Safety and Liveness of Quantitative Automata

Udi Boker
Reichman University, Herzliya, Israel

Thomas A. Henzinger
Institute of Science and Technology Austria (ISTA), Klosterneuburg, Austria

Nicolas Mazzocchi
Institute of Science and Technology Austria (ISTA), Klosterneuburg, Austria

N. Ege Saraç
Institute of Science and Technology Austria (ISTA), Klosterneuburg, Austria

Abstract

The safety-liveness dichotomy is a fundamental concept in formal languages which plays a key role in verification. Recently, this dichotomy has been lifted to quantitative properties, which are arbitrary functions from infinite words to partially-ordered domains. We look into harnessing the dichotomy for the specific classes of quantitative properties expressed by quantitative automata. These automata contain finitely many states and rational-valued transition weights, and their common value functions Inf, Sup, LimInf, LimSup, LimInfAvg, LimSupAvg, and DSum map infinite words into the totally-ordered domain of real numbers. In this automata-theoretic setting, we establish a connection between quantitative safety and topological continuity and provide an alternative characterization of quantitative safety and liveness in terms of their boolean counterparts. For all common value functions, we show how the safety closure of a quantitative automaton can be constructed in PTime, and we provide PSpace-complete checks of whether a given quantitative automaton is safe or live, with the exception of LimInfAvg and LimSupAvg automata, for which the safety check is in ExpSpace. Moreover, for deterministic Sup, LimInf, and LimSup automata, we give PTime decompositions into safe and live automata. These decompositions enable the separation of techniques for safety and liveness verification for quantitative specifications.

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

Keywords and phrases quantitative safety, quantitative liveness, quantitative automata

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2023.17

Related Version *Full Version*: `https://arxiv.org/abs/2307.06016`

Funding This work was supported in part by the ERC-2020-AdG 101020093 and the Israel Science Foundation grant 2410/22.

Acknowledgements We thank Christof Löding for pointing us to some results on PSpace-hardess of universality problems and the anonymous reviewers for their helpful comments.

1 Introduction

Safety and liveness [2] are fundamental concepts in the specification of system behaviors and their verification. While safety characterizes whether a system property can *always* be falsified by a finite prefix of its violating executions, liveness characterizes whether this is *never* possible. A celebrated result shows that *every* property is the intersection of a safety property and a liveness property [2]. This decomposition significantly impacts verification efforts: every verification task can be split into verifying a safety property, which can be solved by lighter methods, such as computational induction, and a liveness property, which requires heavier methods, such as ranking functions.

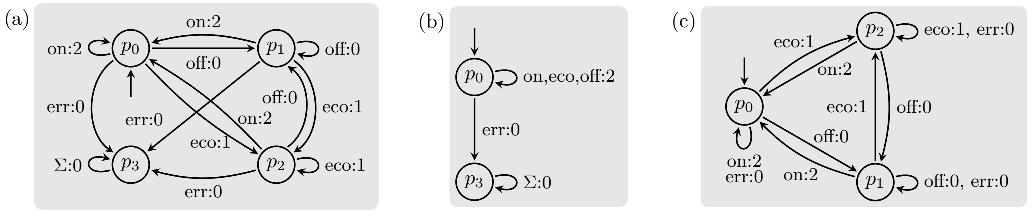

Figure 1 (a) A LimSup-automaton $\mathcal{A}$ modeling the long-term maximal power consumption of a device. **(b)** An Inf-automaton (or a LimSup-automaton) expressing the safety closure of $\mathcal{A}$. **(c)** A LimSup-automaton expressing the liveness component of the decomposition of $\mathcal{A}$.

The notions of safety and liveness consider system properties in full generality: every set of system executions – even the uncomputable ones – can be seen through the lens of the safety-liveness dichotomy. To bring these notions more in line with practical requirements, their projections onto formalisms with desirable closure and decidability properties, such as ω-regular languages, have been studied thoroughly. For example, [3] gives a construction for the safety closure of a Büchi automaton and shows that Büchi automata are closed under the safety-liveness decomposition. In turn, [29] describes an efficient model-checking algorithm for Büchi automata that define safety properties.

Boolean properties define *sets* of system executions or, equivalently, characteristic functions mapping each infinite execution to a binary truth value. Quantitative properties [10] generalize their boolean counterparts; they are *functions* from infinite executions to richer *value domains*, such as the real numbers, allowing the specification and verification of system properties not only for correctness but also for performance and robustness.

As in the boolean case, quantitative extensions of safety and liveness [25] have been defined through the falsifiability, from finite execution prefixes, of quantitative membership hypotheses, which are claims that a given value is a lower or upper bound on the values of certain executions. In particular, quantitative safety (resp. co-safety) characterizes whether every wrong lower (resp. upper) bound hypothesis can *always* be rejected by a finite execution prefix, and quantitative liveness (resp. co-liveness) characterizes whether some wrong lower (resp. upper) bound hypothesis can *never* be rejected by a finite execution prefix. In this setting, the safety closure of a quantitative property maps each execution to the greatest lower bound over the best values that all execution prefixes can have via some continuations; in other words, it is the least safety property that bounds the given property from above [25].

Let us give some examples. Suppose we have three observations `on`, `eco`, and `off`, corresponding to the operational modes of a device, with the power consumption values 2, 1, and 0, respectively. The quantitative property MinPow maps every execution to the minimum among the power consumption values of the modes that occur in the execution, and MaxPow maps them to the corresponding maximum. The property MinPow is safe because, for every execution and power consumption value v, if the MinPow value of the execution is less than v, then there is a finite prefix of the execution in which an operational mode with a power value less than v occurs, and after this prefix, no matter what infinite execution follows, MinPow value cannot be greater. The property MaxPow is live because, for every execution (whose MaxPow value is not the maximal possible value of 2), there is a power value v such that the MaxPow value of the execution is less than v, but for all of its finite prefixes there is an infinite continuation that achieves a MaxPow value of at least v.

Similarly to how boolean automata (e.g., regular and ω-regular automata) define classes of boolean properties amenable to boolean verification, quantitative automata (e.g., limit-average and discounted-sum automata) define classes of quantitative properties amenable to quantitative verification. Quantitative automata generalize standard boolean automata with weighted transitions and a value function that accumulates an infinite sequence of weights into a single value, a generalization of acceptance conditions of ω-regular automata. Let us extend the set of possible observations in the above example with `err`, which denotes an error in the device. In Figure 1a, we describe a quantitative automaton using the value function LimSup to express the long-term maximal power consumption of the device.

In this work, we study the projection of the quantitative safety-liveness dichotomy onto the properties definable by common quantitative automata. First, we show how certain attributes of quantitative automata simplify the notions of safety and liveness. Then, we use these simplifications to the study safety and liveness of the classes of quantitative automata with the value functions Inf, Sup, LimInf, LimSup, LimInfAvg, LimSupAvg, and DSum [10].

In contrast to general quantitative properties, these quantitative automata use functions on the totally-ordered domain of the real numbers (as opposed to a more general partially-ordered domain). In addition, quantitative automata have the restriction that only finitely many weights (those on the automaton transitions) can contribute to the value of an execution. These constraints allow us to provide alternative, simpler characterizations of safety for properties defined by quantitative automata. In particular, we show that, for totally-ordered value domains, a quantitative property is safe iff, for every value v, the set of executions whose value is at least v is safe in the boolean sense. The total-order restriction also allows us to study quantitative safety through the lens of topological continuity. In particular, we characterize safety properties as continuous functions with respect to the left-order topology of their totally-ordered value domain. Moreover, we define the safety of value functions and show that a value function is safe iff every quantitative automaton equipped with this value function expresses a safety property. For example, Inf is a safe value function. Pushing further, we characterize discounting properties and value functions as those that are uniformly continuous and show that it characterizes the conjunction of safety and co-safety. For example, DSum is a discounting value function, therefore both safe and co-safe.

We prove that the considered classes of quantitative automata have the ability to express the least upper bound over their values, namely, they are supremum-closed. Similarly as for safety and the total-order constraint, this ability helps us simplify quantitative liveness. For supremum-closed quantitative properties, we show that a property is live iff for every value v, the set of executions whose value is at least v is live in the boolean sense.

