ON THE SAMPLE COMPLEXITY OF LANGUAGE GENERATION IN THE LIMIT WITH NOISE, PROMPTS, AND FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Language generation in the limit is a learning paradigm in which a learner observes an infinite stream of strings from an unknown target language, chosen from a countable collection, and must eventually produce strings not yet seen. This task is always tractable, i.e., for any countable collection, there exists a learner that eventually succeeds. A natural follow-up question is: how many examples does the learner need before it can do so? Previous work established upper and lower bounds for regular, locally threshold testable (LTT), and context-free languages. However, the sample complexity has not been studied under the extensions of the framework that model noisy input, conditioning on a prefix, and feedback. In this paper, we analyze the sample complexity of language generation in each of these settings. First, we study the sample complexity for partially-ordered (PO) languages, a class not previously analyzed in this framework, and we establish exponential upper and lower bounds. Next, we show that for context-free languages, the sample bounds remain uncomputable in every setting. Finally, we show that for regular, LTT, and PO languages, finite noise increases the sample complexity by only a linear additive term, prompting generally leaves the bounds unchanged, and finite feedback can sometimes reduce them exponentially.

1 INTRODUCTION

Language identification in the limit is an influential learning paradigm (Gold, 1967; Angluin, 1980). In the classic Gold–Angluin model, an adversary enumerates an infinite stream of strings from a target language K, chosen from a countable collection of languages $\mathcal{C}$. At each time step, the learner observes the strings enumerated so far and outputs a hypothesis from $\mathcal{C}$ as its guess for the target language; it succeeds if there exists some finite timestep t after which its hypotheses stabilize to the correct language K. Identification in the limit is impossible in general, and remains so even for relatively simple language classes such as the regular languages (Angluin, 1980). Recently, Kleinberg & Mullainathan (2024) introduced the paradigm of language *generation* in the limit, inspired by modern language models. Rather than identifying the target language K, the learner must produce, after some finite timestep t, a previously unseen string from K. The authors proved that this problem is significantly easier than identification in the limit, showing that it is tractable for every countable collection $\mathcal{C}$.

This tractability result sparked significant follow-up work. Raman et al. (2025) introduced *uniform* generation in the limit, a stricter variant requiring the learner to succeed after seeing a number of strings that is uniform across all languages and enumerations, and showed that this stronger requirement makes the problem intractable for some countable collections. Subsequent work has extended the framework along several axes: prompted generation (Kleinberg & Mullainathan, 2024; Raman et al., 2025), noisy input (Raman & Raman, 2025; Mehrotra et al., 2025; Bai et al., 2026), omissions (Mehrotra et al., 2025; Bai et al., 2026), and feedback (Charikar & Pabbaraju, 2024; Bai et al., 2026), some within the uniform generation setting. Other work compared language generation in the limit with established learning theory frameworks such as PAC or online learning (Raman et al., 2025), or studied language generation in a statistical setting (Kalavasis et al., 2025); we discuss these in §A.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Sample complexity of uniform generation for regular, locally threshold testable (**LTT**), partially-ordered (**PO**), and context-free languages across the base setting of Kleinberg & Mullainathan (2024), generation with noise, prompted generation, and generation with feedback. We sometimes give the bounds in terms of the base bound B. We use s to denote the largest size of the representation of each language, k the noise level, p the prompt, and f the feedback budget. The notation $x \mapsto y$ denotes that x gets replaced by y in the base bound.

Language class	**Base** (B)	**Noise** k	**Prompt** p	**Feedback** f
Regular	$\lvert\Sigma\rvert^{s^{\lvert\mathcal{C}\rvert}}$	$B + k\lvert\mathcal{C}\rvert$	B	$\lvert\Sigma\rvert^{s^{(\lvert\mathcal{C}\rvert - f)}}$
LTT	$\lvert\Sigma\rvert^{3r(r^2+1)^2}, r = s\lvert\mathcal{C}\rvert$	$B + k\lvert\mathcal{C}\rvert$	$r \mapsto (s+\lvert p\rvert)\lvert\mathcal{C}\rvert$	$r \mapsto s(\lvert\mathcal{C}\rvert - f)$
PO	$\lvert\Sigma\rvert^{\lvert\mathcal{C}\rvert(s-1)+1}$	$B + k\lvert\mathcal{C}\rvert$	B	$\lvert\Sigma\rvert^{(\lvert\mathcal{C}\rvert - f)(s-1)+1}$
Context-free	uncomputable	uncomputable	uncomputable	uncomputable

A natural and important question is: how many examples does the learner need before it can generate correctly? This question is most meaningful in the uniform generation setting, where the bound must hold irrespective of the target language or the enumeration. Arenas et al. (2026) answered this question for finite collections, establishing bounds for several language classes: double exponential for regular languages, exponential for locally threshold testable (LTT) languages, and uncomputable for context-free languages. While these classes provide useful theoretical insights, their bounds carry large exponents, and no class is known to do better. Moreover, even for the language classes studied by Arenas et al. (2026), the sample complexity under noise, prompting, and feedback is unknown.

We first analyze the sample complexity of generation for what we call partially-ordered (PO) languages, a class not previously studied in this framework. By exploiting the structure of the automaton we establish single-exponential upper and lower bounds with a smaller exponent than for LTT languages. Next, we study the sample complexity in each extension of the framework. We begin by recasting prompted generation and generation with feedback such that they align closer to how language models are used in practice. Where Raman et al. (2025) treat a prompt as a label attached to an example, we follow Kleinberg & Mullainathan (2024) and treat it as a prefix, with the learner producing a continuation. Charikar & Pabbaraju (2024) allow one feedback query per timestep; instead, we follow Bai et al. (2026) in fixing a total query budget. We then define uniform generation in each of these settings, characterize which collections admit uniform generation, and show that uniform generation is always possible for finite collections. This is a necessary stepping stone without which the sample complexity bounds cannot be derived.

We then analyze the sample complexity across all settings and language classes (see Table 1). For context-free languages, the sample complexity remains uncomputable regardless of the setting. For regular, LTT, and PO languages, the picture is more nuanced: finite noise increases the sample complexity by only a linear additive term, prompted generation generally leaves the bounds unchanged, and finite feedback can sometimes reduce them exponentially.

Our contributions can be summarized as follows: (i) we isolate the arguments of Arenas et al. (2026) into a recipe for deriving sample complexity bounds (§4), (ii) we give single-exponential upper and lower bounds for PO languages (§5); (iii) we reformulate uniform prompted generation and generation with feedback, and we give definitions of uniform generatability in each setting, closure dimensions characterizing which collections admit it, and proofs that finite collections always do (§7,§8), (iv) we give upper and lower bounds for sample complexity of generation with finite noise, prompts, and finite feedback for regular, LTT, and PO languages, and (v) we show that the bounds for context-free languages are uncomputable in each setting.

2 Preliminaries

Prior work has established sample complexity bounds for generation in the limit for several well-known classes and subclasses of formal languages (Arenas et al., 2026). We define these here.

Let Σ be a finite **alphabet** of symbols. A **string** over Σ is a sequence of symbols $x = a_1 a_2 \cdots$, where $a_i \in \Sigma$ for all i. For a string $x = a_1 a_2 \cdots a_k$, we use $|x|$ to denote its length k. Let ε denote the **empty string**. A **formal language** $L \subseteq \Sigma^*$ over the alphabet Σ is a subset of the

Kleene closure Σ^*, i.e., the set of all finite strings over the alphabet Σ. A central quantity in our analysis is the intersection of formal languages. For two languages L_1 and L_2, their **intersection** is $L_1 \cap L_2 = \{x \colon x \in L_1 \text{ and } x \in L_2\}$.

2.1 Regular languages

Regular languages are among the simplest and most extensively studied language class, characterized equivalently by finite automata and regular expressions.

Definition 1. *A **finite automaton** (FA) is a tuple $\mathcal{A} = (\Sigma, Q, I, F, \delta)$, where Σ is a finite **alphabet** of symbols, Q is a finite set of **states**, $I \subseteq Q$ is a set of **initial states**, $F \subseteq Q$ is a set of **final states**, and $\delta \subseteq Q \times \Sigma \cup \times Q$ is a finite set of **transitions**.*

Definition 2. *An FA is called **deterministic FA** (DFA) if I consists of at most one initial state q_0, and, for each pair $(q, a) \in Q \times \Sigma$, there exists at most one state q' such that $(q, a, q') \in \delta$; otherwise, it is called **non-deterministic FA** (NFA).*[1]

A **path** in an FA is a sequence $\pi = (q_0, a_1, q_1), (q_1, a_2, q_2), \ldots, (q_{n-1}, a_n, q_n)$, where $(q_{i-1}, a_i, q_i) \in \delta$ for all $i \in [1, n]$. If $q_0 \in I$ and $q_n \in F$, the path π is called **accepting**. The path π is called **simple** if all its states are distinct; otherwise, it has a **cycle**. FAs recognizing finite languages do not have cycles. The **length** of π is the number of transitions it contains. If $x = a_1 \cdots a_n \in \Sigma^*$ for such an accepting path, then $\mathcal{A}$ **recognizes** the string x. In an FA without cycles, the length of the longest simple path coincides with the length of the longest recognized string. The **language** $L(\mathcal{A})$ of an FA $\mathcal{A}$ is the set of all strings it recognizes. A language is **regular** if there is some FA recognizing it. Any NFA can be converted into an equivalent DFA, but this transformation may incur an exponential blowup in the number of states (Hopcroft et al., 2006). The intersection of two regular languages is also regular, and it can be computed via the **product construction** (Hopcroft et al., 2006), which constructs an NFA that simulates both NFAs in parallel. The resulting product NFA has at most as many states as the product of the sizes of the two original NFAs.

2.2 Locally threshold testable languages

Locally threshold testable (LTT) languages are a well-studied subclass of the regular languages. They admit several equivalent characterizations, including one in terms of profile sequences.

Definition 3. *A **profile** over an alphabet Σ is a tuple $P = (Pr, Su, In)$, where Pr and Su are a prefix and a suffix, respectively, and In is a finite set of constraints of the form $(x, \theta t)$, where $x \in \Sigma^*$, $\theta \in \{\leq, <, =, \geq, >\}$, and $t \in \mathbb{N}$, given in unary notation.*[2]

The language $L(P)$ defined by the profile P is the set of all strings $x \in \Sigma^*$ such that Pr is a prefix of x, Su is a suffix of x, and for every $(y, \theta t) \in In$, the number of occurrences of the substring y satisfies the constraint θt. A language L is **locally threshold testable** (LTT) if there exists a finite sequence of profiles $(P_1, \ldots, P_n)$ such that $L = \bigcup_{i=1}^{n} L(P_i)$. The **size** of a profile is $|P| = |Pr| + |Su| + \sum_{(x,\theta t) \in In} (|x| + t)$. Every string in the language of a profile of size s that defines a finite language has length less than $3s(s^2 + 1)^2$ (Arenas et al., 2026, Lemma 5.2).

2.3 Context-free languages

Context-free languages are another important language class, capturing hierarchical structure that simpler classes cannot express. Context-free grammars are the canonical formalism for defining them; we define them formally and give an example in §B.1. Computing the size of the intersection of two context-free languages is undecidable, and Arenas et al. (2026) show that this carries over to the sample complexity via a reduction from the halting problem: a computable bound would allow deciding whether a Turing machine halts, contradicting undecidability (Hopcroft et al., 2006). We refer the reader to §B.2 for background on Turing machines and the halting problem. We build on this reduction to establish uncomputability in all of our settings.

[1] Generally, NFAs can have transitions of the form (q, ε, q') that do not scan any symbol. Such transitions can be easily eliminated without increasing the number of states.

[2] This means that t is written as a string a^t for some symbol $a \in \Sigma$.

3 UNIFORM LANGUAGE GENERATION IN THE LIMIT

The setup. Let $\mathcal{C} = \{L_1, L_2, \ldots\}$ be a **countable collection** of languages over a **countable domain** $\mathcal{X}$, i.e., each L_i is a subset of $\mathcal{X}$ ($L_i \subseteq \mathcal{X}$). Throughout the rest of the paper, we assume the domain is Σ^*. All languages L_i are *infinite*. An adversary and a learner play the following game. The **adversary** chooses a **target** language $K = L_z \in \mathcal{C}$ and enumerates strings x_t from it, one by one, over a sequence of steps $t = 1, 2, 3, \ldots$. At timestep t, the **enumeration** of strings shown so far is denoted by S_t. Every string in K must appear at some timestep t, but repetitions are allowed. In **noisy generation**, the enumeration may contain noisy examples $x \notin K$, but it still needs to cover all of K. We sometimes use S_t to denote the set of distinct strings in the enumeration S_t. In each step, the **learner** G observes the enumeration S_t and must output one *new* string $a_t = G(S_t)$ from K. In **generation with feedback**, the learner is allowed to ask whether a string x belongs to K and the adversary must answer truthfully. In **prompted generation**, the adversary additionally shows a prompt $p \in \mathcal{P}$ and the learner must output a new continuation x such that $p \cdot x \in K$. Let $S = \bigcup_{t=1}^{\infty} S_t$. A learner **generates in the limit** if for any target language K in $\mathcal{C}$ and any enumeration S, there is a timestep t^* such that for all steps $t \geq t^*$, the algorithm's guess is correct (Kleinberg & Mullainathan, 2024).

