Speculative Decoding Speed-of-Light: Optimal Lower Bounds via Branching Random Walks

Sergey Pankratov
IST Austria

Dan Alistarh
IST Austria

Abstract

Speculative generation has emerged as a promising technique to accelerate inference in large language models (LLMs) by leveraging parallelism to verify multiple draft tokens simultaneously. However, the fundamental limits on the achievable speedup remain poorly understood. In this work, we establish the first "tight" lower bounds on the runtime of any deterministic speculative generation algorithm. This is achieved by drawing a parallel between the token generation process and branching random walks, which allows us to analyze the optimal draft tree selection problem. We prove, under basic assumptions, that the expected number of tokens successfully predicted per speculative iteration is bounded as $\mathbb{E}[X] \leq (\mu + \mu_{(2)}) \log(P)/\mu^2 + O(1)$, where P is the verifier's capacity, μ is the expected entropy of the verifier's output distribution, and $\mu_{(2)}$ is the expected second log-moment. This result provides new insights into the limits of parallel token generation and could guide the design of future speculative decoding systems. Empirical evaluations on Llama models validate our theoretical predictions, confirming the tightness of our bounds in practical settings.

1 Introduction

Speculative decoding (Leviathan et al., 2023; Chen et al., 2023; Stern et al., 2018) has emerged as a standard technique for large language models (LLMs) at scale, as it significantly reduces inference latency without altering the output distribution. The core mechanism involves using a faster **drafting process** M_q to generate a linear sequence or a tree of candidate tokens. These tokens are periodically verified in parallel by a target **verifier** LLM M_p; throughout the paper, we assume M_p is the original model we wish to speed up. By leveraging the fact that the target model can verify up to P tokens efficiently in parallel, this approach allows multiple tokens to be accepted in a single iteration of the verifier M_p.

The efficacy of this technique has spurred significant follow-up research focused on improving the drafting process, notably the EAGLE series of papers (Li et al., 2024b,a, 2025), which introduced feature-level extrapolation and dynamic draft trees, as well as Medusa (Cai et al., 2024). On the industrial side, speculative decoding is now widely supported in high-performance inference frameworks such as vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024).

Despite its widespread adoption and empirical success, the theoretical foundations of speculative decoding are not well understood. In particular, it remains unclear how close existing algorithms are to the fundamental limits of the speedup achievable through this approach. Early theoretical analysis, such as the simplified modeling provided in the original work of Leviathan et al. (2023), offered initial estimates based on the agreement rate between the draft and target models. Recently, Yin et al. (2024) provided a theoretical perspective by conceptualizing the decoding process via Markov chains and establishing instance-dependent lower bounds on the number of "rejected" (not validated) tokens. However, we still lack a framework for characterizing the maximum achievable acceleration relative to the model's properties and system constraints.

Contributions. In this paper, we provide the first tight analysis of speculative decoding, revealing a fundamental limit on the performance of any deterministic speculative generation algorithm. We establish a rigorous runtime lower bound, which is equivalent to a speedup upper bound, by proving a novel relationship between the parallelism of the system (P) and the entropy of the target model's output distribution. Specifically, we show that, given a random initial generated prefix, the expected number of tokens successfully predicted in the next speculative iteration, denoted as $\mathbb{E}[X]$,

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6404–6418

can be bounded as:

$$\mathbb{E}[X] \leq \frac{\mu + \mu_{(2)}}{\mu^2} \log(P) + O(1),$$

where μ is the expectation of the entropy of the output distribution under a random initial prefix, and $\mu_{(2)}$ is the expected second log-moment.

Further, we show that this upper bound on the "acceptance rate" of a given setup is close to the performance of actual practical systems. Specifically, since our modeling is very general, our analysis can be utilized to characterize the efficiency limits of advanced speculative generation techniques like EAGLE-3.

In more detail, we propose a simplified analytical execution model, described in Section 3.1. Assuming a target LLM M_p of fixed latency T and with parallel token capacity P, the execution is iterative: a deterministic algorithm M_q speculates a "draft" tree of tokens up to size P, which are then verified in parallel by the original model M_p. We assume the computational cost of the drafter M_q is negligible (Assumption 1) and assume that the distributions of acceptance probabilities are i.i.d. across different prefixes (Assumption 2) to facilitate the analysis.

Our main results (Theorems 1 and 2) establish the upper bound on $\mathbb{E}[X]$ under these preconditions. We show that the optimal deterministic speculation strategy involves greedily selecting the P most probable token sequences from the speculative tree (Lemma 1). Then, we show that the achievable speedup under this approach scales logarithmically with the parallel capacity P, and roughly inversely with the expected entropy μ, highlighting the diminishing returns of increased parallelism and the inherent difficulty of speculating when the model's output is highly variable.

At the technical level, the bound makes a novel connection between the speculative decoding process and the theory of Branching Random Walks (BRW). By modeling the token tree with log-probabilities, denoted by $\mathcal{U}_{\log}$, as a BRW, we leverage established theoretical tools, such as the Many-to-One Lemma (Shi, 2015), to analyze the distribution of high-probability paths within the constraint of the verification budget P. This connection provides a new analytical framework for understanding the limits of speculative decoding.

We complement our theoretical analysis with practical experiments on Llama models, across standard speculative decoding benchmarks. The results show a clear correlation between the performance upper bound predicted by our bound, and the real-world performance of well-optimized systems such as EAGLE-3. This validates the tightness of our theoretical predictions and demonstrates a new relationship between system's parallel token capacity, model entropy, and the achievable speedup in realistic settings.

2 Related Work

Speculative Decoding. The high latency of autoregressive decoding in Large Language Models (LLMs) is a significant bottleneck (Shazeer, 2019). Speculative decoding mitigates this by reducing the number of sequential calls to the target model. Building on blockwise parallel decoding (Stern et al., 2018), the technique was first formalized concurrently by Leviathan et al. (2023) and Chen et al. (2023).

Subsequent research has focused on improving the efficiency and acceptance rates of the drafting stage. One evolution has been the shift to structured, *tree-based speculation* (Miao et al., 2023), which allows the exploration of multiple potential continuations.

Some draft-generation approaches introduce architectural modifications, such as Medusa (Cai et al., 2024). The EAGLE framework (Li et al., 2024b,a, 2025) introduced feature-level extrapolation, leveraging the target model's internal representations to generate drafts, leading to significantly higher acceptance rates. Other innovations are Staged Speculative Decoding (Spector and Re, 2023) or self-speculation techniques (Elhoushi et al., 2024; Fu et al., 2024).

Theoretical Analysis of Speculative Decoding. The theoretical understanding of speculative decoding is still in its initial stages. First analyses (Leviathan et al., 2023; Chen et al., 2023) provided simplified models based on the expected agreement rate between the draft and target models, using a similar version of the i.i.d. assumption made in this paper. Sun et al. (2023) analyzed the acceptance rate through the lens of optimal transport, while Yin et al. (2024) utilized Markov chain abstractions to establish instance-dependent lower bounds.

Our work contributes to this area by analyzing the fundamental limits imposed by system constraints (parallel token capacity P) and the inherent properties of the target model, which we isolate via the entropy parameter μ. We provide bounds on the expected number of accepted tokens per iteration for any deterministic speculative algorithm, revealing the fundamental speed limits of this technique.

Branching Random Walks. Our analysis introduces a novel connection between speculative decoding and the theory of Branching Random Walks (BRW). BRW is a well-studied field in probability theory (Biggins, 1977; Shi, 2015; Lyons and Peres, 2017) analyzing stochastic processes where particles reproduce and move randomly. By modeling the token generation process as a BRW on the space of log-probabilities, we leverage established theoretical tools, such as the Many-to-One Lemma (related to spinal decomposition and measure changes; see, e.g., Shi, 2015), to analyze the optimal selection of draft trees under a system constraint. This framework allows us to characterize the fundamental trade-offs in speculative generation.

