Martingale Foresight Sampling: A Principled Approach to Inference-Time LLM Decoding

Huayu Li[1*] **ZhengXiao He**[1*] **Siyuan Tian**[2] **Jinghao Wen**[3]
Ao Li[1†]
[1] University of Arizona [2] Microsoft Research [3] Villanova University
`{hl459,zhengxiaohe,aoli1}@arizona.edu`

Abstract

Standard autoregressive decoding in large language models (LLMs) is inherently short-sighted, often failing to find globally optimal reasoning paths due to its token-by-token generation process. While inference-time strategies like foresight sampling attempt to mitigate this by simulating future steps, they typically rely on ad-hoc heuristics for valuing paths and pruning the search space. This paper introduces Martingale Foresight Sampling (MFS), a principled framework that reformulates LLM decoding as a problem of identifying an optimal stochastic process. By modeling the quality of a reasoning path as a stochastic process, we leverage Martingale theory to design a theoretically-grounded algorithm. Our approach replaces heuristic mechanisms with principles from probability theory: step valuation is derived from the Doob Decomposition Theorem to measure a path's predictable advantage, path selection uses Optional Stopping Theory for principled pruning of suboptimal candidates, and an adaptive stopping rule based on the Martingale Convergence Theorem terminates exploration once a path's quality has provably converged. Experiments on six reasoning benchmarks demonstrate that MFS surpasses state-of-the-art methods in accuracy while significantly improving computational efficiency. Code will be released at `https://github.com/miraclehetech/EACL2026-Martingale-Foresight-Sampling`.

1 Introduction

Large Language Models (LLMs) have demonstrated a remarkable capacity for complex reasoning, largely through their ability to generate step-by-step chains of thought (Wei et al., 2022). This technique mimics human deliberation, breaking down daunting problems into a sequence of manageable steps. By generating text one token at a time, the model is fundamentally myopic. It makes a series of locally optimal bets, choosing the most probable next word without any true awareness of the long-term consequences. This process often leads it down paths that, while promising at first, ultimately culminate in logical fallacies or factual dead ends, forcing a stark choice: settle for fast but fragile reasoning, or invest heavily in more computationally intensive methods to guide the model toward a globally coherent solution.

The first wave of solutions approached this challenge as a classical search problem. Pioneering frameworks such as Tree-of-Thought (ToT) (Yao et al., 2023) explicitly cast the reasoning process as navigation through a vast decision tree. By exploring multiple branches simultaneously, evaluating intermediate thoughts and backtracking from unpromising paths, these methods can uncover robust solutions that elude simple greedy decoding. However, this exhaustive exploration comes at a steep and often prohibitive cost. The exponential branching factor of language turns the reasoning space into an immense labyrinth, and traversing it demands computational resources that scale poorly, rendering these methods impractical for many applications.

To strike a more sustainable balance between performance and efficiency, a more recent and promising paradigm has emerged: *Foresight Sampling* (Ma et al.). Instead of blindly exploring the entire search space, this strategy intelligently peers into the future, using short, simulated rollouts to estimate the value of a potential step. The state-of-the-art implementation of this idea, *ϕ-Decoding* (Xu et al., 2025), has shown considerable success. It operationalizes foresight by combining two intuitive but heuristic scores: an Advantage score to reward immediate progress and a clustering-based Alignment score to measure consensus among different possible futures. This approach represents a significant leap forward, offering a more targeted

[*]equal contribution, names are ordered alphabetically by first name.
[†]denotes corresponding author.

and efficient exploration of the solution space.

However, beneath this empirical success lies a fragile foundation. The architecture of these advanced decoders is built upon a collection of ad-hoc design choices and clever rules of thumb. Valuation functions, dynamic pruning criteria, and alignment metrics are the products of sophisticated craft rather than a rigorous scientific principle. This reliance on heuristics creates a ceiling on progress. Such mechanisms can be brittle and may not generalize across different tasks or models and lack the theoretical guarantees required for truly reliable systems. They leave a crucial gap in our understanding of why they work and more importantly, how they might be systematically improved. This state of affairs raises a critical question for the field: *Can we move beyond heuristic-based design and formulate a principled, theoretically-grounded framework for foresight-based decoding?*

In this paper, we respond definitively with an affirmative answer by proposing a fundamental paradigm shift. We reframe LLM decoding entirely: It is not a search problem to be navigated with heuristics but rather the challenge of identifying an optimal stochastic process. For this, we turn to the elegant and powerful mathematics of *Martingale theory*, the study of fair games, and stochastic processes. This new perspective allows us to model the evolving quality of a reasoning path over time. Within this framework, an optimal path emerges as one that behaves like a *submartingale*, a process whose value is expected on average to increase at every step. Our goal is thus transformed into finding the reasoning path that represents the most favorable game.

This theoretical lens empowers us to systematically derive principled mechanisms that directly replace the ad-hoc components of prior work. Our main contributions are: **(1) A New Theoretical Framework**, we reframe LLM decoding from a heuristic search into the problem of identifying an optimal stochastic process. The goal is to find reasoning paths that behave like submartingales, ensuring their quality is expected to increase at each step; **(2) Principled Step Valuation,** we derive a single, principled score for each reasoning step from the Doob Decomposition Theorem. This predictable advantage metric replaces the heuristic Advantage + Alignment combination used in prior work; **(3) Optimal Path Pruning,** we leverage Optional Stopping Theory to create a principled method for pruning suboptimal paths during a beam search. This ensures that computational resources are focused on the most promising candidates; **(4) Adaptive Stopping Rule,** guided by the Martingale Convergence Theorem, we design a dynamic rule that stops the expensive deliberation process as soon as a path's quality has provably converged, preventing wasted computation; **(5) The MFS Algorithm,** we present the Martingale Foresight Sampling (MFS) algorithm, a concrete and efficient implementation of our theoretical framework that puts these principled concepts into practice.