Throughout, we assume the following: (i) the target language K belongs to the collection $\mathcal{C}$, (ii) membership in a language is binary, (iii) every language in $\mathcal{C}$ is infinite, (iv) $\mathcal{C}$ is finite, (v) feedback is truthful, and (vi) the prompt domain $\mathcal{P}$ is finite. Assumptions (iv) and (vi), although restrictive, are essential for deriving the sample complexity bounds.

Uniform generation in the limit. The timestep t^* can depend both on the target language K, as well as the enumeration S. In follow-up work, Raman et al. (2025) introduced **uniform generation**, which requires this timestep to be *uniform* across all languages and enumerations.

Definition 4 (Raman et al., 2025). *A collection $\mathcal{C}$ is **uniformly generatable in the limit** if there is an algorithm G and a sample threshold m such that for any language $K \in \mathcal{C}$, any enumeration of strings $x_1, x_2, \ldots \in K$, if there exists a timestep t^* such that $|S_{t^*}| = m$, then for all $t \geq t^*$ it holds that $a_t \in K \setminus S_t$.*

Unlike generation in the limit, *uniform* generation in the limit is not always possible: Raman et al. (2025) showed which collections admit it using the following characterization.

Definition 5 (Raman et al., 2025). *For any collection of languages $\mathcal{C}$ and enumeration $S_n = x_1, x_2, \ldots, x_n$, let the set $\mathcal{C}(S_n) = \{L \in \mathcal{C} \colon S_n \subseteq L\}$ be the set of **consistent languages** with the enumeration S_n. Then, we define the **closure** $\langle \cdot \rangle_{\mathcal{C}}$ as*

$$\langle S_n \rangle_{\mathcal{C}} = \begin{cases} \bigcap_{L \in \mathcal{C}(S_n)} L, & \text{if } \mathcal{C}(S_n) \neq \emptyset \\ \perp, & \text{if } \mathcal{C}(S_n) = \emptyset. \end{cases} \tag{1}$$

Intuitively, the closure is the intersection of languages that are consistent with the examples seen so far.

Definition 6 (Raman et al., 2025). *The **closure dimension** $\mathrm{C}(\mathcal{C})$ of $\mathcal{C}$ is the supremum over $d \in \mathbb{N}$ for which there exist distinct strings $x_1, x_2, \ldots, x_d \in \mathcal{X}$ such that $\langle \{x_1, x_2, \ldots, x_d\} \rangle_{\mathcal{C}} \neq \perp$ and $|\langle \{x_1, x_2, \ldots, x_d\} \rangle_{\mathcal{C}}| < \infty$. If it is not true for $d = 1$, we set $\mathrm{C}(\mathcal{C}) = 0$, and if the set of such d is unbounded, we set $\mathrm{C}(\mathcal{C}) = \infty$.*

The sample complexity of uniform generation is only well-defined for collections that are uniformly generatable. Raman et al. (2025) gave the following characterization.

Theorem 1 (Raman et al., 2025). *If $\mathrm{C}(\mathcal{C}) < \infty$, then $\mathcal{C}$ is uniformly generatable.*

A finite closure dimension guarantees that once the learner has seen enough distinct examples, the intersection of all consistent languages is always infinite, so the learner can never be trapped with only finitely many valid strings to output. For finite collections, the closure dimension is always finite since there are only finitely many possible intersections (Raman et al., 2025). We say that a collection is **m-generatable** if uniform generation succeeds after observing m distinct examples.

4 A RECIPE FOR DERIVING SAMPLE COMPLEXITY BOUNDS

All the sample complexity bounds for uniform generation rest on the following characterization, which reduces m-generatability to a structural question about the language class.

Proposition 1 (Arenas et al., 2026). *A collection $\mathcal{C}$ of infinite languages is* not *m-generatable if and only if there exists a nonempty subset $\mathcal{S} \subseteq \mathcal{C}$ such that $\bigcap_{L \in \mathcal{S}} L$ is finite and has size at least m.*

It therefore suffices to bound the size of the largest finite intersection of languages in the collection. Every bound in this paper and in Arenas et al. (2026) uses the following four steps.

(i) **Close the representation under intersection.** Fix a representation (e.g., automaton, profile, etc.) for the class and a construction mapping representations of $L_1, \ldots, L_n$ to a representation of $\bigcap_i L_i$, together with a bound $\sigma(s, n)$ on the size of the result when each input has size at most s. For instance, for automata this is the product construction, with $\sigma(s, n) = s^n$ states.

(ii) **Bound the longest string.** Show that, if the intersection is finite, then every string it contains has length at most ℓ, depending on $\sigma(s, n)$. Finiteness is what makes this possible. For automata, it forbids cycles, so the longest accepted string is bounded by the longest simple path.

(iii) **Count strings.** A language over an alphabet Σ with $|\Sigma| \geq 2$, whose strings have length at most ℓ, contains at most $\sum_{i=0}^{\ell} |\Sigma|^i = \frac{|\Sigma|^{\ell+1}-1}{|\Sigma|-1} < |\Sigma|^{\ell+1}$ strings.

(iv) **Apply Prop. 1.** The collection is m-generatable for any m exceeding the bound from item (iii).

Notice that only item (ii) is class-specific; item (i), item (iii) and item (iv) are generic. Moreover, item (i) only applies to representations that are closed under intersection. For a collection $\mathcal{C}$, we write $B(\mathcal{C}) = |\Sigma|^{\ell+1}$, where ℓ is the length bound from item (ii), so that every finite intersection of a non-empty subcollection of $\mathcal{C}$ has size less than $B(\mathcal{C})$. By Prop. 1, $\mathcal{C}$ is $B(\mathcal{C})$-generatable.

The role of the right representation. Because ℓ carries the entire bound, the choice of representation matters more than its compactness. Regular languages are far more succinctly described in monadic second-order (MSO) logic than by automata, but the longest string in a finite intersection of MSO-definable languages is an exponential tower (Hella & Vilander, 2022), so the resulting ℓ is vastly worse than the automaton-based one. For PO languages, we do not change the representation, but exploit additional structure inside it to bound the longest string. Therefore, the right representation is the one that most tightly bounds ℓ, and it depends on the language class.

We briefly summarize the constructions of Arenas et al. (2026) for regular, LTT and context-free languages:

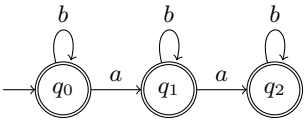
- **Regular languages:** The product of $|\mathcal{C}|$ NFAs with at most s states each has up to $s^{|\mathcal{C}|}$ states. If the intersection is finite, no accepting path can traverse a cycle, so every accepting path is simple and visits each state at most once. The longest accepted string is then bounded by the number of states, i.e., $\ell = s^{|\mathcal{C}|} - 1$. Counting strings of up to this length gives a double exponential bound.
- **LTT languages:** Intersecting profiles amounts to collecting their prefix, suffix and counting constraints together, so $|\mathcal{C}|$ profiles of size at most s yield a profile of size at most $s|\mathcal{C}|$. Finiteness then forces the counting constraints to cap the number of occurrences of every substring, which caps the length of any string in the language: a profile of size r defining a finite language admits no string longer than $3r(r^2+1)^2$ (Arenas et al., 2026, Lemma 5.2). With $r = s|\mathcal{C}|$, the length bound ℓ is polynomial in s and $|\mathcal{C}|$, and the sample complexity drops to a single exponential.
- **Context-free languages:** The size of the intersection of two context-free languages is not computable from the grammars: one can construct two grammars whose intersection has size exactly 2^t, where t is the number of steps a given Turing machine takes to halt (Arenas et al., 2026, Lemma 3.2). A computable length bound would yield a computable bound on t and decide the halting problem.

5 Sample complexity of generation for partially-ordered languages

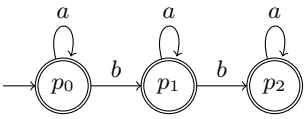
Partially-ordered languages are another subclass of the regular languages that, as we show, admit a single-exponential bound with the smallest exponent in this framework.

The **reachability relation** on a set of states Q is defined $q \leq q'$ if there exists a path from q to q'.

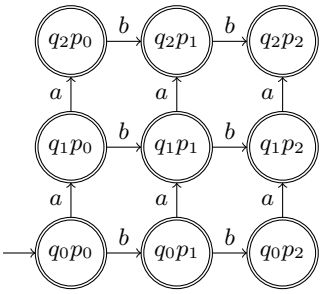
Definition 7. *An NFA is a **partially-ordered NFA** (PONFA) if $\leq$ is a partial order.*



(a) PONFA recognizing language $L_1 = \{x\colon x \text{ has at most two } a\text{s}\}$.



(b) PONFA recognizing language $L_2 = \{x\colon x \text{ has at most two } b\text{s}\}$.



(c) PONFA recognizing $L_1 \cap L_2 = \{x\colon x \text{ has at most two } a\text{s and at most two } b\text{s}\}$.

Figure 1: Two PONFAs and their product PONFA. The longest simple path has length $2 \times 2 = 4$.

We call a language **partially-ordered** (PO) if and only if it can be recognized by a PONFA. The intersection of two PO languages is PO (see Figure 1 for an illustration). Using Prop. 1, we derive the following upper bound.

Theorem 2. *Let $\mathcal{C}$ be a finite collection of infinite PO languages, over an alphabet of size $|\Sigma|$ such that $|\Sigma| \geq 2$, each recognized by a PONFA with at most s states. Then $\mathcal{C}$ is $|\Sigma|^{|\mathcal{C}|(s-1)+1}$-generatable.*

Proof sketch. The product of PONFAs is also PO, with states $(q_1, \ldots, q_{|\mathcal{C}|}) \in Q_1 \times \cdots \times Q_{|\mathcal{C}|}$, where Q_i is the set of states of the i^{th} PONFA. Since the intersection is finite, every accepting path is simple. Because of the partial order, any transition has the form $(q_1, \ldots, q_{|\mathcal{C}|}) \xrightarrow{a} (p_1, \ldots, p_{|\mathcal{C}|})$, where $q_i \leq p_i$ for all i, and $q_i \neq p_i$ for at least one i. That is, the path must at each step transition to a strictly higher tuple of states, therefore at least one automaton must transition to a strictly higher state in its own partial order (see Figure 1 for an example). Since each PONFA has at most s states, each component can increase at most $s-1$ times, so the longest accepted string has length at most $|\mathcal{C}|(s-1)$. The intersection therefore has size less than $|\Sigma|^{|\mathcal{C}|(s-1)+1}$, and the bound follows from Prop. 1. The full proof is in §C. □

In §C, we give a collection whose intersection has more than $|\Sigma|^{|\mathcal{C}|(s-1)}$ strings, so the exponent cannot be lowered.

6 Sample complexity for generation with finite noise

Enumerations may inevitably contain errors, as any real-world corpus would. We study the sample complexity of uniform generation when the learner is guaranteed that at most k of the observed examples are noisy. For regular, LTT, and PO languages, introducing noise adds only a linear term in k to the sample complexity. For context-free languages, the uncomputability barrier is preserved. We build on the machinery of Raman & Raman (2025), who introduced a notion of closure for the noisy setting.

Definition 8 (Raman & Raman, 2025)**.** *For any collection of languages $\mathcal{C}$, a noisy enumeration $S_n = x_1, x_2, \ldots, x_n$, and a noise level k, let $\mathcal{C}(S_n, k) = \{L \in \mathcal{C}\colon |S_n \cap L| \geq |S_n| - k\}$ be the set of consistent languages up to k errors. Then we define the **noisy closure** $\langle \cdot, k \rangle_{\mathcal{C}}$ at noise level k as*

$$\langle S_n, k \rangle_{\mathcal{C}} = \begin{cases} \bigcap_{L \in \mathcal{C}(S_n,k)} L, & \text{if } \mathcal{C}(S_n,k) \neq \emptyset \\ \bot, & \text{if } \mathcal{C}(S_n,k) = \emptyset. \end{cases} \tag{2}$$

A noisy closure at level k is the intersection of all languages consistent with the enumeration up to k errors. We say a collection is m-generatable at noise level k if uniform generation succeeds after observing m distinct examples, at most k of which are noise. We defer the formal definitions and generatability results of Raman & Raman (2025) to §D. We give the following finite-noise analogue of Proposition 2.3 of Arenas et al. (2026).

Proposition 2. *A finite collection $\mathcal{C}$ of infinite languages is* not *uniformly m-generatable at noise level $k \in \mathbb{N}$ if and only if there exists an enumeration $S_t = x_1, x_2, \ldots, x_t$ with $|\{x_1, \ldots, x_t\}| = m$ such that $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k\rangle_{\mathcal{C}}$ is finite and has size at least $m - k|\mathcal{C}|$.*

We give a proof in §D. For all language classes, the noisy bounds follow by applying the same intersection argument as in the noiseless setting. By Prop. 2, generatability fails only when the noisy closure has size at least $m - k|\mathcal{C}|$, so it suffices to take m larger than the noiseless closure bound plus an additive $k|\mathcal{C}|$ term. We formalize this intuition in the following theorem.