These simplifying characterizations of safety and liveness for quantitative automata prove useful for checking the safety and liveness of these automata, for constructing the safety closure of an automaton, and for decomposing an automaton into safety and liveness components. Let us recall the quantitative automaton in Figure 1a. Since it is supremum-closed, we can construct its safety closure in PTime by computing the maximal value it can achieve from each state. The safety closure of this automaton is shown in Figure 1b. For the value functions Inf, Sup, LimInf, LimSup, LimInfAvg, and LimSupAvg, the safety closure of a given automaton is an Inf-automaton, while for DSum, it is a DSum-automaton.

Evidently, one can check if a quantitative automaton $\mathcal{A}$ is safe by checking if it is equivalent to its safety closure, i.e., if $\mathcal{A}(w) = \mathit{SafetyCl}(\mathcal{A})(w)$ for every execution w. This allows for a PSpace procedure for checking the safety of Sup-, LimInf-, and LimSup-automata [10], but not for LimInfAvg- and LimSupAvg-automata, whose equivalence check is undecidable [15]. For these cases, we use the special structure of the safety-closure automaton for reducing safety checking to the problem of whether some other automaton expresses a constant function. We show that the latter problem is PSpace-complete for LimInfAvg- and LimSupAvg-automata, by a somewhat involved reduction to the limitedness problem of distance automata, and obtain an ExpSpace decision procedure for their safety check.

Thanks to our alternative characterization of liveness, one can check if a quantitative automaton $\mathcal{A}$ is live by checking if its safety closure is universal with respect to its maximal value, i.e., if $\mathit{SafetyCl}(\mathcal{A})(w) \geq \top$ for every execution w, where $\top$ is the supremum over the values of $\mathcal{A}$. For all value functions we consider except DSum, the safety closure is an Inf-automaton, which allows for a PSpace solution to liveness checking [10], which we show to be optimal. Yet, it is not applicable for DSum automata, as the decidability of their universality check is an open problem. Nonetheless, as we consider only universality with respect to the maximal value of the automaton, we can reduce the problem again to checking whether an automaton defines a constant function, which we show to be in PSpace for DSum-automata. This yields a PSpace-complete solution to the liveness check of DSum-automata.

Finally, we investigate the safety-liveness decomposition for quantitative automata. Recall the automaton from Figure 1a and its safety closure from Figure 1b. The liveness component of the corresponding decomposition is shown in Figure 1c. Intuitively, it ignores `err` and provides information on the power consumption as if the device never fails. Then, for every execution w, the value of the original automaton on w is the minimum of the values of its safety closure and the liveness component on w. Since we identified the value functions Inf and DSum as safe, their safety-liveness decomposition is trivial. For deterministic Sup-, LimInf-, and LimSup-automata, we provide PTime decompositions, where for Sup and LimInf it extends to nondeterministic automata at the cost of exponential determinization.

We note that our alternative, simpler characterizations of safety and liveness of quantitative properties extend to co-safety and co-liveness. Our results for the specific automata classes are summarized in Table 1. While we focus on automata that resolve nondeterminism by sup, their duals hold for quantitative co-safety and co-liveness of automata that resolve nondeterminism by inf, as well as for deterministic automata. We leave the questions of co-safety and co-liveness for automata that resolve nondeterminism by sup open.

Related Work. The notions of safety and liveness for boolean properties were first presented in [30] and were later formally defined in [2]. The projections of safety and liveness onto properties definable by Büchi automata were studied in [3]. For linear temporal logic, safety and liveness were studied in [37], where checking whether a given formula is safe was shown to be PSpace-complete. The safety-liveness dichotomy also shaped various efforts on verification, such as an efficient model-checking algorithm for safe Büchi automata [29]. A framework for monitorability through the lens of safety and liveness was given in [34], and a monitor model for safety properties beyond ω-regular ones was defined and studied in [18].

Quantitative properties (a.k.a. quantitative languages [10]) generalize their boolean counterparts by moving from a binary domain of truth values to richer value domains such as the real numbers. In the past decades, quantitative properties and automata have been studied extensively in games with quantitative objectives [6, 9], specification and analysis of system robustness [33], measuring the distance between two systems or specifications [13, 23], best-effort synthesis and repair [5, 12], approximate monitoring [26, 24], and more [11, 8, 17].

Safety and liveness of general quantitative properties were defined and studied in [25]. In particular, quantitative safety properties were characterized as upper semicontinuous functions, and every quantitative property was shown to be the pointwise minimum of a safety property and a liveness property. Yet, these definitions have not been studied from the perspective of quantitative finite-state automata.

Other definitions of safety and liveness for nonboolean formalisms were presented in [32, 20]. While [32] focuses on multi-valued formalisms with the aim of providing model-checking algorithms, [20] focuses on the monitorability view of safety and liveness in richer value

Table 1 The complexity of performing the operations on the left column with respect to nondeterministic automata with the value function specified on the top row.

	Inf	Sup, LimInf, LimSup	LimInfAvg, LimSupAvg	DSum
Constructing *SafetyCl*($\mathcal{A}$)	$O(1)$	PTime Theorem 4.18		$O(1)$
Constant-function check	PSpace-complete Proposition 3.2 and Theorems 3.3 and 3.7			
Safety check	$O(1)$	PSpace-complete Theorem 4.22	ExpSpace; PSpace-hard Theorem 4.23 and Lemma 4.21	$O(1)$
Liveness check	PSpace-complete Theorem 5.9			
Safety-liveness decomposition	$O(1)$	PTime if deterministic Theorems 5.10 and 5.11	Open	$O(1)$

domains. The relations between these definitions were investigated in [25]. Notably, a notion of safety was studied for the rational-valued min-plus weighted automata on finite words in [38]. They take a weighted property as v-safe for a given rational v when for every execution w, if the hypothesis that the value of w is strictly less than v is wrong (i.e., its value is at least v), then there is a finite prefix of w to witness it. Then, a weighted property is safe when it is v-safe for *some* value v. Given a nondeterministic weighted automaton $\mathcal{A}$ and an integer v, they show that it is undecidable to check whether $\mathcal{A}$ is v-safe. In contrast, the definition in [25], which we follow, quantifies over *all* values and non-strict lower-bound hypotheses. Moreover, for this definition, we show that checking safety of all common classes of quantitative automata is decidable, even in the presence of nondeterminism. Finally, [4] studies the safety and co-safety of discounted-sum comparator automata. While these automata internally use discounted summation, they are boolean automata recognizing languages, and therefore they only consider boolean safety and co-safety.

Our study shows that determining whether a given quantitative automaton expresses a constant function is a key for deciding safety and liveness, in particular for automata classes in which equivalence or universality checks are undecidable. To the best of our knowledge, this problem has not been studied before.

2 Quantitative Properties and Automata

Let $\Sigma = \{a, b, \dots\}$ be a finite alphabet of letters (observations). An infinite (resp. finite) word is an infinite (resp. finite) sequence of letters $w \in \Sigma^\omega$ (resp. $u \in \Sigma^*$). For a natural number $n \in \mathbb{N}$, we denote by Σ^n the set of finite words of length n. Given $u \in \Sigma^*$ and $w \in \Sigma^* \cup \Sigma^\omega$, we write $u \prec w$ (resp. $u \preceq w$) when u is a strict (resp. nonstrict) prefix of w. We denote by $|w|$ the length of $w \in \Sigma^* \cup \Sigma^\omega$ and, given $a \in \Sigma$, by $|w|_a$ the number of occurrences of a in w. For $w \in \Sigma^* \cup \Sigma^\omega$ and $0 \leq i < |w|$, we denote by $w[i]$ the ith letter of w.

A *value domain* $\mathbb{D}$ is a poset. Unless otherwise stated, we assume that $\mathbb{D}$ is a nontrivial (i.e., $\bot \neq \top$) complete lattice. Whenever appropriate, we write 0 or $-\infty$ instead of $\bot$ for the least element, and 1 or ∞ instead of $\top$ for the greatest element. We respectively use the terms minimum and maximum for the greatest lower bound and the least upper bound of finitely many elements.