By replacing fragile rules with mathematical principles, MFS represents a significant step toward building more robust, efficient, and understandable reasoning algorithms for LLMs. Our work not only introduces a high-performing new algorithm, but also provides a foundational framework for future research on principled decoding.

2 Preliminaries

2.1 Foresight Sampling for LLM Decoding

In standard auto-regressive language generation, the selection of the current token, a_t, is conditioned solely on the input query, x, and the sequence of previously generated tokens, $a_{<t}$. This can be expressed as sampling from the conditional probability distribution learned by the LLM, p_θ:

$$\hat{a}_t \sim p_\theta(a_t|x, a_{<t}) \tag{1}$$

This process is inherently myopic, as the choice of a_t is made without any awareness of its downstream consequences on the full reasoning path. Foresight sampling aims to mitigate this limitation by incorporating an estimate of future outcomes into the decoding process. Generation is conditioned not only on the past $a_{<t}$, but also on simulated future token sequences $a_{>t}$.

This concept is formalized by introducing a step value function, $R(x, a_{\leq t}, a_{>t})$, which scores a candidate token based on the quality of the future it is expected to produce. The decoding objective then becomes sampling from a modified distribution that incorporates this value:

$$\hat{a}_t \sim p_\theta(a_t|x, a_{<t}) \exp[R(x, a_{\leq t}, a_{>t})/\tau] \tag{2}$$

Here, τ is a temperature hyperparameter that controls the sharpness of the distribution. The critical challenge lies in defining the function R. A key quantity for this is the **foresight probability**, which

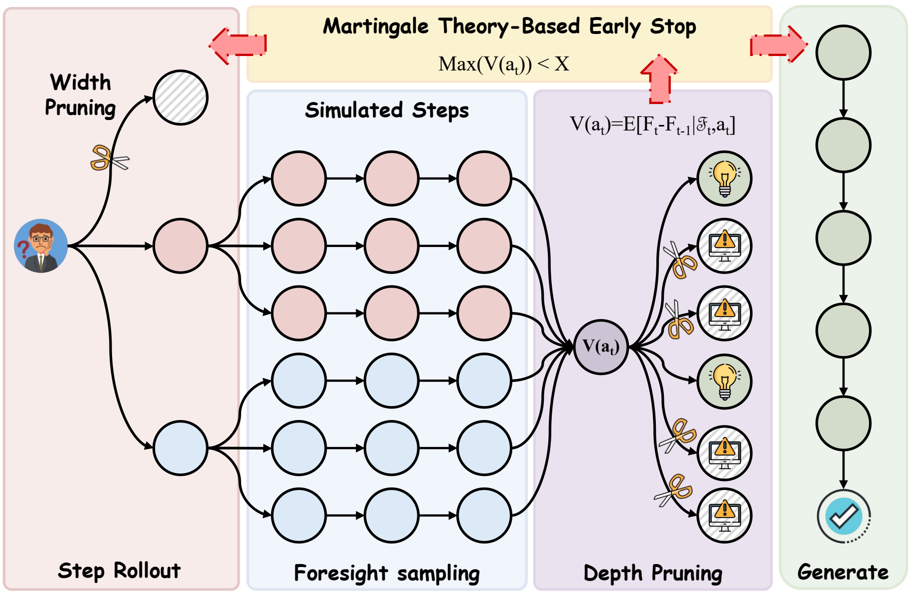

Figure 1: The overall framework of our principled MFS algorithm. We visualized the foresight decoding within one step. For clear visualization, we set the beam size as 3 and rollout number as 3.

estimates the quality of a path at step t by calculating the model's confidence in its simulated future:

$$F_t = p_\theta(a_{>t}|x, a_t, a_{<t}) \tag{3}$$

Current state-of-the-art methods like *ϕ-Decoding* construct the reward function R using heuristics based on this foresight probability, such as combining the change in F_t with scores from clustering different future paths. Our work replaces these heuristics with the principled framework of Martingale theory.

2.2 Martingale Theory

We now introduce the core concepts from Martingale theory that form the mathematical foundation of our approach.

Definition 1 (Filtration and Adapted Process) *A **filtration** is a sequence of increasing σ-algebras $(\mathcal{F}_n)_{n\geq 0}$ where $\mathcal{F}_n \subseteq \mathcal{F}_{n+1} \subseteq \mathcal{F}$ for all n. A stochastic process $(X_n)_{n\geq 0}$ is said to be **adapted** to the filtration if for every n, the random variable X_n is $\mathcal{F}_n$-measurable.*

A filtration $(\mathcal{F}_n)$ represents the accumulation of information over time. At each step n, $\mathcal{F}_n$ contains all the information known up to that point. A process being adapted simply means that its value at time n, X_n, can be determined using only the information available at that time.

Definition 2 (Martingale) *An adapted process $(M_n)_{n\geq 0}$ is a **martingale** with respect to a filtration $(\mathcal{F}_n)_{n\geq 0}$ if it has finite mean ($\mathbb{E}|M_n| < \infty$) and satisfies the condition:*

$$\mathbb{E}[M_{n+1}|\mathcal{F}_n] = M_n \quad a.s., \quad \forall n \geq 0 \tag{4}$$

*If the equality is replaced by '$\geq$' it is a **submartingale** ($\mathbb{E}[M_{n+1}|\mathcal{F}_n] \geq M_n$), and if by '$\leq$' it is a **supermartingale** ($\mathbb{E}[M_{n+1}|\mathcal{F}_n] \leq M_n$).*

A martingale models a fair game, where the expected value of your holdings at the next step, given all past outcomes, is exactly your current holdings. A submartingale models a favorable game, where you expect your holdings to increase (or stay the same), while a supermartingale models an unfavorable one.