Theorem 3. *Let $\mathcal{C}$ be a finite collection of infinite languages that is $B(\mathcal{C})$-generatable. Then $\mathcal{C}$ is $(B(\mathcal{C}) + k|\mathcal{C}|)$-generatable at noise level k.*

Proof sketch. By Prop. 2, it suffices to show that whenever $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k\rangle_{\mathcal{C}}$ is finite, its size is below $m - k|\mathcal{C}|$. The noisy closure is the intersection of the languages in $\mathcal{C}(S_t, k)$, a subset of $\mathcal{C}$, so the noiseless bound applies and $|\langle S_t, k\rangle_{\mathcal{C}}| < B(\mathcal{C})$. Taking $m \geq B(\mathcal{C}) + k|\mathcal{C}|$ makes the condition of Prop. 2 unsatisfiable. We give the full proof in §D. □

Noise is therefore additive and linear in k, uniformly across all classes. Instantiating $B(\mathcal{C})$ with the bounds of Arenas et al. (2026), and our bound for PO languages from §5 gives the following: $|\Sigma|^{s^{|\mathcal{C}|}} + k|\mathcal{C}|$ for regular languages, $|\Sigma|^{3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2} + k|\mathcal{C}|$ for LTT, and $|\Sigma|^{|\mathcal{C}|(s-1)+1} + k|\mathcal{C}|$ for PO languages. For context-free languages, the bound remains uncomputable. We give upper and lower bounds for all language classes in §D.

7 Sample complexity for uniform prompted generation

We study how conditioning on a prefix affects the sample complexity of generation. Rather than treating prompted generation as multiclass classification, where each example carries a prompt label (Raman et al., 2025), we follow Kleinberg & Mullainathan (2024) and treat a prompt as a prefix that the output continues. Thus, the prompt $p \in \mathcal{P}$ selects a language via the projection $L^p = \{x \in \Sigma^* : p \cdot x \in L\}$, the set of valid continuations of p in L. The machinery of Raman et al. (2025) for uniform prompted generation therefore does not transfer; we build it again for our setting below.

Definition 9. *For any collection of languages $\mathcal{C}$, a sequence of examples $S_n = x_1, x_2, \ldots, x_n$, and a prompt $p \in \mathcal{P}$, let $\mathcal{C}(S_n, p) = \{L \in \mathcal{C} : S_n \subseteq L$ and $L^p \neq \emptyset\}$ be the set of languages consistent with the enumeration. We define the* ***prompted closure*** *$\langle \cdot, \cdot \rangle_{\mathcal{C}}$ as*

$$\langle S_n, p\rangle_{\mathcal{C}} = \begin{cases} \bigcap_{L \in \mathcal{C}(S_n, p)} L^p, & \text{if } \mathcal{C}(S_n, p) \neq \emptyset \\ \bot, & \text{if } \mathcal{C}(S_n, p) = \emptyset. \end{cases} \tag{3}$$

Without a guarantee that a prompt admits arbitrarily many continuations, a learner can be trapped with no new string to output. To forbid this, we adopt the robustness condition of Kleinberg & Mullainathan (2024).

Assumption 1. *For every language $L \in \mathcal{C}$ and every prompt $p \in \mathcal{P}$, the language L^p is infinite.*

The language L^p is the standard left quotient of L by p, and whether it is infinite is decidable for all language classes we study. For instance, for PO languages, the left quotient is recognized by the automaton obtained by shifting the initial states to those reachable by reading p, preserving the partial order structure and generally leaving the number of states unchanged.

Formal definitions, a characterization of prompted uniformly generatable collections, and all proofs are deferred to §E. Here we use only the notion of m-prompted uniform generatability: a collection is m-prompted generatable with prompt p if uniform generation succeeds after observing m distinct examples with prompt p.

Corollary 1. *Let $\mathcal{P}$ be finite. Any finite collection $\mathcal{C}$ of infinite languages satisfying Assumption 1 is prompted uniformly generatable.*

The finiteness of the prompt domain $\mathcal{P}$ is essential. When $\mathcal{P}$ is infinite, even a collection consisting of only two languages may fail to be prompted uniformly generatable (see §E).

Proposition 3. *Let $\mathcal{P}$ be finite. A finite collection $\mathcal{C}$ of languages satisfying Assumption 1 is* not *m-prompted uniformly generatable for prompt $p \in \mathcal{P}$ if and only if there exists a non-empty set $\mathcal{S} \subseteq \mathcal{C}$ such that $\bigcap_{L \in \mathcal{S}} L^p$ is finite and has size at least m.*

Theorem 4. *Let $\mathcal{P}$ be finite and let $\mathcal{C}$ be a finite collection of infinite languages satisfying Assumption 1. Then for every $p \in \mathcal{P}$, $\mathcal{C}$ is $B(\{L^p : L \in \mathcal{C}\})$-prompted uniformly generatable with prompt p.*

Proof sketch. By Prop. 3, the sample complexity is governed by the largest finite intersection $\bigcap_{L \in \mathcal{S}} L^p$ over non-empty $\mathcal{S} \subseteq \mathcal{C}$. This is exactly the quantity the recipe of §4 bounds, applied to the collection of left quotients rather than to $\mathcal{C}$ itself, and Assumption 1 guarantees the quotients are infinite, so the recipe is applicable. The full proof is in §E. □

The bound therefore depends only on how the left quotient affects the language representation: for regular and PO languages it is unchanged, since the quotient is recognized by an automaton with at most the same number of states at the original, for LTT languages it gains an additive dependence on $|p|$, and for context-free languages uncomputability is preserved. All constructions and bounds are given in §E.

8 Sample complexity for uniform generation with finite feedback

Bai et al. (2026) prove that generation with finite feedback is equivalent in power to generation without feedback; we show that a generator with a query budget of f can nonetheless achieve uniform generation after seeing fewer examples. We follow Bai et al. (2026) in allowing a total budget of f queries, assuming $|\mathcal{C}| - 1 > f$, since otherwise queries alone suffice to eliminate enough languages and generate without any examples. This differs from Charikar & Pabbaraju (2024), who allow one query per timestep. We define some analogous notions below.

Definition 10. *For any collection of languages $\mathcal{C}$, a sequence of examples $S_n = x_1, x_2, \ldots, x_n$ and a sequence of queries $Q_f = q_1, q_2, \ldots, q_f$ with answers consistent with some target language $K \in \mathcal{C}$, let $\mathcal{C}(S_n, Q_f) = \{L \in \mathcal{C} : S_n \subseteq L$ and $\mathbb{1}\{q_i \in L\} = \mathbb{1}\{q_i \in K\}, \forall i \in [1, f]\}$. We define the* ***feedback-augmented closure*** *$\langle \cdot, \cdot \rangle_{\mathcal{C}}$ as*

$$\langle S_n, Q_f \rangle_{\mathcal{C}} = \begin{cases} \bigcap_{L \in \mathcal{C}(S_n, Q_f)} L, & \text{if } \mathcal{C}(S_n, Q_f) \neq \emptyset \\ \perp, & \text{if } \mathcal{C}(S_n, Q_f) = \emptyset. \end{cases} \tag{4}$$

We adapt the definitions of uniform generatability with feedback and the characterization of uniformly generatable collections from Charikar & Pabbaraju (2024) to our fixed-budget setting. We say a collection is m-generatable with f feedback queries if uniform generation succeeds after observing m distinct examples, using at most f membership queries to the target language. Without this, uniform sample complexity bounds could not be derived. Formal definitions and proofs are given in §F.

Proposition 4. *A finite collection $\mathcal{C}$ of infinite languages is not uniformly m-generatable with f feedback queries iff for every generator G that asks at most f feedback queries, there exist a target $K \in \mathcal{C}$, an enumeration $S_t = x_1, \ldots, x_t$ of K with $|\{x_1, \ldots, x_t\}| = m$, and answers consistent with K to the queries Q_f asked by G, such that $\langle S_t, Q_f \rangle_{\mathcal{C}}$ is finite and has size at least m.*

The role of feedback queries. By Prop. 4, the sample complexity is determined by the size of the feedback-augmented closure. Feedback queries help by eliminating languages from the consistent set, potentially making the intersection infinite. If the query budget suffices to eliminate the right languages regardless of query answers, the generator needs no examples at all. We show that by choosing informative queries that eliminate as many languages as possible from the consistent set, the closure size, and hence the sample complexity, can be reduced substantially. A simple strategy eliminates one language per query, reducing the effective collection size by f. One might hope to do better by halving the consistent set with each query, but such queries need not exist, as we show below.

A simple strategy. Given a query budget f, the generator picks any two languages L_1, L_2 in the consistent set and queries a string $q \in L_1 \triangle L_2 = (L_1 \setminus L_2) \cup (L_2 \setminus L_1)$. Since all languages in $\mathcal{C}$ are distinct, such a string always exists, and regardless of the answer, at least one language is eliminated. Repeating this f times leaves at most $|\mathcal{C}| - f$ languages in the consistent set. We give the following generic theorem; we instantiate it for regular, LTT and PO languages, and prove uncomputability for context-free languages in §F.

Theorem 5. *Let $\mathcal{C}$ be a finite collection of infinite languages and let $f < |\mathcal{C}| - 1$. Then $\mathcal{C}$ is $\max\{B(\mathcal{C}') : \mathcal{C}' \subseteq \mathcal{C}, |\mathcal{C}'| = |\mathcal{C}| - f\}$-generatable with f feedback queries.*

Proof sketch. By Prop. 4, feedback helps only by shrinking the consistent set. At each of the f steps the generator picks two languages L_1, L_2 from the current consistent set and queries a string in $L_1 \triangle L_2$, which is non-empty since the languages are distinct; whatever the answer, at least one of them is eliminated. After f queries at most $|\mathcal{C}| - f$ languages remain consistent, and the recipe of §4 applied to that smaller collection gives the bound. The full proof is in §F. □

A better strategy. Instead of separating two languages at a time, partition the consistent set $\mathcal{C}$ into two roughly equal halves $\mathcal{C}_1$ and $\mathcal{C}_2$, and query a string $q \in (\bigcap_{L \in \mathcal{C}_1} L) \setminus (\bigcup_{L \in \mathcal{C}_2} L)$. Every language in $\mathcal{C}_1$ contains q and none in $\mathcal{C}_2$ does, so whichever way the oracle answers, one half is eliminated entirely, reducing the consistent set to approximately $|\mathcal{C}|/2$ with a single query. Repeating this f times reduces the consistent set to approximately $|\mathcal{C}|/2^f$. However, no such string q may exist, making this strategy impossible. We give an example below for LTT languages.

Theorem 6. *There exists a finite collection of n infinite LTT languages $\mathcal{C}$ over an alphabet of size $|\Sigma| = n \geq 4$, such that $\mathcal{C}$ is* not *2^n-generatable with f feedback queries, for any f such that $f < |\mathcal{C}| - 1$.*

Proof sketch. Let $\mathcal{C}$ consist of n LTT languages over $\Sigma = \{1, \ldots, n\}$, for $n \geq 4$, and $L_i = (\varepsilon, \varepsilon, \{(j, = 1) : j \neq i\})$, for all i. L_i constrains every symbol except i to appear exactly once, and is infinite since i can appear any number of times. For any subset $\mathcal{S} \subseteq \mathcal{C}$ with $|\mathcal{S}| \geq 2$, every symbol is constrained in at least one language, forcing all n symbols to appear once. The intersection is therefore always the same: all permutations of $\{1, \ldots, n\}$, of size $n! \geq 2^n$ for $n \geq 4$.No query eliminates more than one language: a string whose symbols all occur exactly once belongs to every L_i, a string where exactly one symbol i occurs a different number of times belongs to L_i only, and any other string belongs to none. Since $f < |\mathcal{C}| - 1$, at least two languages remain consistent and the closure is the set of permutations. Therefore, $\mathcal{C}$ is not 2^n-generatable with f feedback queries, for any f such that $f < |\mathcal{C}| - 1$. □

Selecting informative queries requires computing differences, and hence complementation, which for NFAs incurs an exponential blowup via determinization (Hopcroft et al., 2006). This overhead affects query selection only. Once languages are eliminated, the remaining collection is bounded using its original representation, so the sample complexity bounds are unaffected.

9 Discussion and conclusion

We studied the sample complexity of uniform language generation for context-free, regular, LTT, and PO languages, in extensions of the language generation in the limit framework that model noisy input, prompting, and feedback. We found that PO languages, whose sample complexity had not been studied before, require an exponential number of samples. A recurring theme is that finite noise adds only a linear term, prompting generally leaves the bounds unchanged, and finite feedback can reduce them exponentially, while the bounds for context-free languages remain uncomputable regardless of the setting.

Some of our assumptions are strong, and a language model trained on a real corpus may not satisfy them. However, these assumptions are reasonable in scenarios such as constrained generation. Here, outputs must be valid against a schema or grammar, membership is indeed binary, there are only a few, fixed candidate languages, and the oracle is a validator rather than a human, so feedback is truthful and cheap. Additionally, our bounds are exponential in $|\mathcal{C}|$, so they are meaningful only when the collection is small. The exponent therefore matters: the smaller one we give for PO languages allows a considerably larger collection before the bound becomes vacuous. The finiteness of the prompt domain is the most restrictive assumption, though it holds when prompts are drawn from a fixed set of templates. The uncomputability result is the most consequential: for any collection at least as expressive as context-free languages, a uniform bound exists but cannot be computed, so one can never know in advance how many examples suffice.