A *quantitative property* is a total function $\Phi : \Sigma^\omega \to \mathbb{D}$ from the set of infinite words to a value domain. A boolean property $P \subseteq \Sigma^\omega$ is a set of infinite words. We use the boolean domain $\mathbb{B} = \{0, 1\}$ with $0 < 1$ and, in place of P, its *characteristic property* $\Phi_P : \Sigma^\omega \to \mathbb{B}$, which is defined by $\Phi_P(w) = 1$ if $w \in P$, and $\Phi_P(w) = 0$ if $w \notin P$. When we say just *property*, we mean a quantitative one.

Given a property $\Phi : \Sigma^\omega \to \mathbb{D}$ and a value $v \in \mathbb{D}$, we define $\Phi_{\sim v} = \{w \in \Sigma^\omega \mid \Phi(w) \sim v\}$ for $\sim \in \{\leq, \geq, \not\leq, \not\geq\}$. The *top value* of a property Φ is $\sup_{w\in\Sigma^\omega} \Phi(w)$, which we denote by $\top_\Phi$, or simply $\top$ when Φ is clear from the context.

A *nondeterministic quantitative*[1] *automaton* (or just automaton from here on) on words is a tuple $\mathcal{A} = (\Sigma, Q, \iota, \delta)$, where Σ is an alphabet; Q is a finite nonempty set of states; $\iota \in Q$ is an initial state; and $\delta \colon Q \times \Sigma \to 2^{(\mathbb{Q}\times Q)}$ is a finite transition function over weight-state pairs. A *transition* is a tuple $(q, \sigma, x, q') \in Q\times\Sigma\times\mathbb{Q} \times Q$, such that $(x, q') \in \delta(q, \sigma)$, also written $q \xrightarrow{\sigma:x} q'$. (There might be finitely many transitions with different weights over the same letter between the same states.[2]) We write $\gamma(t) = x$ for the weight of a transition $t = (q, \sigma, x, q')$. $\mathcal{A}$ is deterministic if for all $q \in Q$ and $a \in \Sigma$, the set $\delta(q, a)$ is a singleton. We require the automaton $\mathcal{A}$ to be *total*, namely that for every state $q \in Q$ and letter $\sigma \in \Sigma$, there is at least one state q' and a transition $q \xrightarrow{\sigma:x} q'$. For a state $q \in Q$, we denote by $\mathcal{A}^q$ the automaton that is derived from $\mathcal{A}$ by setting its initial state ι to q.

A run of $\mathcal{A}$ on a word w is a sequence $\rho = q_0 \xrightarrow{w[0]:x_0} q_1 \xrightarrow{w[1]:x_1} q_2 \dots$ of transitions where $q_0 = \iota$ and $(x_i, q_{i+1}) \in \delta(q_i, w[i])$. For $0 \leq i < |w|$, we denote the ith transition in ρ by $\rho[i]$, and the finite prefix of ρ up to and including the ith transition by $\rho[..i]$. As each transition t_i carries a weight $\gamma(t_i) \in \mathbb{Q}$, the sequence ρ provides a weight sequence $\gamma(\rho) = \gamma(t_0)\gamma(t_1)\dots$ A Val (e.g., DSum) automaton is one equipped with a value function $\mathsf{Val} : \mathbb{Q}^\omega \to \mathbb{R}$, which assigns real values to runs of $\mathcal{A}$. We assume that Val is bounded for every finite set of rationals, i.e., for every finite $V \subset \mathbb{Q}$ there exist $m, M \in \mathbb{R}$ such that $m \leq \mathsf{Val}(x) \leq M$ for every $x \in V^\omega$. Note that the finite set V corresponds to transition weights of a quantitative automaton, and the concrete value functions we consider satisfy this assumption.

The value of a run ρ is $\mathsf{Val}(\gamma(\rho))$. The value of a Val-automaton $\mathcal{A}$ on a word w, denoted $\mathcal{A}(w)$, is the supremum of $\mathsf{Val}(\rho)$ over all runs ρ of $\mathcal{A}$ on w. The *top value* of a Val-automaton $\mathcal{A}$ is the top value of the property it expresses, which we denote by $\top_\mathcal{A}$, or simply $\top$ when $\mathcal{A}$ is clear from the context. Note that when we speak of the top value of a property or an automaton, we always match its value domain to have the same top value.

Two automata $\mathcal{A}$ and $\mathcal{A}'$ are *equivalent*, if they express the same function from words to reals. The size of an automaton consists of the maximum among the size of its alphabet, state-space, and transition-space, where weights are represented in binary.

We list below the value functions for quantitative automata that we will use, defined over infinite sequences $v_0 v_1 \dots$ of rational weights.

- $\mathsf{Inf}(v) = \inf\{v_n \mid n \geq 0\}$
- $\mathsf{Sup}(v) = \sup\{v_n \mid n \geq 0\}$
- $\mathsf{LimInf}(v) = \lim_{n\to\infty} \inf\{v_i \mid i \geq n\}$
- $\mathsf{LimSup}(v) = \lim_{n\to\infty} \sup\{v_i \mid i \geq n\}$
- $\mathsf{LimInfAvg}(v) = \mathsf{LimInf}\left(\frac{1}{n}\sum_{i=0}^{n-1} v_i\right)$
- $\mathsf{LimSupAvg}(v) = \mathsf{LimSup}\left(\frac{1}{n}\sum_{i=0}^{n-1} v_i\right)$
- For a discount factor $\lambda \in \mathbb{Q} \cap (0, 1)$, $\mathsf{DSum}_\lambda(v) = \sum_{i\geq 0} \lambda^i v_i$

Note that (i) when the discount factor $\lambda \in \mathbb{Q} \cap (0, 1)$ is unspecified, we write DSum, and (ii) $\mathsf{LimInfAvg}$ and $\mathsf{LimSupAvg}$ are also called $\underline{\mathsf{MeanPayoff}}$ and $\overline{\mathsf{MeanPayoff}}$ in the literature.

[1] We speak of "quantitative" rather than "weighted" automata, following the distinction made in [7] between the two.

[2] The flexibility of allowing "parallel" transitions with different weights is often omitted, as it is redundant for some value functions, including the ones we focus on in the sequel, while important for others.

The following statement allows us to consider Inf- and Sup-automata as only having runs with nonincreasing and nondecreasing, respectively, sequences of weights and to also consider them as LimInf- and LimSup-automata.

▶ **Proposition 2.1.** *Let* Val $\in \{\mathsf{Inf}, \mathsf{Sup}\}$*. Given a* Val*-automaton, we can construct in* PTime *an equivalent* Val-*,* LimInf- *or* LimSup*-automaton whose runs yield monotonic weight sequences.*

Given a property Φ and a finite word $u \in \Sigma^*$, let $P_{\Phi,u} = \{\Phi(uw) \mid w \in \Sigma^\omega\}$. A property Φ is sup-*closed* (resp. inf-*closed*) when for every finite word $u \in \Sigma^*$ we have that $\sup P_{\Phi,u} \in P_{\Phi,u}$ (resp. $\inf P_{\Phi,u} \in P_{\Phi,u}$) [25].

We show that the common classes of quantitative automata always express sup-closed properties, which will simplify the study of their safety and liveness.

▶ **Proposition 2.2.** *Let* Val $\in$ {Inf, Sup, LimInf, LimSup, LimInfAvg, LimSupAvg, DSum}*. Every* Val*-automaton expresses a property that is* sup*-closed. Furthermore its top value is rational, attainable by a run, and can be computed in* PTime*.*

3 Subroutine: Constant-Function Check

We will show that the problems of whether a given automaton is safe or live are closely related to the problem of whether an automaton expresses a constant function, motivating its study in this section. We first prove the problem hardness by reduction from the universality of nondeterministic finite-state automata (NFAs) and reachability automata.

▶ **Lemma 3.1.** *Let* Val $\in$ {Sup, Inf, LimInf, LimSup, LimInfAvg, LimSupAvg, DSum}*. It is* PSpace*-hard to decide whether a* Val*-automaton* $\mathcal{A}$ *expresses a constant function.*

A simple solution to the problem is to check whether the given automaton $\mathcal{A}$ is equivalent to an automaton $\mathcal{B}$ expressing the constant top value of $\mathcal{A}$, which is computable in PTime by Proposition 2.2. For some automata classes, it is good enough for a matching upper bound.