3 Model

Let $\mathcal{V} = \{x_1, x_2, \ldots, x_{|\mathcal{V}|}\}$ denote the token vocabulary, of size $|\mathcal{V}|$.

Definition 1. *A* ***target LLM*** *is a function* $M_p : \mathcal{V}^P \to \mathcal{P}(\mathcal{V})^P$*:*

- *Takes a prefix* $x_{<t}$ *and outputs the probability distribution for the next token* $p(x_t \mid x_{<t})$*.*
- *has fixed runtime* T *(see Assumption 1).*
- *Processes up to* P *tokens in parallel.* P *here denotes the* parallel token capacity*: the maximum number of tokens that can be verified by the target model before noticeable latency drop-off.*
- *Requires access to the full prior context. To predict token* x_t*, tokens* $x_{<t}$ *must be present within the current or during one of the preceding calls to* M_p*.*

The objective is defined as follows: Given a prefix, generate a sequence of N tokens $\{x_1, \ldots, x_N\}$ identical to the sequence produced by the execution of target LLM M_p. To generate a sequence of N tokens using only M_p, we must run the model N times sequentially. In other words, $\text{Time}_{\text{naive}} = N \times T$.

We can obtain a faster end-to-end runtime δNT ($\delta \leq 1$ is the inverse of speedup), by using speculative generation techniques (Li et al., 2024b; Cai et al., 2024). In this paper, we formally define the limitations of such methods with a lower bound on δ. For example, there exists a trivial bound of $\delta \geq \frac{1}{P}$. That is, all generated tokens need to be verified, and the target model can verify up to P tokens at a time. Therefore, the verifier must be run at least $\frac{N}{P}$ times.

3.1 Timing Model

We analyze algorithm M_q designed to mimic the behavior of a target LLM M_p. M_q must exactly replicate the output of the "black-box" M_p. Hence, any proposed algorithm for M_q must verify tokens by running M_p.

We adopt the following execution model: The execution is divided on iterations. During each iteration, the algorithm M_q speculates k tokens ($k \leq P$). The tokens are then verified by the target model M_p in parallel. During verification, M_p stochastically accepts a sequence. The probability of the accepted sequence is exactly the same as the probability of generating this sequence naturally by M_p. In addition, M_p generates one new token each time the verification is run.

We make a few simplifying idealizations.

Assumption 1 (Simplified Timing Model)**.** *(a) The runtime of* M_p *is a constant* T*. (b) The computational cost of* M_q*, excluding the verification calls to* M_p*, is negligible. (c) Verification calls are sequential.*

These assumptions help us connect expected runtime with Theorem 1. A brief note on the timing assumptions: (a) In practice, T increases linearly with the prefix length. Treating $T = T(N/2)$ as a constant introduces a small deviation for runtime. (b) Since we focus on the asymptotic lower bound, we disregard the overhead incurred by the drafting. Recent work (Liu et al., 2025), however, considers settings where the cost of drafting is not negligible. (C) Asynchronous verification is typically less optimal due to memory allocation overhead. To our knowledge, there are no works that show otherwise. However, it might be possible to process xP tokens faster than $\lceil x \rceil T$ (Sheng et al., 2023). Again, these simplifications affect the runtime by a constant factor.

3.2 The Bounds for Deterministic Drafting

In this section, we formulate lower bounds on δ for any reasonable **deterministic** algorithm. First, we state a simplifying assumption:

Definition 2 (Acceptance Probability). *Let $\beta(x_t|x_{<t})$ be the probability that a speculated token x_t is accepted by M_p, given that its prefix $x_{<t}$ is accepted. For a deterministic M_q, this is exactly $p(x_t|x_{<t})$ — the probability of M_p generating token x_t.*

Assumption 2 (I.i.d. Distributions). *We assume that the acceptance probabilities at each step are drawn i.i.d. across different prefixes $\beta(x_t|x_{<t}) = \beta(x_t)$. Moreover, the probability that the distribution β is point mass is < 1: $\Pr[\beta(x_t) = 1] < 1$.*

While the i.i.d. assumption is not realistic, since the context dependency is inherent in languages, it allows for a tractable proof. These proofs would give insight into the general case. The second part of the assumption excludes scenarios when M_p is deterministic, which, under our timing model, does not have a non-trivial speedup bounds.

Given the i.i.d. assumption, we can apply Wald's equation. Let X_i denote the number of new tokens accepted during iteration i. Let n denote the total number of iterations to generate N tokens. Since M_q is deterministic and the output distributions are i.i.d., X_i are also i.i.d. Then, according to the Wald's equation,

$$N \leq \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \mathbb{E}[n]\,\mathbb{E}[X_1].$$

This lets us lower-bound the speedup δ using the inverse of the expected tokens per iteration.

$$\mathbb{E}[\text{runtime}] = \delta NT \geq \mathbb{E}[\text{n}]\,T \geq \frac{NT}{\mathbb{E}[X_1]}. \quad (1)$$

To lower bound the runtime, we must upper bound $\mathbb{E}[X_1]$.

Optimal deterministic drafting strategy. We now proceed to analyze $\mathbb{E}[X_1]$, the expected number of tokens per iteration.

Definition 3 (Token Tree). *Let $\mathcal{U}$ be a weighted tree of degree $|\mathcal{V}|$ and infinite depth. Each node u in $\mathcal{U}$ represents a token u and each edge represents a possible transition taken by M_p. Each edge (u, v) of the tree is weighted with the corresponding* acceptance probability *β_v: probability of token v being accepted, given its parent is accepted. r is the root of $\mathcal{U}$ and represents the currently known prefix. For a node v, $|v|$ denotes the depth of v in $\mathcal{U}$.*

Let $P(v)$ be the probability that node v is accepted during the current iteration. It is the product of all conditional probabilities along the path from root r to v:

$$P(v) = \prod_{u \in \text{path}(r \to v)} \beta_u.$$

Definition 4 (Draft Tree). *A draft tree $Tree$ is a subtree of $\mathcal{U}$ that has at most P nodes and is rooted at r.*

Let $L(Tree)$ be the longest accepted path of a draft tree $Tree$ (including r). The expected number of accepted tokens is equal to the expected length of the accepted path in $Tree$:

$$\mathbb{E}[X_1] = \mathbb{E}[L(Tree)].$$

For example, if 3 new draft tokens are accepted, then $L(Tree) = 4$ since the root r is counted, by definition. The number of new tokens is also $X = 4$, since the verifier adds one new token to a list of accepted tokens. The optimization problem for M_q is therefore the following:

Choose a draft tree $Tree \subseteq \mathcal{U}$, which maximizes the expected length of its accepted path.

Note that, in our theoretical argument, we allow M_q to have full knowledge of $\mathcal{U}$, therefore it can always find the optimal solution. The above expectation is over the randomness in verification, not the choice of $\mathcal{U}$.

This optimization problem can be solved greedily by the following algorithm.

Lemma 1. *Any draft tree $Tree^*$ that maximizes $\mathbb{E}[L(Tree)]$ contains P nodes of $\mathcal{U}$ with the highest acceptance probability $P(v)$. Moreover, $\mathbb{E}[L(Tree^*)] = \sum_{v \in Tree^*} P(v)$.*

Proof. Recall that $|v|$ denotes the depth of the node v. Fix some solution $Tree$. We can represent the expectation of $L(Tree)$ as a sum over the depth of $Tree$.

$$\mathbb{E}[L(Tree)] = \sum_{d=1}^{\infty} \Pr[L(Tree) \geq d].$$

$L(T) \geq d$ if and only if a node at depth d in $Tree$ is accepted. Since M_p generates a unique sequence, acceptance of two nodes at the same depth are mutually exclusive events. Hence,

$$\Pr[L(Tree) \geq d] = \sum_{v \in Tree, |v| = d} P(v).$$

Which gives us $\mathbb{E}[L(Tree)] = \sum_{v \in Tree} P(v)$. This sum is maximized by $Tree^*$. There always exists a prefix-closed $Tree^*$ since the acceptance probability of a parent is never lower: for a node v and its parent u, $P(u) \geq P(v)$ since $P(v) = pP(u)$ with $p \in [0,1]$. □

Since $Tree^*$ is optimal for every fixed tree $\mathcal{U}$, algorithm M_q that drafts $Tree^*$ at each iteration is therefore optimal under Assumption 1. Lemma 1 does not require Assumption 2. The drafting strategy used by Li et al. (2024a) is identical, but includes a cap on the maximum tree depth and width for practical purposes.