AI USE STATEMENT

We used AI tools to assist with drafting proofs and with checking proofs written by the authors. Every proof appearing in the paper was checked carefully by the authors. We also used AI tools as a writing aid, for improving the phrasing of text drafted by the authors, and for assistance with the formatting. We did not use generative AI tools to generate research ideas or to conduct the literature search. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work is purely theoretical, studying bounds on the number of examples required for language generation in an abstract learning model. We are not aware of any ethical concerns raised by our results, and we foresee no direct pathway from them to harmful applications. The authors declare no conflicts of interest.

REPRODUCIBILITY STATEMENT

Our work is purely theoretical. We state all of our assumptions in §3, and include full proofs for all theoretical results in the appendix.

REFERENCES

Dana Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45(2):117–135, 1980. URL https://www.sciencedirect.com/science/article/pii/S0019995880902855.

Marcelo Arenas, Pablo Barceló, Luis Cofré, and Alexander Kozachinskiy. Language generation: Complexity barriers and implications for learning, 2026. URL https://arxiv.org/abs/2511.05759.

Yannan Bai, Debmalya Panigrahi, and Ian Zhang. Language generation in the limit: Noise, loss, and feedback. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 794–816, 2026. URL https://doi.org/10.1137/1.9781611978971.31.

Moses Charikar and Chirag Pabbaraju. Exploring facets of language generation in the limit, 2024. URL https://arxiv.org/abs/2411.15364.

E Mark Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967. URL https://www.sciencedirect.com/science/article/pii/S0019995867911655.

Lauri Hella and Miikka Vilander. Defining long words succinctly in FO and MSO. In *Revolutions and Revelations in Computability: 18th Conference on Computability in Europe*, pp. 125–138, 2022. URL https://doi.org/10.1007/978-3-031-08740-0_11.

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. 3rd edition, 2006. URL https://dl.acm.org/doi/book/10.5555/1196416.

Alkis Kalavasis, Anay Mehrotra, and Grigoris Velegkas. On the limits of language generation: Trade-offs between hallucination and mode-collapse. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, pp. 1732–1743, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715105. doi: 10.1145/3717823.3718108. URL https://doi.org/10.1145/3717823.3718108.

Jon Kleinberg and Sendhil Mullainathan. Language generation in the limit. In *Advances in Neural Information Processing Systems*, volume 37, pp. 66058–66079, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/7988e9b3876ad689e921ce05d711442f-Paper-Conference.pdf.

Anay Mehrotra, Grigoris Velegkas, Xifan Yu, and Felix Zhou. Language generation with infinite contamination, 2025. URL https://arxiv.org/abs/2511.07417.

Ananth Raman and Vinod Raman. Generation from noisy examples. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=ouaiJlE1hJ.

Vinod Raman, Jiaxun Li, and Ambuj Tewari. Generation through the lens of learning theory. In *Proceedings of Thirty Eighth Conference on Learning Theory*, volume 291 of *Proceedings of Machine Learning Research*, pp. 4740–4776, 2025. URL https://proceedings.mlr.press/v291/raman25a.html.

A Further related work

Generation vs. prediction. Raman et al. (2025) compare uniform language *generation* in the limit with classic *prediction* frameworks such as PAC and online learning. In these frameworks, learnability is characterized by the finiteness of a combinatorial dimension: the VC dimension for PAC learnability, the Littlestone dimension for online learnability. Uniform generation is similar, and the dimension characterizing it is the closure dimension we use in this paper. The authors show that generation and prediction are incomparable: there are classes that are uniformly generatable but neither PAC nor online learnable, and classes that are PAC or online learnable but not generatable. Generation is therefore a different problem, which is why the sample complexity cannot be read off from any of these standard dimensions.

Statistical formulations. A parallel line of work replaces the enumeration with a distribution. Kalavasis et al. (2025) study generation from random samples drawn from a distribution over an unknown target, and ask for consistency and breadth, i.e., as the size of the revealed set increases, the set of strings the learner outputs converges to the set of all unseen strings in the target language. They show that this requirement is much stronger than that of Kleinberg & Mullainathan (2024): it is unachievable for most collections and models, including next-token predictors.

B Further preliminaries

B.1 Context-free languages

Definition 11. *A **context-free grammar** (CFG) is a tuple $\mathcal{G} = (\mathcal{V}, \Sigma, \mathcal{R}, S)$, where $\mathcal{V}$ is a finite set of **nonterminal symbols**, Σ is a finite set of **terminal symbols**, $\mathcal{R} \subseteq \mathcal{V} \times (\mathcal{V} \cup \Sigma)^*$ is a finite set of **production rules**, and $S \in \mathcal{V}$ is a distinguished **start symbol**.*

Using production rules, we can **derive** new symbols by replacing the variable of the left-hand side with the sequence of symbols on the right-hand side, i.e., $\alpha_1 X \alpha_2 \Rightarrow_{\mathcal{G}} \alpha_1 \gamma \alpha_2$, where $X \to \gamma \in \mathcal{R}$ and $\alpha_1, \alpha_2 \in (\mathcal{V} \cup \Sigma)^*$. A CFG $\mathcal{G}$ **derives** a string $x \in \Sigma^*$ from the start symbol S by repeatedly deriving symbols, until only terminal symbols are left, i.e., $S \overset{*}{\underset{\mathcal{G}}{\Rightarrow}} x$. The language generated by $\mathcal{G}$ is $L(\mathcal{G}) = \{x \in \Sigma^* \mid S \overset{*}{\underset{\mathcal{G}}{\Rightarrow}} x\}$. A language is **context-free** if it is generated by some CFG.

Example 1. *Consider the alphabet $\{a, b\}$. The language $L = \{a^n b^n \mid n \geq 0\}$, consisting of all strings with equal number of as and bs, where all as precede the bs, is context-free, and it is generated by the grammar $\mathcal{G} = (\{S\}, \Sigma, \mathcal{R}, S)$, with production rules $\mathcal{R} = \{S \to \varepsilon, S \to aSb\}$.*

B.2 Turing machines and the halting problem

Definition 12. *A **Turing machine** is a tuple $\mathcal{M} = (Q, \Gamma, \delta, q_0, q_F)$, where Q is a finite set of states, Γ is a finite alphabet of tape symbols, including a blank symbol $\perp$, $\delta \colon Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is a transition function, q_0 is the initial state, and q_F is the final state.*

A **configuration** of $\mathcal{M}$ is a triple (q, γ, i), indicating the current state q, the current tape contents $\gamma \in \Gamma^*$, and the position i of the head on the tape. A **computation** of $\mathcal{M}$ is a sequence of configurations $C_0, C_1, \ldots$ such that $C_0 = (q_0, \perp, 0)$ is the initial configuration and C_{n+1} is obtained from C_n by applying δ. We say that $\mathcal{M}$ **halts** if it reaches a configuration such that the current state is the final state.

The **halting problem** asks whether a Turing machine $\mathcal{M}$ ever halts. It is a classic result that the halting problem is undecidable (Hopcroft et al., 2006). More specifically, there is no algorithm that determines whether a Turing machine halts. We use this fact throughout this paper to establish the uncomputability of several sample complexity bounds. Specifically, we show that the existence of a computable bound would allow us to simulate a Turing machine for a finite number of steps and correctly decide whether it halts, contradicting undecidability.

C SAMPLE COMPLEXITY FOR PARTIALLY-ORDERED LANGUAGES

Theorem 2. *Let $\mathcal{C}$ be a finite collection of infinite PO languages, over an alphabet of size $|\Sigma|$ such that $|\Sigma| \geq 2$, each recognized by a PONFA with at most s states. Then $\mathcal{C}$ is $|\Sigma|^{|\mathcal{C}|(s-1)+1}$-generatable.*

Proof. It suffices to prove that for any non-empty subset $\mathcal{S} \subseteq \mathcal{C}$, if the intersection $\bigcap_{L \in \mathcal{S}} L$ is finite, then the size of this intersection is at most our desired bound. Consider the product automaton $\mathcal{A}$ encoding the intersection of each $\mathcal{A}_L$, for $L \in \mathcal{S}$. We assumed that each $\mathcal{A}_L$ is recognized by a PONFA with at most s states. The depth of each PONFA is at most $s-1$. By definition, each $\mathcal{A}_L$ is partially-ordered, therefore $\mathcal{A}$ is also partially-ordered. Its states are of the form $(q_1, \ldots, q_{|\mathcal{S}|})$ and it holds that $(q_1, \ldots, q_n) \leq (q'_1, \ldots, q'_n)$ if $q_i \leq q'_i$ for all i. Since every accepting path in $\mathcal{A}$ is simple, the length of any accepted string is bounded by the length of the longest simple path in $\mathcal{A}$. We now bound this length using the partial order structure of $\mathcal{A}$. At each step of a simple path, the tuple of states must strictly increase in at least one component, otherwise the same tuple would be visited twice, contradicting the definition of simple path. Since each $\mathcal{A}_L$ has at most s states, it can strictly increase at most $s-1$ times. Since there are $|\mathcal{S}| \leq |\mathcal{C}|$ PONFAs, the total number of strict increases is at most $|\mathcal{C}|(s-1)$. Therefore the longest simple path in $\mathcal{A}$ has length at most $|\mathcal{C}|(s-1)$, meaning every string has length at most $|\mathcal{C}|(s-1)$. Since $\mathcal{A}$ accepts only strings of length at most $|\mathcal{C}|(s-1)$, the intersection contains less than $|\Sigma|^{|\mathcal{C}|(s-1)+1}$ strings. □

Theorem 7. *There exists a collection $\mathcal{C}$ of two infinite PO languages over a binary alphabet, each recognized by a PONFA with at most 3 states, such that $\mathcal{C}$ is not $2^{2(3-1)}$-generatable.*

Proof. Let $\mathcal{C} = \{L_1, L_2\}$ be a collection recognized by the PONFAs in Figure 1. Their intersection contains all strings with at most two as and at most two bs, giving 19 strings. By Prop. 1, $\mathcal{C}$ is not $2^4 = 16$-generatable. □

D UNIFORM FINITE-NOISE GENERATION

Definition 13 (Raman & Raman, 2025)**.** *A language collection $\mathcal{C}$ is **uniformly noise-dependent generatable** if there exists a generator G such that, for every noise level k, there exists an $m \in \mathbb{N}$, such that for every $L \in \mathcal{C}$ and any enumeration $x_1, x_2, \ldots$ with $\sum_{t=1}^{\infty} \mathbb{1}\{x_t \notin L\} \leq k$, if there exists $t^* \in \mathbb{N}$ such that $|\{x_1, \ldots, x_{t^*}\}| = m$, then $G(x_1, \ldots, x_t) \in L \setminus \{x_1, \ldots, x_t\}$ for all $t \geq t^*$.*

We call such a language collection uniformly m-generatable at noise level k.

Corollary 2 (Raman & Raman, 2025)**.** *All finite classes are uniformly noise-dependent generatable.*

Proposition 2. *A finite collection $\mathcal{C}$ of infinite languages is* not *uniformly m-generatable at noise level $k \in \mathbb{N}$ if and only if there exists an enumeration $S_t = x_1, x_2, \ldots, x_t$ with $|\{x_1, \ldots, x_t\}| = m$ such that $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k \rangle_{\mathcal{C}}$ is finite and has size at least $m - k|\mathcal{C}|$.*

Proof. ($\Rightarrow$) Suppose $\mathcal{C}$ is not m-generatable at noise level k. Then for any generator G there exists a language $L \in \mathcal{C}$ and an enumeration $x_1, x_2, \ldots$ of L with at most k errors and a timestep t with $|\{x_1, \ldots, x_t\}| = m$ such that $G(x_1, \ldots, x_t) \notin L \setminus \{x_1, \ldots, x_t\}$. Set $S_t = x_1, \ldots, x_t$ and let $X = \langle S_t, k \rangle_{\mathcal{C}}$. Since S_t is valid for L at noise level k, it holds that $L \in \mathcal{C}(S_t, k)$, so $\mathcal{C}(S_t, k)$ is non-empty. In particular, take G to be the generator G^* that outputs an arbitrary unseen element of the noisy closure of its input whenever one exists, and let S_t, L, X be as described above.

X is finite: Suppose X is infinite. Since $L \in \mathcal{C}(S_t, k)$ we have $X \subseteq L$, and $X \setminus \{x_1, \ldots, x_t\}$ is non-empty because $\{x_1, \ldots, x_t\}$ is finite. Then $G^*(S_t) \in L \setminus \{x_1, \ldots, x_t\}$, contradicting that G^* fails. So X is finite.

$|X| \geq m - k|\mathcal{C}|$: Since $X = \bigcap_{L' \in \mathcal{C}(S_t, k)} L'$, we have $\{x_1, \ldots, x_t\} \setminus X = \bigcup_{L' \in \mathcal{C}(S_t, k)} (\{x_1, \ldots, x_t\} \setminus L')$. Each $L' \in \mathcal{C}(S_t, k)$ excludes at most k of these m strings, so $|\{x_1, \ldots, x_t\} \setminus X| \leq k|\mathcal{C}|$. We get $|X| \geq m - k|\mathcal{C}|$.

