and $s \in \Sigma_k^*$, the counter x_i stores $|s|_i - |s|_{i+1}$, and x_k stores $|s|$. Moreover, because we need $k - 1$ counters to recognize P_k (see Thm. 4.3 in [11]) and one more to store the output, p_k is not universally ℓ-counter monitorable for $\ell < k$.

Let $\ell < k$, and take a monotonically decreasing ℓ-counter verdict function v_ℓ that approximately monitors p_k from below. Note that, for every $s \in \Sigma_k^*$, if the generating counter machine does not store a linear function $\alpha(s) \leq |s|$, then v_ℓ can be either the constant 0 function or a function that switches from ∞ to 0 and never misses a violation. Then, we construct an $(\ell+1)$-counter machine that stores $|s|$ in the new counter. If v_ℓ is constant, it uses the rest to count $|s|_i - |s|_{i+1}$ for $1 \leq i < \ell$ and catch violations, similarly as v_k above; otherwise, it outputs $|s|$ instead of 0. The resulting verdict $v_{\ell+1}$ is monotonically decreasing and approximately monitors p_k from below. It is also more precise than v_ℓ. Now, suppose the generating ℓ-counter machine counts a linear function $\alpha(s) \leq |s|$. Since this machine cannot recognize P_k, there exists $f \in P_k$ such that $\lim v_\ell(f) < \infty$, i.e., v_ℓ incorrectly concludes that $|s|_i < |s|_{i+1}$ for some $s \prec f$ and $1 \leq i < k$. We construct an $(\ell+1)$-counter machine M where the additional counter keeps track of $|s|_i - |s|_{i+1}$ for every $s \in \Sigma_k^*$, the output register stores $|s|$, and the rest behave the same as in v_ℓ. Moreover, whenever v_ℓ concludes that $|s|_i < |s|_{i+1}$, the behavior of M is determined by the correct value of $|s|_i - |s|_{i+1}$ stored in the new counter. It yields that the verdict $v_{\ell+1}$ generated by M is more precise than v_ℓ because $\lim v_\ell(f) < \lim v_{\ell+1}(f)$ for some trace $f \in \Sigma^\omega$. The case for monitoring from above is similar. □

Since monitoring p_k in the proof above involves recognizing a boolean property P_k, the counter machine for p_k must be able to distinguish traces with respect to P_k. To achieve this, intuitively, the machine needs a counter for each "independent" quantity. Also, the use of a variable-size alphabet Σ_k is merely a convenience. We can encode every word over Σ_k in binary with the help of an additional separator symbol. More explicitly, we can take a ternary alphabet $\Sigma = \{0, 1, \#\}$ to represent every letter in Σ_k as a binary sequence and separate the sequences by #. We combine these observations to construct a quantitative property for which counting does not suffice for universal monitoring no matter the number of registers, but each additional register gives a better approximation.

Theorem 10. *There exists a quantitative property p such that for every $k > 1$ and every k-counter verdict function v_k that approximately monitors p from below (resp. above), there exists a $(k+1)$-counter verdict function v_{k+1} that approximately monitors p from below (resp. above) and is more precise than v_k.*

Proof. Let $\Sigma = \{0, 1, \#\}$. For every $s \in \Sigma^*$ and $i \in \mathbb{N}$, let $n_i(s)$ denote the number of occurrences of the binary sequence that corresponds to i in the longest prefix of s that ends with #. For example, if we have $s = 001\#10$, then $n_1(s) = 1$ and $n_2(s) = 0$. Similarly as in the proof of Theorem 9, consider counter machines with instructions $\langle 0, +1, -1, \geq 0 \rangle$ and the following boolean safety property: $P = \{f \in \Sigma^\omega \mid \forall i \in \mathbb{N} : \forall s \prec f : n_i(s) \geq n_{i+1}(s)\}$. We define the quantitative property p as $p(f) = \infty$ if $f \in P$, and $p(f) = |r|_\#$ where r is the shortest prefix of f that negatively determines P. Observe that P is a generalization of P_k in the proof of Theorem 9, and it requires counting infinitely many distinct quantities. Therefore, one can show that, to universally monitor p, one needs infinitely many counter registers. However, for every $k > 1$, there exists a k-counter verdict function v_k that approximately monitors p from below or above, for instance, by keeping track of $n_i(s) - n_{i+1}(s)$ for every $1 \leq i < k$ and counting #'s in the remaining register.