▶ **Proposition 3.2.** *Deciding whether an* Inf-*,* Sup-*,* LimInf-*, or* LimSup*-automaton expresses a constant function is* PSpace*-complete.*

Yet, this simple approach does not work for DSum-automata, whose equivalence is an open problem, and for limit-average automata, whose equivalence is undecidable [15].

For DSum-automata, our alternative solution removes "non-optimal" transitions from the automaton and then reduces the problem to the universality problem of NFAs.

▶ **Theorem 3.3.** *Deciding whether a* DSum*-automaton expresses a constant function is* PSpace*-complete.*

The solution for limit-average automata is more involved. It is based on a reduction to the limitedness problem of distance automata, which is known to be in PSpace [21, 36, 22, 31]. We start with presenting Johnson's algorithm, which we will use for manipulating the transition weights of the given automaton, and proving some properties of distance automata, which we will need for the reduction.

A *weighted graph* is a directed graph $G = \langle V, E\rangle$ equipped with a weight function $\gamma : E \to \mathbb{Z}$. The cost of a path $p = v_0, v_1, \ldots, v_k$ is $\gamma(p) = \sum_{i=0}^{k-1} \gamma(v_i, v_{i+1})$.

▶ **Proposition 3.4** (Johnson's Algorithm [28, Lem. 2 and Thms. 4 and 5])**.** *Consider a weighted graph* $G = \langle V, E\rangle$ *with weight function* $\gamma : E \to \mathbb{Z}$*, such that* G *has no negative cycles according to* γ*. We can compute in* PTime *functions* $h : V \to \mathbb{Z}$ *and* $\gamma' : E \to \mathbb{N}$ *such that for every path* $p = v_0, v_1, \ldots, v_k$ *in* G *it holds that* $\gamma'(p) = \gamma(p) + h(v_0) - h(v_k)$*.*

▶ Remark. Proposition 3.4 is stated for graphs, while we will apply it for graphs underlying automata, which are multi-graphs, namely having several transitions between the same pairs of states. Nevertheless, to see that Johnson's algorithm holds also in our case, one can change every automaton to an equivalent one whose underlying graph is a standard graph, by splitting every state into several states, each having a single incoming transition.

A *distance automaton* is a weighted automaton over the tropical semiring (a.k.a., min-plus semiring) with weights in $\{0, 1\}$. It can be viewed as a quantitative automaton over finite words with transition weights in $\{0, 1\}$ and the value function of summation, extended with accepting states. A distance automaton is of *limited distance* if there exists a bound on the automaton's values on all accepted words.

Lifting limitedness to infinite words, we have by König's lemma that a total distance automaton of limited distance b, in which all states are accepting, is also guaranteed to have a run whose weight summation is bounded by b on every infinite word.

▶ **Proposition 3.5.** *Consider a total distance automaton $\mathcal{D}$ of limited distance b, in which all states are accepting. Then for every infinite word w, there exists an infinite run of $\mathcal{D}$ on w whose summation of weights (considering only the transition weights and ignoring the final weights of states), is bounded by b.*

Lifting nonlimitedness to infinite words, it may not suffice for our purposes to have an infinite word on which all runs of the distance automaton are unbounded, as their limit-average value might still be 0. Yet, thanks to the following lemma, we are able to construct an infinite word on which the limit-average value is strictly positive.

▶ **Lemma 3.6.** *Consider a total distance automaton $\mathcal{D}$ of unlimited distance, in which all states are accepting. Then there exists a finite nonempty word u, such that $\mathcal{D}(u) = 1$ and the possible runs of $\mathcal{D}$ on u lead to a set of states U, such that the distance automaton that is the same as $\mathcal{D}$ but with U as the set of its initial states is also of unlimited distance.*

Using Propositions 3.4 and 3.5 and Lemma 3.6 we are in position to solve our problem by reduction to the limitedness problem of distance automata.

▶ **Theorem 3.7.** *Deciding whether a* LimInfAvg- *or* LimSupAvg-*automaton expresses a constant function, for a given constant or any constant, is PSPACE-complete.*

4 Quantitative Safety

The membership problem for quantitative properties asks, given a property $\Phi : \Sigma^\omega \to \mathbb{D}$, a word $w \in \Sigma^\omega$, and a value $v \in \mathbb{D}$, whether $\Phi(w) \geq v$ holds [10]. Safety of quantitative properties is defined from the perspective of membership queries [25]. Intuitively, a property is safe when each wrong membership hypothesis has a finite prefix to witness the violation. The safety closure of a given property maps each word to the greatest lower bound over its prefixes of the least upper bound of possible values.

▶ **Definition 4.1** (Safety [25]). *A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* safe *when for every $w \in \Sigma^\omega$ and value $v \in \mathbb{D}$ with $\Phi(w) \not\geq v$, there is a prefix $u \prec w$ such that $\sup_{w' \in \Sigma^\omega} \Phi(uw') \not\geq v$. The* safety closure *of a property Φ is the property defined by $SafetyCl(\Phi)(w) = \inf_{u \prec w} \sup_{w' \in \Sigma^\omega} \Phi(uw')$ for all $w \in \Sigma^\omega$.*

We remark that (i) a property is safe iff it defines the same function as its safety closure [25, Thm. 9], and (ii) the safety closure of a property is the least safety property that bounds the given property from above [25, Prop. 6]. Co-safety of quantitative properties and the co-safety closure is defined symmetrically.

▶ **Definition 4.2** (Co-safety [25]). *A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* co-safe *when for every $w \in \Sigma^\omega$ and value $v \in \mathbb{D}$ with $\Phi(w) \not\leq v$, there exists a prefix $u \prec w$ such that $\inf_{w' \in \Sigma^\omega} \Phi(uw') \not\leq v$. The* co-safety closure *of a property Φ is the property defined by $\mathit{CoSafetyCl}(\Phi)(w) = \sup_{u \prec w} \inf_{w' \in \Sigma^\omega} \Phi(uw')$ for all $w \in \Sigma^\omega$.*

Consider the case of a server that processes incoming requests and approves them accordingly. The quantitative property for the minimal response time of such a server is safe, while its maximal response time is co-safe [25, Examples 3 and 26]. Although these are sup- and inf-closed properties, safety and co-safety are independent of sup- and inf-closedness. To witness, consider the alphabet $\Sigma = \{a, b\}$ and the value domain $\mathbb{D} = \{x, y, \bot, \top\}$ where x and y are incomparable, and define $\Phi(w) = x$ if $a \prec w$ and $\Phi(w) = y$ if $b \prec w$.

▶ **Proposition 4.3.** *There is a property Φ that is safe and co-safe but neither* sup- *nor* inf*-closed.*

The Cantor space of infinite words is the set Σ^ω with the metric $\mu : \Sigma^\omega \times \Sigma^\omega \to [0, 1]$ such that $\mu(w, w) = 0$ and $\mu(w, w') = 2^{-|u|}$ where $u \in \Sigma^*$ is the longest common prefix of $w, w' \in \Sigma^\omega$ with $w \neq w'$. Given a boolean property $P \subseteq \Sigma^\omega$, the topological closure $\mathit{TopolCl}(P)$ of P is the smallest closed set (i.e., boolean safety property) that contains P, and the topological interior $\mathit{TopolInt}(P)$ of P is the greatest open set (i.e., boolean co-safety property) that is contained in P.

We show the connection between the quantitative safety (resp. co-safety) closure and the topological closure (resp. interior) through sup-closedness (resp. inf-closedness). Note that the sup-closedness assumption makes the quantitative safety closure values realizable. This guarantees that for every value v, every word whose safety closure value is at least v belongs to the topological closure of the set of words whose property values are at least v.