($\Leftarrow$) Suppose there exists an enumeration S_t with $|\{x_1, \ldots, x_t\}| = m$ such that $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k \rangle_{\mathcal{C}}$ is finite and has size at least $m - k|\mathcal{C}|$. Let G be a generator. Consider $G(S_t)$. We consider two cases:

$G(S_t) \in \langle S_t, k \rangle_{\mathcal{C}}$: G outputs something inside the noisy closure. The adversary continues with the remaining elements of the closure, which leave the consistent set unchanged. Since the closure is finite, it is exhausted after finitely many steps and G fails.

$G(S_t) \notin \langle S_t, k \rangle_{\mathcal{C}}$: Then there exists some language $L \in \mathcal{C}(S_t, k)$ such that $G(S_t) \notin L$. So G fails on L. □

Theorem 3. *Let $\mathcal{C}$ be a finite collection of infinite languages that is $B(\mathcal{C})$-generatable. Then $\mathcal{C}$ is $(B(\mathcal{C}) + k|\mathcal{C}|)$-generatable at noise level k.*

Proof. Let $m \geq B(\mathcal{C}) + k|\mathcal{C}|$. By Prop. 2, $\mathcal{C}$ fails to be m-generatable at noise level k only if there is an enumeration S_t with $|\{x_1, \ldots, x_t\}| = m$ such that $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k \rangle_{\mathcal{C}}$ is finite of size at least $m - k|\mathcal{C}|$. We show no such enumeration exists.

Suppose S_t is an enumeration with $\mathcal{C}(S_t, k) \neq \emptyset$ and $\langle S_t, k \rangle_{\mathcal{C}}$ finite. Write $\mathcal{S} = \mathcal{C}(S_t, k)$. By Def. 8,

$$\langle S_t, k \rangle_{\mathcal{C}} = \bigcap_{L \in \mathcal{S}} L, \tag{5}$$

and $\mathcal{S}$ is a non-empty subset of $\mathcal{C}$. The noisy closure is therefore a finite intersection of languages of $\mathcal{C}$, which is exactly the object bounded in the noiseless setting. Hence

$$|\langle S_t, k \rangle_{\mathcal{C}}| < B(\mathcal{C}) \leq m - k|\mathcal{C}|, \tag{6}$$

where the second inequality is the choice of m. The size condition of Prop. 2 thus fails for every enumeration, and $\mathcal{C}$ is m-generatable at noise level k. □

D.1 Uncomputable bound for context-free languages

Arenas et al. (2026) use the following lemma to establish that no computable bound on the sample complexity exists for context-free languages. Their proof reduces from the halting problem via a construction that encodes the computation of a Turing machine into the intersection of two CFGs.

Lemma 1 (Lemma 3.2, Arenas et al., 2026)**.** *There exists an algorithm that, given a Turing Machine $\mathcal{M}$ with empty input, constructs two CFGs $\mathcal{G}_1$ and $\mathcal{G}_2$ such that, if M halts in exactly t computational steps, then*

$$|L(\mathcal{G}_1) \cap L(\mathcal{G}_2)| = 2^t. \tag{7}$$

Corollary 3. *For any noise level k, there is no algorithm that, given two infinite CFGs $\mathcal{G}_1$ and $\mathcal{G}_2$, outputs $m \in \mathbb{N}$ such that $\{L(\mathcal{G}_1), L(\mathcal{G}_2)\}$ is uniformly m-generatable at noise level k. The algorithm may diverge if either language is finite.*

Proof. Fix some noise level k and, for contradiction, assume there exists an algorithm such that, given two infinite CFGs G_1 and G_2, outputs m such that the collection $\{L(G_1), L(G_2)\}$ is m-generatable at noise level k. Given a Turing Machine M, we construct CFGs G_1 and G_2 as in Lemma 1. We know that the finiteness of context-free languages is decidable. We distinguish three cases:

$L(G_1)$ is finite: Choose t such that $2^t > |L(G_1)| \geq |L(G_1) \cap L(G_2)|$. Then M cannot halt in t or more steps. Simulate M for $t - 1$ steps and conclude that it does not halt if it does not reach a halting configuration.

$L(G_2)$ is finite: Analogous to previous case.

Both $L(G_1)$ and $L(G_2)$ are infinite: Apply the algorithm and get m such that $\{L(G_1), L(G_2)\}$ is m-generatable at noise level k. Since m-generatability implies m'-generatability for every $m' \geq m$, we may assume $m \geq 2k$. Choose t such that $2^t \geq m - 2k$. We claim that M cannot halt in $s \geq t$ steps.

Suppose M halts in $s \geq t$ steps. Then $|L(G_1) \cap L(G_2)| = 2^s$. Since both $L(G_1)$ and $L(G_2)$ are infinite and their intersection is finite, both $L(G_1) \setminus L(G_2)$ and $L(G_2) \setminus L(G_1)$ are infinite. We construct an enumeration S consisting of $m - 2k$ distinct elements from $L(G_1) \cap L(G_2)$, k elements from $L(G_1) \setminus L(G_2)$, and k elements from $L(G_2) \setminus L(G_1)$, giving $|S| = m$. We now verify the conditions of Prop. 2 with S and m.

Since $|S \cap L(G_1)| = m - k$ and $|S \cap L(G_2)| = m - k$, it holds that $\mathcal{C}(S, k) = \{L(G_1), L(G_2)\}$, which is non-empty. Furthermore, $\langle S, k \rangle_{\mathcal{C}} = L(G_1) \cap L(G_2)$, which is finite of size 2^s. Finally, we need $|\langle S, k \rangle_{\mathcal{C}}| \geq m - k|\mathcal{C}|$. Trivially, $2^s \geq 2^t \geq m - 2k$. Therefore, $\{L(G_1), L(G_2)\}$ is not m-generatable at noise level k, contradicting the output of the algorithm. Therefore, M cannot halt in $s \geq t$ steps and it suffices to simulate it for $t - 1$ steps to decide whether it halts. In all cases, we have decided the halting problem, therefore the algorithm cannot exist. □

D.2 Bounds for regular languages

Corollary 4. *Let $\mathcal{C}$ be a finite collection of infinite regular languages over an alphabet of size $|\Sigma| \geq 2$ such that each $L \in \mathcal{C}$ is accepted by an NFA with at most s states. Then $\mathcal{C}$ is $(|\Sigma|^{s^{|\mathcal{C}|}} + k|\mathcal{C}|)$-generatable at noise level k.*

Proof. By Prop. 2, it suffices to show that for every enumeration $S_t = x_1, \ldots, x_t$ with $|\{x_1, \ldots, x_t\}| = m$, if $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k \rangle_{\mathcal{C}}$ is finite, then $|\langle S_t, k \rangle_{\mathcal{C}}| < m - k|\mathcal{C}|$.

Suppose $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k \rangle_{\mathcal{C}}$ is finite. By the product construction, $\langle S_t, k \rangle_{\mathcal{C}}$ is accepted by an NFA with at most $s^{|\mathcal{C}|}$ states. Since $\langle S_t, k \rangle_{\mathcal{C}}$ is finite, every word in it has length less than $s^{|\mathcal{C}|}$, otherwise the NFA would have a cycle, contradicting the finiteness assumption. The number of words of length less than $s^{|\mathcal{C}|}$ over an alphabet of size $|\Sigma|$ is $|\Sigma|^{s^{|\mathcal{C}|}}$. So $|\langle S_t, k \rangle_{\mathcal{C}}| < |\Sigma|^{s^{|\mathcal{C}|}}$. If $m - k|\mathcal{C}| \geq |\Sigma|^{s^{|\mathcal{C}|}}$, then $|\langle S_t, k \rangle_{\mathcal{C}}| < m - k|\mathcal{C}|$ and the condition in Prop. 2 is not satisfied. Therefore $\mathcal{C}$ is m-uniformly generatable at noise level k for any $m \geq |\Sigma|^{s^{|\mathcal{C}|}} + k|\mathcal{C}|$. □

Now we show that this bound is tight with an example of collection of regular languages which actually requires this number of examples for uniform generation in noisy setting.

Theorem 8. *There exists $c \in \mathbb{N}$ such that for every $k, l, n \in \mathbb{N}$ with $l \geq 1$ and $n \geq 1$, there exists a collection $\mathcal{C}$ of infinite binary regular languages such that $|\mathcal{C}| = 2^l$, each $L \in \mathcal{C}$ is accepted by a DFA with at most $c \cdot l \cdot n$ states, and $\mathcal{C}$ is not $(2^{n^{|\mathcal{C}|}} + k|\mathcal{C}|)$-generatable at noise level k.*

Proof. We use the same collection $\mathcal{C} = \{L_w^n \mid w \in \{0,1\}^l\}$ as Arenas et al. (2026). By their construction, each L_w^n is accepted by a DFA with at most $c \cdot l \cdot n$ states, each language is infinite and

$$\left| \bigcap_{w \in \{0,1\}^l} L_w^n \right| \geq 2^{n^{|\mathcal{C}|}}, \tag{8}$$

and this intersection is finite.

We use Prop. 2 to prove that $\mathcal{C}$ is not $(2^{n^{|\mathcal{C}|}} + k \cdot 2^l)$-generatable at noise level k. Let $m = 2^{n^{|\mathcal{C}|}} + k \cdot 2^l$ and choose S such that it contains $2^{n^{|\mathcal{C}|}}$ distinct elements from the intersection and, for each $w \in \{0,1\}^l$, k elements from $\bigcap_{w' \neq w} L_{w'}^n$. We can prove that the set $\bigcap_{w' \neq w} L_{w'}^n$ is infinite, therefore we can always pick k elements from it. To show this, consider a binary string $b \in \{0,1\}^l$ and a string of the form $s_b = wbwb \cdots w$, with $j \geq n+1$ odd blocks equal to w, and even blocks equal to b. This violates the condition of language L_w^n, therefore $s_b \notin L_w^n$. For every $w' \neq w$, the string s_b has zero odd w' blocks, so the condition of language $L_{w'}^n$ is trivially satisfied. Because the length of such strings can become arbitrarily large, there are infinitely many strings. We now verify the conditions of Prop. 2.

For each language L_w^n, there are k samples outside of it. So for every $L_w^n \in \mathcal{C}$, $|S \cap L_w^n| = m - k$. Every language is consistent with S at noise level k, by construction. Therefore $\mathcal{C}(S, k) = \mathcal{C}$.

Since $\mathcal{C}(S, k) = \mathcal{C}$, it holds that the noisy closure $\langle S, k \rangle_{\mathcal{C}} = \bigcap_{L \in \mathcal{C}(S,k)} L = \bigcap_{w \in \{0,1\}^l} L_w^n$ is finite by the proof of Arenas et al. (2026).

We need to show that $|\langle S, k \rangle_{\mathcal{C}}| \geq m - k|\mathcal{C}|$. The right-hand side equals $2^{n^{|\mathcal{C}|}} + k|\mathcal{C}| - k|\mathcal{C}| = 2^{n^{|\mathcal{C}|}}$, and $|\langle S, k \rangle_{\mathcal{C}}| \geq 2^{n^{|\mathcal{C}|}}$. So $\mathcal{C}$ is not m-generatable at noise level k. □

D.3 Bounds for LTT languages

Corollary 5. *Let $\mathcal{C}$ be a finite collection of infinite languages over an alphabet of size $|\Sigma| \geq 2$ such that each language $L \in \mathcal{C}$ is defined by an LTT of size at most s. Then $\mathcal{C}$ is $(|\Sigma|^{3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2}+k|\mathcal{C}|)$-generatable at noise level k.*

Proof. By Prop. 2, it suffices to show that for every enumeration $S_t = x_1, \ldots, x_t$ with $|\{x_1, \ldots, x_t\}| = m$, if $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k\rangle_{\mathcal{C}}$ is finite, then $|\langle S_t, k\rangle_{\mathcal{C}}| < m - k|\mathcal{C}|$. Suppose $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k\rangle_{\mathcal{C}}$ is finite. We bound the size of $\langle S_t, k\rangle_{\mathcal{C}}$. Since each $L \in \mathcal{C}$ is an LTT language of size at most s, each L can be written as a finite union of profiles of size at most s. By distributivity of intersection over union, $\langle S_t, k\rangle_{\mathcal{C}}$ can be written as a union of profiles, each obtained by intersecting one profile from each $L \in \mathcal{C}(S, k)$. Each such intersection of profiles of size at most s can be represented as a single profile of size at most $s|\mathcal{C}|$. Each of these profiles defines a finite language. By Lemma 5.2 of Arenas et al. (2026), every string in a profile of size r that defines a finite language has length less than $3r(r^2+1)^2$. Using $r = s|\mathcal{C}|$, every string in $\langle S_t, k\rangle_{\mathcal{C}}$ has length less than $3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2$. The number of strings of length less than $3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2$ over an alphabet of size $|\Sigma|$ is at most $|\Sigma|^{3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2}$. Therefore, $|\langle S_t, k\rangle_{\mathcal{C}}| < |\Sigma|^{3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2}$. If $m - k|\mathcal{C}| \geq |\Sigma|^{3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2}$, then $|\langle S_t, k\rangle_{\mathcal{C}}| < m - k|\mathcal{C}|$ and the condition in Prop. 2 is not satisfied. Therefore $\mathcal{C}$ is m-uniformly generatable at noise level k for any $m \geq |\Sigma|^{3s|\mathcal{C}|((s|\mathcal{C}|)^2+1)^2} + k|\mathcal{C}|$. □

Now we show that this bound is tight by providing an example of collection of LTT languages which actually requires this number of examples for uniform generation in noisy setting.