Theorem 1 (Doob Decomposition) *Any adapted L^1 process $(X_n)_{n\geq 0}$ can be uniquely decomposed into a martingale $(M_n)_{n\geq 0}$ and a predictable process $(A_n)_{n\geq 0}$ such that $X_n = X_0 + M_n + A_n$. The increment of the predictable process is given by:*

$$A_k - A_{k-1} = \mathbb{E}[X_k - X_{k-1}|\mathcal{F}_{k-1}] \tag{5}$$

This powerful theorem allows us to dissect any stochastic process into two parts: its underlying, knowable trend (the **predictable process** A_n) and its purely random, unpredictable fluctuations (the **martingale** M_n). A process is a submartingale if and only if its predictable part, or drift is non-decreasing. This decomposition is central to our method for valuing reasoning steps.

Theorem 2 (Doob's Forward Convergence) *A supermartingale that is bounded in L^1 converges almost surely to a finite limit.*

This theorem implies that a process modeling a favorable game (our submartingale quality process) that cannot grow infinitely and is bounded from above must eventually settle down. If the process stops showing a clear upward trend and begins to behave like a martingale, it has likely converged to its limit. Continuing the process beyond this point is inefficient, as no significant improvement is expected. This provides a principled basis for an early-stopping mechanism.

Definition 3 (Stopping Time) *A random variable T is a **stopping time** if the event $\{T \leq n\}$ is $\mathcal{F}_n$-measurable for all n.*

A stopping time is a rule for deciding when to stop a process, where the decision at any time n must be based solely on the information available up to that point. You cannot peek into the future to make your decision. This concept is fundamental to making principled decisions about pruning search paths during decoding. The properties of martingales at stopping times are described by the **Optional Stopping Theorem**, which justifies that a path that has fallen significantly behind is unlikely to recover, making its pruning a sound decision.

3 Problem Formulation: Decoding as a Stochastic Process

We formally model the LLM decoding process within the Martingale framework, transforming the task of generating a sequence of tokens into the problem of identifying an optimal stochastic process. This allows us to leverage the powerful analytical tools of probability theory to guide the decoding procedure. The core components of our model are defined as follows.

- **Filtration** ($\mathcal{F}_t$): The information revealed up to step t is the sequence of generated tokens $a_{<t}$. This generates the filtration $\mathcal{F}_t = \sigma(a_0, a_1, \dots, a_{t-1})$.
  Interpretation: The filtration $\mathcal{F}_t$ is the mathematical formalization of the history available to the language model at each step of the auto-regressive process. Any decision made at step t, including the choice of the next token a a_t, must be a function of the information contained within $\mathcal{F}_t$.
- **Quality Process** (F_t): For a given reasoning path, we define its quality at step t using the foresight probability F_t. The sequence of these quality estimates, $\{F_t\}_{t \geq 0}$, forms a stochastic process adapted to the filtration $\mathcal{F}_t$.
  Interpretation: By treating the sequence of foresight probabilities as a stochastic process, we can analyze its trajectory over time. This process, F_t, quantifies the model's confidence in the future success of its current reasoning path. Our goal is to steer this process in a favorable direction.

With this formulation, the objective of inference-time optimization can be reframed from a heuristic search into a clear, mathematical goal: *To find and extend a reasoning path such that its corresponding quality process F_t is a submartingale with the steepest possible ascent.*

This objective implies that at each step t, we must select the next token a_t that ensures the quality process adheres to the submartingale property:

$$\mathbb{E}[F_{t+1} | \mathcal{F}_t] \geq F_t \tag{6}$$

This condition states that the expected quality at the next step, given the information we will have then, should be greater than or equal to our current quality. We are explicitly searching for favorable game paths, the reasoning paths that are expected to improve.

Furthermore, we aim for the steepest possible ascent. This transforms the selection of the next token into a well-defined optimization problem. Given a set of candidate tokens for the current step, we seek the token a_t^* that maximizes the expected increase in quality:

$$a_t^* = \arg\max_{a_t} \mathbb{E}[\, F_{t+1} - F_t \mid \mathcal{F}_t] \tag{7}$$

This expected increase is precisely the predictable advantage or the drift of the quality process, which we will later show can be estimated using the Doob Decomposition. By maximizing this quantity, we greedily select the path that is most like a submartingale at every step. This principled objective replaces the heuristic valuation functions of prior work and provides a solid theoretical foundation for the decoding mechanisms introduced in the following sections.

4 Martingale Foresight Sampling

Having formulated LLM decoding as the search for an optimal stochastic process, we now construct the MFS algorithm. Any practical decoder must answer three fundamental questions: (1) How should we value each potential step? (2) How do we efficiently manage multiple candidate paths in a search? (3) When should the resource-intensive deliberation process end? Our framework allows us to answer each of these questions not with heuristics but with principled mechanisms derived directly from Martingale theory. The overall framework is illustrated in Fig. 1. Once the Martingale-theory-based early stopping criterion is activated, only the remaining candidate paths are retained. These surviving paths are executed until completion and their predictions are aggregated via majority voting to produce the final answer. This design ensures that no additional output tokens are generated while also mitigating cases where the model fails to elicit an explicit answer in the output.

4.1 Principled Step Value via Doob Decomposition

The first and most critical question for any guided decoder is how to score a potential next token. Our problem formulation seeks to find the path of reasoning whose quality process, F_t, behaves as a submartingale with the steepest possible ascent. This requires a value function that accurately reflects this objective.

