Monitoring Event Frequencies

Thomas Ferrère
IST Austria

Thomas A. Henzinger
IST Austria

Bernhard Kragl
IST Austria

Abstract

The monitoring of event frequencies can be used to recognize behavioral anomalies, to identify trends, and to deduce or discard hypotheses about the underlying system. For example, the performance of a web server may be monitored based on the ratio of the total count of requests from the least and most active clients. Exact frequency monitoring, however, can be prohibitively expensive; in the above example it would require as many counters as there are clients. In this paper, we propose the efficient probabilistic monitoring of common frequency properties, including the mode (i.e., the most common event) and the median of an event sequence. Our main contribution is an algorithm that, under suitable probabilistic assumptions, can be used to monitor these important frequency properties with four counters, independent of the number of different events. Our algorithm samples longer and longer subwords of an infinite event sequence. We prove the almost-sure convergence of our algorithm by generalizing ergodicity theory from increasing-length prefixes to increasing-length subwords of an infinite sequence. A similar algorithm could be used to learn a connected Markov chain of a given structure from observing its outputs, to arbitrary precision, for a given confidence.

2012 ACM Subject Classification Software and its engineering → Software organization and properties; Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases monitoring, frequency property, Markov chain

1 Introduction

The safety and security of computerized systems are ensured by a chain of methods that enables via the use of logic and formal semantics to assert and check the correct operation of system, real or simulated. Runtime monitoring [4] happens at the end of this chain and is employed as a complement to rigorous design and verification practices to catch malfunctions as they occur in a live system. In addition to critical functional aspects, softer performance metrics also need to be monitored to ensure a suitable quality of service. Monitoring system properties takes place in parallel with the execution of the system itself. A dedicated component, called monitor, is employed to observe the system behavior as input and generate a verdict about the system behavior as output. Due to reactivity considerations, the monitor is often required to perform its observations in real-time, and not being the main computational artifact, should consume limited resources.

In this paper we propose a formal language for describing quantitative properties based on frequencies, and study study their monitoring problem. While all such *frequency properties* are theoretically monitorable using counter registers, we do not know of efficient algorithms in the case of large or infinite input alphabets. As a motivating example we use the *mode* of a sequence over a finite alphabet Σ. By definition, $a \in \Sigma$ is the mode of an ω-word w if there exists a length n such that each prefix of w longer than n contains more occurrences of a's than occurrences of any other letter $b \in \Sigma$. This frequency property can be monitored using a separate counter for every event in Σ. However the alphabet Σ is typically too large for this to be a practical model.[1] We show that there is no shortcut to monitor the mode exactly and in real time: in general $|\Sigma|$ counters are needed for this task.

However, we are not always interested in monitoring exactly and in real time the mode after every new event, and sometimes wish to estimate what the mode is expected to be in the future. Perhaps surprisingly, we can then do much better. Let us assume that the past, finite, observed behavior of an event sequence is representative of the future, infinite, unknown behavior. This is the case for stochastic systems, for instance if the observation sequence is generated by a Markov chain. We move from the *real-time monitoring* problem, asking to compute or approximate, in real time, the value of a frequency property for each observed prefix, to the *limit monitoring* problem, asking to estimate the future limit value of the frequency property, if it exists. In particular, for the mode of a connected Markov chain, the longer we observe a behavior, the higher our confidence in predicting its mode. While every real-time monitor can be used as limit monitor, there can be limit monitors that use dramatically fewer resources.

We present a simple, memory-efficient strategy to limit monitor frequency properties of random ω-words. In particular, our mode monitor uses four counters only. Two of the counters keep track of the number of occurrences of two letters at a time. The first letter is the current mode prediction, say a. The second letter is the mode replacement candidate, say b. We count the number of a's and b's over a given subword, until a certain number of events, say 10, has been processed. The most frequent letter out of a and b in this 10-letter subword, say a, wins the round and becomes the new mode prediction. The other letter loses the round and is replaced by a letter sampled at random, say c. In the next round the subword length will be increased, say to 11, and a will compete against c over the next subword. We reuse two counters for the two letters, and the other two counters to keep track of the current subword length and to stop counting when that length is reached. By repeating the process we get increasingly higher confidence that a is indeed the mode. Even if by random perturbation the mode a of the generating Markov chain was no longer the current prediction, it would eventually get sampled again and statistically reappear, and eventually remain, as the prediction.

The algorithm of our mode monitor easily transfers to an efficient monitor for the median. Indeed, we also show that our results generalize to any property expressible as Boolean combination of linear inequalities over frequencies of events. An application of our algorithmic ideas is to learn the transition probabilities of a connected Markov chain of known structure through the observation of subword frequencies.

The main result of this paper is that, assuming the monitored system is a connected Markov chain, our monitoring algorithm converges almost surely. The proof of this fact calls for a new ergodic theory based on subwords as opposed to prefixes. This theory uses as its main building block a variant of the law of large numbers over so-called triangular random arrays of the form $X_{1,1}, X_{2,1}, X_{2,2}, X_{3,1}, \ldots$ and hinges on deep results from matrix theory. The correctness of the algorithm can also be understood, in a weaker form, by showing convergence in probability of its output. Assuming that the Markov chain starts in a stationary distribution, the probability of a given word u occurring as subword of an ω-word w at position i is independent of i. As a result, when the value of a function over prefixes converges probabilistically, then the same limit is reached probabilistically over arbitrary subwords.

In short, the main conceptual and technical contributions of this paper are the following:

1. We propose the novel setting of limit monitoring (Section 3).

[1] Consider the IPv4 protocol alphabet with its 4,294,967,296 letters (addresses) and the UTF-8 encoding alphabet with its 1,112,064 letters (code points).

2. We provide a generic scheme for efficient limit monitoring (Section 5) and instantiate it to specialized monitoring algorithms for the mode (Section 5.2) and median (Section 5.3), as well as a general class of frequency properties (Section 6).
3. We develop a new ergodicity theory for connected Markov chains (Section 5.1) to prove our monitoring algorithms correct.

1.1 Related Work

In the area of formal verification, probabilistic model checking [14, 15] and quantitative verification [11] are concerned with the white-box static analysis of a probabilistic system. Statistical model checking [1] tries to learn the probabilistic structure of a system by sampling *many* executions, and thus also applies to black-box systems. These are in contrast to our monitoring setting where a *single* execution of a black-box system is dynamically observed during execution. Our work belongs specifically to the field of runtime verification [4], which is concerned with the evaluation of temporal properties over program traces. While much of the research in this domain assumes finite-state monitors, in this work we study an infinite-state problem based on the model of counter monitors. The expressiveness of different register machines and resource trade-offs for monitoring safety properties involving counters and arithmetic registers is studied in [9]. Another infinite-state model for monitoring is that of quantified event automata [3], which combine finite automata specifications with first-order quantification. Other quantitative automata machines are surveyed in [7].

The computation of aggregates over an ongoing system execution in real-time was considered in various areas of computer science. Stream expressions [8] and quantitative regular expressions [2] provide frameworks for the specification of transducers over data streams. The work on runtime verification and stream processing can be seen as solving real-time monitoring problems, and very rarely assumes a probabilistic model. A notable exception can be found in [21], who propose to use hypothesis testing to provide an interval of confidence on the monitor outcome when evaluating some probabilistic property. In the vast literature from runtime verification to online algorithms, the problem of limit monitoring as defined, solved, and applied in this paper was, to the best of our knowledge, not studied before.

It is well-known that certain common statistical indicators can be computed in real time. For example, the average can be computed by simply maintaining the sum and sample size. Perhaps more surprisingly, the variance and covariance of a sequence can also be computed in one pass through classical online algorithms [23]. However, other indicators, like the median, are hard or impossible to compute in real time. Offline algorithms for the median include selection algorithms (e.g., quickselect [12]) with $O(n)$ run-time (versus $O(n \log n)$ for sorting), median of medians [5] (which is approximate), and the randomized algorithm of Mitzenmacher & Upfal [17]. The best known online algorithm uses two heaps to store the lower and higher half of values (i.e., all samples have to be stored), with an amortized cost of $O(\log n)$ per input. To the best of our knowledge, no real-time algorithm to compute the median exactly was proposed in the literature.

Statistical properties of subword frequencies in Markov chains are studied in [6]. In Markov chain theory, the existence, uniqueness, and convergence results for stationary distributions are among the most fundamental results [18]. The rate of convergence towards a stationary distribution is called mixing time [16]. In general, the mixing time is controlled by the spectral gap of the transition matrix, with precise results only know for particular random processes, like card shuffling. These result do not lead to bounds on the convergence rate of frequencies of events in labeled Markov chains.