▶ **Theorem 4.4.** *Consider a property $\Phi : \Sigma^\omega \to \mathbb{D}$ and a threshold $v \in \mathbb{D}$. If Φ is* sup*-closed, then $(\mathit{SafetyCl}(\Phi))_{\geq v} = \mathit{TopolCl}(\Phi_{\geq v})$. If Φ is* inf*-closed, then $(\mathit{CoSafetyCl}(\Phi))_{\leq v} = \mathit{TopolInt}(\Phi_{\leq v})$.*

For studying the safety of automata, we first provide alternative characterizations of quantitative safety through threshold safety, which bridges the gap between the boolean and the quantitative settings, and continuity of functions. These hold for all properties on totally-ordered value domains, and in particular for those expressed by quantitative automata. Then, we extend the safety notions from properties to value functions, allowing us to characterize families of safe quantitative automata. Finally, we provide algorithms to construct the safety closure of a given automaton $\mathcal{A}$ and to decide whether $\mathcal{A}$ is safe.

4.1 Threshold Safety and Continuity

In this section, we define threshold safety to connect the boolean and the quantitative settings. It turns out that quantitative safety and threshold safety coincide on totally-ordered value domains. Furthermore, these value domains enable a purely topological characterization of quantitative safety properties in terms of their continuity.

▶ **Definition 4.5** (Threshold safety). *A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* threshold safe *when for every $v \in \mathbb{D}$ the boolean property $\Phi_{\geq v} = \{w \in \Sigma^\omega \mid \Phi(w) \geq v\}$ is safe (and thus $\Phi_{\not\geq v}$ is co-safe). Equivalently, for every $w \in \Sigma^\omega$ and $v \in \mathbb{D}$ if $\Phi(w) \not\geq v$ then there exists $u \prec w$ such that for all $w' \in \Sigma^\omega$ we have $\Phi(uw') \not\geq v$.*

▶ **Definition 4.6** (Threshold co-safety). *A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* threshold co-safe *when for every $v \in \mathbb{D}$ the boolean property $\Phi_{\not\leq v} = \{w \in \Sigma^\omega \mid \Phi(w) \not\leq v\}$ is co-safe (and thus $\Phi_{\leq v}$ is safe). Equivalently, for every $w \in \Sigma^\omega$ and $v \in \mathbb{D}$ if $\Phi(w) \not\leq v$ then there exists $u \prec w$ such that for all $w' \in \Sigma^\omega$ we have $\Phi(uw') \not\leq v$.*

In general, quantitative safety implies threshold safety, but the converse need not hold with respect to partially-ordered value domains. To witness, consider the value domain $\mathbb{D} = [0,1] \cup \{x\}$ where x is such that $0 < x$ and $x < 1$, but it is incomparable with all $v \in (0,1)$, while within $[0,1]$ there is the standard order. Let Φ be a property defined over $\Sigma = \{a,b\}$ as follows: $\Phi(w) = x$ if $w = a^\omega$, $\Phi(w) = 2^{-|w|_a}$ if $w \in \Sigma^* b^\omega$, and $\Phi(w) = 0$ otherwise. We show that Φ is threshold safe but not safe.

▶ **Proposition 4.7.** *Every safety property is threshold safe, but there is a threshold-safety property that is not safe.*

While for a fixed threshold, safety and threshold safety do not necessarily overlap even on totally-ordered domains, once quantifying over all thresholds, they do.

▶ **Theorem 4.8.** *Let $\mathbb{D}$ be a totally-ordered value domain. A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is safe iff it is threshold safe.*

We move next to the relation between safety and continuity. We recall some standard definitions; more about it can be found in textbooks, e.g., [19, 27]. A *topology* of a set X can be defined to be its collection τ of open subsets, and the pair (X, τ) stands for a *topological space.* It is *metrizable* when there exists a distance function (metric) d on X such that the topology induced by d on X is τ.

Recall that we take Σ^ω as a Cantor space with the metric μ defined as in Section 4. Consider a totally-ordered value domain $\mathbb{D}$. For each element $v \in \mathbb{D}$, let $L_v = \{v' \in \mathbb{D} \mid v' < v\}$ and $R_v = \{v' \in \mathbb{D} \mid v < v'\}$. The *order topology* on $\mathbb{D}$ is generated by the set $\{L_v \mid v \in \mathbb{D}\} \cup \{R_v \mid v \in \mathbb{D}\}$. Moreover, the *left order topology* (resp. *right order topology*) is generated by the set $\{L_v \mid v \in \mathbb{D}\}$ (resp. $\{R_v \mid v \in \mathbb{D}\}$). For a given property $\Phi : \Sigma^\omega \to \mathbb{D}$ and a set $V \subseteq \mathbb{D}$ of values, the *preimage* of V on Φ is defined as $\Phi^{-1}(V) = \{w \in \Sigma^\omega \mid \Phi(w) \in V\}$.

A property $\Phi : \Sigma^\omega \to \mathbb{D}$ on a topological space $\mathbb{D}$ is *continuous* when for every open subset $V \subseteq \mathbb{D}$ the preimage $\Phi^{-1}(V) \subseteq \Sigma^\omega$ is open. In [25, 26], a property Φ is defined as upper semicontinuous when $\Phi(w) = \lim_{u \prec w} \sup_{w' \in \Sigma^\omega} \Phi(uw')$, extending the standard definition for functions on extended reals to functions from infinite words to complete lattices. This characterizes safety properties since it is an equivalent condition to a property defining the same function as its safety closure [25, Thm. 9]. We complete the picture by providing a purely topological characterization of safety properties in terms of their continuity in totally-ordered value domains.

▶ **Theorem 4.9.** *Let $\mathbb{D}$ be a totally-ordered value domain. A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is safe (resp. co-safe) iff it is continuous with respect to the left (resp. right) order topology on $\mathbb{D}$.*

Observe that a property is continuous with respect to the order topology on $\mathbb{D}$ iff it is continuous with respect to both left and right order topologies on $\mathbb{D}$. Then, we immediately obtain the following.

▶ **Corollary 4.10.** *Let $\mathbb{D}$ be a totally-ordered value domain. A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is safe and co-safe iff it is continuous with respect to the order topology on $\mathbb{D}$.*

Now, we shift our focus to totally-ordered value domains whose order topology is metrizable. We provide a general definition of discounting properties on such domains.

▶ **Definition 4.11** (Discounting). *Let $\mathbb{D}$ be a totally-ordered value domain for which the order topology is metrizable with a metric d. A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* discounting *when for every $\varepsilon > 0$ there exists $n \in \mathbb{N}$ such that for every $u \in \Sigma^n$ and $w, w' \in \Sigma^\omega$ we have $d(\Phi(uw), \Phi(uw')) < \varepsilon$.*

Intuitively, a property is discounting when the range of potential values for every word converges to a singleton. As an example, consider the following discounted safety property: Given a boolean safety property P, let Φ be a quantitative property such that $\Phi(w) = 1$ if $w \in P$, and $\Phi(w) = 2^{-|u|}$ if $w \notin P$, where $u \prec w$ is the shortest bad prefix of w for P. We remark that our definition captures the previous definitions of discounting given in [14, 1].

▶ Remark. Notice that the definition of discounting coincides with uniform continuity. Since Σ^ω equipped with Cantor distance is a compact space [16], every continuous property is also uniformly continuous by Heine-Cantor theorem, and thus discounting.

As an immediate consequence, we obtain the following.

▶ **Corollary 4.12.** *Let $\mathbb{D}$ be a totally-ordered value domain for which the order topology is metrizable. A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is safe and co-safe iff it is discounting.*

4.2 Safety of Value Functions

In this section, we focus on the value functions of quantitative automata, which operate on the value domain of real numbers. In particular, we carry the definitions of safety, co-safety, and discounting to value functions. This allows us to characterize safe (resp. co-safe, discounting) value functions as those for which all automata with this value function are safe (resp. co-safe, discounting). Moreover, we characterize discounting value functions as those that are safe and co-safe.

Recall that we consider the value functions of quantitative automata to be bounded from below and above for every finite input domain $V \subset \mathbb{Q}$. As the set V^ω can be taken as a Cantor space, just like Σ^ω, we can carry the notions of safety, co-safety, and discounting from properties to value functions.