We now construct a $(k+1)$-counter verdict function v_{k+1} from v_k. Suppose v_k approximately monitors p from below. Note that the generating machine of v_k lacks the resources to distinguish traces with respect to P correctly. Therefore, regardless of the monotonicity of v_k, a similar reasoning as in the proof of Theorem 9 applies. We can use the additional counter of v_{k+1} to keep track of $n_i(s) - n_{i+1}(s)$ or $|s|_\#$ for every $s \in \Sigma^*$ while the rest operate the same as in v_k. It yields that $\limsup v_k(f) < \limsup v_{k+1}(f)$; thus v_{k+1} is more precise than v_k. One can similarly show the case for monitoring from above. □

VI. Conclusion and Future Work

We argued for the need of a quantitative semantic framework for runtime verification which supports monitors that over- or under-approximate quantitative properties, and we provided such a framework.

An obvious direction for future work is to systematically explore precision-resource tradeoffs for different monitor models and property classes. For example, a quantitative property class that we have not considered in this work is the limit monitoring of statistical indicators [12]. Other interesting resources that play a role in precision-resource trade-offs are the "speed" or rate of convergence of monitors, that is, how quickly a monitor reaches the desired property value, and "assumptions", that is, prior knowledge about the system or the environment that can be used by the monitor [42], [43]. We also plan to consider the reliability of communication channels [44] and how it relates to monitoring precision.

Another question is the synthesis problem: given a quantitative property p and a register machine template (instruction set and number of registers), does there exist a register machine M generating a verdict function v that universally or existentially monitors p from below or above?

Building on our definitions of continuous and co-continuous quantitative properties, one can define a generalization of the safety-progress hierarchy [10] to obtain a Borel classification of quantitative properties.

Lastly, a logical extension of monitoring is *enforcement* [45], [46], [6], that is, manipulating the observed system's behavior to prevent undesired outcomes. We aim to extend the notion of enforceability from the boolean to the quantitative setting and explore precision-resource trade-offs for enforcement monitors (a.k.a. *shields* [47]).

Acknowledgment

We thank the anonymous reviewers for their helpful comments. This research was supported in part by the Austrian Science Fund (FWF) under grant Z211-N23 (Wittgenstein Award).

References

[1] E. Bartocci, Y. Falcone, A. Francalanza, and G. Reger, "Introduction to runtime verification," in *Lectures on Runtime Verification - Introductory and Advanced Topics*, ser. Lecture Notes in Computer Science, E. Bartocci and Y. Falcone, Eds. Springer, 2018, vol. 10457, pp. 1–33. [Online]. Available: https://doi.org/10.1007/978-3-319-75632-5_1

[2] M. Kwiatkowska, "Quantitative verification: Models techniques and tools," in *Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, ser. ESEC-FSE '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 449–458. [Online]. Available: https://doi.org/10.1145/1287624.1287688

[3] T. A. Henzinger, "Quantitative reactive modeling and verification," *Comput. Sci.*, vol. 28, no. 4, p. 331–344, Nov. 2013. [Online]. Available: https://doi.org/10.1007/s00450-013-0251-7

[4] U. Boker and T. A. Henzinger, "Exact and approximate determinization of discounted-sum automata," *Log. Methods Comput. Sci.*, vol. 10, no. 1, 2014. [Online]. Available: https://doi.org/10.2168/LMCS-10(1:10)2014

[5] M. Kim, S. Kannan, I. Lee, O. Sokolsky, and M. Viswanathan, "Computational analysis of run-time monitoring: Fundamentals of java-mac1 1this research was supported in part by onr n00014-97-1-0505, nsf ccr-9988409, nsf ccr-0086147, nsf cise-9703220, and aro daad19-01-1-0473." *Electronic Notes in Theoretical Computer Science*, vol. 70, no. 4, pp. 80 – 94, 2002, rV'02, Runtime Verification 2002 (FLoC Satellite Event). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1571066104805784

[6] Y. Falcone, J.-C. Fernandez, and L. Mounier, "What can you verify and enforce at runtime?" *International Journal on Software Tools for Technology Transfer*, vol. 14, no. 3, pp. 349–382, 2012.

[7] L. Aceto, A. Achilleos, A. Francalanza, A. Ingólfsdóttir, and K. Lehtinen, "An operational guide to monitorability," in *Software Engineering and Formal Methods*, P. C. Ölveczky and G. Salaün, Eds. Cham: Springer International Publishing, 2019, pp. 433–453.

[8] A. Pnueli and A. Zaks, "Psl model checking and run-time verification via testers," in *FM 2006: Formal Methods*, J. Misra, T. Nipkow, and E. Sekerinski, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 573–586.