An indirect (and somewhat degenerate) approach to monitoring would be to first learn the monitored system, and then perform offline verification on the learned model. Learning probabilistic generators was studied in the setting of automata learning [19], but requires more powerful oracle queries like membership and equivalence. Rudich showed that the structure and transition probabilities of a Markov chain can, in principle, be learned from a single input sequence [20]. However, the algorithm is impractical as it essentially enumerates all possible structures.

2 Definitions

Let Σ be a finite alphabet of events. Given a finite or infinite word or ω-word $w \in \Sigma^* \cup \Sigma^\omega$ and a position i, $1 \leq i \leq |w|$, we denote by w_i its i'th value. Given a pair of positions i and j, $i \leq j$, we denote by $w_{i..j}$ the *infix* of w from i to j, such that $|w_{i..j}| = j - i + 1$ and $(w_{i..j})_k = w_{i+k-1}$ for all $1 \leq k \leq j - i + 1$. We denote by $w_{..i} = w_{1..i}$ the *prefix* of w of length i. For any word $w \in \Sigma^*$ and letter $a \in \Sigma$ we write $|w|_a$ for the number of occurrences of a in w.

2.1 Sequential Statistics

We define a statistic to be any function that outputs an indicator for a given input word.

▶ **Definition 1** (Statistic). *Let Σ be a finite alphabet and Λ be an output domain. A* statistic *is a function $\mu : \Sigma^* \to \Lambda$.*

In this paper we focus on statistics that are based on the frequency, or number of occurrence, of events. Two typical examples are the *mode*, i.e. the most frequent event, and the *median*, i.e., the value separating as evenly as possible the upper half from the lower half of a data sample.

▶ **Example 2** (Mode). We say that $a \in \Sigma$ is the *mode* of w when $|w|_a > |w|_\sigma$ for all $\sigma \in \Sigma \setminus \{a\}$. We denote by $\mathsf{mode} : \Sigma^* \to \Sigma \uplus \{\bot\}$ the statistic that maps a word to its mode if it exists, or to $\bot$ otherwise.

▶ **Example 3** (Median). Let Σ be ordered by $\prec$. We say that $a \in \Sigma$ is the *median* of w when $\sum_{\sigma \succ a} |w|_\sigma < \sum_{\sigma \preccurlyeq a} |w|_\sigma$ and $\sum_{\sigma \prec a} |w|_\sigma < \sum_{\sigma \succcurlyeq a} |w|_\sigma$. We denote by $\mathsf{median} : \Sigma^* \to \Sigma \uplus \{\bot\}$ the statistic that maps a word to its median if it exists, or to $\bot$ otherwise.

An example of a statistic that takes into account the order of events in a word is the most frequent event that occurs right after some dedicated event.

2.2 Counter Monitors

The task of a monitor is to compute a statistic in real time. We define a variant of monitor machines that allows us to classify a monitor based on the amount of resources it uses. We adapt the definition of counter monitors set in [9] to our setting of monitoring frequencies.

Let X be a set of integer variables, called *registers* or *counters*. Registers can be read and written according to relations and functions in the signature $S = \langle 0, +1, \leq \rangle$ as follows:

- An *update* is a mapping from variables to terms over S;
- A *test* is a conjunction of atomic formulas over S and their negation.

The set of updates and test over X are denoted $\Gamma(X)$ and $\Phi(X)$, respectively.

▶ **Definition 4** (Counter Monitor). *A* counter monitor *is a tuple* $\mathcal{A} = (\Sigma, \Lambda, X, Q, \lambda, s, \Delta)$, *where* Σ *is an input alphabet,* Λ *is an output alphabet,* X *is a set of registers,* Q *is a set of control locations,* $\lambda : Q \times \mathbb{N}^X \to \Lambda$ *is an output function,* $s \in Q$ *is the initial location, and* $\Delta \subseteq Q \times \Sigma \times \Phi(X) \times \Gamma(X) \times Q$ *is a transition relation such that for every location* $q \in Q$, *event* σ, *and valuation* v *there exists a unique edge* $(q, \sigma, \phi, \gamma, q') \in \Delta$ *such that* $v \models \phi$ *is satisfied. The sets* Σ, X, Q, Δ *are assumed to be finite.*

A *run* of the monitor $\mathcal{A}$ over a word $w \in \Sigma^* \cup \Sigma^\omega$ is a sequence of transitions $(q_1, v_1) \xrightarrow{w_1} (q_2, v_2) \xrightarrow{w_2} \ldots$ labeled by w such that $q_1 = s$ and $v_1(x) = 0$ for all $x \in X$. Here we write $(q, v) \xrightarrow{\sigma} (q', v')$ when there exists an edge $(q, \sigma, \phi, \gamma, q') \in \Delta$ such that $v \models \phi$ and $v'(x) = v(\gamma(x))$ for all $x \in X$. There exists exactly one run of a given counter monitor $\mathcal{A}$ over a given word w.

▶ **Definition 5** (Monitor Semantics). *Every counter monitor* $\mathcal{A}$ *computes a statistic* $[\![\mathcal{A}]\!] : \Sigma^* \to \Lambda$, *such that* $[\![\mathcal{A}]\!](w) = \lambda(q, v)$ *for* (q, v) *the final state in the run of* $\mathcal{A}$ *over* $w \in \Sigma^*$.

We remark that the term "counter machine" has various different meanings in the literature and designates machines with varying computational power. In our definition we note the use of constant 0 that enables resets. Such resets cannot be simulated in real time. On the contrary, arbitrary increments are w.l.o.g., as shown in [10].

2.3 Probabilistic Generators

In this work we model systems as labeled Markov chains, whose executions generate random ω-words.

▶ **Definition 6** (Markov Chain). *A (finite, connected, labeled)* Markov chain *is a tuple* $\mathcal{M} = (\Sigma, Q, \lambda, \pi, p)$, *where* Σ *is a finite set of events,* Q *is a set of states,* $\lambda : Q \to \Sigma$ *is a labeling,* π *is an initial-state distribution over* Q, *and* $p : Q \times Q \to [0, 1]$ *is a transition distribution with* $\sum_{q' \in Q} p(q, q') = 1$ *for all* $q \in Q$ *and whose set of edges* (q, q') *such that* $p(q, q') > 0$ *forms a strongly connected graph.*

In the rest of this paper, even when not explicitly stated, every Markov chain is assumed to be finite and connected.

Let $\mathcal{M} = (\Sigma, Q, \lambda, \pi, p)$ be a Markov chain. A random infinite sequence $(X_i)_{i \geq 1}$ of states is an execution of $\mathcal{M}$, *Markov*($\mathcal{M}$) for short, if (i) X_1 has distribution π and (ii) conditional on $X_i = q$, X_{i+1} has distribution $q' \mapsto p(q, q')$ and is independent of $X_1, \ldots, X_{i-1}$. By extension, a random ω-word w is *Markov*($\mathcal{M}$) if $w_i = \lambda(X_i)$ for all $i \geq 1$.

We denote by $V_q(k) = \sum_{i=1}^{k} 1_{\{X_i = q\}}$ the *number of visits* to state q within k steps, and by $T_q = \inf\{i > 1 \mid X_i = q\}$ the *first time of visiting* state q (after the initial state). Then $m_q = \mathbb{E}(T_q \mid X_1 = q)$ is the *expected return time* to state q. The ergodic theorem for Markov chains states that the long-run proportion of time spent in each state q is the inverse of m_q. Thus we call $f_q = \frac{1}{m_q}$ the (long-run) *frequency* of q.

▶ **Theorem 7** (Ergodic Theorem [18]). *Let* $\mathcal{M}$ *be a finite connected Markov chain. If* $(X_i)_{i \geq 1}$ *is Markov*($\mathcal{M}$) *then* $V_q(n)/n \xrightarrow{a.s.} f_q$ *as* $n \to \infty$ *for every state* q.

Now summing the frequencies of all states mapped to a letter σ gives the expected frequency of σ, $f_\sigma = \sum_{\substack{q \in Q \\ \lambda(q) = \sigma}} f_q$, as characterized by the following corollary.

▶ **Corollary 8.** *Let* $\mathcal{M}$ *be a finite connected Markov chain. If* w *is Markov*($\mathcal{M}$) *then* $|w_{..n}|_\sigma / n \xrightarrow{a.s.} f_\sigma$ *as* $n \to \infty$ *for every letter* σ.