▶ **Definition 4.13** (Safety and co-safety of value functions). *A value function $\mathsf{Val} : \mathbb{Q}^\omega \to \mathbb{R}$ is* safe *when for every finite subset $V \subset \mathbb{Q}$, infinite sequence $x \in V^\omega$, and value $v \in \mathbb{R}$, if $\mathsf{Val}(x) < v$ then there exists a finite prefix $z \prec x$ such that $\sup_{y \in V^\omega} \mathsf{Val}(zy) < v$. Similarly, a value function $\mathsf{Val} : \mathbb{Q}^\omega \to \mathbb{R}$ is* co-safe *when for every finite subset $V \subset \mathbb{Q}$, infinite sequence $x \in V^\omega$, and value $v \in \mathbb{R}$, if $\mathsf{Val}(x) > v$ then there exists a finite prefix $z \prec x$ such that $\inf_{y \in V^\omega} \mathsf{Val}(zy) > v$.*

▶ **Definition 4.14** (Discounting value function). *A value function $\mathsf{Val} : \mathbb{Q}^\omega \to \mathbb{R}$ is* discounting *when for every finite subset $V \subset \mathbb{Q}$ and every $\varepsilon > 0$ there exists $n \in \mathbb{N}$ such that for every $x \in V^n$ and $y, y' \in V^\omega$ we have $|\mathsf{Val}(xy) - \mathsf{Val}(xy')| < \varepsilon$.*

We remark that by [25, Thms. 20 and 27], the value function Inf is safe and Sup is co-safe; moreover, the value function DSum is discounting by definition. Now, we characterize the safety (resp. co-safety) of a given value function by the safety (resp. co-safety) of the automata family it defines. We emphasize that the proofs of the two statement are not dual. In particular, exhibiting a finite set of weights that falsifies the safety of a value function from a nonsafe automaton requires a compactness argument.

▶ **Theorem 4.15.** *Consider a value function Val. All Val-automata are safe (resp. co-safe) iff Val is safe (resp. co-safe).*

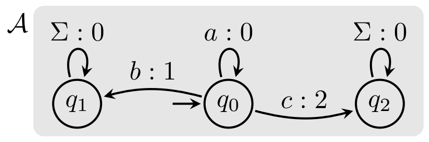

Figure 2 A Sup-automaton whose safety closure cannot be expressed by a Sup-automaton.

Thanks to the remark following Definition 4.11, for any finite set of weights $V \subset \mathbb{Q}$, a value function is discounting iff it is continuous on the Cantor space V^ω. We leverage this observation to characterize discounting value functions as those that are both safe and co-safe.

▶ **Theorem 4.16.** *A value function is discounting iff it is safe and co-safe.*

As a consequence of Corollary 4.12 and Theorems 4.15–4.16, we obtain the following.

▶ **Corollary 4.17.** *All* Val*-automata are discounting iff* Val *is discounting.*

4.3 Safety of Quantitative Automata

We now switch our focus from generic value functions to families of quantitative automata defined by the common value functions Inf, Sup, LimInf, LimSup, LimInfAvg, LimSupAvg, and DSum. As remarked in Section 4.2, the value functions Inf and DSum are safe, thus all Inf-automata and DSum-automata express a safety property by Theorem 4.15. Below, we focus on the remaining value functions of interest.

Given a Val-automaton $\mathcal{A}$ where Val is one of the nonsafe value functions above, we describe (i) a construction of an automaton that expresses the safety closure of $\mathcal{A}$, and (ii) an algorithm to decide whether $\mathcal{A}$ is safe.

For these value functions, we can construct the safety closure as an Inf-automaton.

▶ **Theorem 4.18.** *Let* $\mathsf{Val} \in \{\mathsf{Sup}, \mathsf{LimInf}, \mathsf{LimSup}, \mathsf{LimInfAvg}, \mathsf{LimSupAvg}\}$*. Given a* Val*-automaton* $\mathcal{A}$*, we can construct in* PTime *an* Inf*-automaton that expresses its safety closure.*

For the prefix-independent value functions we study, the safety-closure automaton we construct in Theorem 4.18 can be taken as a deterministic automaton with the same value function.

▶ **Theorem 4.19.** *Let* $\mathsf{Val} \in \{\mathsf{LimInf}, \mathsf{LimSup}, \mathsf{LimInfAvg}, \mathsf{LimSupAvg}\}$*. Given a* Val*-automaton* $\mathcal{A}$*, we can construct in* PTime *a* Val*-automaton that expresses its safety closure and can be determinized in* ExpTime*.*

In contrast, this is not possible in general for Sup-automata, as Figure 2 witnesses.

▶ **Proposition 4.20.** *Some* Sup*-automaton admits no* Sup*-automata that expresses its safety closure.*

We first prove the problem hardness by reduction from constant-function checks.

▶ **Lemma 4.21.** *Let* $\mathsf{Val} \in \{\mathsf{Sup}, \mathsf{LimInf}, \mathsf{LimSup}, \mathsf{LimInfAvg}, \mathsf{LimSupAvg}\}$*. It is* PSpace*-hard to decide whether a* Val*-automaton is safe.*

For automata classes with PSpace equivalence check, a matching upper bound is straightforward by comparing the given automaton and its safety-closure automaton.

▶ **Theorem 4.22.** *Deciding whether a* Sup-, LimInf-, *or* LimSup-*automaton expresses a safety property is* PSPACE*-complete.*

On the other hand, even though equivalence of limit-average automata is undecidable [15], we are able to provide a decision procedure using as a subroutine our algorithm to check whether a given limit-average automaton expresses a constant function (see Theorem 3.7). The key idea is to construct a limit-average automaton that expresses the constant function 0 iff the original automaton is safe. Our approach involves the determinization of the safety-closure automaton, resulting in an ExpSpace complexity.

▶ **Theorem 4.23.** *Deciding whether a* LimInfAvg- *or* LimSupAvg-*automaton expresses a safety property is in* ExpSpace*.*

5 Quantitative Liveness

The definition of quantitative liveness, similarly to that of quantitative safety, comes from the perspective of the quantitative membership problem [25]. Intuitively, a property is live when for every word whose value is less than the top, there is a wrong membership hypothesis without a finite prefix to witness the violation.

▶ **Definition 5.1** (Liveness and co-liveness [25]). *A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* live *when for all $w \in \Sigma^\omega$, if $\Phi(w) < \top$, then there exists a value $v \in \mathbb{D}$ such that $\Phi(w) \not\geq v$ and for all prefixes $u \prec w$, we have $\sup_{w' \in \Sigma^\omega} \Phi(uw') \geq v$. Similarly, a property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* co-live *when for all $w \in \Sigma^\omega$, if $\Phi(w) > \bot$, then there exists a value $v \in \mathbb{D}$ such that $\Phi(w) \not\leq v$ and for all prefixes $u \prec w$, we have $\inf_{w' \in \Sigma^\omega} \Phi(uw') \leq v$.*

As an example, consider a server that receives requests and issues grants. The server's maximum response time is live, while its minimum response time is co-live, and its average response time is both live and co-live [25, Examples 41 and 42].

First, we provide alternative characterizations of quantitative liveness for sup-closed properties by threshold liveness, which bridges the gap between the boolean and the quantitative settings, and top liveness. Then, we provide algorithms to check liveness of quantitative automata, and to decompose them into a safety automaton and a liveness automaton.

5.1 Threshold Liveness and Top Liveness

Threshold liveness connects a quantitative property and the boolean liveness of the sets of words whose values exceed a threshold value.

▶ **Definition 5.2** (Threshold liveness and co-liveness). *A property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* threshold live *when for every $v \in \mathbb{D}$ the boolean property $\Phi_{\geq v} = \{w \in \Sigma^\omega \mid \Phi(w) \geq v\}$ is live (and thus $\Phi_{\not\geq v}$ is co-live). Equivalently, Φ is threshold live when for every $u \in \Sigma^*$ and $v \in \mathbb{D}$ there exists $w \in \Sigma^\omega$ such that $\Phi(uw) \geq v$. Similarly, a property $\Phi : \Sigma^\omega \to \mathbb{D}$ is* threshold co-live *when for every $v \in \mathbb{D}$ the boolean property $\Phi_{\not\leq v} = \{w \in \Sigma^\omega \mid \Phi(w) \not\leq v\}$ is co-live (and thus $\Phi_{\leq v}$ is live). Equivalently, Φ is threshold co-live when for every $u \in \Sigma^*$ and $v \in \mathbb{D}$ there exists $w \in \Sigma^\omega$ such that $\Phi(uw) \leq v$.*

A set $P \subseteq \Sigma^\omega$ is dense in Σ^ω when its topological closure equals Σ^ω, i.e., $\mathit{TopolCl}(P) = \Sigma^\omega$. We relate threshold liveness with the topological denseness of a single set of words.