Branching random walks (BRW). A branching random walk (BRW) is usually governed by a 1D point process Ξ on $\mathbb{R}$: a random variable whose support is a set of locally finite subsets of $\mathbb{R}$. Then BRW is defined as a process, where

- A root (single node of generation 0) is located in position 0.
- Each node of generation i creates children that move according to the point process Ξ from the position of the parent.

The generation i of node u is given by $|u|$ and its position by $V(u)$.

The theory of BRW is well-studied, and we will use it to characterize $\mathcal{U}$. First, let us introduce the (log-)Laplace transform for the point process Ξ. For $\theta > 0$ it is given by:

$$\psi(\theta) = \ln \mathbb{E}\left[\sum_{u:|u|=1} e^{-\theta V(u)}\right]. \qquad (2)$$

We introduce a simplified version of the many-to-one lemma.

Lemma 2 (The Many-to-One Lemma). *For a BRW with point process Ξ and any $\theta > 0$, such that $\psi(\theta) < \infty$, there exists a random walk S_n with $S_0 = 0$, such that for any $n \geq 1$, and any measurable function $g : \mathbb{R} \to [0, \infty)$, we have*

$$\mathbb{E}\left[\sum_{v:|v|=n} g(V(v))\right] = \mathbb{E}\left[e^{\theta S_n + n\psi(\theta)} g(S_n)\right].$$

Branching random walks for $\mathcal{U}$. The token tree $\mathcal{U}$ defined earlier (Definition 4) is closely related to the BRWs. That is, let $\mathcal{U}_{\log}$ be a tree $\mathcal{U}$, where the node weights β_u are replaced by $-\log(\beta_u)$. Under Assumption 2 (i.i.d. distributions), $\mathcal{U}_{\log}$ is modeled as a BRW with point process

$$\Xi_{\log} = \{-\log \beta_1, \ldots, -\log \beta_{|\mathcal{V}|}\}.$$

The position of the point v is then $V(v) = \sum_{u \in \text{path}(r \to v)} -\log(\beta_u) = -\log P(v)$. The (log-)Laplace transform defined in Equation (2) simplifies to

$$\psi(\theta) = \ln \mathbb{E}\left[\sum_{u:|u|=1} \beta_u^\theta\right].$$

The upper bound of $\mathbb{E}[X]$. Let $N(t)$ be a random variable that counts the number of nodes in $\mathcal{U}_{\log}$, whose value is $\leq t$:

$$N(t) = \#\{u \in \mathcal{U}_{\log} : V(u) \leq t\}.$$

Let μ denote the expected entropy of the output distribution and $\mu_{(2)}$ denote the expected second moment of $-\log$:

$$\mu = \mathbb{E}\left[-\sum_{u:|u|=1} \beta_u \log \beta_u\right],$$

$$\mu_{(2)} = \mathbb{E}\left[\sum_{u:|u|=1} \beta_u \log^2(\beta_u)\right].$$

By the non-deterministic assumption (Assumption 2), we have that $\mu > 0$.

Claim 1. *Assume $\mu > 0$, $\mu_{(2)} < \infty$. Let $N(t) = \#\{u \in \mathcal{U}_{\log} : V(u) \leq t\}$ denote the number of nodes of $\mathcal{U}_{\log}$ with value $\leq t$. Let S_d be a spine random walk (from Lemma 2 with $\theta = 1$) for $\mathcal{U}_{\log}$, $\mathbb{E}[h(S_1)] = \mathbb{E}[\sum_{u \in \Xi_{\log}} e^{-u} h(u)]$. Denote $U(x) = \sum_{d=0}^{\infty} \Pr[S_d \leq x]$. Then,*

$$\mathbb{E}[N(t)] = \int_0^t e^x dU(x).$$

Proof. Let $N_d(t) = \#\{u \in \mathcal{U}_{\log} : |u| = d, V(u) \leq t\}$ denote the number of nodes at layer d with value $\leq t$. So,

$$\mathbb{E}[N(t)] = \mathbb{E}\left[\sum_{d=0}^{\infty} N_d(t)\right] = \sum_{d=0}^{\infty} \mathbb{E}[N_d(t)].$$

By applying the Many-to-One lemma (Lemma 2) with identity function $\mathbb{1}(V(u) \leq t)$ and $\theta = 1$ (so $\psi(\theta) = 0$),

$$\mathbb{E}[N_d(t)] = \mathbb{E}\left[\sum_{u:|u|=d} \mathbb{1}(V(u) \leq t)\right] = \mathbb{E}\left[e^{S_d} \mathbb{1}(S_d \leq t)\right].$$

Moreover, $\mathbb{E}[S_1] = \mathbb{E}\left[\sum_{u:|u|=1} e^{-V(u)} V(u)\right] = \mu$.

Denote CDF of S_d as $F_d(x) = \Pr[S_d \le x]$. We can expand the expectation by definition:

$$\mathbb{E}\left[e^{S_d}\mathbb{1}\left(S_d \le t\right)\right] = \int_0^t e^x dF_d(x).$$

$U(x) = \sum_{d=0}^{\infty} F_d(x)$. Thus,

$$\mathbb{E}\left[N(t)\right] = \sum_{d=0}^{\infty} \int_0^t e^x dF_d(x) = \int_0^t e^x dU(x).$$

□

Lemma 3. *Assume $\mu > 0$ and $\mu_{(2)} < \infty$. For any threshold $t > 0$, the expected number of nodes in $\mathcal{U}_{\log}$, whose value is $\le t$ is:*

$$\mathbb{E}\left[N(t)\right] \le \frac{\mu + \mu_{(2)}}{\mu^2} e^t - \frac{1}{\mu} + 1.$$

Proof. By Claim 1,

$$\mathbb{E}\left[N(t)\right] = \int_0^t e^x dU(x).$$

Integrate by parts. $U(0-) = 0$ and $U(x)$ is right-continuous ($U(x+) = U(x)$), so:

$$\mathbb{E}\left[N(t)\right] = e^t U(t) - \int_0^t U(x)\, e^x dx.$$

$U(x)$ has a known bound from renewal theory (see Appendix A). Therefore, we can write $U(x) = x/\mu + C(x)$ with $C(x) \in [1, 1 + \mu_{(2)}/\mu^2]$. Substituting $U(x)$ into $\mathbb{E}\left[N(t)\right]$, we get

$$\mathbb{E}\left[N(t)\right] = \frac{\mu + \mu_{(2)}}{\mu^2} e^t - \frac{1}{\mu} + 1.$$

□

Theorem 1 (Bound on Speculative Generation)**.** *Let M_p be a LLM with latency T and parallel token capacity P. Let M_q be any deterministic algorithm that generates $\ge N$ tokens, verified by M_p. Let X denote the number of tokens successfully predicted by M_q during an iteration. $\mu > 0$ and $\mu_{(2)} < \infty$. Under Assumption 2 and for $P \ge 1 + \frac{\mu_{(2)}}{\mu^2}$, the expectation of X is*

$$\mathbb{E}[X] \le \frac{\mu + \mu_{(2)}}{\mu^2} \log(P) + O(1).$$

Or, more precisely,

$$\mathbb{E}[X] \le a \ln\left(\frac{P-b}{a}\right) + a + b,$$

where $a = \frac{\mu+\mu_{(2)}}{\mu^2}$ and $b = 1 - \frac{1}{\mu}$.