The Doob Decomposition Theorem, introduced in our preliminaries, provides a perfect theoretical tool for this task. The theorem states that any adapted process can be uniquely separated into its predictable trend (or drift) and its unpredictable fluctuations. A process is a submartingale if and only if its predictable trend is nondecreasing. To find the submartingale with the steepest ascent, we must therefore choose the step that maximizes this predictable, positive trend.

This insight directly yields our step value function. The value of a candidate step a_t is its predictable advantage, defined as the expected increase of the predictable component of the quality process:

$$V(a_t) = \mathbb{E}[F_t - F_{t-1} | \mathcal{F}_t, a_t] \tag{8}$$

Although this theoretical quantity is an expectation and cannot be computed exactly, it can be efficiently estimated via Monte Carlo simulation using foresight rollouts. For each candidate token a_t, we generate N simulated future paths (rollouts) and average their resulting qualities to approximate the expectation.

This single, theoretically-derived value, $V(a_t)$, replaces the ad-hoc combination of Advantage and Alignment scores used in prior work like ϕ-Decoding. It provides a unified and principled metric that is directly tied to our objective of finding a path with a positive trend.

4.1.1 Empirical Validation of Step Values

We further validate the step value by analyzing token-level log-probability statistics on correct vs. incorrect reasoning paths (LLaMA-3.1-8B-Instruct, ReClor). The results reveal a clear contrast: correct paths exhibit substantially higher predictable advantage (-0.458 vs. -1.027) and markedly lower variance (0.249 vs. 0.994). These trends confirm that successful reasoning behaves as a stable, positive-drift process, whereas incorrect reasoning is volatile. This empirically supports our submartingale formulation and justifies pruning high-variance trajectories in MFS.

4.2 Optimal Path Selection via Optional Stopping Theory

To effectively explore the vast reasoning space, multiple candidate paths must be maintained in parallel, a technique commonly known as a beam search. This introduces the second key question: How do we decide when to prune a suboptimal path to focus computational resources on more promising candidates?

We frame this decision as a stopping-time problem, as defined in our preliminaries. For each competing path i in our beam, we are deciding the optimal time to terminate further investment in its exploration. To formalize this, we define the deficit process

$$D_t^i = F_t^{\text{best}} - F_t^i, \tag{9}$$

which measures how far the path i lags behind the current best-scoring path at step t. If the best path follows a strong submartingale, while path i does not, the deficit D_t^i itself behaves as a positive drift

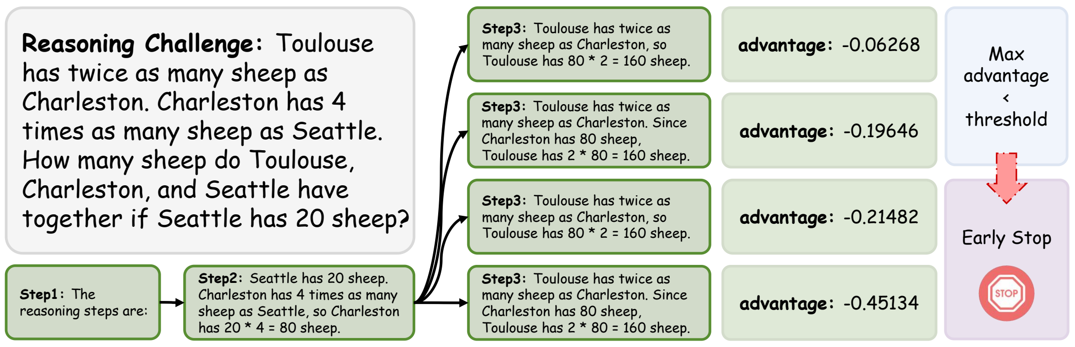
Reasoning Challenge: Toulouse has twice as many sheep as Charleston. Charleston has 4 times as many sheep as Seattle. How many sheep do Toulouse, Charleston, and Seattle have together if Seattle has 20 sheep?
Step1: The reasoning steps are:
Step2: Seattle has 20 sheep. Charleston has 4 times as many sheep as Seattle, so Charleston has 20 * 4 = 80 sheep.
Step3: Toulouse has twice as many sheep as Charleston, so Toulouse has 80 * 2 = 160 sheep.
Step3: Toulouse has twice as many sheep as Charleston. Since Charleston has 80 sheep, Toulouse has 2 * 80 = 160 sheep.
Step3: Toulouse has twice as many sheep as Charleston, so Toulouse has 80 * 2 = 160 sheep.
Step3: Toulouse has twice as many sheep as Charleston. Since Charleston has 80 sheep, Toulouse has 2 * 80 = 160 sheep.
advantage: -0.06268
advantage: -0.19646
advantage: -0.21482
advantage: -0.45134
Max advantage < threshold
Early Stop
STOP

Figure 2: Illustration of martingale-based early stopping on a GSM8K example. The deficit process triggers a stopping time once the candidate path's gap exceeds the adaptive threshold, pruning unpromising trajectories and reducing unnecessary token generation. This principled criterion ensures efficient beam management while preserving solution quality.

submartingale, implying that the gap is likely to widen as reasoning progresses.