[9] A. Bauer, M. Leucker, and C. Schallhart, "Runtime verification for ltl and tltl," *ACM Trans. Softw. Eng. Methodol.*, vol. 20, no. 4, Sep. 2011. [Online]. Available: https://doi.org/10.1145/2000799.2000800

[10] E. Chang, Z. Manna, and A. Pnueli, "The safety-progress classification," in *Logic and Algebra of Specification.* Springer, 1993, pp. 143–202.

[11] T. Ferrère, T. A. Henzinger, and N. E. Saraç, "A theory of register monitors," in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, 2018, pp. 394–403.

[12] T. Ferrère, T. A. Henzinger, and B. Kragl, "Monitoring event frequencies," in *28th EACSL Annual Conference on Computer Science Logic (CSL 2020).* Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[13] A. Francalanza, L. Aceto, A. Achilleos, D. P. Attard, I. Cassar, D. Della Monica, and A. Ingólfsdóttir, "A foundation for runtime monitoring," in *Runtime Verification*, S. Lahiri and G. Reger, Eds. Cham: Springer International Publishing, 2017, pp. 8–29.

[14] L. Aceto, A. Achilleos, A. Francalanza, A. Ingólfsdóttir, and K. Lehtinen, "Adventures in monitorability: from branching to linear time and back again," *Proceedings of the ACM on Programming Languages*, vol. 3, no. POPL, pp. 1–29, 2019.

[15] A. Bauer, M. Leucker, and C. Schallhart, "Monitoring of real-time properties," in *International Conference on Foundations of Software Technology and Theoretical Computer Science.* Springer, 2006, pp. 260–272.

[16] ——, "The good, the bad, and the ugly, but how ugly is ugly?" in *International Workshop on Runtime Verification.* Springer, 2007, pp. 126–138.

[17] N. Decker, M. Leucker, and D. Thoma, "Impartiality and anticipation for monitoring of visibly context-free properties," in *Runtime Verification*, A. Legay and S. Bensalem, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 183–200.

[18] H. Barringer, Y. Falcone, K. Havelund, G. Reger, and D. Rydeheard, "Quantified event automata: Towards expressive and efficient runtime monitors," in *FM 2012: Formal Methods*, D. Giannakopoulou and D. Méry, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 68–84.

[19] K. Havelund, G. Reger, D. Thoma, and E. Zălinescu, *Monitoring Events that Carry Data.* Cham: Springer International Publishing, 2018, pp. 61–102. [Online]. Available: https://doi.org/10.1007/978-3-319-75632-5_3

[20] L. de Alfaro, T. A. Henzinger, and R. Majumdar, "Discounting the future in systems theory," in *Automata, Languages and Programming*, J. C. M. Baeten, J. K. Lenstra, J. Parrow, and G. J. Woeginger, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 1022–1037.

[21] P. Bouyer, N. Markey, and R. M. Matteplackel, "Averaging in ltl," in *International Conference on Concurrency Theory.* Springer, 2014, pp. 266–280.

[22] B. Bonakdarpour, S. Navabpour, and S. Fischmeister, "Time-triggered runtime verification," *Formal Methods in System Design*, vol. 43, no. 1, pp. 29–60, 2013.

[23] S. Pinisetty, T. Jéron, S. Tripakis, Y. Falcone, H. Marchand, and V. Preoteasa, "Predictive runtime verification of timed properties," *Journal of Systems and Software*, vol. 132, pp. 353–365, 2017.

[24] A. Bakhirkin, T. Ferrère, O. Maler, and D. Ulus, "On the quantitative semantics of regular expressions over real-valued signals," in *Formal Modeling and Analysis of Timed Systems*, A. Abate and G. Geeraerts, Eds. Cham: Springer International Publishing, 2017, pp. 189–206.

[25] S. Jakšić, E. Bartocci, R. Grosu, T. Nguyen, and D. Ničković, "Quantitative monitoring of stl with edit distance," *Formal methods in system design*, vol. 53, no. 1, pp. 83–112, 2018.

[26] R. Alur, K. Mamouras, and C. Stanford, "Automata-based stream processing," in *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017).* Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[27] ——, "Modular quantitative monitoring," *Proc. ACM Program. Lang.*, vol. 3, no. POPL, Jan. 2019. [Online]. Available: https://doi.org/10.1145/3290363

[28] R. Alur, D. Fisman, K. Mamouras, M. Raghothaman, and C. Stanford, "Streamable regular transductions," *Theoretical Computer Science*, vol. 807, pp. 15–41, 2020.