Proof. By Lemma 1, $\mathbb{E}[X] \le \mathbb{E}\left[\sum_{v \in Tree^*} P(v)\right]$. We can rewrite $P(v)$ as an integral of threshold indicator function of $V(v)$ by substituting $t = -\log x$:

$$P(v) = \int_0^{\infty} e^{-t}\mathbb{1}\left(V(v) \le t\right) dt.$$

Let $N(t)$ be a random variable that counts the number of nodes in $\mathcal{U}_{\log}$, whose value is $\le t$. Then, since $Tree^*$ contains P nodes with smallest values:

$$\begin{aligned}\sum_{v \in Tree^*} P(v) &= \int_0^{\infty} e^{-t} \sum_{v \in Tree^*} \mathbb{1}\left(V(v) \le t\right)\, dt \\ &= \int_0^{\infty} e^{-t} \min\left(P, N(t)\right)\, dt.\end{aligned}$$

Since $N(t) \ge 0$ for all $t \ge 0$, we can swap expectation and integral and then apply Jensen's ($\min\left(P, \cdot\right)$ is concave):

$$\begin{aligned}&\mathbb{E}\left[\int_0^{\infty} e^{-t} \min\left(P, N(t)\right) dt\right] \\ &= \int_0^{\infty} e^{-t} \mathbb{E}\left[\min\left(P, N(t)\right)\right] dt \\ &\le \int_0^{\infty} e^{-t} \min\left(P, \mathbb{E}\left[N(t)\right]\right) dt.\end{aligned}$$

Finally, applying Lemma 3, and denoting $a = \frac{\mu+\mu_{(2)}}{\mu^2}$ and $b = 1 - \frac{1}{\mu}$, we have:

$$\mathbb{E}[X] \le \int_0^{\infty} e^{-t} \min\left(P, ae^t + b\right) dt.$$

Let $t^* = \ln\left(\left(P - b\right)/a\right)$. Then:

$$\begin{aligned}&\int_0^{\infty} e^{-t} \min\left(P, ae^t + b\right) dt \\ &= \int_0^{t^*} a + be^{-t} dt \ + P \int_{t^*}^{\infty} e^{-t} dt \\ &= a \ln\left(\frac{P-b}{a}\right) + a + b.\end{aligned}$$

□

Theorem 1 shows that the maximum speedup of any deterministic speculative decoding algorithm M_q grows logarithmically with parallelism P. The slope of this scaling is determined by the expected next-token entropy of the target model μ: a lower entropy produces deeper speculative trees and larger speedups, while a higher entropy quickly limits the benefit of parallelism. This bound exposes a core trade-off: parallelism can increase expected latency, but its gains are fundamentally capped by the model's output uncertainty.

3.3 Asymptotic bounds

This section analyzes the behavior of $\mathbb{E}[X]$, the expected number of tokens per iteration, when $P \to \infty$. A detailed derivation is given in Appendix B.

Theorem 2 (Bound on Speculative Generation)**.** *Let M_p be a LLM with latency T and parallel token capacity P. Let M_q be an optimal deterministic algorithm that generates $\geq N$ tokens, verified by M_p. Let X denote the number of tokens successfully predicted by M_q during an iteration. $\mu > 0$ and $\mu_{(2)} < \infty$. Under Assumption 2 and assuming that the point process is non-arithmetic, the expectation of X is*

$$\mathbb{E}[X] \approx \frac{\log(P)}{\mu}.$$

Proof Sketch. We show that $\log(P)/\mu - o(\log P) \leq \mathbb{E}[X] \leq \log(P)/\mu + o(\log P)$. The upper bound follows the proof on Theorem 1, where the $\mathbb{E}[N(t)]$ estimate is replaced by its limit. Then we substitute this estimate for $t \lesssim \log(P)$. For the lower bound on $\mathbb{E}[X]$ we bound the P-th smallest value in $\mathcal{U}_{\log}$ by the same limit estimate of $\mathbb{E}[N(t)]$ and Wald's argument. We provide the complete proof in Appendix B. □

Lower bound under imperfect knowledge In previous sections, we assumed that M_q has complete knowledge of the output probabilities of the target model ($\mathcal{U}$). This allows us to derive the hard limits for any algorithm that predicts the target model M_p even with access to an oracle.

However, this assumption does not reflect how these systems operate in practice. Namely, in standard speculative decoding Leviathan et al. (2023), we would normally have a smaller draft model, trained to predict the target model's output. In turn, this motivates the question: how does the relation between the drafter predictions q and target's distribution p affect the maximum achievable acceptance rate? In this section, we analyze the performance of an algorithm M_q that only has access to distribution $q(x_t|x_{<t})$. This yields a lower bound on $\mathbb{E}[X]$ similar to Theorem 2, with cross-entropy $\mu_{CE} = \mathbb{E}[H(P||Q) \mid Q > 0] = \mathbb{E}[-\sum p \log q \mid Q > 0]$.

Informally, we obtain the following lower bound for an optimal deterministic algorithm M_q with imperfect knowledge:

$$\mathbb{E}[X] \gtrsim \min\left(\frac{1}{\mathbb{E}[Pr[q=0]]}, \frac{\log P}{\mu_{CE}}\right).$$

We formally state and prove this result in Appendix C (see Theorem 5).

In practice, $\mathbb{E}\big[\Pr[q=0]\big]$ is typically negligible. This gives the following estimate for $\mathbb{E}[X]$ for speculation without full knowledge:

$$\frac{\log P}{\mu_{CE}} \lesssim \mathbb{E}[X] \lesssim \frac{\log P}{\mu}.$$

4 Experiments

Setup and Objectives. To validate our theoretical results, we first evaluate the key expected entropy (μ) and expected second log-moment ($\mu_{(2)}$) for popular models such as Llama 3.1 8B (L3-8B) Instruct and Llama 3.3 70B (L3-70B) Instruct (Grattafiori et al., 2024), DeepSeek R1 Distill Llama 8B (DS-8B) (DeepSeek-AI, 2025), and Qwen3 8B (Q3-8B) (Team, 2025). We evaluate these parameters across several task types: code generation, conversation, math problem solving, summarization, and question answering. For robustness, we chose the benchmarks used by the EAGLE Li et al. (2024b) series of papers: HumanEval (Chen et al., 2021), MT-bench (Zheng et al., 2023), GSM8K (Cobbe et al., 2021), CNN/Daily Mail (Nallapati et al., 2016), and Natural Questions (NQ) (Kwiatkowski et al., 2019). Full experiments take a few hours on a standard multi-GPU server.

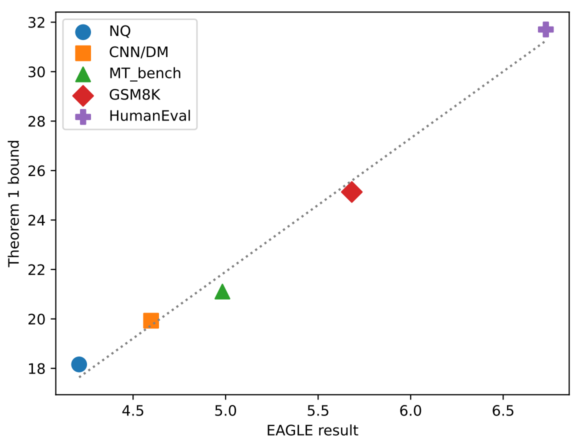

Figure 1: Empirical validation of Theorem 1 (EAGLE-3, LLaMA 3.1 Instruct (8B)) with fixed speculation size $P = 60$, across a diverse set of tasks. The interpolation line shows the near-linear correspondence between the EAGLE speedup results and our bound with a fixed constant gap of $\times 5$.