3 The Limit-Monitoring Problem

We want to monitor the value of a given statistic $\mu : \Sigma^* \to \Lambda$ over the execution of some (probabilistic) process $\mathcal{P}$. This execution is potentially infinite, forming a word $w \in \Sigma^\omega$. In practice, the statistic μ is often used as an estimator of some parameter $v \in \Lambda$ of process $\mathcal{P}$. Such a parameter is always well-defined in the case where μ converges to v as follows.

▶ **Definition 9** (Convergence). *A statistic $\mu : \Sigma^* \to \Lambda$* (almost surely) converges *to a value $v \in \Lambda$ over a random process $\mathcal{P}$, written $\mu(\mathcal{P}) = v$, if $\mathbb{P}_{w\sim\mathcal{P}}(\lim_{n\to\infty} \mu(w_{..n}) = v) = 1$.*

Computing the value of the statistic μ over every finite prefix of w can be an objective in itself. It gives us the most precise estimate of the parameter v when defined. A monitor fulfilling this requirement is called *real-time*. Such a monitor is past-oriented, and is concerned with computing accurately the value $\mu(w_{..n})$ of the statistic at step n, for all n.

▶ **Definition 10** (Real-Time Monitoring). *A monitor $\mathcal{A}$ is a* real-time monitor *of statistic μ, if $[\![\mathcal{A}]\!] = \mu$.*

However, if the aim of the monitor is to serve as an estimator of the parameter v, then it may not be strictly required to output the exact value of μ at every step, as long is its output almost surely converges to v. A monitor that almost surely converges to v is qualified as *limit*. Such a monitor is future-oriented, and is concerned with the asymptotic value of the statistic μ as time tends to infinity, not necessarily computing its precise value over each prefix of the computation.

▶ **Definition 11** (Limit Monitoring). *A monitor $\mathcal{A}$ is a* limit monitor *of statistic $\mu : \Sigma^* \to \Lambda$ on process $\mathcal{P}$, when $[\![\mathcal{A}]\!](\mathcal{P}) = v$ if and only if $\mu(\mathcal{P}) = v$ for all $v \in \Lambda$.*

In words, if the statistic converges then the limit monitor converges to the same value, and if the statistic does not converge then neither does the monitor. To the best of our knowledge, the notion of limit monitoring was not previously considered. By definition, every real-time monitor is trivially also a limit monitor for the corresponding statistic. However, in this paper we show that dedicated limit monitors can be much more efficient.

▶ **Proposition 12.** *Every real-time monitor of some statistic μ is also a limit monitor of μ, on arbitrary generating processes.*

This is in clear contrast with much related work on runtime verification, where past-oriented monitoring (inherently deterministic) often turns out to be computationally easier than future-oriented monitoring (requiring nondeterministic simulation).

4 Precise Real-Time Monitoring

In this section we study the real-time monitoring of statistics by counter monitors. Real-time monitors can be seen as monitoring the past in a precise manner. We show that for some common statistics such as the *mode* and *median* statistics this problem is inherently resource-intensive. More precisely, we identify a class of statistical quantities requiring at least as many counters as there are events in the input alphabet.

To illustrate the difficulty of monitoring certain statistics in real time, recall the *mode* as defined in Example 2. A straightforward real-time monitor for the mode counts the number of occurrences of each letter σ in a separate counter x_σ. Then σ is the mode if and only if $x_\sigma > x_\rho$ for all $\rho \in \Sigma \setminus \{\sigma\}$. Hence $|\Sigma|$ counters suffice to monitor the mode.

But can we do better? Intuitively it seems necessary to keep track of the exact number of occurrences for each individual letter. Indeed, we show in this section that for real-time monitors this number is tight: any real-time counter monitor of the mode must use at least $|\Sigma|$ counters. In many application where the alphabet Σ is large this may be beyond the amount of resources available for a monitor. While Proposition 12 implies that the mode can also be limit monitored using $|\Sigma|$ counters, we show in the next section that limit monitoring can be much more resource-sparing.

To capture the hardness of real-time monitoring for a whole class of statistics, we start by defining an equivalence relation over words relative to a statistic. Two words are μ-equivalent if it is impossible for μ to distinguish them, even with an arbitrary suffix appended to both words.

▶ **Definition 13** (μ-Equivalence). *Let μ be a statistic over Σ. Two words $w_1, w_2 \in \Sigma^*$ are* μ-equivalent*, denoted $w_1 \equiv_\mu w_2$, if $\mu(w_1 u) = \mu(w_2 u)$ for all words $u \in \Sigma^*$.*

Now we define the notion of a Σ-counting statistic, which states that two equivalent words must have exactly the same number of occurrences per letter, modulo a constant shift across all letters. Intuitively a Σ-counting statistic induces many equivalence classes, too many to be possibly tracked by a counter monitor with less than $|\Sigma|$ counters.

▶ **Definition 14** (Σ-Counting). *A statistic μ is* Σ-counting *if $w \equiv_\mu w'$ implies that there exists $n \in \mathbb{Z}$ such that $|w|_\sigma = |w'|_\sigma + n$ for all $\sigma \in \Sigma$.*

▶ **Proposition 15.** *For any Σ such that $|\Sigma| > 1$ both the mode and the median statistics are Σ-counting.*

To illustrate the definition of Σ-counting, consider the mode-equivalent words $aabc$ and a over the alphabet $\Sigma = \{a, b, c\}$. The distance for all letter counts is one. Over the alphabet with an additional letter d the two words are not mode-equivalent (for example, consider the extensions $aabcd$ and ad), since the distance for the count of d is zero.

Our proof that Σ-counting statistics are expensive to monitor follows the argument in [9] that separates $(k + 1)$-counter monitors from k-counter monitors. In particular, we show that for large n, the number of μ-inequivalent words of length less or equal to n is greater than the number of possible configurations reachable by an $O(k)$-counter monitor over words of length less or equal to n.

▶ **Theorem 16.** *Real-time counter monitors of a Σ-counting statistic require $\Omega(|\Sigma|)$ counters.*

As a corollary of Proposition 15 and Theorem 16, we have that precisely monitoring the mode and the median in real time requires roughly as many counters as the size of the alphabet, which is prohibitive in many practical applications.

5 Efficient Limit Monitoring

In this section we develop a new algorithmic framework for efficient limit monitoring of frequency-based statistics. We first present a general monitoring scheme and then instantiate it to derive efficient monitoring algorithms for both mode (Section 5.2) and median (Section 5.3). In Section 6 we present a monitoring algorithm for a general class of frequency properties. While corresponding real-time monitors require a number of counters proportional to the size of the input alphabet, our limit monitors only use a constant number of counters (e.g., four for the mode), independent of the alphabet size. The algorithmic ideas in our monitoring scheme are simple and intuitive, which makes our algorithms easy to understand, implement, and deploy. However, the correctness proofs are surprisingly hard and required us to develop a new ergodicity theory for Markov chains that takes limits over arbitrary subwords (Section 5.1).

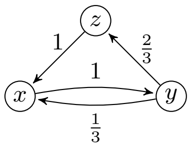
Our high-level monitoring strategy comprises the following points:

1. Split the input sequence into subwords of increasing length.
2. In every subword, acquire partial information about the statistic.
3. Assemble global information about the statistic across different subwords.

The idea behind splitting the input sequence into subwords is that when the monitored property involves frequencies of many events, then different events can be counted separately over different subwords, which enables us to reuse registers. Because of the probabilistic nature of the generator we can still ensure that, in the long run, the monitor value converges to the limit of the statistic. As we will see, there is great flexibility in how exactly the sequence is partitioned. In principle, the subwords can overlap or leave gaps arbitrarily, as long as the length of the considered subwords grows "fast enough".

5.1 The Ergodic Theorem over Infixes

Consider the following Markov chain on the left-hand side, and a random ω-word generated by this Markov chain in the table on the right-hand side.



ω-word	x	y	z	x	y	z	x	y	x	y	z	x	y	z	x	y	$\dots$
Prefixes	0	.5	.33	.25	.4	.33	.29	.38	.33	.4	.36	.33	.38	.36	.33	.38	$\xrightarrow{\text{a.s.}} \frac{3}{8}$
Infixes	0		.5			.33				.5					.2		$\xrightarrow{\text{a.s.}} \frac{3}{8}$

The second row of the table shows the frequency of state y in prefixes of increasing length. For example, after $xyzx$ we have frequency $\frac{1}{4}$. The classic ergodic theorem (Theorem 7) tells us that this frequency almost surely converges to $f_y = \frac{3}{8}$, the inverse of the expected return time to y. However, this theorem does not apply to take a limit over arbitrary subwords, for example, the infixes of increasing length (indicated by vertical lines) in the third row of the table. We prove a result that shows that also in this much more general case the limit frequency of y is $\frac{3}{8}$.