This motivates a principled pruning rule based on an adaptive stopping time. Specifically, we define

$$T^i_{\text{prune}} = \inf t \geq 1 \mid D^i_t \geq c_{\text{prune}}(t), \tag{10}$$

where the pruning threshold $c_{\text{prune}}(t)$ is recomputed at every step t according to the empirical score distribution across the beam. Concretely, we use

$$c_{\text{prune}}(t) = \mu_F(t) + \lambda_1 \times \sigma_F(t), \tag{11}$$

where $\mu_F(t)$ and $\sigma_F(t)$ denote the mean and standard deviation of candidate scores at step t, and λ_1 is a hyperparameter controlling the sensitivity of pruning. This dynamic formulation ensures that a path is pruned only if its deficit is statistically significant relative to the contemporaneous distribution of scores.

The soundness of this mechanism is supported by Doob's Submartingale Inequality. Once a deficit D^i_t surpasses $c_{\text{prune}}(t)$, the probability that path i will ever catch up to the best path is provably bounded and decays rapidly as the gap widens. Thus, pruning is not heuristic but a theoretically principled application of optional stopping theory, providing a robust, in-search alternative to ad-hoc pre-filtering of candidates.

4.3 Adaptive Pruning via Martingale Convergence

Finally, a crucial aspect of efficiency is knowing when to stop the expensive, step-by-step foresight process altogether and revert to standard autoregressive generation to complete the answer. Continuing sampling for too long when the process is already stable is wasteful, while stopping too early harms performance. This raises our third question: what is the optimal moment to cease deliberation?

The Martingale Convergence Theorem provides a clear answer. It tells us that a bounded submartingale (our favorable game) must eventually converge to a finite limit. The practical implication is that if our quality process F_t ceases to show a clear upward trend and begins to behave like a martingale (a fair game), it has likely plateaued. Continuing the foresight process offers no further expected improvement and would waste computational resources.

This theoretical insight translates directly into an adaptive, in-depth pruning rule. Deliberation should cease when the predictable advantage of even the best possible next step falls below a small threshold, $\epsilon_{\text{stop}} \geq 0$,

$$\max_{a_t} V(a_t) \leq \epsilon_{\text{stop}} \tag{12}$$

This condition formally indicates the departure from a strict submartingale behavior, thereby yielding a principled and adaptive stopping criterion. Unlike heuristic cut-offs such as fixed search depths or clustering-based rules, our criterion dynamically adjusts to problem complexity. To further illustrate the early-stopping mechanism, we present Fig. 2. As shown, once the model has confidently derived the intermediate quantities for Toulouse, Charleston, and Seattle, the final step of summation becomes trivial and deterministically resolved. At this point, additional sampling is redundant; the framework halts exploration, substantially reducing unnecessary token consumption while preserving predictive accuracy.

Models	GSM8K	Math-500	GPQA	ReClor	LogiQA	ARC-c	Avg.	FLOPS
				LLaMA3.1-8B-Instruct				
Auto-Regressive (CoT)	70.28	31.00	26.56	49.40	33.33	58.91	44.91	1.34×10^{16}
Tree-of-Thoughts	75.74	31.60	31.25	59.00	45.93	80.72	54.04	7.03×10^{17}
MCTS	80.44	34.40	24.11	61.40	42.70	79.95	53.83	1.79×10^{18}
Guided Decoding	75.51	31.20	30.58	60.20	43.47	81.74	53.78	6.54×10^{17}
Predictive Decoding	81.43	34.00	31.03	64.00	46.70	84.56	56.95	6.89×10^{17}
ϕ-Decoding	86.58	38.20	34.60	64.00	48.39	85.41	59.53	6.43×10^{17}
Ours	**87.64**	**38.20**	**34.90**	**65.20**	**48.61**	**86.73**	**60.21**	$\mathbf{4.38 \times 10^{17}}$
				Mistral-v0.3-7B-Instruct				
Auto-Regressive (CoT)	49.05	12.20	23.88	52.20	37.02	69.54	40.65	0.81×10^{16}
Tree-of-Thoughts	53.90	10.80	26.34	55.60	41.63	73.63	43.65	4.99×10^{17}
MCTS	60.12	10.80	22.77	56.80	40.71	74.74	44.32	9.33×10^{17}
Guided Decoding	53.90	10.80	27.46	53.20	36.71	73.55	42.60	7.03×10^{17}
Predictive Decoding	58.00	11.00	22.10	54.20	39.78	73.55	43.11	4.73×10^{17}
ϕ-Decoding	60.42	16.40	29.24	58.20	43.01	78.16	47.57	3.55×10^{17}
Ours	**61.64**	**17.60**	**30.58**	**59.00**	**43.16**	**79.95**	**48.63**	$\mathbf{2.86 \times 10^{17}}$

Table 1: Reasoning benchmark results of different decoding strategies. Best results for each model are highlighted in bold.

5 Experiments

5.1 BenchMarks And Reported Metric

To comprehensively evaluate the LLM performances on diverse tasks and fight head-to-head with some of the latest methods like Predictive Decoding (Ma et al.) and ϕ-decoding (Xu et al., 2025). We evaluate on six representative reasoning benchmarks: GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), GPQA (Rein et al.), ReClor (Yu et al.), LogiQA (Liu et al., 2021), and ARC-Challenge (Clark et al., 2018). Together, these datasets span arithmetic word problems, competition-level mathematics, graduate-level factual reasoning, formal logic, and commonsense multiple-choice evaluation. We report the Pass@1 accuracy (Acc.) and FLOPS (metric for efficiency) for each benchmark as the evalutation metric. The computation formula for FLOPS is as follows (Kaplan et al., 2020): FLOPS $\approx 6nP$, where n represents the total number of output tokens and P is the number of LLM parameters.