[29] P. Caspi and A. Benveniste, "Toward an approximation theory for computerised control," in *Embedded Software*, A. Sangiovanni-Vincentelli and J. Sifakis, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 294–304.

[30] R. Bloem, K. Chatterjee, T. A. Henzinger, and B. Jobstmann, "Better quality in synthesis through quantitative objectives," in *Computer Aided Verification*, A. Bouajjani and O. Maler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 140–156.

[31] K. Chatterjee, L. Doyen, and T. A. Henzinger, "Quantitative languages," *ACM Trans. Comput. Logic*, vol. 11, no. 4, Jul. 2010. [Online]. Available: https://doi.org/10.1145/1805950.1805953

[32] R. Bloem, K. Chatterjee, and B. Jobstmann, *Graph Games and Reactive Synthesis.* Cham: Springer International Publishing, 2018, pp. 921–962. [Online]. Available: https://doi.org/10.1007/978-3-319-10575-8_27

[33] P. Bouyer, N. Markey, M. Randour, K. G. Larsen, and S. Laursen, "Average-energy games," *Acta Informatica*, vol. 55, no. 2, pp. 91–127, 2018.

[34] K. Chatterjee, T. A. Henzinger, and J. Otop, "Quantitative monitor automata," in *International Static Analysis Symposium.* Springer, 2016, pp. 23–38.

[35] E. Paul, "Monitor logics for quantitative monitor automata," in *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017).* Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[36] R. Calinescu, C. Ghezzi, M. Kwiatkowska, and R. Mirandola, "Self-adaptive software needs quantitative verification at runtime," *Communications of the ACM*, vol. 55, no. 9, pp. 69–77, 2012.

[37] N. Fulton and A. Platzer, "Safe reinforcement learning via formal methods: Toward safe control through proof and learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[38] T. A. Henzinger, A. Lukina, and C. Schilling, "Outside the box: Abstraction-based monitoring of neural networks," in *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, ser. Frontiers in Artificial Intelligence and Applications, G. D. Giacomo, A. Catalá, B. Dilkina, M. Milano, S. Barro, A. Bugarín, and J. Lang, Eds., vol. 325. IOS Press, 2020, pp. 2433–2440. [Online]. Available: https://doi.org/10.3233/FAIA200375

[39] K. Weihrauch, *Complete Partial Orders.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 404–432. [Online]. Available: https://doi.org/10.1007/978-3-642-69965-8_28

[40] N. Piterman and A. Pnueli, *Temporal Logic and Fair Discrete Systems.* Cham: Springer International Publishing, 2018, pp. 27–73. [Online]. Available: https://doi.org/10.1007/978-3-319-10575-8_2

[41] A. Khalimov, B. Maderbacher, and R. Bloem, "Bounded synthesis of register transducers," in *Automated Technology for Verification and Analysis*, S. K. Lahiri and C. Wang, Eds. Cham: Springer International Publishing, 2018, pp. 494–510.

[42] T. A. Henzinger and N. E. Saraç, "Monitorability under assumptions," in *Runtime Verification*, J. Deshmukh and D. Ničković, Eds. Cham: Springer International Publishing, 2020, pp. 3–18.

[43] L. Aceto, A. Achilleos, A. Francalanza, A. Ingólfsdóttir, and K. Lehtinen, "The best a monitor can do," in *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, January 25-28, 2021, Ljubljana, Slovenia (Virtual Conference)*, ser. LIPIcs, C. Baier and J. Goubault-Larrecq, Eds., vol. 183. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 7:1–7:23. [Online]. Available: https://doi.org/10.4230/LIPIcs.CSL.2021.7

[44] S. Kauffman, K. Havelund, and S. Fischmeister, "What can we monitor over unreliable channels?" *International Journal on Software Tools for Technology Transfer*, pp. 1–24, 2020.

[45] J. Ligatti and S. Reddy, "A theory of runtime enforcement, with results," in *Computer Security – ESORICS 2010*, D. Gritzalis, B. Preneel, and M. Theoharidou, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 87–100.

[46] Y. Falcone, L. Mounier, J.-C. Fernandez, and J.-L. Richier, "Runtime enforcement monitors: composition, synthesis, and enforcement abilities," *Formal Methods in System Design*, vol. 38, no. 3, pp. 223–262, 2011.

[47] B. Könighofer, M. Alshiekh, R. Bloem, L. Humphrey, R. Könighofer, U. Topcu, and C. Wang, "Shield synthesis," *Formal Methods in System Design*, vol. 51, no. 2, pp. 332–361, 2017.