The strong law of large numbers states that the empirical average of i.i.d. random variables converges to their expected value, i.e., $(X_1 + \cdots + X_n)/n \xrightarrow{\text{a.s.}} \mathbb{E}(X_1)$ as $n \to \infty$. The fact that random variables are "reused" from the n'th to the $(n+1)$'st sample does matter in this statement. Otherwise the mere existence of a mean value is not sufficient to guarantee convergence. However, when the variance (or higher-order moment) is bounded, then this "reuse" is no longer required. We now prove such a variant of the law of large numbers.[2]

▶ **Theorem 17.** *Let $\{X_{n,i} : n, i \geq 1\}$ be a family of identically distributed random variables with $\mathbb{E}(X_{1,1}) = \mu$ and $\mathbb{E}(X_{1,1}^4) < \infty$, such that $\{X_{n,i} : i \geq 1\}$ are mutually independent for every $n \geq 1$. Let $(s_n)_{n\geq 1}$ be a sequence of indices with $s_n \geq an$ for every $n \geq 1$ and fixed $a > 0$. Set $S_n = \sum_{i=1}^{s_n} X_{n,i}$. Then $S_n/s_n \xrightarrow{a.s.} \mu$ as $n \to \infty$.*

In our proof the combination of the fourth-moment bound and the linear increase of s_n leads to a converging geometric series. We believe that these assumptions could be slightly

---

[2] Such a setting is sometimes called array of rowwise independent random variables in the literature, see [13] in particular.

relaxed to a second-moment bound or to sublinearly increasing sequences. Theorem 17 already gives a basis to reason about infix-convergence for i.i.d. processes. We now use it to derive a corresponding result for Markov chains.

Let $\mathcal{M}$ be a Markov chain and $(X_i)_{i\geq 1}$ be *Markov*$(\mathcal{M})$. Given an *offset function* $s : \mathbb{N} \to \mathbb{N}$, we refer to $X_{s(n)+1}X_{s(n)+2}\cdots$ as the *n'th suffix* of X. We denote by $V_q^n(k) = \sum_{i=1}^{k} 1_{\{X_{s(n)+i}=q\}}$ the *number of visits* to state q within k steps in the n'th suffix. We generalize the classic ergodic theorem for Markov chains (Theorem 7) to take the limit over arbitrary subwords.

▶ **Theorem 18.** *Let $\mathcal{M}$ be a finite connected Markov chain and s an offset function. If $(X_i)_{i\geq 1}$ is Markov$(\mathcal{M})$ then $V_q^n(n)/n \xrightarrow{a.s.} f_q$ as $n \to \infty$ for every state q.*

Our proof applies Theorem 17 to the i.i.d. excursion times between visiting state q within the n'th suffix. This requires bounding the moments of excursion times and showing that the time until visiting q for the first time in every subword becomes almost surely negligible for increasing size subwords. As a corollary of Theorem 18 we get the following characterization for the long-run frequencies of letters over infixes.

▶ **Corollary 19.** *Let $\mathcal{M}$ be a finite connected Markov chain and s an offset function. If w is Markov$(\mathcal{M})$ then $|w_{s(n)+1..s(n)+n}|_\sigma/n \xrightarrow{a.s.} f_\sigma$ as $n \to \infty$ for every letter σ.*

5.2 Monitoring the Mode

As we saw in Section 4, precisely monitoring the mode in real-time requires at least $|\Sigma|$ counters. By contrast, we show now that the mode can be limit monitored using only four counter registers. For convenience we also use two registers to store event letters; since we assume Σ to be finite they can be emulated in the finite state component of the monitor.

The core idea of our monitoring algorithm is to split w into chunks, and for each chunk only count the number of occurrences of two letters x and y. Letter x is considered the current candidate for the mode and y is a randomly selected contender. If x does not occur more frequently than y in the current chunk, y becomes the mode candidate for the next chunk. The success of the monitor relies on two points: (i) it must be repeatably possible for the true mode to end up in x, and (ii) it must be likely for the true mode to eventually remain in x. The first point is achieved by taking y randomly, and the second point is achieved by gradually increasing the chunk size. It is sufficient to increase the chunk size by one and decompose w as follows:

$$\underbrace{\sigma_1}\ \underbrace{\sigma_2\sigma_3}\ \underbrace{\sigma_4\sigma_5\sigma_6}\ \underbrace{\sigma_7\sigma_8\sigma_9\sigma_{10}}\ \underbrace{\sigma_{11}\cdots}$$

Formally, the decomposition of w into chunks is given by an offset function $s : \mathbb{N} \to \mathbb{N}$ with $s(n) = \frac{n(n-1)}{2}$, such that the n'th chunk starts at $s(n)+1$ and ends at $s(n)+n$. For convenience, we introduce a double indexing of w by $n \geq 1$ and $1 \leq i \leq n$, such that $w_{n,i} = w_{s(n)+i}$ is the i'th letter in the n'th chunk.

Algorithm 1: Mode monitor

```
1  Function Init(σ):
2      x, y := σ, σ
3      c_x, c_y := 0, 0
4      n, i := 2, 1
5      return x
6  Function Next(σ):
7      if i = 1 then
8          if c_x ≤ c_y then  x := y
9          y := σ
10         c_x, c_y := 0, 0
11
12     if x = σ then c_x := c_x + 1
13     if y = σ then c_y := c_y + 1
14
15     if i = n then n, i := n + 1, 1
16              else i := i + 1
17     return x
```

Algorithm 2: Median monitor

```
1  Function Init(σ):
2      x := σ
3      c_1, c_2, c_3, c_4 := 0, 0, 0, 0
4      n, i := 2, 1
5      return x
6  Function Next(σ):
7      if i = 1 then
8          if c_1 ≥ c_2 then x := pre_≺(x)
9          if c_3 ≥ c_4 then x := succ_≺(x)
10         c_1, c_2, c_3, c_4 := 0, 0, 0, 0
11     if σ < x then c_1 := c_1 + 1
12     if σ ≥ x then c_2 := c_2 + 1
13     if σ > x then c_3 := c_3 + 1
14     if σ ≤ x then c_4 := c_4 + 1
15     if i = n then n, i := n + 1, 1
16              else i := i + 1
17     return x
```

A formal description of our mode monitor is given in Algorithm 1. The counters n and i keep track of the decomposition of w. For the very first letter σ, `Init` initializes both registers x and y to σ (line 2). Then, for every subsequent letter, `Next` counts an occurrence of x and y using counters c_x and c_y, respectively (line 12-13). At the beginning of every chunk, x is replaced by y if it did not occur more frequently in the previous chunk (line 8), and y is set to the first letter of the chunk (line 9). At every step, x is the current estimate of the mode.

▶ **Example 20.** For alphabet $\Sigma = \{a, b, c\}$ and probability distribution p with $p(a) = 0.5$, $p(b) = 0.3$, and $p(c) = 0.2$, the following table shows a word w where every letter was independently sampled from p, and the corresponding mode at every position in w.

w	c b b a b a c a a b c a c a a a ⋯
mode	c - b b b b b - a - - a a a a a ⋯

In this example, mode first switches between the different letters and undefined, but then eventually seems to settle on a. We show that this is not an accident, but happens precisely because a is the unique letter that p assigns the highest probability.

Now the following table shows the execution of Algorithm 1 on the same random word.

n	1	2	3	4	5	6 ⋯
i	1	1 2	1 2 3	1 2 3 4	1 2 3 4 5	1 ⋯
σ	c	b b	a b a	c a a b	c a c a a	a ⋯
x	c	c	b	a	a	a ⋯
y	c	b	a	c	c	a ⋯
c_x	1	0 0	0 1 1	0 1 2 2	0 1 1 2 3	1 ⋯
c_y	1	1 2	1 1 2	1 1 1 1	1 1 2 2 2	1 ⋯

Initially c is considered the mode and compared to b in the second chunk, where b occurs more frequently. Thus b is considered the mode and compared to a in the third chunk, where a occurs more frequently. In the fourth and fifth chunk a is compared to c, where a occurs more frequently in both chunks. Again, the algorithm seems to settle on a, the true mode.