Theorem 9. *For every $n \in \mathbb{N}$ with $n \geq 4$, there exists a collection $\mathcal{C}$ consisting of two infinite languages over an alphabet of size n such that each language $L \in \mathcal{C}$ is defined by a profile of size at most $7(n+1)$, and $\mathcal{C}$ is not $(2^n + 2k)$-generatable at noise level k.*

Proof. We use the same two languages $L(P_1)$ and $L(P_2)$ from Arenas et al. (2026), defined over the alphabet $\Sigma = \{1, 2, \ldots, n\}$, such that

$$P_1 = (\varepsilon, \varepsilon, \{(i, \leq 1) \colon i \text{ is odd}\} \cup \{(i, \geq 1) \colon i \text{ is even}\}), \tag{9}$$

$$P_2 = (\varepsilon, \varepsilon, \{(i, \geq 1) \colon i \text{ is odd}\} \cup \{(i, \leq 1) \colon i \text{ is even}\}). \tag{10}$$

Both $L(P_1)$ and $L(P_2)$ are infinite, $|L(P_1) \cap L(P_2)| = n!$, and each profile has length at most $7(n+1)$. Since $n! \geq 2^n$ for $n \geq 4$, the intersection has size at least 2^n. Since the languages are infinite and their intersection is finite, both $L(P_1) \setminus L(P_2)$ and $L(P_2) \setminus L(P_1)$ are infinite. Choose an enumeration S consisting of $m = 2^n + 2k$ distinct elements such that 2^n elements are chosen from $L(P_2) \cap L(P_1)$, k elements are chosen from $L(P_1) \setminus L(P_2)$ and k elements are chosen from $L(P_2) \setminus L(P_1)$. We now verify the conditions of Prop. 2.

There are exactly k elements chosen from $L(P_2) \setminus L(P_1)$, and these are the only elements of S outside $L(P_1)$, therefore $|S \cap L(P_1)| = m - k$. Similarly, $|S \cap L(P_2)| = m - k$. Therefore, $\mathcal{C}(S, k) = \{L(P_1), L(P_2)\}$.

Since $\mathcal{C}(S, k) = \{L(P_1), L(P_2)\}$, it holds that $\langle S, k\rangle_{\mathcal{C}} = L(P_1) \cap L(P_2)$, which is finite of size $n!$.

We need to show that $|\langle S, k\rangle_{\mathcal{C}}| \geq m - k|\mathcal{C}|$. It holds that $n! \geq 2^n + 2k - 2k = 2^n$ for any $n \geq 4$.

By Prop. 2, $\mathcal{C}$ is not $(2^n + 2k)$-generatable at noise level k. □

D.4 Bound for partially-ordered languages

Corollary 6. *Let $\mathcal{C}$ be a finite collection of infinite PO languages, over an alphabet of size $|\Sigma| \geq 2$, each recognized by a PONFA with at most s states. Then $\mathcal{C}$ is $(|\Sigma|^{|\mathcal{C}|(s-1)+1} + k|\mathcal{C}|)$-generatable at noise level k.*

Proof. It suffices to show that for every enumeration $S_t = x_1, \ldots, x_t$ such that $|\{x_1, \ldots, x_t\}| = m$, if $\mathcal{C}(S_t, k)$ is non-empty and $\langle S_t, k\rangle_{\mathcal{C}}$ is finite, then $|\langle S_t, k\rangle_{\mathcal{C}}| < m - k|\mathcal{C}|$.

Since each language $L \in \mathcal{C}$ is a PO language recognized by a PONFA with at most s states, the noisy closure $\langle S_t, k\rangle_{\mathcal{C}} = \bigcap_{L\in\mathcal{C}(S_t,k)} L$ is recognized by the intersection automaton of the PONFAs. Since the noisy closure is finite, every path in this automaton is simple, therefore we can bound the longest string. As we've proven in the proof of Theorem 2, the longest string can have length at most $|\mathcal{C}|(s-1)$. Therefore, $|\langle S_t, k\rangle_{\mathcal{C}}| < |\Sigma|^{|\mathcal{C}|(s-1)+1}$. If $m - k|\mathcal{C}| \geq |\Sigma|^{|\mathcal{C}|(s-1)+1}$, then $|\langle S_t, k\rangle_{\mathcal{C}}| < m - k|\mathcal{C}|$, and the condition in Prop. 2 is not satisfied. Therefore $\mathcal{C}$ is m-uniformly generatable at noise level k for any $m \geq |\Sigma|^{|\mathcal{C}|(s-1)+1} + k|\mathcal{C}|$. □

Theorem 10. *There exists a collection $\mathcal{C}$ of two infinite PO languages over a binary alphabet, each recognized by a PONFA with at most 3 states, such that $\mathcal{C}$ is not $(2^{2(3-1)} + 2k)$-generatable at noise level k.*

Proof. We use the same two languages L_1 and L_2 from the noiseless lower bound. Their intersection has size at least $|\Sigma|^{|\mathcal{C}|(s-1)} = 16$. Since both $L_1 \setminus L_2$ and $L_2 \setminus L_1$ are infinite, we can choose k elements from each to construct a noisy enumeration of size $|\Sigma|^{|\mathcal{C}|(s-1)} + 2k$. The conditions from Prop. 2 are satisfied, therefore $\mathcal{C}$ is not $(|\Sigma|^{|\mathcal{C}|(s-1)} + 2k)$-generatable at noise level k. □

E PROMPTED UNIFORM GENERATION

Definition 14. *A language collection $\mathcal{C}$ is **prompted uniformly generatable** if there exists a generator G such that, for every prompt $p \in \mathcal{P}$, there exists an $m \in \mathbb{N}$, such that for every $K \in \mathcal{C}$ and any enumeration $x_1, x_2, \ldots$ of K, if there exists a $t^* \in \mathbb{N}$ such that $|\{x_1, \ldots, x_{t^*}\}^p| = m$, then for all $t \geq t^*$, G outputs a string a such that $p \cdot a \in K \setminus \{x_1, \ldots, x_t\}$.*

Definition 15. *For a prompt $p \in \mathcal{P}$, the **prompted closure dimension** $\mathrm{PC}(\mathcal{C}, p)$ of a collection $\mathcal{C}$ at p is the supremum over $d \in \mathbb{N}$ satisfying: there exists a language $K \in \mathcal{C}$ and an enumeration S of K, such that there exists a timestep t with $|S_t{}^p| \geq d$ and*

$$\left(\bigcap_{L\in\mathcal{C}(S_t,p)} L^p\right) \setminus S_t{}^p = \emptyset. \tag{11}$$

Theorem 11. *A collection $\mathcal{C}$ is prompted uniformly generatable if and only if $\mathrm{PC}(\mathcal{C}, p) < \infty$ for every $p \in \mathcal{P}$.*

Proof. ($\Rightarrow$) We prove that if $\mathrm{PC}(\mathcal{C}, p) = \infty$ for some $p \in \mathcal{P}$, then $\mathcal{C}$ is not prompted uniformly generatable. Fix such a p, any generator G, and any $m \in \mathbb{N}$. We must find a language $K \in \mathcal{C}$ and an enumeration of K such that G fails after seeing m distinct strings with prompt p.

Since $\mathrm{PC}(\mathcal{C}, p) = \infty$, there exist a language $K \in \mathcal{C}$ and an enumeration S of K with timestep t such that $|S_t{}^p| \geq m$ and $\left(\bigcap_{L\in\mathcal{C}(S_t,p)} L^p\right) \setminus S_t{}^p = \emptyset$. Let $X = \bigcap_{L\in\mathcal{C}(S_t,p)} L^p$, so that $X \subseteq S_t{}^p$, and consider $\hat{x} = G(S_t, p)$.

If $\hat{x} \in S_t{}^p$, then $p \cdot \hat{x} \in S_t$, so G outputs a continuation already seen and fails on K at timestep t.

If $\hat{x} \notin S_t{}^p$, then $X \subseteq S_t{}^p$, so $\hat{x} \notin X$. Then there exists $L \in \mathcal{C}(S_t, p)$ with $\hat{x} \notin L^p$. Since $L \in \mathcal{C}(S_t, p)$, we have $S_t \subseteq L$, so S_t is a valid segment of an enumeration of L. Since L is infinite, we can extend S_t by listing the remaining strings in some order, to obtain a full enumeration of L whose first strings are exactly S_t. Now consider running G with this enumeration, target L, and prompt p. At timestep t, G fails to produce a continuation a with $p \cdot a \in L \setminus S_t$.

($\Leftarrow$) Suppose $\mathrm{PC}(\mathcal{C}, p) < \infty$ for every $p \in \mathcal{P}$. Fix any $p \in \mathcal{P}$ and let $d_p = \mathrm{PC}(\mathcal{C}, p)$. We construct a generator that succeeds with prompt p after $m_p = d_p + 1$ distinct examples.

Given an enumeration S_t and prompt p, define G as follows. Compute $X = \bigcap_{L\in\mathcal{C}(S_t,p)} L^p$ and output an arbitrary element of $X \setminus S_t{}^p$ if this set is non-empty; otherwise output an arbitrary element of Σ^*.

Fix any $K \in \mathcal{C}$ and any enumeration $x_1, x_2, \ldots$ of K. Suppose there exists t^* with $|\{x_1, \ldots, x_{t^*}\}^p| = d_p + 1$. We show that for all $t \geq t^*$, $G(S_t, p) \in K^p \setminus S_t{}^p$. Fix any $t \geq t^*$. Since $|S_t{}^p| \geq d_p + 1 >$

$\mathrm{PC}(\mathcal{C}, p)$, the condition of Def. 15 is not satisfied at t, so $X \setminus {S_t}^p \neq \emptyset$ and G outputs an element of it. Since $K \in \mathcal{C}(S_t, p)$ we have $X \subseteq K^p$, so this element lies in $K^p \setminus {S_t}^p$. As p was arbitrary, $\mathcal{C}$ is prompted uniformly generatable. □

Corollary 1. *Let $\mathcal{P}$ be finite. Any finite collection $\mathcal{C}$ of infinite languages satisfying Assumption 1 is prompted uniformly generatable.*

Proof. It suffices to show that $\mathrm{PC}(\mathcal{C}) < \infty$. Since both $\mathcal{C}$ and $\mathcal{P}$ are finite, the set

$$\mathcal{I} = \left\{ \bigcap_{L \in \mathcal{S}} L^p : \mathcal{S} \subseteq \mathcal{C}, p \in \mathcal{P}, \mathcal{S} \neq \emptyset \right\} \tag{12}$$

is finite. Let D be the maximum size of any element of $\mathcal{I}$ that is finite, or 0 if there is no such element.

We prove that $\mathrm{PC}(\mathcal{C}) \leq D$. Assume for contradiction that $\mathrm{PC}(\mathcal{C}) > D$. Then there exist $K \in \mathcal{C}$, $p \in \mathcal{P}$, and an enumeration S of K with a timestep t such that $|{S_t}^p| \geq D + 1$, and

$$\left(\bigcap_{L \in \mathcal{C}(S_t, p)} L^p \right) \setminus {S_t}^p = \emptyset. \tag{13}$$

Let $X = \bigcap_{L \in \mathcal{C}(S_t, p)} L^p$. Since each $L \in \mathcal{C}(S_t, p)$ contains S_t, we have ${S_t}^p \subseteq L^p$, thus ${S_t}^p \subseteq X$. By combining this with $X \subseteq {S_t}^p$ from the failure condition, we get $X = {S_t}^p$.

Since $|{S_t}^p| \geq D + 1$, we have $|X| \geq D + 1$. But $X \in \mathcal{I}$ and X is finite, so $|X| \leq D$ by the definition of D. This leads to a contradiction. □

The prompt domain $\mathcal{P}$ being finite is essential. The following example shows a finite collection consisting of only two infinite languages that is not prompted uniformly generatable. This example is adapted from Raman et al. (2025) to match our prompted generation setting.

Example 2. *Let $\Sigma = \{1, \#, a\}$ and $\mathcal{P} = \{1\#, 11\#, 111\#, \ldots\}$. We use p_n to denote the prompt $1^n\#$. Note that no prompt is a prefix of another.*

For each prompt p_n, let $A_n = \{a^{n^2+1}, \ldots, a^{n^2+n}\}$ be a set of n suffixes. Notice that the sets $A_1, A_2, \ldots$ are pairwise disjoint.

Now define the two languages

$$L_1 = \{p_n \cdot x : n \in \mathbb{N}, x \in A_n\} \cup \{p_n \cdot a^k : n \in \mathbb{N}, k > n^2 + n, k \textit{ even}\}, \tag{14}$$

$$L_2 = \{p_n \cdot x : n \in \mathbb{N}, x \in A_n\} \cup \{p_n \cdot a^k : n \in \mathbb{N}, k > n^2 + n, k \textit{ odd}\}. \tag{15}$$

Both languages are infinite, and for every prompt p_n, both projections $L_1^{p_n}$ and $L_2^{p_n}$ are infinite. Since the prompts are prefix-free, every string of L_1 or L_2 beginning with p_n is of the form $p_n \cdot y$ for a unique n, so $L_1^{p_n} \cap L_2^{p_n} = A_n$, which is finite of size n.