5.2 Baselines and Backbone LLMs

We benchmark ϕ-Decoding against five representative reasoning paradigms: 1) **Auto-Regressive (CoT)** (Wei et al., 2022) is standard chain-of-thought reasoning via auto-regressive decoding; 2) **Tree-of-Thought (ToT)** (Yao et al., 2023) constructs a reasoning tree where each node denotes a step; we adopt the breadth-first search (BFS) variant; 3) **Monte Carlo Tree Search (MCTS)** (Hao et al., 2023) builds a search tree with iterative expansion and backtracking, following the *Reasoning as Planning (RaP)* framework; 4) **Guided Decoding** (Xie et al., 2023) performs stochastic beam search with self-evaluation at each step; 5) **Predictive Decoding** (Ma et al.) incorporates look-ahead strategies via model predictive control to reweight LLM distributions and mitigate myopic generation; 6) **ϕ-Decoding** (Xu et al., 2025) estimates the step value based on the joint distribution derived from foresight paths. Both in-depth and in-width pruning strategies are introduced to alleviate the overthinking issue without requiring external auxiliary models.

For evaluation, we consider six widely used reasoning benchmarks. All baselines are implemented on two mid-scale instruction-tuned LLMs: **LLaMA3.1-8B-Instruct** (Grattafiori et al., 2024) and **Mistral-v0.3-7B-Instruct** (Jiang et al., 2023). To validate our method in the latest model, we extend the experiments to **Qwen2.5-3B-Instruct** (Yang et al., 2025).

5.3 Experimental Environments And Basic Settings

All experiments were performed on two NVIDIA RTX 3090 GPUs. For inference and sampling, we adopt vLLM (v0.9.1) in conjunction with PyTorch (v2.7.0) to ensure efficient large-scale decoding. Unless otherwise specified, the decoding temperature of the LLMs of the backbone is fixed at 0.7, a widely adopted setting that balances exploration and determinism among various reasoning tasks.

Models	GSM8K	ReClor	FLOPs
LLaMA3.1-8B-Instruct			
Ours	87.64	65.20	3.70×10^{17}
w/o martingale-based early-stop	85.75	64.00	5.85×10^{17}
w/o optional path selection	86.36	61.40	4.12×10^{18}

Table 2: Ablation study on **LLaMA3.1-8B-Instruct**. FLOPs are estimated using $6nP$ (Kaplan et al., 2020), with P=8B. Step beam size and number of rollouts per step are set to 8.

For our martingale-based early stopping criterion, we set the pruning threshold at 10^{-6}, so that once all candidate trajectories cease exhibiting an upward martingale trend, further sampling is terminated. This configuration provides a principled trade-off between computational efficiency and predictive stability. As λ_1 controls the aggressiveness of variance-based pruning, we additionally examine its effect on accuracy and FLOPs (Appendix C).

5.4 Main Results

Table 1 presents the performance of our method against five representative decoding strategies on six reasoning benchmarks, using both LLaMA3.1-8B-Instruct and Mistral-v0.3-7B-Instruct as backbone models. The results clearly demonstrate that our principled approach consistently establishes a new state-of-the-art in both accuracy and computational efficiency. In LLaMA3.1-8B-Instruct, our method achieves a new state-of-the-art average accuracy of 60.21, representing a 34.1% relative improvement over the standard Auto-Regressive (CoT) baseline. It also surpasses the strongest existing foresight sampling method, ϕ-Decoding. This superior accuracy is achieved with remarkable efficiency; our method requires only 4.38×10^{17} FLOPS, which makes it 31.9% more efficient than ϕ-decoding and over 35% more efficient than MCTS. This dual advantage is consistent on Mistral-v0.3-7B-Instruct, where our method's average accuracy of 48.63 marks a 19.63% relative gain over the CoT baseline. In terms of computational cost, our approach is the most efficient among all search-based strategies, requiring 19.4% fewer FLOPS than ϕ-Decoding and 42.7% fewer than Tree-of-Thoughts.

5.5 Ablation Studies and Analysis

Ablation Studies To isolate the contribution of the core components of our framework, we conducted a critical ablation study focusing on our martingale-based early stopping mechanism, as it is central to achieving the balance between performance and computational efficiency. In Table 2, we compare our full MFS framework against a variant where this principled stopping criterion is removed or optional path selection is removed, forcing the model to continue its foresight sampling process for a fixed, extended duration. The results clearly demonstrate the dual benefit of our approach. As expected, the early stop mechanism and optional path selection produce a substantial efficiency gain, reducing the required computational cost by **36.8%** and more in terms of FLOPS. More importantly, the results show that removing this criterion also degrades accuracy on the GSM8K and ReClor benchmarks. This suggests that our principled stopping rule does more than just saving computation; it actively prevents the model from overthinking a problem, a phenomenon where continued, unguided exploration can introduce noise and lead the model away from an already-converged optimal solution. This study confirms that the adaptive stopping rule, guided by the Martingale Convergence Theorem, is essential for achieving both state-of-the-art accuracy and efficiency.

Probing Alternative Design Choices To better understand whether components of the foresight-sampling framework better than prior methods, we examine an alternative design choice at the core of candidate pruning: the step-valuation function. Prior work commonly relies on heuristic ϕ-decoding scores to prune intermediate reasoning paths, but these heuristics lack a principled connection to the underlying stochastic decision process. We instead introduce a **step-value estimator**, a Doob-style advantage function $V(a_t)$ that predicts the utility of each partial reasoning trajectory.

To isolate the effect of valuation quality, we keep all other components, including beam size, rollout depth, stopping rule, and pruning thresholds fixed. As shown in Table 3, the step-value estimator consistently improves both accuracy and computational efficiency. On **ReClor**, accuracy increases from **64.00** to **65.20** with a **1.47**$\times$ FLOPs reduction. On **MATH500**, accuracy improves from **37.20** to **38.20** alongside a **1.48**$\times$ FLOPs reduction. These results demonstrate that a valuation signal grounded in advantage estimation provides a more reliable pruning criterion than heuristic scoring, enabling more effective allocation of computation toward promising reasoning paths.