To prove the correctness of our algorithm according to Definition 11 requires us to first characterize when a Markov chain has a mode, i.e., under which conditions the mode statistic almost surely converges. For this it is illustrative to instantiate Definition 9 for the mode, which states that a is the mode of an ω-word w if there exists a length n, such that for every length $n' \geq n$, $|w_{..n'}|_a > |w_{..n'}|_b$ for every $b \neq a$. In a Markov chain the ergodic theorem characterizes the long-run frequencies of states, and thus the long-run frequencies of letters (see Corollary 8). Hence a Markov chain has a mode if and only if its random ω-word almost surely has a unique letter that occurs most frequently.

▶ **Theorem 21.** *Over Markov chains, the mode statistic converges to a if and only if $f_a > f_b$ for all $b \neq a$.*

Proof. Let $\mathcal{M}$ be a Markov chain and w be *Markov*$(\mathcal{M})$. According to Corollary 8, $|w_{..n}|_\sigma/n \xrightarrow{\text{a.s.}} f_\sigma$ as $n \to \infty$ for every $\sigma \in \Sigma$

Now assuming $f_a > f_b$ for all $b \neq a$, we have for sufficiently large n that $|w_{..n}|_a > |w_{..n}|_b$ for all $b \neq a$, and thus a is the mode of w almost surely.

Conversely, if there are two distinct letters a, a' with equal maximal frequencies $f_a, f_{a'}$, then almost surely the mode switches infinitely often between a and a', thus neither a nor a' is the mode of w, and thus w does not have a mode. ◀

Now we can prove that Algorithm 1 is a limit monitor for the mode. The core of the argument is that the probability of the true mode eventually staying in register x is lower-bounded by the probability of a eventually being the most frequent letter in *every* subword and a being eventually selected into y, which happens almost surely.

▶ **Theorem 22.** *Algorithm 1 limit-monitors the mode over Markov chains.*

Proof. Let w be *Markov*$(\mathcal{M})$ and let a be the mode of w (the other case where w does not have a mode is obvious). Let γ_n be the function that maps every letter to the number of its occurrences in the n'th subword, i.e., $\gamma_n(\sigma) = |w_{s(n)+1..s(n)+n}|_\sigma$. To capture Algorithm 1 mathematically, we define the random variables

$$Y_n = w_{n,1}; \qquad X_1 = w_{1,1}; \qquad X_{n+1} = \begin{cases} X_n, & \text{if } \gamma_n(X_n) > \gamma_n(Y_n); \\ Y_n, & \text{if } \gamma_n(X_n) \leq \gamma_n(Y_n). \end{cases}$$

That is, X_n and Y_n are the values of x and y throughout the n'th subword. We need to show that almost surely, eventually $X_n = a$ forever, i.e., $\mathbb{P}(\Diamond\Box X_n = a) = 1$.[3]

It is more likely that a eventually stays in x forever as that a eventually is the most frequent letter in *every* subword and that a is also eventually sampled into y:

$$\begin{aligned} &\mathbb{P}(\Diamond\Box X_n = a) \\ \geq\ &\mathbb{P}(\Diamond(\Box\forall b \neq a : \gamma_n(b) < \gamma_n(a)) \wedge (\Diamond Y_n = a)) \\ \geq\ &\mathbb{P}((\Box_{\geq n_0}\forall b \neq a : \gamma_n(b) < \gamma_n(a)) \wedge (\Diamond_{\geq n_0} Y_n = a)) \end{aligned}$$

The last lower bound holds for any fixed n_0 and we show that it converges to 1 as $n_0 \to \infty$.

$$\begin{aligned} &\mathbb{P}((\Box_{\geq n_0}\forall b \neq a : \gamma_n(b) < \gamma_n(a)) \wedge (\Diamond_{\geq n_0} Y_n = a)) \\ \geq\ &\mathbb{P}(\Box_{\geq n_0}\forall b \neq a : \gamma_n(b) < \gamma_n(a)) \cdot \mathbb{P}(\Diamond_{\geq n_0} Y_n = a) \\ =\ &\mathbb{P}(\Box_{\geq n_0}\forall b \neq a : \gamma_n(b) < \gamma_n(a)) \end{aligned}$$

[3] In the interest of readability we use temporal (modal) logic notation $\Diamond$ and $\Box$ meaning *eventually* and *forever*, respectively.

Since $\gamma_n(\sigma)/n \xrightarrow{\text{a.s.}} f_\sigma$ by Corollary 19 and a is the unique letter with highest frequency f_a by Theorem 21, we have $\mathbb{P}(\Box_{\geq n_0} \forall b \neq a : \gamma_n(b) < \gamma_n(a)) = 1$ for sufficiently large n_0. Thus, $\mathbb{P}(\Diamond\Box X_n = a) = 1$. ◀

Note that our policy of always selecting the mode contender y from the input is an optimization, since we expect to see the mode often in the input. Our proof requires that the true mode is selected into y infinitely often, which is the case because we update y at irregular positions. Two other policies to update y would be (i) to always uniformly sample from Σ, or (ii) to cycle deterministically through all elements of Σ.

5.3 Monitoring the Median

Recall from Example 3 that a is the *median* of a word w over a $\prec$-ordered alphabet Σ when

$$\sum_{\sigma \succ a} |w|_\sigma < \sum_{\sigma \preccurlyeq a} |w|_\sigma \tag{1}$$

on the one hand, and

$$\sum_{\sigma \prec a} |w|_\sigma < \sum_{\sigma \succcurlyeq a} |w|_\sigma \tag{2}$$

on the other hand. These equations readily lead to our median limit-monitoring algorithm shown in Algorithm 2, which we display next to out mode monitor to highlight their common structure. The idea of the algorithm is to maintain a median candidate x and then use four counters c_1, c_2, c_3, c_4 to compute the sums in inequality (1) and (2), for $a = x$, in every subword (line 11-14). Whenever any of the two inequalities is not satisfied at the end of a subword, a new median candidate is selected into x for the next subword. In particular, if inequality (1) is violated then the next lower value in the ordering $\prec$ is selected (line 8), and if inequality (2) is violated then the next higher value is selected (line 9). Notice that we could eliminate the counters c_3, c_4, by alternating the computation of inequality (1) and (2) over different subwords, and thus reusing c_1, c_2 to compute inequality (2).

▶ **Theorem 23.** *Algorithm 2 limit-monitors the median over Markov chains.*

6 Monitoring General Frequency Properties

In the previous section we presented high-level principles for efficient limit monitoring and designed specialized monitoring algorithms for the mode and median statistic, which are both derived from event frequencies. We postulate that our algorithmic ideas are straightforward to adapt to obtain monitors for many other frequency-based statistics. However, we did not yet precisely defined what we mean by *frequency property*, nor demonstrated how efficiently these can be limit-monitored in the general setting. In this section we provide a first step in this direction by defining a simple language to specify frequency-based Boolean statistics, and showing that all statistics definable in this language can be limit-monitored over Markov chains with four counters only.

From the defining equations of the mode and median we observe that a characteristic construction is the formation of linear inequalities over the frequencies (or equivalently, occurrence counts) of specific events. The key part of the argument for the correctness of our monitoring algorithms is that since event frequencies almost surely converge, both over prefixes and infixes, also these inequalities almost surely "stabilize". We use the same construction at the core of a language to define general frequency-based statistics. For simplicity we focus on statistics that output a Boolean value.

▶ **Definition 24.** *A* frequency formula *over alphabet* Σ *is a Boolean combination of atomic formulas of the form*

$$\sum_{\sigma \in \Sigma} \alpha_\sigma \cdot f_\sigma > \alpha \tag{3}$$

where all α*'s are integer coefficients.*

A frequency formula ϕ is built from linear inequalities over frequencies of events. The evaluation of a frequency formulas is as expected (we write $w \models \phi$ if ϕ evaluates to true over w). Hence we see ϕ as defining the Boolean statistic $[\![\phi]\!] : \Sigma^* \to \mathbb{B}$, where

$$[\![\phi]\!](w) = \begin{cases} 1, & \text{if } w \models \phi; \\ 0, & \text{if } w \not\models \phi. \end{cases}$$

▶ **Example 25.** The existence of a mode is expressed as the frequency formula

$$\bigvee_{a \in \Sigma} \bigwedge_{\substack{\sigma \in \Sigma \\ \sigma \neq a}} f_a > f_\sigma \,.$$

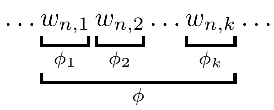
▶ **Example 26.** Consider the detection of the malfunction of a web server, which which would favour certain client requests over others. Such a malfunction could be observed by detecting that certain events are disproportionately more frequent than others. The following frequency formula specifies that no event can occur 100-times more frequent than any other event:

$$\bigwedge_{\substack{a,b \in \Sigma \\ a \neq b}} f_a < 100 \cdot f_b \,.$$

A frequency formula ϕ can be limit monitored by simply evaluating ϕ repeatedly over longer and longer subwords. However, the key to save resources is to evaluate different atomic subformulas of ϕ over different subwords, and thus only evaluating one subformula at a time.