▶ **Proposition 5.3.** *A property Φ is threshold live iff the set $\{w \in \Sigma^\omega \mid \Phi(w) = \top\}$ is dense in Σ^ω.*

Liveness is characterized by the safety closure being strictly greater than the property whenever possible [25, Thm. 37]. Top liveness puts an additional requirement on liveness: the safety closure of the property should not only be greater than the original property but also equal to the top value.

▶ **Definition 5.4** (Top liveness and bottom co-liveness). *A property Φ is* top live *when* $\mathit{SafetyCl}(\Phi)(w) = \top$ *for every* $w \in \Sigma^\omega$. *Similarly, a property Φ is* bottom co-live *when* $\mathit{CoSafetyCl}(\Phi)(w) = \bot$ *for every* $w \in \Sigma^\omega$.

We provide a strict hierarchy of threshold-liveness, top-liveness, and liveness.

▶ **Proposition 5.5.** *Every threshold-live property is top live, but not vice versa; and every top-live property is live, but not vice versa.*

Top liveness does not imply threshold liveness, but it does imply a weaker form of it.

▶ **Proposition 5.6.** *For every top-live property Φ and value $v < \top$, the set $\Phi_{\geq v}$ is live in the boolean sense.*

While the three liveness notions differ in general, they do coincide for sup-closed properties.

▶ **Theorem 5.7.** *A* sup*-closed property is live iff it is top live iff it is threshold live.*

5.2 Liveness of Quantitative Automata

We start with the problem of checking whether a quantitative automaton is live, and continue with quantitative safety-liveness decomposition.

We first provide a hardness result by reduction from constant-function checks.

▶ **Lemma 5.8.** *Let* $\mathsf{Val} \in \{\mathsf{Inf}, \mathsf{Sup}, \mathsf{LimInf}, \mathsf{LimSup}, \mathsf{LimInfAvg}, \mathsf{LimSupAvg}, \mathsf{DSum}\}$. *It is PSpace-hard to decide whether a* Val*-automaton $\mathcal{A}$ is live.*

For automata classes whose safety closure can be expressed as Inf-automata, we provide a matching upper bound by simply checking the universality of the safety closure with respect to its top value. For DSum-automata, whose universality problem is open, our solution is based on our constant-function-check algorithm (see Theorem 3.3).

▶ **Theorem 5.9.** *Deciding whether an* Inf-*,* Sup-*,* LimInf-*,* LimSup-*,* LimInfAvg-*,* LimSupAvg- *or* DSum*-automaton expresses a liveness property is PSpace-complete.*

We turn to safety-liveness decomposition, and start with the simple case of Inf- and DSum-automata, which are guaranteed to be safe. Their decomposition thus consists of only generating a liveness component, which can simply express a constant function that is at least as high as the maximal possible value of the original automaton $\mathcal{A}$. Assuming that the maximal transition weight of $\mathcal{A}$ is fixed, it can be done in constant time.

Considering Sup-automata, recall that their safety closure might not be expressible by Sup-automata (Proposition 4.20). Therefore, our decomposition of deterministic Sup-automata takes the safety component as an Inf-automaton. The key idea is to copy the state space of the original automaton and manipulate the transition weights depending on how they compare with the safety-closure automaton.

▶ **Theorem 5.10.** *Given a deterministic* Sup*-automaton $\mathcal{A}$, we can construct in PTime a deterministic safety* Inf*-automaton $\mathcal{B}$ and a deterministic liveness* Sup*-automaton $\mathcal{C}$, such that $\mathcal{A}(w) = \min(\mathcal{B}(w), \mathcal{C}(w))$ for every infinite word w.*

Using the same idea, but with a slightly more involved reasoning, we show a safety-liveness decomposition for deterministic LimInf- and LimSup-automata.

▶ **Theorem 5.11.** *Let* Val $\in$ {LimInf, LimSup}. *Given a deterministic* Val*-automaton* $\mathcal{A}$*, we can construct in* PTIME *a deterministic safety* Val*-automaton* $\mathcal{B}$ *and a deterministic liveness* Val*-automaton* $\mathcal{C}$*, such that* $\mathcal{A}(w) = \min(\mathcal{B}(w), \mathcal{C}(w))$ *for every infinite word* w*.*

Considering nondeterministic Sup- and LimInf-automata, they can be decomposed by first determinizing them at an exponential cost [10, Thm. 14]. For nondeterministic LimSup-automata, which cannot always be determinized, we leave the problem open. We also leave open the question of whether LimInfAvg- and LimSupAvg-automata are closed under safety-liveness decomposition.

6 Conclusions

We studied, for the first time, the quantitative safety-liveness dichotomy for properties expressed by Inf, Sup, LimInf, LimSup, LimInfAvg, LimSupAvg, and DSum automata. To this end, we characterized the quantitative safety and liveness of automata by their boolean counterparts, connected them to topological continuity and denseness, and solved the constant-function problem for these classes of automata. We presented automata-theoretic constructions for the safety closure of these automata and decision procedures for checking their safety and liveness. We proved that the value function Inf yields a class of safe automata and DSum both safe and co-safe. For some automata classes, we provided a decomposition of an automaton into a safe and a live component. We emphasize that the safety component of our decomposition algorithm is the safety closure, and thus the best safe approximation of a given automaton.

We focused on quantitative automata [10] because their totally-ordered value domain and their sup-closedness make quantitative safety and liveness behave in particularly natural ways; a corresponding investigation of weighted automata [35] remains to be done. We left open the problems of the safety-liveness decomposition of limit-average automata, the complexity gap in the safety check of limit-average automata, and the study of co-safety and co-liveness for nondeterministic quantitative automata, which is not symmetric to safety and liveness due to the nonsymmetry in resolving nondeterminism by the supremum value of all possible runs.

References

1 Shaull Almagor, Udi Boker, and Orna Kupferman. Discounting in LTL. In Erika Ábrahám and Klaus Havelund, editors, *Tools and Algorithms for the Construction and Analysis of Systems – 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8413 of *Lecture Notes in Computer Science*, pages 424–439. Springer, 2014. doi:10.1007/978-3-642-54862-8_37.

2 Bowen Alpern and Fred B. Schneider. Defining liveness. *Inf. Process. Lett.*, 21(4):181–185, 1985. doi:10.1016/0020-0190(85)90056-0.

3 Bowen Alpern and Fred B. Schneider. Recognizing safety and liveness. *Distributed Comput.*, 2(3):117–126, 1987. doi:10.1007/BF01782772.

4 Suguman Bansal and Moshe Y. Vardi. Safety and co-safety comparator automata for discounted-sum inclusion. In Isil Dillig and Serdar Tasiran, editors, *Computer Aided Verification – 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I*, volume 11561 of *Lecture Notes in Computer Science*, pages 60–78. Springer, 2019. doi:10.1007/978-3-030-25540-4_4.

5 Roderick Bloem, Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Better quality in synthesis through quantitative objectives. In Ahmed Bouajjani and Oded Maler, editors, *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 – July 2, 2009. Proceedings*, volume 5643 of *Lecture Notes in Computer Science*, pages 140–156. Springer, 2009. doi:10.1007/978-3-642-02658-4_14.

6 Roderick Bloem, Krishnendu Chatterjee, and Barbara Jobstmann. Graph games and reactive synthesis. In Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem, editors, *Handbook of Model Checking*, pages 921–962. Springer, 2018. doi:10.1007/978-3-319-10575-8_27.

7 Udi Boker. Quantitative vs. weighted automata. In *Proc. of Reachbility Problems*, pages 1–16, 2021.

8 Udi Boker and Karoliina Lehtinen. Token games and history-deterministic quantitative automata. In Patricia Bouyer and Lutz Schröder, editors, *Foundations of Software Science and Computation Structures – 25th International Conference, FOSSACS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*, volume 13242 of *Lecture Notes in Computer Science*, pages 120–139. Springer, 2022. doi:10.1007/978-3-030-99253-8_7.