Upper bound VS EAGLE-3 Next, in Figure 1, we evaluate how well Theorem 1 upper bound

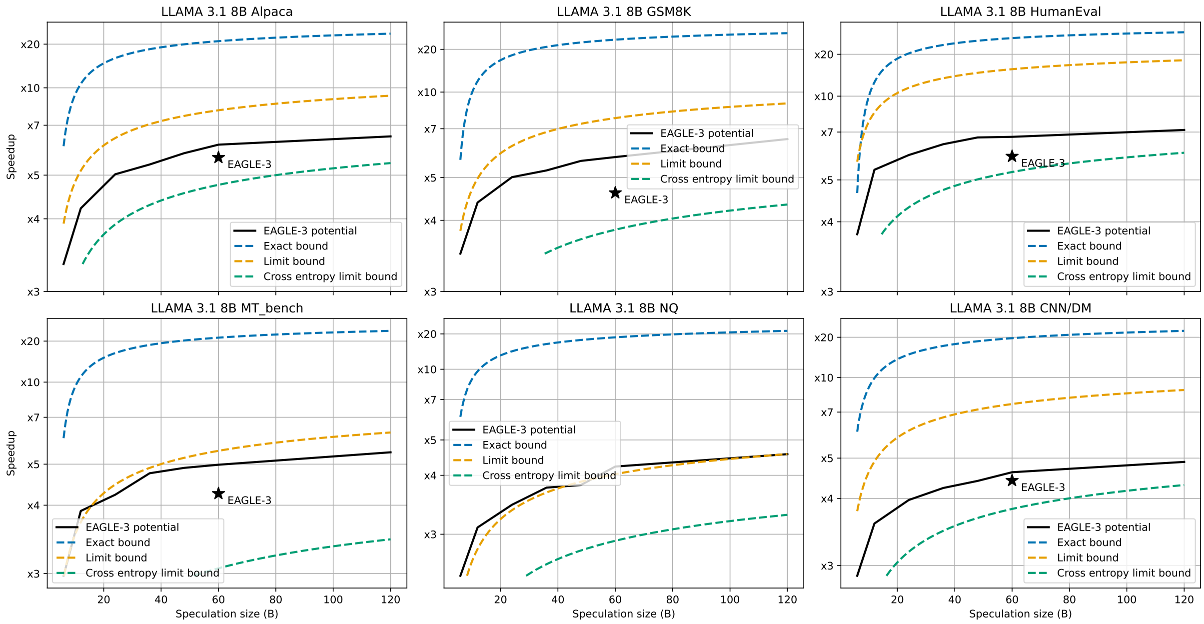

Figure 2: Comparing Theorem 2 and Theorem 5 with (EAGLE-3, LLaMA 3.1 Instruct (8B)) the $\pm o(\log P)$ term is neglected for this plot. The blue line represent the precise bound fromTheorem 1

on expected accepted tokens captures the expected number of tokens successfully speculated by EAGLE-3 on the LLaMA 3.1 Instruct (8B) model. For each iteration, we run EAGLE-3 with the sampling method described in Lemma 1, which corresponds to the original EAGLE-2 sampling procedure, but without a limit on the depth and width of the tree. This illustrates the potential of the algorithm if we neglect the speculation time (τ from Li et al. (2025)). We fix the speculation size to $P = 60$. The plot shows a clear linear relationship between our bound and the EAGLE-3 results, indicating that actual speedup scales predictably with the Theorem 1 bound across dataset. However, the bound is roughly $\times 5$ larger, indicating a constant gap between our bound and the actual performance of EAGLE-3. This is explained by our simplifying assumptions: in particular, EAGLE-3 has higher speculation error than the "oracle" speculator assumed by our lower bound.

Scaling with parallel capacity Finally, Figure 2 shows how the speedup achieved by the EAGLE-3 algorithm varies with the speculation size P when speculating on the LLaMA 3.1 8B Instruct, on 6 different benchmarks. The solid black line represents the best achievable speedup for EAGLE-3 under our simplifying assumptions. Specifically, we compute this bound by assuming a negligible speculation overhead, but use the actual EAGLE-3 acceptance rate, given their trained speculators. Further, the dashed lines corresponds to the theoretical bounds on maximal speedup. Specifically, the blue lines shows the exact upper bounds from Theorem 1; the orange lines represent the limit bounds ($P \to \infty$) established in Theorem 2; and the green line shows the lower speedup limit for the imperfect knowledge drafting of Theorem 5 (also for $P \to \infty$). For both orange and green lines, we ignore the $o(\log P)$ error, and hence these graphs should be viewed with caution. The star marker denotes the performance of EAGLE-3 according to Li et al. (2025) (configured for practical use). The star is missing from a Natural Questions benchmark since it was not presented in the EAGLE-3 paper.

First, we notice that the gap between the real EAGLE-3 performance under realistic condition (star point) and the potential bound (solid black line) is relatively low. However, EAGLE-3 still leaves a large gap to the theoretical exact lower bound, represented by the blue dashed lines. Specifically, we postulate that this stems from the gap between the optimal speculator assumed in our analytical argument and the one adopted by the EAGLE-3 implementation. Secondly, the gap between the limit line (orange) and the imperfect knowledge line (green) is mostly due to an oracle speculator assumption. Generally, this suggests that the approximately $\times 2$ difference between the theoretical bound and EAGLE-3 is caused by a speculation error. This suggests that there is still significant room for improvement with regards to designing optimal practical speculation algorithms.

5 Discussion

We established theoretical limits on the speedup achievable by deterministic speculative decoding algorithms, by drawing a novel connection to the theory of Branching Random Walks (BRW). Our main result (Theorem 1) shows that the expected number of accepted tokens per iteration scales logarithmically with the verification capacity P, which in turn implies significant diminishing returns from increasing system parallelism. Specifically, exponentially increasing the computational budget P will only yield a linear improvement in speedup, highlighting that massive parallelism alone cannot overcome the probabilistic bottlenecks of speculation.

Roughly, the bound is inversely related to the expected entropy μ of the target model. This quantifies the intuition that speculation is inherently difficult when the model's output is uncertain. Furthermore, the constant factor $(\mu + \mu_{(2)})/\mu^2$ closely bounds the impact of variability in the entropy distribution. More precisely, a higher expected second log-moment $\mu_{(2)}$ (greater variance in log-probabilities) reduces the maximum achievable speedup, indicating that the specific distribution of uncertainty impacts performance beyond just the average entropy. More generally, modeling the token tree of log-probabilities as a BRW provides a new analytical lens for understanding speculative generation. This allows us to rigorously analyze the distribution of high-probability paths under resource constraints, moving beyond previous simplified models based on average acceptance rates.

Finally, the empirical evaluations demonstrate a strong correlation between our theoretical bounds and the performance of state-of-the-art systems like EAGLE-3. This suggests that current deterministic algorithms are approaching the fundamental limits under the assumptions studied, and can guide future system design, emphasizing the importance of model characteristics (reducing effective entropy) over merely increasing the parallelism budget P.

6 Limitations

Our theoretical framework relies on a number of simplifying assumptions.

I.i.d. Assumption. The analysis depends on Assumption 2, which states that the distributions of acceptance probabilities are independent and identically distributed across prefixes. This is not exact for language models, where context dependency is inherent and output entropy will vary with the preceding text. This assumption is needed for the application of BRW theory and Wald's equation. Extending the analysis to stationary and ergodic processes, which better capture the dynamics of language modeling, remains an important direction for future work.

Simplified Timing Model. We adopt a simplified timing model (Assumption 1) that assumes a constant latency T for the verifier and negligible computational cost for the drafter. In practice, the verifier's latency increases with the prefix length (due to KV cache growth); at the same time, highly-efficient drafting mechanisms such as the one in EAGLE-3 do have negligible overhead. These assumptions allow us to isolate the fundamental limits imposed by the probabilistic verification process, but abstract away practical engineering trade-offs.

Perfect Knowledge and Deterministic Drafting. Our upper bounds are derived for the optimal deterministic drafting strategy, assuming that the drafter has perfect knowledge of the target model's output probabilities ($\mathcal{U}$). This is done for analytical purposes, as we wish to bound the performance of an ideal process. However, in reality, drafters only approximate the target distribution. We address this deeper in Appendix C by introducing a lower bound for an optimal drafter. Furthermore, the analysis is focused on deterministic drafting and does not address stochastic speculation strategies.