▶ **Theorem 27.** *Over Markov chains, every frequency formula can be limit-monitored using 4 counters.*

Proof. Let ϕ be a frequency formula with k atomic subformulas $\phi_1, \ldots, \phi_k$ of the form (3). The monitor partitions the input word w into infixes $w_{n,i}$ with $|w_{n,i}| = n$, for $n \geq 1$ and $1 \leq i \leq k$, as follows:



Keeping track of the increasing infix length n and the current position within an infix requires two counters. Then over every infix $w_{n,i}$ the monitor uses two counters to compute ϕ_i, one for positive and one for negative increments. At the end of $w_{n,i}$ we have a truth value for ϕ_i that is used to partially evaluate ϕ. This evaluation is implemented in the final-state component of the monitor, and the two counters are reused across all infixes. Then after every k'th infix we have a new "estimate" of ϕ that in the long run converges the same way as $[\![\phi]\!]$. Hence the resulting automaton is a limit monitor of ϕ: by Corollary 19, the frequency of each event over infixes of increasing length tends to its respective asymptotic frequency, so that strict inequalities holding over empirical frequencies almost surely hold over infixes of increasing length. ◀

7 Conclusion

In this paper we have studied the monitoring of frequency properties of event sequences. We observed that real-time monitoring can be surprisingly hard (i.e., resource-intensive) for such properties, and introduced the alternative notion of limit monitoring. In this limit monitoring setting we showed that a simple algorithmic idea leads to resource-efficient monitoring algorithms for frequency properties. To prove the correctness of our algorithms we generalized the ergodic theory of Markov chains.

The results in this paper are a first indicator of the relevance and potential of limit monitoring. We hope that future research broadens the understanding of this problem and we close with a number of interesting directions.

First, we are interested in a tighter characterization of properties that can be efficiently limit monitored. Let us remark that the results in this paper immediately generalize from counting individual events to counting the occurrences of regular event patterns. This is the case because regular expression matching can be performed in real time by the finite state component of a counter monitor. We extended our frequency formulas with free variables to support non-Boolean statistics, and quantification to reason about unknown alphabet symbols. However, the shape and efficiency of a generic monitoring algorithm is not yet clear. For examples, we saw that there are different policies to partition the input sequence and different policies to obtain candidate values for the monitor output. Certain forms of existential quantification can be translated to random sampling, but this does not seem to hold in general since not all events in the alphabet may occur in the execution under consideration. Going even further, it would be interesting to consider limit monitoring of properties with temporal aspects (such as *always* and *eventually* modalities).

Second, it is well known (see e.g. [6]) that the asymptotic frequencies of k-long subwords fully characterize a k-state connected Markov chain. Hence the transition probabilities of a Markov chain (of known structure) can be inferred from the conditional probabilities of events. Thus, assuming the structure of a Markov chain is known, frequency queries and the algorithmic ideas in this paper can be used to learn its transition probabilities to an arbitrary precision. It would be interesting to study more broadly "how much" of a system can be learned from frequency properties (and similar observations).

Third, throughout this paper we used the term efficient to mean resource-efficient in the amount of memory used by a monitor. However, there is the orthogonal question of time-efficiency. For a limit monitor this means how quickly a monitor converges in relation to the monitored statistic. We hope that future research can provide numerical guarantees or estimates for convergence rates. For the simple setting of an i.i.d. word over a two-letter alphabet, we proved that the mode statistic converges exponentially fast. More precisely, if w is a random ω-word where every letter is i.i.d. (that is, independent and identically distributed) according to a probability distribution p over $\{a, b\}$ with $p(a) > p(b)$, then $\mathbb{P}(\mathsf{mode}(w_{..n}) = a) \geq 1 - (4p(a)p(b))^{\lfloor \frac{n}{2} \rfloor}$. Since this depends on the exact probabilities, the analytical expressions of the confidence value seem to become intractable for three letters or more. In probability theory, there exist several different notions of convergence of random variables. The results in this paper use the notion of almost-sure convergence of a statistic μ (Definition 9), that is, $\mathbb{P}_{w\sim\mathcal{P}}(\lim_{n\to\infty} \mu(w_{n..}) = v) = 1$. It would be interesting to study also other notions, for example convergence in probability, that is, $\lim_{n\to\infty} \mathbb{P}_{w\sim\mathcal{P}}(\mu(w_{n..}) = v) = 1$.

Fourth, the correctness results we derived for our monitoring algorithms hold for systems modeled as connected Markov chains. However, we believe that the algorithmic ideas of this paper are more widely applicable. Thus it would be interesting to study limit monitoring for other types of systems, for example, Markov decision processes which are challenging for our monitoring scheme because nondeterminism allows certain events to *always* occur deliberately when the monitor is not watching for them. In the security context a monitored system is usually assumed to be adversarial, not probabilistic. It could be interesting to turn our deterministic monitors of probabilistic systems into probabilistic monitors for nondeterministic systems.

References

1. Gul Agha and Karl Palmskog. A survey of statistical model checking. *ACM Trans. Model. Comput. Simul.*, 28(1):6:1–6:39, 2018.
2. Rajeev Alur, Dana Fisman, and Mukund Raghothaman. Regular programming for quantitative properties of data streams. In *ESOP*, volume 9632 of *Lecture Notes in Computer Science*, pages 15–40. Springer, 2016.
3. Howard Barringer, Yliès Falcone, Klaus Havelund, Giles Reger, and David E. Rydeheard. Quantified event automata: Towards expressive and efficient runtime monitors. In *FM*, 2012.
4. Ezio Bartocci and Yliès Falcone, editors. *Lectures on Runtime Verification - Introductory and Advanced Topics*, volume 10457 of *Lecture Notes in Computer Science*. Springer, 2018.
5. Manuel Blum, Robert W. Floyd, Vaughan R. Pratt, Ronald L. Rivest, and Robert Endre Tarjan. Time bounds for selection. *J. Comput. Syst. Sci.*, 7(4):448–461, 1973.
6. Taylor L. Booth. Statistical properties of random digital sequences. *IEEE Trans. Computers*, 17(5):452–461, 1968.
7. Krishnendu Chatterjee, Thomas A. Henzinger, and Jan Otop. Quantitative monitor automata. In *SAS*, 2016.
8. Ben D'Angelo, Sriram Sankaranarayanan, César Sánchez, Will Robinson, Bernd Finkbeiner, Henny B. Sipma, Sandeep Mehrotra, and Zohar Manna. LOLA: runtime monitoring of synchronous systems. In *TIME*, pages 166–174. IEEE Computer Society, 2005.
9. Thomas Ferrère, Thomas A. Henzinger, and N. Ege Saraç. A theory of register monitors. In *LICS*, pages 394–403. ACM, 2018.
10. Patrick C. Fischer, Albert R. Meyer, and Arnold L. Rosenberg. Counter machines and counter languages. *Mathematical Systems Theory*, 2(3):265–283, 1968.
11. Thomas A. Henzinger. Quantitative reactive modeling and verification. *Computer Science - R&D*, 28(4):331–344, 2013.
12. C. A. R. Hoare. Algorithm 65: find. *Commun. ACM*, 4(7):321–322, 1961.
13. Tien-Chung Hu, F Moricz, and R Taylor. Strong laws of large numbers for arrays of rowwise independent random variables. *Acta Mathematica Hungarica*, 54(1-2):153–162, 1989.
14. Marta Kwiatkowska, Gethin Norman, and David Parker. Probabilistic model checking: Advances and applications. In Rolf Drechsler, editor, *Formal System Verification: State-of the-Art and Future Trends*, pages 73–121. Springer, 2018.
15. Marta Z. Kwiatkowska. Quantitative verification: models techniques and tools. In *ESEC/SIGSOFT FSE*, pages 449–458. ACM, 2007.
16. David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. Markov chains and mixing times, second edition, 2017. URL: https://pages.uoregon.edu/dlevin/MARKOV/mcmt2e.pdf.
17. Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
18. James R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998.
19. Dana Ron. *Automata Learning and its Applications*. PhD thesis, Hebrew University, 1995.
20. Steven Rudich. Inferring the structure of a markov chain from its output. In *FOCS*, pages 321–326. IEEE Computer Society, 1985.