Scoring Method	**Acc** $\uparrow$	**FLOPs** ($\times 10^{17}$) $\downarrow$	**Speedup**
		ReClor	
Step-value (Ours)	**65.20**	**2.73**	**1.47**$\times$
ϕ-Decoding Score	64.00	4.01	1$\times$
		MATH500	
Step-value (Ours)	**38.20**	**10.22**	**1.48**$\times$
ϕ-Decoding Score	37.20	15.15	1$\times$

Table 3: Comparison of step-value vs. ϕ-decoding scores for candidate pruning (Beam=8, $\lambda_1 = 0.8$).

Generalization experiments on Qwen2.5-3B-Instruct To further assess the generalization ability of our framework, we evaluate MFS on the recent compact model **Qwen2.5-3B-Instruct** (Team, 2024), which is known for its strong reasoning capability. The results of two classic benchmarks, ReClor and ARC-c, are reported in Table 4. MFS yields substantial gains over both the standard autoregressive baseline and the state-of-the-art **ϕ-Decoding** (Xu et al., 2025). Notably, MFS improves ReClor by **+10.18** and ARC-c by **+7.58** over the baseline, highlighting its robustness across distinct model architectures. The improvements become even more pronounced as model quality increases, suggesting that MFS benefits more from advanced reasoning capabilities. For completeness, Algorithms 1 and 2 present side-by-side pseudocode for MFS and ϕ-Decoding.

Method	**ReClor**	**ARC-c**
AR (CoT)	53.60	77.47
Predictive Decoding	60.00 (+6.40)	78.69 (+1.22)
Guided Decoding	60.40 (+6.80)	78.34 (+0.87)
Tree-of-Thoughts	60.40 (+6.80)	78.23 (+0.76)
ϕ-Decoding	59.40 (+5.80)	79.69 (+2.22)
Ours	**63.78** **(+10.18)**	**85.05** **(+7.58)**

Table 4: Generalization experiments on Qwen2.5-3B-Instruct. Values in parentheses denote improvement over the AR (CoT) baseline.

6 Conclusion

In summary, we introduced MFS, a principled framework that reframes LLM decoding from a heuristic search into the problem of identifying an optimal stochastic process. By moving beyond ad-hoc design choices, we demonstrated that the core components of an advanced decoder can be derived directly from foundational theorems in Martingale theory. Our approach models the quality of a reasoning path as a stochastic process with the objective of finding and extending paths that behave like submartingales, those whose value is expected to increase at every step. Experiments conducted on six reasoning benchmarks show that MFS surpasses state-of-the-art methods in accuracy while simultaneously improving computational efficiency.

7 Limitations

While MFS establishes a principled and theoretically grounded decoding framework, several limitations merit discussion.

First, our current evaluation focuses primarily on convergent reasoning tasks (e.g., mathematics, symbolic logic). This is inherent to the theoretical foundations of MFS: our formulation relies on the Martingale Convergence Theorem (Theorem 2), which guarantees stabilization toward a well-defined terminal value (the correct answer). Open-ended or creative generation does not provide such a limit, making martingale-style convergence inappropriate for optimizing diversity or stylistic quality.

Second, although MFS is far more computationally efficient than classical search-based methods, it still introduces additional overhead compared with standard autoregressive decoding due to the need to simulate future rollouts. Our adaptive stopping rule,derived from martingale convergence, substantially reduces redundant computation, but the inherent foresight cost remains a constraint.

Third, the effectiveness of MFS depends on the quality of the base model's foresight probability F_t. The Doob decomposition used in MFS assumes a meaningful separation between predictable drift (reasoning signal) and the martingale residual (noise). If the underlying LLM is poorly calibrated or generates weak predictive signals, the advantage estimation and pruning decisions may degrade. Improving foresight calibration is therefore an important direction for future research.

Finally, our empirical results span six structured reasoning benchmarks. While these demonstrate strong gains, the behavior of MFS on long-form, multi-turn tasks remains unexplored.

References

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Dongsheng Jiang, Yuchen Liu, Songlin Liu, Jin'e Zhao, Hao Zhang, Zhen Gao, Xiaopeng Zhang, Jin Li, and Hongkai Xiong. 2023. From clip to dino: Visual encoders shout in multi-modal large language models. *arXiv preprint arXiv:2310.08825*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2021. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3622–3628.

Chang Ma, Haiteng Zhao, Junlei Zhang, Junxian He, and Lingpeng Kong. Non-myopic generation of language models for reasoning and planning. In *The Thirteenth International Conference on Learning Representations*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Xie. 2023. Self-evaluation guided beam search for reasoning. *Advances in Neural Information Processing Systems*, 36:41618–41650.