7 LLM Use

LLMs were used for minor editing and proofreading of the draft.

References

John D Biggins. 1977. Martingale convergence in the branching random walk. *Journal of Applied Probability*, 14(1):25–37.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In *International Conference on Machine Learning*. PMLR.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, Ahmed Aly, Beidi Chen, and Carole-Jean Wu. 2024. Layerskip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12622–12642.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. In *International Conference on Machine Learning*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Geoffrey Grimmett and David Stirzaker. 2020. *Probability and random processes*. Oxford university press.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024a. EAGLE-2: Faster inference of language models with dynamic draft trees. In *Empirical Methods in Natural Language Processing*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024b. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *International Conference on Machine Learning*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2025. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. In *Annual Conference on Neural Information Processing Systems*.

Tianyu Liu, Yun Li, Qitan Lv, Kai Liu, Jianchen Zhu, and Winston Hu. 2025. PEARL: parallel speculative decoding with adaptive draft length. In *International Conference on Learning Representations*. OpenReview.

Gary Lorden. 1970. On excess over the boundary. *The Annals of Mathematical Statistics*, 41(2):520–527.

Russell Lyons and Yuval Peres. 2017. *Probability on Trees and Networks*, volume 42. Cambridge University Press.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Rae Ying Yee Wong, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. 2023. Specinfer: Accelerating generative LLM serving with speculative inference and token tree verification. *arXiv preprint arXiv:2305.09781*.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. 2023. Flexgen: High-throughput generative inference of large language models with a single GPU. In *International Conference on Machine Learning*, pages 31094–31116. PMLR.

Zhan Shi. 2015. *Branching Random Walks*, volume 2151 of *Lecture Notes in Mathematics*. Springer.

Benjamin Spector and Chris Re. 2023. Accelerating LLM inference with staged speculative decoding. *arXiv preprint arXiv:2308.04623*.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. In *Advances in Neural Information Processing Systems*.

Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. 2023. Spectr: Fast speculative decoding via optimal transport. In *Advances in Neural Information Processing Systems*.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Ming Yin, Minshuo Chen, Kaixuan Huang, and Mengdi Wang. 2024. A theoretical perspective for speculative decoding algorithm. In *Advances in Neural Information Processing Systems*, pages 128082–128117.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. 2024. Sglang: Efficient execution of structured language model programs. In *Advances in neural information processing systems*, pages 62557–62583.

A Renewal bound

Let $X_1, X_2, \ldots$ be i.i.d. positive random variables and define the partial sums $S_d = \sum_{i=1}^{d} X_i$ with $S_0 = 0$. For $x \geq 0$, set $U(x) = \sum_{d=0}^{\infty} \Pr[S_d \leq x]$.

Claim 2. *For all $x \geq 0$,*

$$\frac{x}{\mathbb{E}[X_1]} + 1 \;\leq\; U(x) \;\leq\; \frac{x}{\mathbb{E}[X_1]} + 1 + \frac{\mathbb{E}[X_1^2]}{(\mathbb{E}[X_1])^2}.$$

Proof. Let $\widetilde{N}_{\geq}(x) = \min\{d \geq 0 : S_d \geq x\}$ denote the number of steps to reach x. Then

$$\begin{aligned} U(x) &= \sum_{d=0}^{\infty} \Pr[S_d \leq x] \\ &= \sum_{d=0}^{\infty} \Pr[\widetilde{N}_{\geq}(x) \geq d] = 1 + \mathbb{E}[\widetilde{N}_{\geq}(x)]. \end{aligned}$$

By Wald's identity,

$$\mathbb{E}[\widetilde{N}_{\geq}(x)] = \frac{\mathbb{E}[S_{\widetilde{N}_{\geq}(x)}]}{\mathbb{E}[X_1]} \;\geq\; \frac{x}{\mathbb{E}[X_1]}.$$

Lorden's inequality (Lorden, 1970) gives

$$\mathbb{E}[\widetilde{N}_{\geq}(x)] \;\leq\; \frac{x}{\mathbb{E}[X_1]} + \frac{\mathbb{E}[X_1^2]}{(\mathbb{E}[X_1])^2}.$$

Combining the two bounds completes the proof. □

B The Limit Bounds

In this section we prove Theorem 2. That is, we show that an optimal drafting algorithm under our model assumptions would successfully generate $\approx \log P/\mu$ tokens per iteration in expectation.

Non-arithmetic assumption We would call a random variable or a point *arithmetic* if its values are taken from a scaled integer set (a grid).

Definition 5. *A random variable X is arithmetic if there exist $\lambda > 0$ and c_0, such that the sample space of X is a subset of $c_0 + \lambda\mathbb{Z}$ with probability* 1.

Likewise, a point process Ξ is arithmetic if for some $\lambda > 0$ and c_0, its sample space is a subset of $c_0 + \lambda\mathbb{Z}^\infty$ with probability 1.

Assumption 3 (Non-arithmetic)**.** *The point process $\Xi_{\log}$ (defined in Section 3.2) is non-arithmetic.*

This assumption simplifies our statements. A similar result can be demonstrated in the arithmetic case, which will introduce an additional constant dependent on λ.

Upper bound To prove the upper bound, we would use a key renewal theorem.

Theorem 3 (Key Renewal Theorem (Grimmett and Stirzaker, 2020))**.** *Let $X_1, X_2, \ldots$ be i.i.d. non-arithmetic positive random variables and $S_d = \sum_{i=1}^{d} X_i$ with $S_0 = 0$. Denote $m(t) = \mathbb{E}[\max(n : S_n \leq t)]$, the renewal function. Let $g : [0, \infty) \to [0, \infty)$ be a monotonically decreasing function such that $\int_0^\infty g(x)dx < \infty$, then*

$$\lim_{t\to\infty} \int_0^t g(t-x)dm(x) = \frac{1}{\mathbb{E}[X_1]} \int_0^\infty g(x)dx.$$

Throughout this section, we are using the spinal random walk defined in Section 3.2 and Claim 1. In short, for a BRW $\mathcal{U}_{\log}$ there exists a random walk S_d called a spine random walk, whose expectation has the many-to-one property of Lemma 2. In our case, S_d can simply be imagined as a branch of $\mathcal{U}_{\log}$ chosen at random. (for more details, refer to the Spinal Decomposition Theorem, e.g., from Shi (2015)).

Lemma 4. *Assume $\mu > 0$, $\mu_{(2)} < \infty$, and $\Xi_{\log}$ is not arithmetic. Then,*

$$\mathbb{E}\left[N(t)\right] = \frac{e^t}{\mu} + o(e^t).$$

The $o(e^t)$ in the lemma can be replaced with $O(1)$, but that unnecessarily complicates the proof.