21 Usa Sammapun, Insup Lee, and Oleg Sokolsky. RT-MaC: Runtime monitoring and checking of quantitative and probabilistic properties. In *RTCSA*, pages 147–153. IEEE Computer Society, 2005.

22 Zbigniew S. Szewczak. On moments of recurrence times for positive recurrent renewal sequences. *Statistics & Probability Letters*, 78(17):3086–3090, 2008.

23 B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

A Proofs of Section 3

▶ **Proposition 12.** *Every real-time monitor of some statistic μ is also a limit monitor of μ, on arbitrary generating processes.*

Proof. We have $[\![\mathcal{A}]\!] = \mu$, hence if μ converges to some value v over a generating process $\mathcal{P}$ then $[\![\mathcal{A}]\!]$ also converges to v. ◀

B Proofs of Section 4

▶ **Definition 28.** *Two configurations (q_1, v_1) and (q_2, v_2) of a counter monitor over the alphabet Σ are* equivalent *if for all finite words $u \in \Sigma^*$ we have $(q_1, v_1) \xrightarrow{u} (q, v)$ if and only if $(q_2, v_2) \xrightarrow{u} (q, v)$. As customary, $(q, v) \xrightarrow{u} (q', v')$ denotes the existence of a sequence of transitions from (q, v) to (q', v') labeled by u.*

▶ **Proposition 15.** *For any Σ such that $|\Sigma| > 1$ both the mode and the median statistics are Σ-counting.*

Proof. We prove the statement for the case of the mode. Let w and w' be words such that $w \equiv_{\mathsf{mode}} w'$. Assume, towards a contradiction, that for all $n \in \mathbb{Z}$ there exists $\sigma \in \Sigma$ such that $|w|_\sigma \neq |w'|_\sigma + n$. By assumption w and w' have the same mode ρ, otherwise they are trivially not equivalent according to $\equiv_{\mathsf{mode}}$. Let $k = |w|_\rho - |w'|_\rho$. There exists $\sigma \in \Sigma$ such that $|w|_\sigma \neq |w'|_\sigma + k$. We have in particular $\sigma \neq \rho$. Let $l = |w|_\rho - |w|_\sigma$ and $l' = |w'|_\rho - |w'|_\sigma$. But then $\mathsf{mode}(w\sigma^l) \neq \mathsf{mode}(w'\sigma^l)$ or $\mathsf{mode}(w\sigma^{l'}) \neq \mathsf{mode}(w'\sigma^{l'})$, which contradicts $w \equiv_{\mathsf{mode}} w'$. ◀

▶ **Theorem 16.** *Real-time counter monitors of a Σ-counting statistic require $\Omega(|\Sigma|)$ counters.*

Proof. Let $\Sigma = \{0, 1, \ldots, k\}$ be a finite alphabet of size $k \geq 3$. Let μ be a Σ-counting statistic, and $\mathcal{A}$ be a $k - 2$ counter monitor with m states. We show that for large enough n, the number of μ-inequivalent words of length less or equal to n is strictly greater than the number of possible configurations reachable by a k-counter monitor over words of length less or equal to n.

By Σ-counting hypothesis, if $u_1 \equiv_\mu u_2$ then there is an integer p for which $|u_1|_i = |u_2|_i + p$ holds for all $0 \leq i \leq k$. We can thus represent each equivalence class of $\equiv_\mu$ by a string u such that $|u|_i = 0$ for (at least) one $i \in \Sigma$. The number of equivalence classes of prefixes of length up to n is $\binom{n+k+1}{k+1} - \binom{n}{k+1}$.

We assume without loss of generality that counter values are incremented in $\mathcal{A}$ by at most one at every event [10]. There are mn^{k-2} possible configurations of a counter monitors over words of length up to n. Yet we have that $\binom{n+k+1}{k+1} - \binom{n}{k+1} > mn^{k-2}$ for sufficiently large n. Hence for some integer n, there are more μ-inequivalent words of length less or equal to n than configurations a $(k-2)$-counter monitor can possibly reach after reading such words. It follows that no $(k-2)$-counter monitor can compute μ. ◀

C Proofs of Section 5

▶ **Theorem 17.** *Let $\{X_{n,i} : n, i \geq 1\}$ be a family of identically distributed random variables with $\mathbb{E}(X_{1,1}) = \mu$ and $\mathbb{E}(X_{1,1}^4) < \infty$, such that $\{X_{n,i} : i \geq 1\}$ are mutually independent for every $n \geq 1$. Let $(s_n)_{n\geq 1}$ be a sequence of indices with $s_n \geq an$ for every $n \geq 1$ and fixed $a > 0$. Set $S_n = \sum_{i=1}^{s_n} X_{n,i}$. Then $S_n/s_n \xrightarrow{a.s.} \mu$ as $n \to \infty$.*

Proof. Let $\mathbb{E}(X_{1,1}^4) = M$. W.l.o.g. $\mu = 0$ (consider $Y_{n,i} = X_{n,i} - \mu$). We expand $\mathbb{E}(S_n^4)$ and observe that, by independence, $\mathbb{E}(X_{n,i}X_{n,j}^3) = \mathbb{E}(X_{n,i}X_{n,j}X_{n,k}^2) = \mathbb{E}(X_{n,i}X_{n,j}X_{n,k}X_{n,l}) = 0$ for distinct indices i, j, k, l. Hence,

$$\mathbb{E}(S_n^4) = \mathbb{E}\left(\sum_{1\leq i\leq s_n} X_{n,i}^4 + 6 \sum_{1\leq i<j\leq s_n} X_{n,i}^2 X_{n,j}^2\right).$$

Now for $i \leq j$, by independence and the Cauchy-Schwarz inequality

$$\mathbb{E}(X_{n,i}^2 X_{n,j}^2) = \mathbb{E}(X_{n,i}^2)\mathbb{E}(X_{n,j}^2) \leq \mathbb{E}(X_{n,i}^4)^{\frac{1}{2}}\mathbb{E}(X_{n,j}^4)^{\frac{1}{2}} = M\,.$$

So we get the bound

$$\mathbb{E}(S_n^4) \leq s_n M + 3s_n(s_n - 1)M \leq 3s_n^2 M\,.$$

Thus

$$\mathbb{E}\left(\sum_{n\geq 1}(S_n/s_n)^4\right) \leq 3M \sum_{n\geq 1} 1/s_n^2 \leq \tfrac{3M}{a^2} \sum_{n\geq 1} 1/n^2 < \infty$$

which implies

$$\sum_{n\geq 1}(S_n/s_n)^4 < \infty \text{ a.s.}$$

and hence $S_n/s_n \xrightarrow{\text{a.s.}} 0$. ◀

▶ **Theorem 18.** *Let $\mathcal{M}$ be a finite connected Markov chain and s an offset function. If $(X_i)_{i\geq 1}$ is Markov($\mathcal{M}$) then $V_q^n(n)/n \xrightarrow{a.s.} f_q$ as $n \to \infty$ for every state q.*

To prove the result, we first introduce some notation and supporting lemmas.

We denote by $T_q^{n(r)}$ (for $r \geq 0$) the r'th time of visiting state q in the n'th subword, and by $S_q^{n(r)}$ (for $r \geq 1$) the length of the r'th excursion to state q in the n'th subword:

$$T_q^{n(0)} = \inf\{i \geq 1 \mid X_{s(n)+i} = q\};$$
$$T_q^{n(r+1)} = \inf\{i > T_q^{n(r)} \mid X_{s(n)+i} = q\};$$
$$S_q^{n(r+1)} = T_q^{n(r+1)} - T_q^{n(r)}.$$

Let $\overline{SS}_q^n(k)$ be the length of the first k excursions to state q, and $\overline{TS}_q^n(k)$ additionally includes the time to visit q for the first time:

$$\overline{SS}_q^n(k) = \sum_{i=1}^{k} S_q^{n(i)}; \qquad \overline{TS}_q^n(k) = T_q^{n(0)} + \overline{SS}_q^n(k)\,.$$

For a state q we establish the following connection between the time it takes to visit q a certain number of times, and the number of times q is visited within a certain time bound.