Proposition 3. *Let $\mathcal{P}$ be finite. A finite collection $\mathcal{C}$ of languages satisfying Assumption 1 is* not *m-prompted uniformly generatable for prompt $p \in \mathcal{P}$ if and only if there exists a non-empty set $\mathcal{S} \subseteq \mathcal{C}$ such that $\bigcap_{L \in \mathcal{S}} L^p$ is finite and has size at least m.*

Proof. ($\Rightarrow$) Suppose $\mathcal{C}$ is not m-prompted uniformly generatable for p. In particular, take G to be the generator G^* that outputs an arbitrary unseen element of the prompted closure of its input whenever one exists. Then there exists a language $K \in \mathcal{C}$, an enumeration $x_1, x_2, \ldots$ of K, and a timestep t with $|{S_t}^p| = m$ such that $G^*(S_t, p) \notin K^p \setminus {S_t}^p$. Let $\mathcal{S} = \mathcal{C}(S_t, p)$ and $X = \bigcap_{L \in \mathcal{S}} L^p$. Since $S_t \subseteq K$ and $K^p \neq \emptyset$ by Assumption 1, we have $K \in \mathcal{S}$, so $\mathcal{S}$ is non-empty.

X is finite: Suppose X is infinite. Since $K \in \mathcal{S}$ we have $X \subseteq K^p$, and $X \setminus {S_t}^p$ is non-empty because ${S_t}^p$ is finite. Then $G^*(S_t, p) \in K^p \setminus {S_t}^p$, contradicting that G^* fails. So X is finite.

$|X| \geq m$: Every $L \in \mathcal{S}$ satisfies $S_t \subseteq L$, hence ${S_t}^p \subseteq L^p$, and therefore ${S_t}^p \subseteq X$. Since $|{S_t}^p| = m$, we get $|X| \geq m$.

($\Leftarrow$) Suppose there is a non-empty $\mathcal{S} \subseteq \mathcal{C}$ such that $X = \bigcap_{L \in \mathcal{S}} L^p$ is finite with $|X| \geq m$. Fix $K \in \mathcal{S}$ and let G be any generator. The adversary enumerates K so that the first strings are $p \cdot x$ for

$x \in X$, which is possible since $X \subseteq K^p$, and then the remaining strings of K in any order. Let t be the first timestep at which $|S_t{}^p| = m$; since $|X| \geq m$ such a t exists and $S_t{}^p \subseteq X$. Every $L \in \mathcal{S}$ contains $p \cdot x$ for each $x \in S_t{}^p$, so $\mathcal{S} \subseteq \mathcal{C}(S_t, p)$ and hence $\bigcap_{L \in \mathcal{C}(S_t,p)} L^p \subseteq X$. We distinguish two cases.

$G(S_t, p) \notin X$: Then there is some $L \in \mathcal{S}$ with $G(S_t, p) \notin L^p$, and the same enumeration is valid for target L, so G fails on L.

$G(S_t, p) \in X$: The adversary continues presenting the elements of $X \setminus S_t{}^p$, which leaves $\mathcal{C}(S_t, p)$ unchanged. Since X is finite, after finitely many steps $S_t{}^p = X$ and G must output a string outside X, which is the previous case. □

Theorem 4. *Let $\mathcal{P}$ be finite and let $\mathcal{C}$ be a finite collection of infinite languages satisfying Assumption 1. Then for every $p \in \mathcal{P}$, $\mathcal{C}$ is $B(\{L^p : L \in \mathcal{C}\})$-prompted uniformly generatable with prompt p.*

Proof. Fix $p \in \mathcal{P}$ and write $\mathcal{C}^p = \{L^p : L \in \mathcal{C}\}$ for the collection of left quotients by p.

The recipe of §4 applies to $\mathcal{C}^p$. The collection $\mathcal{C}^p$ is finite, with $|\mathcal{C}^p| \leq |\mathcal{C}|$, and by Assumption 1 every $L^p \in \mathcal{C}^p$ is infinite. Moreover, the representation of each class we study is effectively closed under left quotient. The constructions are given below. Hence $\mathcal{C}^p$ is a finite collection of infinite languages of the same class, so item (i) and item (ii) of the recipe apply to it and $B(\mathcal{C}^p)$ is well defined: every finite intersection of a non-empty subcollection of $\mathcal{C}^p$ has size less than $B(\mathcal{C}^p)$.

Apply Prop. 3. Let $m = B(\mathcal{C}^p)$ and suppose, for contradiction, that $\mathcal{C}$ is not m-prompted uniformly generatable with prompt p. By Prop. 3, there exists a non-empty $\mathcal{S} \subseteq \mathcal{C}$ such that $\bigcap_{L \in \mathcal{S}} L^p$ is finite and has size at least m. But $\{L^p : L \in \mathcal{S}\}$ is a non-empty subcollection of $\mathcal{C}^p$, so by the previous paragraph, this intersection, being finite, has size less than $B(\mathcal{C}^p) = m$, a contradiction. Therefore $\mathcal{C}$ is $B(\mathcal{C}^p)$-prompted uniformly generatable with prompt p. □

E.1 Computing the left quotient

Regular languages. Given an NFA $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ recognizing L, the left quotient L^p is recognized by the NFA $\mathcal{A}' = (\Sigma, Q, I_p, F, \delta)$, where $I_p = \{q \in Q : \exists q_0 \in I \text{ s.t. there is a path from } q_0 \text{ to } q \text{ reading } p\}$. Some states can become unreachable and trimming can remove them, but in general the number of states is unchanged.

LTT languages. Given a profile $P = (Pr, Su, In)$ of size s, the left quotient $L(P)^p$ is a finite union of profiles obtained as follows. The new prefix is obtained by consuming p from Pr. If p is a prefix of Pr, then the new prefix is the portion of Pr beyond p, of length at most s; otherwise, the new prefix is ε. If neither p nor Pr is a prefix of the other, then no string of $L(P)$ begins with p, so $L(P)^p = \emptyset$ and this profile is omitted from the union. Then, each counting constraint $(w, \theta t) \in In$ must account for occurrences of w in $p \cdot x$ that either fall entirely in p, are at the boundary between p and x, or fall within x. The number of occurrences of w in p are determined entirely by w and p. The constraints can be adjusted accordingly. The occurrences at the boundaries depend on what x starts with, giving rise to a union of profiles. The number of profiles in the union is at most $(s+1)^s$, independent of p. Each profile in the union has size at most $s + |p|$. Since LTT languages are closed under finite union, $L(P)^p$ is an LTT language defined by a finite union of at most $(s+1)^s$ profiles, each of size $s + |p|$.

PO languages. Given a PONFA $\mathcal{A}$ recognizing L, the left quotient L^p is recognized by the automaton obtained from $\mathcal{A}$ by shifting the initial states to I_p, just like for regular languages. Partial-orderedness is preserved by the shift. The number of states is unchanged in general.

Context-free languages. Given a CFG generating L, the left quotient L^p is also context-free. However, the uncomputability barrier transfers without modification.

E.2 Sample complexity bounds

The corollaries below follow almost immediately by applying the corresponding noiseless bounds and Prop. 3.

Corollary 7. *Let $\mathcal{C}$ be a finite collection of regular languages, with a finite prompt domain, satisfying Assumption 1. If each language is recognized by an NFA with at most s states, over an alphabet Σ, then $\mathcal{C}$ is $|\Sigma|^{s^{|\mathcal{C}|}}$-prompted uniformly generatable.*

Corollary 8. *Let $\mathcal{C}$ be a finite collection of LTT languages, with a finite prompt domain, satisfying Assumption 1. If each language is recognized by a profile of size at most s, over an alphabet Σ, then $\mathcal{C}$ is $|\Sigma|^{3(s+|p|)|\mathcal{C}|(((s+|p|)|\mathcal{C}|)^2+1)^2}$-prompted uniformly generatable with prompt p.*

Corollary 9. *For any prompt p in a finite prompt domain, there is no algorithm that, given CFGs encoding the infinite languages L_1 and L_2, outputs m such that $\{L_1, L_2\}$ is m-prompted uniformly generatable with prompt p.*

Corollary 10. *Let $\mathcal{C}$ be a finite collection of PO languages, with a finite prompt domain, satisfying Assumption 1. If each language is recognized by a PONFA with at most s states, over an alphabet Σ, then $\mathcal{C}$ is $|\Sigma|^{|\mathcal{C}|(s-1)+1}$-prompted uniformly generatable.*

F Uniform generation with feedback

Definition 16. *A language collection $\mathcal{C}$ is **uniformly generatable with feedback** if there exists a generator G such that, for every query budget $f \in \mathbb{N}$, there exists an $m \in \mathbb{N}$, such that for every $L \in \mathcal{C}$ and any enumeration $x_1, x_2, \ldots$ of L, if there exists $t^* \in \mathbb{N}$ such that $|\{x_1, \ldots, x_{t^*}\}| = m$, then for all $t \geq t^*$, G outputs a string in $L \setminus \{x_1, \ldots, x_t\}$ using at most f feedback queries.*

We adapt the definition of feedback closure dimension of Charikar & Pabbaraju (2024), which they call feedback-generation closure, to our setup.

Definition 17. *The **feedback closure dimension** $\mathrm{FC}(\mathcal{C}, f)$ of a collection $\mathcal{C}$ at query budget f is the supremum over $d \in \mathbb{N}$ satisfying: For every generator G that asks at most f feedback queries, there exists a language $K \in \mathcal{C}$ and an enumeration S of K such that there exists a timestep t with $|S_t| \geq d$ and*

$$\left(\bigcap_{L \in \mathcal{C}(S_t, Q_f)} L \right) \setminus S_t = \emptyset, \tag{16}$$

where Q_f is the query set.

Charikar & Pabbaraju (2024) give a characterization of the collections that are uniformly generatable with feedback. However, their setup is different, therefore we give the following analogous characterization.

Theorem 12. *A collection is uniformly generatable with feedback if and only if the feedback closure dimension is finite.*

Proof. ($\Rightarrow$) Suppose $\mathrm{FC}(\mathcal{C}, f) = \infty$ for some number of queries f. We need to show that $\mathcal{C}$ is not m-uniformly generatable with f queries for any m. Fix any generator G that uses at most f queries and any $m \in \mathbb{N}$. We will find a language $L \in \mathcal{C}$, an enumeration $S_t = x_1, x_2, \ldots$ of L, and a timestep t with $|\{x_1, \ldots, x_t\}| = m$, such that G fails after t timesteps.

Since $\mathrm{FC}(\mathcal{C}, f) = \infty$, there exists a $d \geq m$, a sequence of distinct examples $x_1, \ldots, x_d$, and a query set Q_f such that $\mathcal{C}(\{x_1, \ldots, x_d\}, Q_f)$ is non-empty and $\left(\bigcap_{L \in \mathcal{C}(S_t, Q_f)} L\right) \setminus S_t = \emptyset$.

Let $\hat{x} = G(\{x_1, \ldots, x_d\}, Q_f)$. Assume $\hat{x} \in \{x_1, \ldots, x_d\}$. Then G fails to output a new string. Now assume $\hat{x} \notin \{x_1, \ldots, x_d\}$. Since $\left(\bigcap_{L \in \mathcal{C}(S_t, Q_f)} L\right)$ is empty outside of $\{x_1, \ldots, x_d\}$, we have $\hat{x} \notin \bigcap_{L \in \mathcal{C}(S_t, Q_f)} L$, so there exists some language $L' \in \mathcal{C}(S_t, Q_f)$ with $\hat{x} \notin L'$. Since $L' \in \mathcal{C}(S_t, Q_f)$, it holds that $S_t \subseteq L'$. Since L' is infinite, let $x_{d+1}, x_{d+2}, \ldots$ be any enumeration of $L' \setminus \{x_1, \ldots, x_d\}$. Then $x_1, x_2, \ldots$ is a valid enumeration of L'. At timestep $t = d$, we have $|\{x_1, \ldots, x_t\}| = d \geq m$ and $G(\{x_1, \ldots, x_d\}, Q_f) = \hat{x}$. Since $\hat{x} \notin L'$, we have $\hat{x} \notin L' \setminus \{x_1, \ldots, x_d\}$, therefore G fails at timestep $t = d$. Since G, m, and Q_f were arbitrary, $\mathcal{C}$ is not m-uniformly generatable with a feedback budget of f for any m.

($\Leftarrow$) Suppose $\mathrm{FC}(\mathcal{C}, f) < \infty$ for every query budget f. We need to show that $\mathcal{C}$ is m-uniformly generatable with feedback. Fix any query budget $f \in \mathbb{N}$. Let $d = \mathrm{FC}(\mathcal{C}, f) < \infty$. We claim that $m = d + 1$ suffices.