Proof. By Claim 1,

$$\mathbb{E}\left[N(t)\right] = \int_0^t e^x dU(x).$$

Denote $m(x) = \mathbb{E}[\max(d : S_d \leq x)]$. Since $m(x) = \sum_{d=1}^{\infty} F_d(x)$, we have $U(x) = m(x) + 1$ and

$$\mathbb{E}\left[N(t)\right] = e^t \int_0^t e^{x-t} dm(x).$$

Applying Theorem 3, we get

$$\lim_{t\to\infty} \frac{\mathbb{E}[N(t)]}{e^t} = \frac{1}{\mu}.$$

□

Theorem 4 (Bound on Speculative Generation)**.** *Let M_p be a LLM with latency T and parallel token capacity P. Let M_q be a deterministic algorithm that generates $\geq N$ tokens, verified by M_p. Let X denote the number of tokens successfully predicted by M_q during an iteration. $\mu > 0$ and $\mu_{(2)} < \infty$. Under Assumptions 2 and 3 and for $P \geq 1 + \frac{\mu_{(2)}}{\mu^2}$, the expectation of X is*

$$\mathbb{E}[X] \leq \frac{\log(P)}{\mu} + o(\log P).$$

Proof. By Lemma 1, $\mathbb{E}[X] \leq \mathbb{E}\left[\sum_{v\in Tree^*} P(v)\right]$. We can rewrite $P(v)$ as an integral of threshold indicator function of $V(v)$ by substituting $t = -\log x$:

$$P(v) = \int_0^\infty e^{-t} \mathbb{1}\left(V(v) \leq t\right) dt.$$

Let $N(t)$ be a random variable that counts the number of nodes in $\mathcal{U}_{\log}$, whose value is $\leq t$. Then, since $Tree^*$ contains P nodes with smallest values:

$$\begin{aligned}\sum_{v\in Tree^*} P(v) &= \int_0^\infty e^{-t} \sum_{v\in Tree^*} \mathbb{1}\left(V(v) \leq t\right)\, dt \\ &= \int_0^\infty e^{-t} \min\left(P, N(t)\right)\, dt.\end{aligned}$$

Since $N(t) \geq 0$ for all $t \geq 0$, we can swap expectation and integral and then apply Jensen's ($\min(P, \cdot)$ is concave):

$$\begin{aligned}&\mathbb{E}\left[\int_0^\infty e^{-t} \min\left(P, N(t)\right) dt\right] \\ &= \int_0^\infty e^{-t} \mathbb{E}\left[\min\left(P, N(t)\right)\right] dt \\ &\leq \int_0^\infty e^{-t} \min\left(P, \mathbb{E}\left[N(t)\right]\right) dt.\end{aligned}$$

By Lemma 4, $\mathbb{E}\left[N(t)\right] = e^t/\mu + o(e^t)$. Substituting gives us:

$$\mathbb{E}[X] \leq \frac{\log P}{\mu} + o\left(\log P\right).$$

□

Lower bound Denote T_P as the P-th smallest value in $\mathcal{U}_{\log}$. First, we give a rough asymptotic bound on $\mathbb{E}[T_P]$.

Lemma 5. *Assume $\mu > 0$, $\mu_{(2)} < \infty$, and $\Xi_{\log}$ is not arithmetic. Then,*

$$\mathbb{E}\left[T_P\right] \geq \log P - \log\log P - O(1).$$

Proof. Throughout this proof, assume P is large enough. Using the tail-sum formula for expectations,

$$\begin{aligned}\mathbb{E}[T_P] &= \int_0^\infty \Pr\left[T_P > t\right] dt \\ &= \int_0^\infty \Pr\left[N(t) < P\right] dt.\end{aligned}$$

Let $t^* = \log(\mu P) - \log\log(P)$.

$$\int_0^\infty \Pr\left[N(t) < P\right] dt \geq t^* \Pr\left[N(t^*) < P\right].$$

Next, we apply Markov inequality,

$$\Pr\left[N(t^*) < P\right] \geq 1 - \frac{\mathbb{E}[N(t^*)]}{P}.$$

Finally, by Lemma 4,

$$\frac{\mathbb{E}[N(t^*)]}{P} \leq \frac{1 + o(1)}{\log P}.$$

Which gives us:

$$\begin{aligned}\mathbb{E}[T_P] &\geq t^* \left(1 - \frac{1 + o(1)}{\log P}\right) \\ &= \log P - \log\log P - O(1).\end{aligned}$$

□

Lemma 6. *Assume $\mu > 0$, $\mu_{(2)} < \infty$, and $\Xi_{\log}$ is not arithmetic.*

$$\mathbb{E}[X] \geq \frac{\mathbb{E}[T_P]}{\mu}.$$

Proof. Recall the notation from Section 3.2. For $u \in \mathcal{U}_{\log}$, β_u is the accepted probability for node u, given its parent is accepted. $P(u)$ is the accepted probability of node u. $V(u) = -\log P(u)$ is the position of u in the branching random walk.

Let $Tree^*$ be some optimal subtree of $\mathcal{U}_{\log}$ of size P (from Lemma 1). Let $\delta(Tree^*) = \{u \in \mathcal{U}_{\log} : u \notin Tree^*, parent(u) \in Tree^*\}$ denote the *frontier* of $Tree^*$. $\delta(Tree^*)$ is a set of all possible candidates for the root of the next iteration. More precisely, $\{P(u) : u \in \delta(Tree^*)\}$ is a distribution, and $P(u)$ is the probability that u is the next token accepted by the verifier M_p. Hence, $\sum_{u\in\delta(Tree^*)} P(u) = 1$.

The maximum value in $Tree^*$ is denoted by $T_P = \max_{v\in Tree^*}(P(v))$. By construction of $Tree^*$, for every $u \in \delta(Tree^*)$, $P(u) \geq T_P$. Hence, we have

$$\sum_{u\in\delta(Tree^*)} P(u)V(u) \geq T_P.$$

Finally, by Wald's, we get

$$\mathbb{E}[\sum_{u\in\delta(Tree^*)} P(u)V(u)] = \mu\mathbb{E}[X].$$

□

As a corollary of Lemmas 5 and 6 and Theorem 4, we get Theorem 2:

$$\mathbb{E}[X] \approx \frac{\log P}{\mu}.$$

There is a nice way to interpret the formula. A static tree with P nodes and average degree e^{μ} would have depth $\mathbb{E}[X]$. That is, the relation between e^{μ} and $\mathbb{E}[X]$ is approximately the same as between the degree of a static balanced tree and its depth.

C The Bound for Deterministic Drafter with Imperfect Knowledge

In this section, we keep the model from Section 3 unchanged. The only modification is the information available to the drafter M_q. Instead of observing the verifiers distribution $p(x_t \mid x_{<t})$, the drafter sees *speculated distributions* $q(x_t \mid x_{<t})$. In particular, the drafter algorithm is oblivious to the distribution p (even from the previous iteration).

Under Assumption 2, we also assume that q are i.i.d., or formally:

Assumption 4 (I.i.d. Speculated Distributions)**.** *The speculated distributions q are i.i.d. across different prefixes $q(x_t \mid x_{<t}) = q(x_t)$. Moreover, the probability that the distribution q is point mass is < 1: $\Pr[q(x_t) = 1] < 1$.*

Let $\mathcal{U}_q$ denote a token tree with edges labeled with q instead of p. We simply write $q(v)$ to denote the probability on an edge (u, v). The *speculated probability* of a node is therefore:

$$Q(v) = \prod_{u\in \text{path}(r\to v)} q(u).$$

The *value* of a node in $\mathcal{U}_q$ is $V_q(v) = -\log(Q(v))$. The output of the optimal drafter M_q is $Tree_q^*$ — $Tree^*$ from Lemma 1 for $\mathcal{U}_q$. $T_q(P)$ denotes the P-th smallest value in U_q.