9 Patricia Bouyer, Nicolas Markey, Mickael Randour, Kim G. Larsen, and Simon Laursen. Average-energy games. *Acta Informatica*, 55(2):91–127, 2018. doi:10.1007/s00236-016-0274-1.

10 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Quantitative languages. *ACM Trans. Comput. Log.*, 11(4):23:1–23:38, 2010. doi:10.1145/1805950.1805953.

11 Krishnendu Chatterjee, Thomas A. Henzinger, and Jan Otop. Quantitative monitor automata. In Xavier Rival, editor, *Static Analysis – 23rd International Symposium, SAS 2016, Edinburgh, UK, September 8-10, 2016, Proceedings*, volume 9837 of *Lecture Notes in Computer Science*, pages 23–38. Springer, 2016. doi:10.1007/978-3-662-53413-7_2.

12 Loris D'Antoni, Roopsha Samanta, and Rishabh Singh. Qlose: Program repair with quantitative objectives. In Swarat Chaudhuri and Azadeh Farzan, editors, *Computer Aided Verification – 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17-23, 2016, Proceedings, Part II*, volume 9780 of *Lecture Notes in Computer Science*, pages 383–401. Springer, 2016. doi:10.1007/978-3-319-41540-6_21.

13 Luca de Alfaro, Marco Faella, and Mariëlle Stoelinga. Linear and branching metrics for quantitative transition systems. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings*, volume 3142 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 2004. doi:10.1007/978-3-540-27836-8_11.

14 Luca de Alfaro, Thomas A. Henzinger, and Rupak Majumdar. Discounting the future in systems theory. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands, June 30 – July 4, 2003. Proceedings*, volume 2719 of *Lecture Notes in Computer Science*, pages 1022–1037. Springer, 2003. doi:10.1007/3-540-45061-0_79.

15 Aldric Degorre, Laurent Doyen, Raffaella Gentilini, Jean-François Raskin, and Szymon Torunczyk. Energy and mean-payoff games with imperfect information. In Anuj Dawar and Helmut Veith, editors, *Computer Science Logic, 24th International Workshop, CSL 2010, 19th Annual Conference of the EACSL, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6247 of *Lecture Notes in Computer Science*, pages 260–274. Springer, 2010. doi:10.1007/978-3-642-15205-4_22.

16 Volker Diekert and Martin Leucker. Topology, monitorable properties and runtime verification. *Theor. Comput. Sci.*, 537:29–41, 2014. doi:10.1016/j.tcs.2014.02.052.

17 Uli Fahrenberg. A generic approach to quantitative verification. *CoRR*, abs/2204.11302, 2022. doi:10.48550/arXiv.2204.11302.

18 Thomas Ferrère, Thomas A. Henzinger, and N. Ege Saraç. A theory of register monitors. In Anuj Dawar and Erich Grädel, editors, *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 394–403. ACM, 2018. doi:10.1145/3209108.3209194.

19 Theodore W Gamelin and Robert Everist Greene. *Introduction to topology*. Courier Corporation, 1999.

20 Felipe Gorostiaga and César Sánchez. Monitorability of expressive verdicts. In Jyotirmoy V. Deshmukh, Klaus Havelund, and Ivan Perez, editors, *NASA Formal Methods – 14th International Symposium, NFM 2022, Pasadena, CA, USA, May 24-27, 2022, Proceedings*, volume 13260 of *Lecture Notes in Computer Science*, pages 693–712. Springer, 2022. doi:10.1007/978-3-031-06773-0_37.

21 K. Hashiguchi. Limitedness theorem on finite automata with distance functions. *Journal of computer and system sciences*, 24(2):233–244, 1982.

22 K. Hashiguchi. New upper bounds to the limitedness of distance automata. *Theoretical Computer Science*, 233(1-2):19–32, 2000.

23 Thomas A. Henzinger. Quantitative reactive modeling and verification. *Comput. Sci. Res. Dev.*, 28(4):331–344, 2013. doi:10.1007/s00450-013-0251-7.

24 Thomas A. Henzinger, Nicolas Mazzocchi, and N. Ege Saraç. Abstract monitors for quantitative specifications. In Thao Dang and Volker Stolz, editors, *Runtime Verification – 22nd International Conference, RV 2022, Tbilisi, Georgia, September 28-30, 2022, Proceedings*, volume 13498 of *Lecture Notes in Computer Science*, pages 200–220. Springer, 2022. doi:10.1007/978-3-031-17196-3_11.

25 Thomas A. Henzinger, Nicolas Mazzocchi, and N. Ege Saraç. Quantitative safety and liveness. In Orna Kupferman and Pawel Sobocinski, editors, *Foundations of Software Science and Computation Structures – 26th International Conference, FoSSaCS 2023, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2023, Paris, France, April 22-27, 2023, Proceedings*, volume 13992 of *Lecture Notes in Computer Science*, pages 349–370. Springer, 2023. doi:10.1007/978-3-031-30829-1_17.

26 Thomas A. Henzinger and N. Ege Saraç. Quantitative and approximate monitoring. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 – July 2, 2021*, pages 1–14. IEEE, 2021. doi:10.1109/LICS52264.2021.9470547.

27 Hendrik Jan Hoogeboom and Grzegorz Rozenberg. Infinitary languages: Basic theory an applications to concurrent systems. In J. W. de Bakker, Willem P. de Roever, and Grzegorz Rozenberg, editors, *Current Trends in Concurrency, Overviews and Tutorials*, volume 224 of *Lecture Notes in Computer Science*, pages 266–342. Springer, 1986. doi:10.1007/BFb0027043.

28 D. B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM (JACM)*, 24(1):1–13, 1977.

29 Orna Kupferman and Moshe Y. Vardi. Model checking of safety properties. *Formal Methods Syst. Des.*, 19(3):291–314, 2001. doi:10.1023/A:1011254632723.

30 Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Trans. Software Eng.*, 3(2):125–143, 1977. doi:10.1109/TSE.1977.229904.

31 H. Leung and V. Podolskiy. The limitedness problem on distance automata: Hashiguchi's method revisited. *Theoretical Computer Science*, 310(1-3):147–158, 2004.

32 Yongming Li, Manfred Droste, and Lihui Lei. Model checking of linear-time properties in multi-valued systems. *Inf. Sci.*, 377:51–74, 2017. doi:10.1016/j.ins.2016.10.030.

33 Dejan Nickovic, Olivier Lebeltel, Oded Maler, Thomas Ferrère, and Dogan Ulus. AMT 2.0: qualitative and quantitative trace analysis with extended signal temporal logic. *Int. J. Softw. Tools Technol. Transf.*, 22(6):741–758, 2020. doi:10.1007/s10009-020-00582-z.

34 Doron Peled and Klaus Havelund. Refining the safety-liveness classification of temporal properties according to monitorability. In Tiziana Margaria, Susanne Graf, and Kim G. Larsen, editors, *Models, Mindsets, Meta: The What, the How, and the Why Not? – Essays Dedicated to Bernhard Steffen on the Occasion of His 60th Birthday*, volume 11200 of *Lecture Notes in Computer Science*, pages 218–234. Springer, 2018. doi:10.1007/978-3-030-22348-9_14.

35 Marcel Paul Schützenberger. On the definition of a family of automata. *Inf. Control.*, 4(2-3):245–270, 1961. doi:10.1016/S0019-9958(61)80020-X.

36 I. Simon. On semigroups of matrices over the tropical semiring. *RAIRO-Theoretical Informatics and Applications*, 28(3-4):277–294, 1994.

37 A. Prasad Sistla. Safety, liveness and fairness in temporal logic. *Formal Aspects Comput.*, 6(5):495–512, 1994. doi:10.1007/BF01211865.

38 Sigal Weiner, Matan Hasson, Orna Kupferman, Eyal Pery, and Zohar Shevach. Weighted safety. In Dang Van Hung and Mizuhito Ogawa, editors, *Automated Technology for Verification and Analysis – 11th International Symposium, ATVA 2013, Hanoi, Vietnam, October 15-18, 2013. Proceedings*, volume 8172 of *Lecture Notes in Computer Science*, pages 133–147. Springer, 2013. doi:10.1007/978-3-319-02444-8_11.