▶ **Lemma 29.** *For $a \geq 0$ and arbitrary b, we have*

$$\overline{TS}_q^n(k) \leq n + a \implies V_q^n(n) \geq k - a\,; \quad (4)$$

$$\overline{TS}_q^n(k) \geq n - b \implies V_q^n(n) \leq k + 1 + |b|\,. \quad (5)$$

Proof. $\overline{TS}_q^n(k)$ is the time of visiting q for the $(k+1)$'th time. In (4) this is at most $\lfloor a \rfloor$ steps beyond n. If we walk back $\lfloor a \rfloor$ steps to be within n, then at worst every step is a q and thus $V_q^n(n) \geq k + 1 - \lfloor a \rfloor \geq k - a$. In (5), for $b \geq 0$, $\overline{TS}_q^n(k)$ is at most $\lfloor b \rfloor$ steps before n. If we walk forward $\lfloor b \rfloor$ steps to be beyond n, we can visit q at most $\lfloor b \rfloor$ more times before crossing n and thus $V_q^n(n) \leq k + 1 + \lfloor b \rfloor \leq k + 1 + b$. The case $b < 0$ is trivial. ◀

As final ingredients we need that the moments of the recurrence times are finite, and that the "setup time" to visit state q for the first time is negligible over increasing length infixes.

▶ **Lemma 30.** *The moments of $T_q^{n\,(0)}$ and $S_q^{n\,(r)}$ are finite.*

Proof. By Perron's theorem for positive matrices, the explicit formulas for the recurrence moments derived in [22] converge for finite Markov chains. ◀

▶ **Lemma 31.** $T_q^{n\,(0)}/n \xrightarrow{a.s.} 0$ *as* $n \to \infty$.

Proof. By Lemma 30 the second moments of $T_q^{n\,(0)}$ are finite, and because there are finitely many states there is a constant C such that $\mathbb{E}((T_q^{n\,(0)})^2) \leq C$ for all $n \geq 1$. Thus

$$\mathbb{E}\left(\sum_{n\geq 1} (T_q^{n\,(0)}/n)^2\right) \leq C \sum_{n\geq 1} 1/n^2 < \infty$$

which implies

$$\sum_{n\geq 1} (T_q^{n\,(0)}/n)^2 < \infty \text{ a.s.}$$

and hence $T_q^{n\,(0)}/n \xrightarrow{\text{a.s.}} 0$. ◀

Now we are ready to prove our generalized ergodic theorem.

Proof of Theorem 18. For every n, the $S_q^{n\,(r)}$'s are i.i.d. with expected value m_q. Thus, by Theorem 17,

$$\overline{SS}_q^n(\lceil \tfrac{n}{m_q} \rceil)/\lceil \tfrac{n}{m_q} \rceil \xrightarrow{\text{a.s.}} m_q \qquad \text{as } n \to \infty,$$

i.e., almost surely

$$\forall \varepsilon > 0 \exists \delta \forall n > \delta : m_q - \varepsilon \leq \overline{SS}_q^n(\lceil \tfrac{n}{m_q} \rceil)/\lceil \tfrac{n}{m_q} \rceil \leq m_q + \varepsilon\,.$$

Inside the quantifiers we multiply $\lceil \frac{n}{m_q} \rceil$, add $T_q^{n\,(0)}$, and derive

$$n - (\tfrac{n\varepsilon}{m_q} + \varepsilon - T_q^{n\,(0)}) \leq \overline{TS}_q^n(\lceil \tfrac{n}{m_q} \rceil) \leq n + (\tfrac{n\varepsilon}{m_q} + m_q + \varepsilon + T_q^{n\,(0)})\,.$$

By applying Lemma 29 and dividing by n we obtain

$$\frac{1}{m_q} - \left(\frac{\varepsilon}{m} + \frac{m+\varepsilon}{n} + \frac{T_q^{n\,(0)}}{n}\right) \leq V_q^n(n)/n \leq \frac{1}{m_q} + \left(\frac{\varepsilon}{m} + \frac{2+\varepsilon}{n} + \frac{T_q^{n\,(0)}}{n}\right)\,.$$

By Lemma 31 and suitably chosen ε, the terms in parenthesis can be made arbitrarily small for sufficiently large n. Thus, for any given $\varepsilon' > 0$, almost surely $\frac{1}{m_q} - \varepsilon' \leq V_q^n(n)/n \leq \frac{1}{m_q} + \varepsilon'$ for sufficiently large n, proving the theorem. ◀

D Convergence Rate of the Mode

▶ **Proposition 32.** *Let p be a probability distribution over the alphabet $\{a,b\}$ with $p(a) > p(b)$. Let w be a random ω-word where every letter is i.i.d. according to p. Then $\mathbb{P}(\mathsf{mode}(w_{..n}) = a) \geq 1 - \rho^{\lfloor \frac{n}{2} \rfloor}$, for $\rho = 1 - (2p(a) - 1)^2$.*

Proof. Let us define the series $p_i = \mathbb{P}(|w_{..i}|_a \leq |w_{..i}|_b)$, $q_i = \mathbb{P}(|w_{..i}|_a = |w_{..i}|_b)$, and $r_i = \mathbb{P}(|w_{..i}|_a = |w_{..i}|_b + 1)$, giving the probabilities that a is not more frequent than b, a occurs as often as b, and a occurs once more than b among the first i letters in w, respectively. By definition we have, for all $i \geq 0$:

$$
\begin{aligned}
p_{2i} &= \sum_{j=0}^{i} \binom{2i}{j} (1-p(a))^{2i-j} p(a)^j \\
p_{2i+1} &= \sum_{j=0}^{i} \binom{2i+1}{j} (1-p(a))^{2i+1-j} p(a)^j \\
q_{2i} &= \binom{2i}{i} (1-p(a))^i p(a)^i \\
q_{2i+1} &= 0 \\
r_{2i} &= 0 \\
r_{2i+1} &= \binom{2i+1}{i+1} (1-p(a))^i p(a)^{i+1} .
\end{aligned}
$$

Furthermore, observe that

$$
p_{i+1} = p_i + (1-p(a)) r_i - p(a) q_i . \tag{6}
$$

We show that for each $i > 0$, p_{2i} is dominated by a partial sum of the geometric series with initial value q_{2i} and rate $0 \leq \frac{1-p(a)}{p(a)} < 1$:

$$
\begin{aligned}
p_{2i} &= \sum_{j=0}^{i} \binom{2i}{j} (1-p(a))^{2i-j} p(a)^j \\
&\leq \binom{2i}{i} \sum_{k=0}^{i} (1-p(a))^{i+k} p(a)^{i-k} \\
&= \binom{2i}{i} (1-p(a))^i p(a)^i \sum_{k=0}^{i} \left(\frac{1-p(a)}{p(a)}\right)^k \\
&\leq \binom{2i}{i} (1-p(a))^i p(a)^i \sum_{k=0}^{\infty} \left(\frac{1-p(a)}{p(a)}\right)^k \\
&= q_{2i} \frac{1}{1 - \frac{1-p(a)}{p(a)}} \\
&= q_{2i} \frac{p(a)}{2p(a) - 1} .
\end{aligned}
$$

Thus

$$
q_{2i} \geq \frac{2p(a) - 1}{\sigma} p_{2i} . \tag{7}
$$

We also have $r_{2i+1} = \frac{2i+1}{i+1} p(a) q_{2i}$, which gives us

$$r_{2i+1} \leq 2p(a)q_{2i} \, . \tag{8}$$

Now we show that p_{2i} decreases at a constant rate:

$$\begin{aligned}
p_{2i+2} &= p_{2i} + (1-p(a))r_{2i+1} - p(a)q_{2i} && \text{by (6)} \\
&\leq p_{2i} + 2(1-p(a))p(a)q_{2i} - p(a)q_{2i} && \text{by (8)} \\
&= p_{2i} + p(a)q_{2i} - 2p(a)^2 q_{2i} \\
&= p_{2i} - (2p(a)-1)p(a)q_{2i} \\
&\leq p_{2i} - (2p(a)-1)^2 p_{2i} && \text{by (7)} \\
&= (1-(2p(a)-1)^2)p_{2i} \, .
\end{aligned}$$

Let $\rho = 1 - (2\sigma - 1)^2$. Since $p_0 = 1$ and $0 \leq \rho < 1$, for all $i \geq 0$ we get

$$p_{2i} \leq \rho^i \, , \tag{9}$$

and knowing that $r_{2i} = 0$, we get

$$p_{2i+1} \leq p_{2i} \, , \tag{10}$$

which concludes our proof. ◀