Lower bound Let $\Xi_{(q>0)} = \{u : |u| = 1,\ q(u) > 0\}$ be a set of nodes in the point process with non-zero probability of speculation. Let μ_{CE} denote the expected cross-entropy of the output distribution p and the speculated distribution q, conditioned on $q > 0$. Let $\Pr[q = 0]$ denote the probability of $\{q = 0\}$. Formally,

$$\begin{aligned}\mu_{CE} &= \mathbb{E}[-\log q(u) \mid q > 0] \\ &= \mathbb{E}\left[\frac{-\sum_{\Xi_{(q>0)}} p(u)\, \log q(u)}{\sum_{\Xi_{(q>0)}} p(u)}\right],\end{aligned}$$

$$\mathbb{E}[\Pr[q = 0]] = \mathbb{E}\left[\sum_{u:\,|u|=1} p(u)\, \mathbb{1}(q(u) = 0)\right].$$

It is common in speculative decoding to have a drafting model with a smaller vocabulary than the target model. Hence, we separate two scenarios: a) $p(x) > 0$, while $q(x) = 0$ (usually caused by out-of-vocabulary misses) and $q(x) > 0$ (normal speculative decoding). Under this setup, we can prove a lower bound for the expected number of tokens per iteration $\mathbb{E}[X]$:

Theorem 5. *Let M_p be a LLM with latency T and parallel token capacity P. Let M_q be an algorithm that generates $\geq N$ tokens. Each iteration M_q selects P most likely nodes from $Tree_q^*$ and verifies them with M_p. Let X denote the number of tokens successfully predicted by M_q during an iteration.*

Table 1: Values of the parameters μ and $\mu_{(2)}$ across models and tasks, estimated with temperature 1.0, for Llama 3.1 8B Instruct, Llama 3.3 70B Instruct, DeepSeek R1 Distill Llama 8B, and Qwen3 8B

Model	Dataset	Parameters
L3-8B	HumanEval	$\mu = 0.279, \mu_{(2)} = 0.777$
	MT-bench	$\mu = 1.088, \mu_{(2)} = 6.654$
	GSM8K	$\mu = 0.636, \mu_{(2)} = 2.912$
	CNN/DM	$\mu = 0.577, \mu_{(2)} = 1.531$
	NQ	$\mu = 0.837, \mu_{(2)} = 2.926$
	Mean	$\mu = 0.683, \mu_{(2)} = 2.960$
L3-70B	HumanEval	$\mu = 0.136, \mu_{(2)} = 0.238$
	MT-bench	$\mu = 0.179, \mu_{(2)} = 0.383$
	GSM8K	$\mu = 0.126, \mu_{(2)} = 0.199$
	CNN/DM	$\mu = 0.146, \mu_{(2)} = 0.266$
	NQ	$\mu = 0.179, \mu_{(2)} = 0.382$
	Mean	$\mu = 0.153, \mu_{(2)} = 0.29$
DS-8B	HumanEval	$\mu = 0.530, \mu_{(2)} = 1.252$
	MT-bench	$\mu = 0.633, \mu_{(2)} = 1.951$
	GSM8K	$\mu = 0.233, \mu_{(2)} = 0.531$
	CNN/DM	$\mu = 0.601, \mu_{(2)} = 1.650$
	NQ	$\mu = 0.771, \mu_{(2)} = 2.397$
	Mean	$\mu = 0.554, \mu_{(2)} = 1.556$
Q3-8B	HumanEval	$\mu = 0.178, \mu_{(2)} = 0.313$
	MT-bench	$\mu = 0.369, \mu_{(2)} = 0.814$
	GSM8K	$\mu = 0.202, \mu_{(2)} = 0.369$
	CNN/DM	$\mu = 0.328, \mu_{(2)} = 0.691$
	NQ	$\mu = 0.518, \mu_{(2)} = 1.297$
	Mean	$\mu = 0.319, \mu_{(2)} = 0.697$

$\mu > 0$ *and* $\mu_{CE} > 0$. *Under Assumptions 2 to 4,*

$$\mathbb{E}[X] \geq \min\left(\frac{1}{\mathbb{E}\big[\Pr[q=0]\big]}, \frac{\log P}{\mu_{CE}} - o(\log P)\right).$$

Proof. Recall the notation from the proof of Lemma 6. $\delta(Tree)$ is the frontier of $Tree$.

First, separate $\mathbb{E}[X]$ into two cases. Let $\widetilde{w} \in \delta(Tree_q^*)$ denote the last accepted token in this iteration. Then

$$\mathbb{E}[X] \geq \min\big(\mathbb{E}[X \big| q(\widetilde{w}) = 0], \mathbb{E}[X \big| q(\widetilde{w}) > 0]\big).$$

Case A. This can be modeled as a series of i.i.d. steps I_i. Let p_i and q_i be two distributions generated in step i. Then, $I_i = \mathbb{1}\,(q_i(Y_i) = 0)$, where Y_i is drawn with distribution p_i. Let $\tau_{q=0}$ be the first time $I_i = 1$. We can use Wald's equation to find its expectation:

$$\mathbb{E}[\tau_{q=0}]\mathbb{E}\left[I_1\right] = 1.$$

Finally, $\mathbb{E}[X \mid q(\widetilde{w}) = 0] = \mathbb{E}[\tau_{q=0}]$ and $\mathbb{E}\left[\Pr[q=0]\right] = \mathbb{E}[I_1]$.

Case B. Similar to Lemma 6, we observed that for every $u \in \delta(Tree_q^*)$, $V_q(u) \geq T_q(P)$. Then applying Wald's, we get

$$\mathbb{E}[X \mid q > 0] \geq \frac{\mathbb{E}[T_q(P)]}{\mu_{CE}}.$$

Finally, with Assumption 4, we can apply Lemma 5 to $T_q(P)$ to obtain the desired bound. □

D Entropy measurements

Table 1 shows the estimated expected entropies (μ) and expected second log-moments ($\mu_{(2)}$), for Theorem 1. We compute these values for each model and benchmark with temperature 1 for all runs. The parameters are computed over the output distributions of the target model running on the corresponding dataset (80 samples per dataset). When substituted in Theorem 1 or Theorem 2, these results provide evaluations for which models and tasks offer the highest potential for parallelization.

The results in Table 1 show both lower and more stable entropy and second-moment bounds for the larger and more accurate Llama 3.3 70B model, across all tasks. By contrast, the 8B models have both less stable and higher parameter values, across all tasks, but especially so in the multi-turn MT-Bench benchmark. In terms of our bounds, this implies a higher parallelization potential for the larger model.

Across different architectures, we observe that Qwen3 consistently exhibits a lower expected entropy (μ) across almost all benchmarks compared to Llama 3.1. According to our bounds, this suggests that Qwen3 is better suited for speculation.

E Cross-entropy lower bound VS EAGLE-3

In Figure 3, we evaluate how well Theorem 5, our lower bound on$\mathbb{E}[X]$, captures the expected number of tokens successfully speculated by EAGLE-3 across different 8B models. For each iteration, we

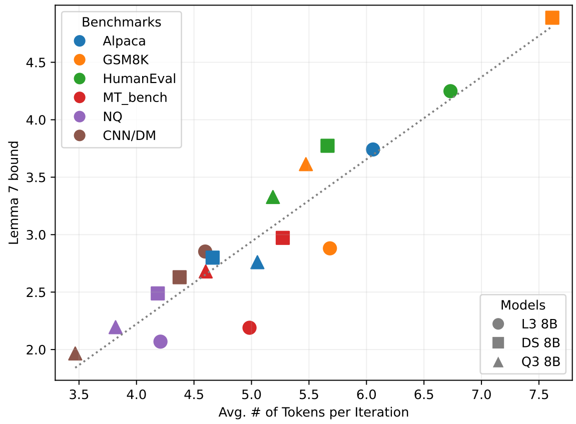

Figure 3: Validating Theorem 5 with EAGLE-3 across different models. The $\pm o(\log P)$ term is neglected for this plot. Speculation size is set to $P = 60$. The interpolation line shows the near-linear correspondence between the EAGLE speedup results and lower bound.

run EAGLE-3 with the sampling method described in Lemma 1, which corresponds to the original EAGLE-2 sampling procedure, but without a limit on the depth and width of the tree. This illustrates the potential of the algorithm if we neglect the speculation time (τ from Li et al. (2025)). We fix the speculation size to $P = 60$. The plot shows a clear linear relationship between our bound and the EAGLE-3 results, indicating that actual speedup scales predictably with the Theorem 1 bound across dataset. However, the bound is smaller, indicating a constant gap between our bound and the actual performance of EAGLE-3. This is likely the $\pm o(\log P)$ term.

Moreover, on Figure 3 the interpolation lines between different speculation models remain consistent, which shows that Theorem 5 serves as a more precise indicator of the quality of the drafters. Meanwhile, Theorem 1 and Theorem 2 establish the theoretical maximum speedup of any deterministic algorithm M_q for a given target model M_p.