Since d is the supremum over all numbers satisfying the condition in Def. 17, the property isn't satisfied for $d' = d + 1$. Then there exists a generator G using at most f queries such that for every $L \in \mathcal{C}$, every enumeration S_t and every timestep t with $|S_t| \geq d + 1$, for every query set Q_f, it holds that $\left(\bigcap_{L \in \mathcal{C}(S_t, Q_f)} L\right) \setminus S_t \neq \emptyset$. Hence G can always find a new string to output. Since the enumeration S_t of L must eventually satisfy $|S_t| \geq d + 1$, G generates correctly from that point onward. Thus $\mathcal{C}$ is $d + 1$-uniformly generatable with feedback, since f was arbitrary. □

Proposition 5. *Any finite collection $\mathcal{C}$ is uniformly generatable with feedback.*

Proof. Let $\mathcal{C}$ be a finite collection such that $|\mathcal{C}| = q$. By Theorem 12, it suffices to show that $\mathrm{FC}(\mathcal{C}, f) < \infty$ for all $f \in \mathbb{N}$.

By Theorem 36 of Raman et al. (2025), all finite collections are uniformly generatable. Therefore, it holds that $\mathrm{C}(\mathcal{C}) = \mathrm{FC}(\mathcal{C}, 0) < \infty$. With no feedback queries, the feedback closure reduces to the closure dimension of Raman et al. (2025).

Fix any $f \in \mathbb{N}$. Since more queries shrink the consistent set, i.e., $\mathcal{C}(S_t, Q_f) \subseteq \mathcal{C}(S_t, \emptyset)$, the intersection $\bigcap_{L \in \mathcal{C}(S_t, Q_f)} L$, is larger, making it harder for the adversary to force $(\bigcap_{L \in \mathcal{C}(S_t, Q_f)} L) \setminus S_t = \emptyset$. Therefore, $\mathrm{FC}(\mathcal{C}, f) \leq \mathrm{FC}(\mathcal{C}, 0) < \infty$ for all $f \in \mathbb{N}$. □

Proposition 4. *A finite collection $\mathcal{C}$ of infinite languages is not uniformly m-generatable with f feedback queries iff for every generator G that asks at most f feedback queries, there exist a target $K \in \mathcal{C}$, an enumeration $S_t = x_1, \ldots, x_t$ of K with $|\{x_1, \ldots, x_t\}| = m$, and answers consistent with K to the queries Q_f asked by G, such that $\langle S_t, Q_f \rangle_{\mathcal{C}}$ is finite and has size at least m.*

Proof. ($\Rightarrow$) Suppose $\mathcal{C}$ is not m-generatable with f feedback queries. Let G be any generator asking at most f feedback queries. Then there exists a language $K \in \mathcal{C}$, an enumeration $S = x_1, x_2, \ldots$ of K, answers consistent with K to the queries Q_f asked by G, and a timestep t with $|\{x_1, \ldots, x_t\}| = m$ such that $G(S_t, Q_f) \notin K \setminus S_t$. Let $X = \langle S_t, Q_f \rangle_{\mathcal{C}}$. Since S_t is a valid enumeration of K and Q_f has answers consistent with K, it holds that $K \in \mathcal{C}(S_t, Q_f)$, so $\mathcal{C}(S_t, Q_f)$ is non-empty.

X is finite: Suppose X is infinite. Since $K \in \mathcal{C}(S_t, Q_f)$ we have $X \subseteq K$, and $X \setminus S_t$ is non-empty because S_t is finite. Applying the above to the generator G^* that outputs an arbitrary unseen element of the feedback-augmented closure of its input whenever one exists, we get $G^*(S_t, Q_f) \in K \setminus S_t$, contradicting that G^* fails. So X is finite.

$|X| \geq m$: Every $L \in \mathcal{C}(S_t, Q_f)$ satisfies $S_t \subseteq L$, hence $S_t \subseteq X$. Since $|S_t| = m$, we have $|X| \geq m$.

($\Leftarrow$) Let G be any generator asking at most f feedback queries. By hypothesis there exist $K \in \mathcal{C}$, an enumeration S_t of K with $|\{x_1, \ldots, x_t\}| = m$, and answers consistent with K to the queries Q_f asked by G, such that $\langle S_t, Q_f \rangle_{\mathcal{C}}$ is finite and has size at least m. We distinguish two cases:

$G(S_t, Q_f) \notin \langle S_t, Q_f \rangle_{\mathcal{C}}$: Then there exists some language $L \in \mathcal{C}(S_t, Q_f)$ such that $G(S_t, Q_f) \notin L$. So G fails on L.

$G(S_t, Q_f) \in \langle S_t, Q_f \rangle_{\mathcal{C}}$: The adversary continues with the remaining elements of the closure, which lie in every consistent language and so leave the consistent set and the closure unchanged. The budget is exhausted, so no further queries are available. Since the closure is finite, it is exhausted after finitely many steps and G must output a string outside it, which is the previous case.

Since G was arbitrary, $\mathcal{C}$ is not m-generatable with f feedback queries. □

Theorem 5. *Let $\mathcal{C}$ be a finite collection of infinite languages and let $f < |\mathcal{C}| - 1$. Then $\mathcal{C}$ is $\max\{B(\mathcal{C}') : \mathcal{C}' \subseteq \mathcal{C}, |\mathcal{C}'| = |\mathcal{C}| - f\}$-generatable with f feedback queries.*

Proof. Let $m \geq B(|\mathcal{C}| - f)$. The generator queries as follows: while the consistent set contains two distinct languages L, L' and queries remain, it queries a string $x \in L \triangle L'$, which is non-empty since the languages of $\mathcal{C}$ are distinct. Exactly one of L, L' contains x, so whatever the oracle answers, one of the two becomes inconsistent with the target and is eliminated. After f queries at most $|\mathcal{C}| - f$ languages remain consistent.

By Prop. 4, generatability fails only if the feedback-augmented closure is finite of size at least m. That closure is the intersection of the languages still consistent, hence a finite intersection of at most $|\mathcal{C}| - f$ languages of $\mathcal{C}$, so its size is below $B(|\mathcal{C}| - f) \leq m$ by the definition of B. The condition of Prop. 4 is therefore never satisfied. □

F.1 Uncomputable bound for context-free languages

We showed that if the number of feedback queries is large enough, a collection can trivially become 0-generatable with feedback. However, if the generator still needs samples after using all the feedback budget, the number of samples is not computable.

Theorem 13. *There exist collections $\mathcal{C}$ of infinite context-free languages and numbers f of feedback queries such that there is no algorithm that outputs an m such that $\mathcal{C}$ is m-generatable with f feedback queries. The algorithm may diverge if any language in $\mathcal{C}$ is finite.*

Proof. Fix some feedback budget f such that the consistent set contains at least two languages after exhausting all f queries. We assume the consistent set is $\{L_1, L_2\}$, but the same argument can be used for more languages. Now assume, for contradiction, that there exists an algorithm that outputs m such that $\mathcal{C}$ is m-generatable with f feedback queries.

Given a Turing machine $\mathcal{M}$, we construct CFGs $\mathcal{G}_1$ and $\mathcal{G}_2$ s.t. $L(\mathcal{G}_1) = L_1$ and $L(\mathcal{G}_2) = L_2$ as in Lemma 1. We know that the finiteness of context-free languages is decidable. We distinguish three cases:

L_1 *is finite*: Choose t such that $2^t \geq |L_1| \geq |L_1 \cap L_2|$. Then $\mathcal{M}$ cannot halt in t or more steps. Simulate $\mathcal{M}$ for $t - 1$ steps and conclude that it does not halt if no halting configuration is reached.

L_2 *is finite*: Analogous to the previous case.

Both L_1 and L_2 are infinite: Apply the algorithm to obtain the number m. Choose t such that $2^t \geq m$. We claim that $\mathcal{M}$ cannot halt in $s \geq t$ steps. Suppose for contradiction that $\mathcal{M}$ halts in $s \geq t$ steps. Then, by Lemma 1, $|L_1 \cap L_2| = 2^s \geq 2^t \geq m$. This means $\mathcal{C}$ is not m-generatable with f feedback queries, contradicting the output of the algorithm. Therefore $\mathcal{M}$ cannot halt in $s \geq t$ steps, so simulating it for $t - 1$ steps suffices.

In all cases, we have decided the halting problem, so no such algorithm can exist. □

F.2 Bound for regular languages

Theorem 14. *Let $\mathcal{C}$ be a finite collection of infinite regular languages over an alphabet of size $|\Sigma| \geq 2$, such that each language $L \in \mathcal{C}$ is accepted by an NFA with at most s states. Let $f < |\mathcal{C}| - 1$. Then $\mathcal{C}$ is $|\Sigma|^{s^{|\mathcal{C}|-f}}$-generatable with f feedback queries.*

Proof. The generator has a budget of f feedback queries. It can use the following strategy. For each $i = 1, \ldots, f$, pick any two languages $L, L' \in \mathcal{C}(S_t, Q_i)$ from the current consistent set, and query a string $x \in L \triangle L'$ in their symmetric difference. The symmetric difference is non-empty whenever $L \neq L'$. Since L and L' are encoded as NFAs, we first determinize them and convert them to DFAs, which may incur an exponential blowup in the number of states. The symmetric difference can be encoded as a DFA as well. Once we have a DFA for $L \triangle L'$, finding a string in this language amounts to checking reachability in the DFA: we check whether any accepting state is reachable from the initial state. After querying this string, we can safely eliminate one language from the consistent set. After f queries, the size of the consistent set is at most $|\mathcal{C}| - f$. We therefore need $|\Sigma|^{s^{|\mathcal{C}|-f}}$ samples. □

F.3 Bounds for LTT languages

Theorem 15. *Let $\mathcal{C}$ be a finite collection of infinite LTT languages over an alphabet of size $|\Sigma| \geq 2$, such that each language $L \in \mathcal{C}$ is defined by an LTT of size at most s. Let $f < |\mathcal{C}| - 1$. Then $\mathcal{C}$ is $|\Sigma|^{3s(|\mathcal{C}|-f)((s(|\mathcal{C}|-f))^2+1)^2}$-generatable with f feedback queries.*

Proof. The generator has a budget of f feedback queries. It can use the following strategy. For each $i = 1, \ldots, f$, pick any two languages $L, L' \in \mathcal{C}(S_t, Q_i)$ from the current consistent set, and query a string $x \in L \triangle L'$ in their symmetric difference. The symmetric difference is non-empty whenever $L \neq L'$. Since L and L' are encoded as unions of profiles, their symmetric difference is expressible as a finite union of profiles, since LTT languages are closed under symmetric difference. Finding a string in this language is decidable since membership in a union of profiles is directly checkable by checking each profile in turn. After querying this string, we can safely eliminate one language from the consistent set. After f queries, the size of the consistent set is at most $|\mathcal{C}| - f$. We therefore need $|\Sigma|^{3s(|\mathcal{C}|-f)(((s(|\mathcal{C}|-f))^2+1)^2}$ samples. □

Theorem 6. *There exists a finite collection of n infinite LTT languages $\mathcal{C}$ over an alphabet of size $|\Sigma| = n \geq 4$, such that $\mathcal{C}$ is* not *2^n-generatable with f feedback queries, for any f such that $f < |\mathcal{C}| - 1$.*

Proof. Let $\mathcal{C}$ be a collection of n languages over the alphabet $\Sigma = \{1, \ldots, n\}$, where $L_i = (\varepsilon, \varepsilon, \{(j, = 1) \colon j \neq i\})$, i.e., it only leaves the symbol i unconstrained and contains exactly one occurrence of the other symbols. Each language is infinite since the symbol i can appear any number of times.

For any subset $\mathcal{S} \subseteq \mathcal{C}$ such that $|\mathcal{S}| \geq 2$, the intersection $\bigcap_{L \in \mathcal{S}} L$ forces every symbol that is constrained in at least one language to appear exactly once. Since for any two languages, every symbol is constrained in at least one of them, all n symbols must appear exactly once. Therefore, the intersection is the set of all permutations of $\{1, \ldots, n\}$, which has size $n!$.

No query eliminates more than one language: a string whose symbols all occur exactly once belongs to every L_i, a string where exactly one symbol i occurs a different number of times belongs to L_i only, and any other string belongs to none. In the first and third cases the answer is consistent with every remaining language, and in the second the adversary answers that the string is not in the target, eliminating only L_i. Hence after $f < |\mathcal{C}| - 1$ queries at least two languages remain consistent, and by the previous paragraph the feedback-augmented closure is the set of permutations, of size $n! \geq 2^n$ for $n \geq 4$. Then $\mathcal{C}$ is not 2^n-generatable with f feedback queries. □

F.4 Bound for PO languages

Theorem 16. *Let $\mathcal{C}$ be a finite collection of infinite PO languages over an alphabet of size $|\Sigma| \geq 2$, such that each language $L \in \mathcal{C}$ is accepted by a PONFA with at most s states. Let $f < |\mathcal{C}| - 1$. Then $\mathcal{C}$ is $|\Sigma|^{(|\mathcal{C}|-f)(s-1)+1}$-generatable with f feedback queries.*

Proof. Analogous to the proof for regular languages, except that after f queries the remaining consistent set of at most $|\mathcal{C}| - f$ PO languages is bounded using Theorem 2, giving $|\Sigma|^{(|\mathcal{C}|-f)(s-1)+1}$ instead. □