Fangzhi Xu, Hang Yan, Chang Ma, Haiteng Zhao, Jun Liu, Qika Lin, and Zhiyong Wu. 2025. ϕ-decoding: Adaptive foresight sampling for balanced inference-time exploration and exploitation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13214–13227.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. Reclor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations*.

```
1: Input: Query $x$, Initial beams $B_0$, Step beam size $M$, Num rollouts $N$
2: Output: Final reasoning path
3: Initialize: beams $\leftarrow B_0$, $t \leftarrow 0$
4: while not convergence_condition_met do
5:     candidate_paths $\leftarrow$ Generate $M \times N$ rollouts
6:                 ▷ —- Valuation Stage: The Core Difference —-
   // For MFS (Principled Valuation)
7:     for each path $p$ in candidate_paths do
8:                 ▷ Estimate trend via Doob Decomposition
9:         $F_t \leftarrow$ Estimate future quality
10:        $V(p) \leftarrow F_t - F_{t-1}$    ▷ Predictable Advantage
11:    end for
   // For $\phi$-Decoding (Heuristic Valuation)
12:        for each path $p$ in candidate_paths do
13:            $F_t \leftarrow$ Estimate future quality
14:            Advantage $\leftarrow F_t - F_{t-1}$
15:            Alignment $\leftarrow$ Score from clustering paths
16:            $R(p) \leftarrow$ Combine(Advantage, Alignment)
17:        end for
18:    end while
```

Algorithm 1: Decoding Loop (Part 1)

```
1:                 ▷ —- Path Selection Stage —-
   // For MFS
2: weights $\leftarrow$ softmax([$V(p)$ for $p$])
3: beams $\leftarrow$ sample($M$, paths, weights)
   // For $\phi$-Decoding
4: weights $\leftarrow$ softmax([$R(p)$ for $p$])
5: beams $\leftarrow$ sample($M$, paths, weights)
6:                 ▷ —- Convergence Check —-
   // For MFS (Martingale Convergence)
7: if $\max(V(p)) \leq \epsilon_{\text{stop}}$ then
8:     convergence_condition_met $\leftarrow$ True
9: end if
   // For $\phi$-Decoding (Consensus)
10:    if max_cluster_size/num_paths $\geq \delta$ then
11:        convergence_condition_met $\leftarrow$ True
12:    end if
13:    $t \leftarrow t + 1$
14:    Return best path from beams
```

Algorithm 2: Decoding Loop (Part 2)

A Detailed comparison versus ϕ-decoding

This section compares the proposed MFS and ϕ-decoding in detail as shown in Algorithms 1 and 2. This comparison demonstrates the fundamental architectural difference: MFS replaces the **heuristic-based components** of prior work with **principled mechanisms** derived directly from Martingale theory. Specifically, the algorithms highlight how our method substitutes the ad-hoc *Advantage + Alignment* scoring with a theoretically grounded valuation based on the *Doob Decomposition* and replaces the consensus-based stopping rule with an adaptive criterion guided by the *Martingale Convergence Theorem*. This illustrates how our principled design directly translates to superior empirical performance and efficiency.

B Parameters

This section details the hyperparameter configurations used for our MFS experiments. The complete settings for each task and the backbone model are presented in Table 5. Our framework is notably robust, requiring minimal task-specific tuning. For all experiments across all models, we kept the core search parameters constant: The size of the step beam (M), was set to 8, and the number of rolls per step beam (N) was also set to 8. The primary hyperparameter we tuned was λ_1, which controls the sensitivity of our principal pruning mechanism. As shown in the table, this parameter was adjusted within a narrow range (0.6 to 1.0) to optimize performance for specific model-task pairs.

Task	Hyper-Parameter Setup
LLaMA3.1-8B-Instruct	
GSM8K	$M = 8, N = 8, \lambda_1 = 0.8$
MATH-500	$M = 8, N = 8, \lambda_1 = 0.6$
GPQA	$M = 8, N = 8, \lambda_1 = 0.6$
ReClor	$M = 8, N = 8, \lambda_1 = 0.8$
LogiQA	$M = 8, N = 8, \lambda_1 = 0.6$
ARC-C	$M = 8, N = 8, \lambda_1 = 0.8$
Mistralv0.3-7B-Instruct	
GSM8K	$M = 8, N = 8, \lambda_1 = 0.6$
MATH-500	$M = 8, N = 8, \lambda_1 = 0.8$
GPQA	$M = 8, N = 8, \lambda_1 = 0.8$
ReClor	$M = 8, N = 8, \lambda_1 = 0.6$
LogiQA	$M = 8, N = 8, \lambda_1 = 1.0$
ARC-C	$M = 8, N = 8, \lambda_1 = 1.0$
Qwen2.5-3B-Instruct	
ReClor	$M = 8, N = 8, \lambda_1 = 0.8$
ARC-C	$M = 8, N = 8, \lambda_1 = 0.8$

Table 5: Experimental setup of our martingale-principled decoding. M denotes the step beam size, N is the number of rollouts per step beam, and λ_1 is the parameter for principled pruning.

C Analysis of the Pruning Coefficient λ_1

The coefficient λ_1 controls the aggressiveness of our variance-based pruning rule (Eq. 11). Lower λ_1 retains more candidate trajectories, leading to higher computational cost but stable accuracy; higher λ_1 tightens the pruning threshold, removing noisy trajectories earlier.

Table 6 reports the accuracy–efficiency trade-off across ReClor and MATH500 under different λ_1 and beam sizes. We observe that $\lambda_1 = 0.8$ achieves the best balance, reducing FLOPs by approximately

λ_1	b	**ReClor Acc** (↑)	**ReClor FLOPs** (↓)	**MATH500 Acc** (↑)	**MATH500 FLOPs** (↓)
0.6	4	61.0	0.70	35.0	2.72
0.6	6	62.4	1.81	37.0	5.65
0.6	8	64.0	3.42	**38.2**	10.22
0.8	4	61.8	0.62	33.8	1.80
0.8	6	64.3	1.52	34.4	4.19
0.8	8	**65.2**	**2.73**	37.2	6.92

Table 6: Ablation on pruning coefficient λ_1 and beam size b for LLaMA3.1-8B-Instruct on ReClor and MATH500. ↑ higher is better, ↓ lower is better.

20% while slightly improving accuracy. This supports using a moderately strict variance threshold to prevent wasted compute on unstable reasoning paths.